NEXOGENESIS
STREAMLINING WATER RELATED POLICIES

D4.1 Self-learning nexus engine specifications and technical design

Lead: Lluís Echeverria (EUT)

Date : 30/06/2022

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Project Deliverable

Project Number	Project Acronym	Project Title
101003881	NEXOGENESIS	Facilitating the next generation of effective and intelligent water-related policies, utilizing artificial intelligence and reinforcement learning to assess the water-energy-food-ecosystem (WEFE) nexus

Instrument:	Thematic Priority
H2020 RIA	LC-CLA-14-2020

Title
Self-learning nexus engine specifications and technical design

Contractual Delivery Date	Actual Delivery Date
M10: June 2022	M10

Start Date of the project	Duration
01 September 2021	48 months

Organisation name of lead contractor for this deliverable	Document version
EUT	1.0

Dissemination level	Deliverable Type
Public	**Report**

Authors (organisations)
Lluís Echeverria (EUT), Nuria Nievas (EUT) and Josep Pijuan (EUT)
Reviewers (organisations)
Lydia Vamvakeridou-Lyroudia (KWR)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Abstract

The NEXOGENESIS (NXG) objective in WP4 is to develop the Self Learning Nexus Assessment Engine (SLNAE) platform to enable an intelligent assessment in the WEFE policy-making scenario. In this line, the present document reports the first steps towards this goal.

The six WP4 pillars are introduced, which identify the fundamental Self-Learning Nexus Assessment Engine (SLNAE) components and link them with other NGX WPs outputs: i) the WEFE Policy framework, ii) WEFE Goals, Targets and Indicators, iii) the Nexus complexity modelling, iv) the Decision Support System functionalities, v) the Data Sharing Tools, and vi) the Graphical User Interface.

In order to proceed with the SLNAE design, first, the use case methodology has been implemented to collect user stories and later to identify the required functionalities the platform must offer. On this basis, the general SLNAE platform architecture and all the components have been designed, and the required technologies to achieve so are identified. A deeper analysis has been carried out to define the algorithmic methodologies and NXG WP4 research in the self-learning engine, which will be based on novel AI, ML and control theory techniques.

Finally, the NXG Data Sharing Tools system has been also designed following the same approach (use cases, requirements and final design) to enable a flexible cross-WP data exchange.

The present document corresponds to the first version (v1.0) of deliverable **D4.1 Self-learning nexus engine specifications and technical design.** In M24, a second version (v2.0) will be presented where the stakeholders' views and feedback will be considered to improve the use cases, requirements and ultimately the designs presented in v1.0. With this, it is expected to generate a more useful and valuable tool for SHs and final users.

Related Deliverables:
There are no related deliverables

Keywords
SLNAE, AI, DSS, UI, design

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Abbreviation/Acronyms

AI	Artificial Intelligence
CS	Case Study
CSS	Cascading Style Sheets
DL	Deep Learning
DMP	Data Management Plan
DNN	Deep Neural Networks
DLR	Deep Reinforcement Learning
DSS	Decision Support System
GUI/UI	Graphical User Interface/User Interface
HTML	Hyper Text Markup Language
HTTPS	Hypertext Transfer Protocol Secure
ICT	Information and Communication Technologies
IDR	Internal Data Repository
JS	JavaScript
JSON	JavaScript Object Notation
JWT	Json Web Tokens
ML	Machine Learning
MS	Milestone
NXG	Nexogenesis project
NXGT	Nexogenesis tool
ORM	Object Relational Mapper
SDGs	Sustainable Development Goals
SDM	System Dynamic Model
SLNAE	Self-Learning Nexus Assessment Engine
SH	Stakeholder
REST	Representational State Transfer
RL	Reinforcement Learning
RP	Reference Pathway
WEFE	Water-Energy-Food-Ecosystem

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Contents

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Figures

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Tables

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

1. Introduction

The Self Learning Nexus Assessment Engine (SLNAE), for simplification also known as the Nexogenesis Tool (NXGT), is a platform aimed to provide support in WEFE (Water, Energy, Food and ecosystems) policy-decision-making scenarios. It considers the Nexogenesis (NXG) holistic approach for nexus governance to propose integrative policies with the aim to maximise global policy goals. At its core, Artificial Intelligence (AI) and Machine Learning (ML) algorithms enable the possibility to effectively operate the complex WEFE interlinkages and provide optimal policy advice for improved resources management.

In this context, the SLNAE tool will be used both during the NXG project time and after its finalization. In the first case, the engine will be used along with the proposed NXG co-creation framework for nexus governance and WEFE resource management. Figure 1 presents this stage of the cocreation framework, where the following iterative pipeline is defined: i) Nexus policies and targets are defined together with WP1 and SHs (T1.4), ii) Proposed policies are integrated into the complexity science tools developed in WP3 (T3.3), iii) Optimal policy packages are built by the self-learning engine in WP4 (T4.4), and iv) SHs and WP1 validate policy packages recommendations (T1.4). When possible (e.g. in front-runners CSs), two rounds of this methodology will be applied. Further information regarding this process can be found in WP1 deliverables.

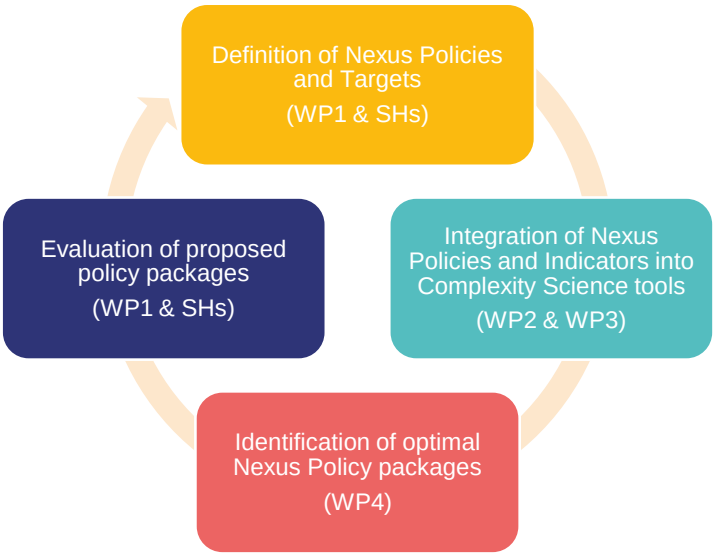

Figure 1. NXG co-creation framework for Nexus Policy packages identification

At a later stage, during the project, the SLNAE capabilities will be extended with ICT technology and functionalities (e.g. a Graphical User Interface or an Open Semantic Repository) to let the final users autonomously interact with the NXG research and developments linked to the NXG CSs. In the end, a completely online platform will be developed which will integrate all the NXG outcomes to help the community to understand the WEFE nexus interlinkages and policy impacts and implications.

In parallel, the SLNAE will also provide support to the NXG project in terms of data-sharing tools. A common data repository will be deployed to centralize all the data created during the NXG project, which considers, at least, the following entities: i) policies designed in WP1 and

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

WP5, ii) outputs of nexus scenario simulations generated in WP2, iii) complexity science tools developed in WP3, and iv) the definition of policy goals and targets, WEFE footprint indicators, and CS specific indicators in WP1 and WP5. This system will contribute to a better cross-WPs organization and synchronization. Furthermore, all this information and data will be considered under the Data Management Plan (DMP) umbrella, and will be public (in those cases where no restrictions are identified) through the NGX open repositories.

With the previous general objectives in mind, it is of utmost importance to proceed with a detailed disaggregation and description of the functional and non-functional requirements required by the proposed solution. Later, once identified, those requirements are translated into system functionalities and ultimately incorporated into the platform design. In this scenario, the design of the SLNAE tool also takes advantage of the NXG cocreation methodology, since both consortium expert partners and SH are taken into account in this process.

In this context, this document represents the first version of the deliverable D4.1, and includes all the use cases, requirements and designs obtained till now. During the first half of the project, this design will be shared with the SHs and their vision and feedback will be collected (mainly in workshops WS2 and WS3). Consequently, this document will be accordingly iterated. As a result, in M24 the final version of D4.1 will be generated, which will cover the whole SLNAE platform design by taking also into account the SHs view and advice (M24). Mainly, the topics that are most likely to be extended are those where SHs' feedback is considered, such as the SLNAE GUI or the DSS functionalities.

1.1. Structure

The document is structured as follows: Section 2 introduces the WP4 pillars, those key components that will guide the SLNAE platform development. Next, Section 3 presents the methodology used to collect and define the use cases, which are later used in Section 4 to identify all the SLNAE requirements. On this basis, Section 5 develops the basis of the SLNAE platform. Finally, Section 6 ends with the conclusions and next steps.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

2. Pillars

In order to start the collection of requirements for the SLNAE and ultimately proceed with its design, the fundamental components of the tool have been classified into pillars. A pillar represents a principal element that constitutes a basis to support the structure of the engine. Each pillar has its own needs and functionalities. Most of the pillars are represented by a single component, however, the GUI pillar is compounded by several ones.

The six WP4 pillars are introduced, which identify the fundamental Self-Learning Nexus Assessment Engine (SLNAE) components and link them with other NGX WPs outputs: i) the WEFE Policy framework, ii) WEFE Goals, Targets and Indicators, iii) the Nexus complexity modelling, iv) the Decision Support System functionalities, v) the Data Sharing Tools, and vi) the Graphical User Interface.
The identified pillars are presented below.

4.1. WEFE Policy Framework

The NXG cross-sectorial policy-making framework is a critical input and component in WP4. It defines the available set of policies that will be used by the AI self-learning engine to generate optimal WEFE policy packages. Furthermore, in a second phase, it will also contain all the CSs' policies that will be available in the SLNAE platform to be used by the users in the WEFE simulations.

Policies will have different parameters for the definition of their applicability and logic. The qualitative impacts of policies in the nexus sectors will be translated into quantitative effects through the SDMs. In this context, the SDMs will be able to simulate any policy, no matter if it is regional or not. However, there will be a limited number of combinations of policies since not all of them make sense. Therefore, a list of available policies and permitted combinations should be made per CS. Moreover, depth analysis of the effect of policy packages on the SDMs variables should be presented in order to understand the implications and relation between nexus sectors and impacts. In summary, this pillar covers the integration of policies per each CS, the best policy packages per CS, the internal policy parameters to be considered in the application of each policy, and the external policy parameters that will define the simulation logic.

Some of the questions that need to be solved in collaboration with WPs, CSs and SHs regarding the WEFE policy framework component are: Which policies are relevant per sector and CS? The policies to be implemented should be policies that already exist, or new policies/instruments can be proposed? Which are the capabilities of the simulations regarding policy implementation? What type of policies will be implemented? Which combination of policies (policy package) is reasonable and coherent to be implemented? And finally, which type of policy parameters will be available for the end-users' policy package selection and definition? All this information will be provided by CS working in close collaboration with WP1 and WP5.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

4.2. WEFE Goals, Targets & Indicators

Another critical element in WP4 is the WEFE Goals, Targets & Indicators pillar. A policy goal defines a strategic objective for a particular policy or set of policies, usually related only to one nexus sector. Policy goals are found in official policy documents and may be different per CS. Targets make policy goals measurable by quantifying them. A target will be a reference point to see how users are getting closer to a certain ambition. Moreover, users may select more ambitious targets than the ones established by regulations. The achievement of targets will be assessed by a group of metrics (indicators) previously associated with the specific targets. There will be a set of common indicators across CSs and specific indicators per each CS. Common indicators will be selected by the project partners to allow comparison between CSs, and the case-specific indicators will be co-decided with SHs in each CS based on what is prescribed by policies and on SHs' preferences. Indicators will be calculated using the outputs from the SDMs in the SDM.

Furthermore, composite indicators will be considered to combine one or more indicators on a more general level to summarize the status of each nexus sector. There is no need to involve SH at this stage; it is the job of the project to come up with meaningful composite indicators.

Finally, the WEFE footprint indicators represent another element in the WEFE Goals, Targets & Indicators pillar that will consider the ecosystem to yield a composite index that utilises the nexus as its conceptual framework related to Sustainable Development Goals (SDGs). These indicators will be common for all CSs in the SLNAE, and will enable cross CSs comparisons. Data will be normalized in order to allow this comparison. The availability of data, what data is available, and what data is relevant needs to be considered in the development of these indicators (WP3).

Goals, WEFE targets, WEFE indicators, composite indicators, and WEFE footprint indicators must be well defined to evaluate the applications of policy packages, and allow comparison between different situations, reference pathways, and CSs. All this information will be provided by WP1, WP3 and WP5.

4.3. Nexus complexity modelling

This pillar encompasses all the issues related to the implementation of functionalities in the SDMs. SDMs functionalities should be defined in the early stages of the project since STELLA SDMs will be translated into Python language in an automatic procedure for their integration and execution in the SLNAE tool. The main mechanisms that will be used in the SDMs have been specified to be stocks, flows, converters, and an array of converters [1]; but others like delays may be also analyzed. All mechanisms must be known and considered in the translation process.

Moreover, policy package implications will be represented in SDMs. They will affect some SDMs variables or projections. However, higher logic of policy behaviour will be implemented in the SLNAE whenever it becomes easier. For that, policy parameters and policy typologies used to define a policy should be confirmed in advance.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Similarly, all variables needed to compute indicators and low-level indicators will be provided by the SDMs. However, high-level indicators and targets will be computed outside the SDMs, in the SLNAE, according to end-users requests.

Finally, uncertainty must be considered in the SDMs and the SLNAE (both in the GUI and the self-learning engine processes).

Therefore, the SDM component will comprise all use cases related to the STELLA mechanisms, policy packages implementation and logic, SDMs needs for the computation of indicators, and the implementation of uncertainty through the SDMs. In the latter case, for each SDM input variable, there will be a data distribution specification (e.g. mean and standard deviation in a normal distribution) instead of a time series.

4.4. DSS functionalities

The Decision Support System (DSS) pillar is based on the self-learning engine, and has two main functionalities: the policy identification in the SH co-creation process to evaluate and support the policy package definition (DSS functionalities - co-creation process), and the end-user decision support system through the SLNAE UI (DSS functionalities - SLNAE).

The first functionality aim is to help in the iterative process to identify the relevant policy packages for their evaluation together with SHs under the co-creation framework. This functionality will be considered during the project execution.

The second functionality is to recommend valuable policy packages to the users in order to assist their analysis and decision-making process. Recommendations will be made based on the default targets and the user-defined targets in the tool. Policies may be applied with different start and end parameters, range of application, and other additional decision variables. Every freedom in the definition of policies complicates the most favourable selection of parameters to accomplish and satisfy all trade-offs in policy goals. All use cases related to desired recommendations must be specified in the DSS component such as the number of recommendations expected, the details that are given per each recommendation, and the desired response time per execution.

4.5. Data Sharing Tools

The Data Sharing Tools pillar is defined to allow communication between all partners in the sense of data and model sharing, and to automate the updating of models and shared data used by each of the WPs that make up the SLNAE tool. A protocol will be needed to manage the process to upload new data versions to the platform, like file versioning and folder structure, and to notify the required people in the NXG pipeline.

4.6. Graphical User interface (GUI)

The GUI pillar represents an interactive system with visual components and graphical representations that enable the SHs, and more specifically, the end-users to access the SLNAE platform. Thus, the objectives and sections of the GUI are identified based on end-user needs.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Therefore, the first step has been the recognition of potential users in which two particular groups have been identified based on specific presentation of the information: technical users and strategic users. The two types of views (related to each group of potential viewers) should be interlinked, and the users should have the option to move from one view to the other in each scenario. Each type has a different level of detail and a different way of visualization. The information is available to everybody with no restriction on different users. The two types of views are:

- *Technical view*: This view is given for users with some level of knowledge and expectation of an extensive and detailed study. It enables a depth analysis of the implications and impacts of policies in the WEFE sectors and variables, numerical evaluations, cause-effect relations, and statistics for scientific reasoning. Examples of technical users are scientists and academics.
- *Strategic view*: This view is given for the users that expect synthesized and clear information with colourful simple graphics and diagrams. Visualization should comprise as few words and numbers as possible of the simulation results and present the indicators evaluation and the required comparisons. Examples of technical users are policymakers, authorities, associations, and general users.

Based on the specific aims of the GUI, use cases will be classified into different components:

- *User interface – General*: This component is referring to the initial user interface where an explanation of the project and its methodologies will be given, information about the CSs will be presented, and the user will be able to sign up, log in, and manage its profile, setting, and other user's functionalities.
- *User interface –Simulations Management*: This user interface presents the option to start new simulations, access the saved ones, duplicate/edit/remove saved simulations, and presents the link to the results and comparisons of simulations.
- *User interface – Simulations Configuration*: There will be two stages to configuring and launching a simulation. The configuration of a simulation is based on the selection of CS, the selection of the starting point or reference pathway, the specification of whether uncertainty is applied or not in the simulation process, and the selection of user-defined goals, targets, and indicators for the evaluation of policy packages. In each CS, there will be fixed goals and targets either mandated by law or defined by SHs.
- *User interface – Simulation*: The other stage is the simulation itself. It will be a new screen with the simulation view where the user will be able to select available policy packages and define the simulation step. The Decision Support System (DSS) will be considered at this stage given recommendations under the user's requests.
- *User interface – Simulation Results*: The affected variables from the SDMs, indicators, and visualizations should be provided at each step of the simulation process presenting how the application of a policy from a nexus sector affects another nexus sector. All variables' relations should be integrated into the SDMs.
- *User interface – Simulations Comparison*: Another important component to consider in the GUI is the comparability between different policy package executions and their implications in the indicators and targets to achieve the policy goals. The GUI workflow

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

may be a mix between a step-by-step visualization and a dashboard to present results and comparisons. Comparisons between different simulations in the same CS and against different CSs are needed.

This is the pillar that may suffer more modifications in its design during the first two years of the project (M24). This is because the SHs' contributions and views are expected with the aim to generate a tool useful for them. WP4 will focus workshops WS2 and WS3 on these topics. In terms of planning, there will be no delays since T4.5 is expected to start at M24.

Additionally, during the September 2022 annual project meeting in Riga (M12), there will be a workshop to co-design in more detail the visualization tool, which will be detailed in the next version of this Deliverable. Moreover, mock-ups will be presented to help with the SLNAE GUI definition process.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

3. Use cases

This section describes the use cases methodology implemented in WP4 to facilitate the identification of needed features in the development of the SLNAE. It aims to identify, clarify and organize system requirements with three main essential elements:

- *The actor*. The system user, being a single or group of users
- *The goal*. The successful outcome after completing a process
- *The system*. The needed steps to achieve the goal, including necessary preconditions or other requirements

A complete use case includes one main but also alternative flows to achieve a goal. For this reason, for each of the flows, it is important to highlight what triggers it and which are the preconditions needed. The definition of use cases should be technology agnostic, referring only to the business logic needed by actors to achieve the goal, allowing developers to use the best technology according to the list of requirements. The most detailed the use cases can be provided, the most completed list of requirements will be achieved. In this scenario, Figure 2 presents the schema of the proposed methodology to collect the required information.

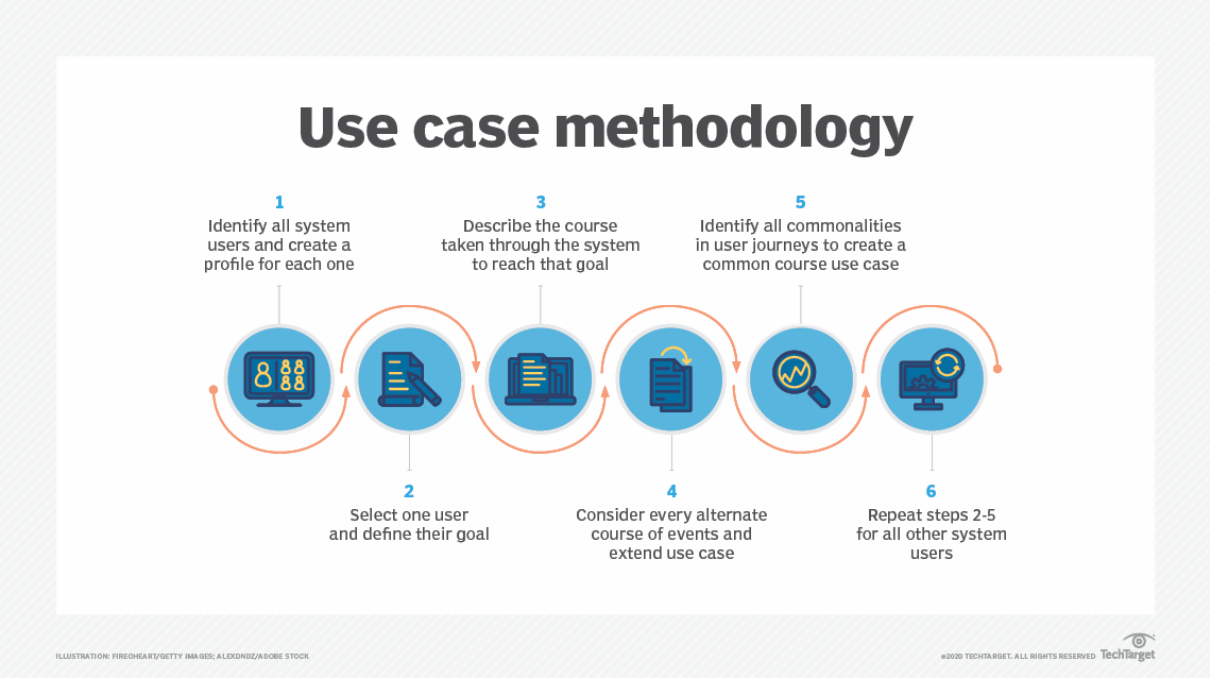

Figure 2. Use case methodological approach workflow

3.1. The actors(users)

The definition of the use cases is centred on the actors, the users. The potential users of the SLNAE have been distinguished into two main groups according to the main functionalities as end-users: *Strategic users* (users who need simple but clear UIs and functionalities, which aim is to take strategic decisions) and *Technical users* (users with a deeper technical knowledge that aim to realize complex analysis or simulations). The detailed definition of both groups, even new user identifications, is ongoing in the scope of the co-creation framework and workshops

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

organized by WP1 team in coordination with the CSs (WP5). Furthermore, the users within these groups can be directly linked to the tree-tier SHs analysis carried out by WP5 (MS6 Stakeholder register, resulting stakeholder mapping).
On the other side, there is a potential third group of users, called *data providers*, that will feed the SLNAE with different sources of data. At the moment, it is expected data from WP2 related to the biophysical and socio-economic modelling, the SDMs from WP3 to be integrated into the SLNAE, Nexus indicators and policies definitions from CSs, WP1 and WP5, or CSs information from WP5. All these aspects are further introduced in section 5.3 Nexogenesis Data sharing tools.
Finally, the last type of user expected is the *administrator*, which will have the responsibility to configure, update and maintain any of the SLNAE modules and infrastructure.

Table 1. Groups of users involved in the final SLNAE operation

User group	Description	Examples
Strategic users	People using the tool as a DSS to understand policy effects and impacts, and to create or prioritize policy-making strategies	Policy makers
Technical users	People using the tool to investigate at a low level the NEXUS interlinkages and policy impacts, and the implication of transboundary decisions	Project partners, nexus researchers or technical staff from stakeholder bodies
Data providers	Project partners or River basin data responsible with aims to update needed data to run the tool	CMCC, UHE, UTH staff
Administrator	Administrators of the tool.	EUT staff

3.2. The use cases template

The consortium has been working on the definition of uses cases and user requirements based on a set of meetings in the scope of WP4 along with all deliverables and documentation provided by the consortium. To get a complete list of use cases, it has been shared and discussed with the project partners, mainly WPLs and CS leaders, a template for its definition. The template includes the columns described in Table 2 to define each SLNAE use case.

Table 2. Descriptive fields for use case definition

Field	Content
ID	[Unique ID of this use case]
Description	[Describe the goal and context of this use case. This is usually an expanded version of what you entered in the "Title" field.]

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Component	[Select a component of SLNAE (The list of components can be updated in the sheet List of items), for what the use case described is focused on]
Goal	[Enter the goal of the use case - preferably as a short, active verb phrase]
Preconditions	[Describe the state the system is in before the first event in this use case. What is needed to allow this use case]
Triggers	[Which is the event, the real need, the action that fires this use case usage]
Success post condition	[Describe the state the system is in after all the events in this use case have taken place.]
Primary Actor	[A person or a WP partner that interacts with your component to achieve the goal of this use case.]
Main course	[Brief description and list of steps the actor should do to achieve the success post conditions, from the preconditions. This is the key point in the use cases description]
Extensions	[Describe all the other scenarios for this use case - including exceptions and error cases.]
Frequency of Use	[How often will this use case be used?]
Status	[Development status]
Owner	[Who owns this use case in the project team? Please add the partner acronym]
Priority	[Priority of this use case. Use: High, Medium, Low]

The SLNAE components organization is based on the WP4 pillars definition (section 2 Pillars). The following elements have been identified:

- SLNAE-User Interface
    - General
    - Simulation configuration
    - Simulation management
    - Simulation
    - Simulation results
    - Simulation comparison
- SLNAE-Backend
- SLNAE-DSS
- SLNAE-Data Sharing Tool

The SLNAE-User Interface has been divided into six subcategories in order to achieve better granularity, since it was expected to contain many of the use cases due to the stories definition (i.e. final functionalities) mainly start there.

After a first iteration of feedback request, it has been created a simpler template (without some columns) in order to facilitate a quick assessment of the required use cases by non-technical

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

partners. On this basis, the complete table is generated. The basic template contains the following columns:

- ID/code
- Description
- Component
- Goal
- System users
- Main course(steps)
- Potential requirements (used to identify non-functional requirements)
- Comments

Table AI.1, in Annex I, shows the complete list of use cases describing the main functionalities that the SLNAE and its components have to include according to their requirements. So far, thirty-six (36) use cases have been identified, but new others can be identified during the first two years of the project based on SHs feedback (collected in workshops WS2 or WS3). The final version of the use cases will be presented in the next version of this deliverable.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

4. Requirements

The IEEE Standard Glossary of Software Engineering Terminology [2] defines a requirement as:

1) A condition or capability needed by a stakeholder to solve a problem or achieve an objective.
2) A condition or capability that must be met or possessed by a solution or solution component to satisfy a contract, standard, specification, or other formally imposed documents.
3) A documented representation of a condition or capability as in (1) or (2).

In this scenario, a system requirements specification collects information on the requirements for a tool or service, and describes what the software will do and how it will be expected to perform. It is composed by functional and non-functional requirements. Both types are essential for a clear system characteristics and components identification and, thus, for a successful and smooth development stage. The former group covers all those functionalities a system must offer and what it must do. In case the system does not meet a functional requirement, it will fail in achieving their final objective. In parallel, the later group collects all those requirements that describe how the system works, and are focused on how it goes about delivering a specific function. If non-functional requirements are not met, final users may become frustrated with how the system works and, consequently, it will also fail in achieving their objectives.

Thus, following the methodology proposed in WP4, the system requirements specification is extracted from the Use Cases definition (section 3 Use cases). Here, WP4 software technicians have analysed the proposed use cases and have generated the system requirements specification table of functional and non-functional system requirements the SLNAE tool has to provide and implement.

Table 3. Descriptive fields for requirements definition

Field	Content
ID	[Unique ID of the requirement. FR-X for functional requirements, an NFR-Y for non-functional requirements]
Use Case ID	[Use Case from where the requirement has been identified]
Component	[Select a component of SLNAE (The list of components can be updated in the sheet List of items), for what the requirement described is focused on]
Description	[Describe the goal and context of requirement]
Details	[Further details and clarifications]
Prerequisites	[Important constraint that must be accomplished]
To be iterated	[Indicated whether a requirement can be modified based on SHs' feedback]

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Table 3 describes the information required to define a system requirement. It is important to note that some requirements are candidates to be modified during the first half of the project based on the SHs' feedback collected in WS2 and WS3. This possibility is indicated in the last column of Table 3.

So far, seventy-seven (77) requirements have been identified and extracted from the proposed use cases. Sixty-eight functional requirements (68) and nine (9) non-functional requirements.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

5. SLNAE Design

The final goal of task T4.1 is to establish the basis and provide the design for the implementation and development of the SLNAE. To achieve so, first, use cases and requirements have been identified, as shown in the previous sections, by considering the SHs (including also NXG partners) necessities.

Below, the technical design of the SLNAE platform is presented, which provides a response to all the system requirements identified so far. This design includes both, a high-level architecture schema where all components and information flows are represented, and a low-level description of such elements. Finally, the required software technologies necessary to construct the systems are also proposed.

It is important to recap that the SLNAE tool will be used both during the NXG project and after its finalization. In the first case, the engine will be used to identify optimal policy packages to maximise goals and targets defined by SHs in the co-creation framework cycle. This necessity forces WP4 to start working first on the AI engine in order to be able to have the system ready when needed. Later, the online web platform will be built around the engine to offer the final SLNAE framework.

5.1. General overview

Figure 3 presents a general overview of the SLNAE platform where all the actors involved in the NXG WEFE approach can be identified. The proposed SLNAE global architecture is organized into four inter-related components, each one representing a key element of the NXG project. Furthermore, each component is aligned with one, or various, WP4 pillars.

The first component corresponds to the *Stakeholders & co-creation framework*, which represents WP1 and WP5 responsibilities and outcomes, and is aligned to the *WEFE Policy Framework* and *WEFE Targets & Indicators* pillars. The second component covers the *Nexus integration* aspect of the NXG project. It integrates the technical WPs (WP2 and WP3), and is aligned to the *Nexus complexity modelling* pillar. Moreover, it is also indirectly linked to the *WEFE Policy Framework* and *WEFE Targets & Indicators* pillars since these concepts will be embedded into the SDMs.

The third component corresponds to the SLNAE itself, developed in WP4. It is mainly linked to all the pillars, since all of them will be finally integrated there, although it will particularly focus on the *DSS functionalities* and *Graphical User interface* ones. Finally, as an extension of the SLNAE platform, the *NXG Data Repositories & Dara Sharing Tools component* is related to the Data Sharing Tools pillar, and contains data from all WPs (from WP1 to WP5). The WPs generated information will be stored in the NXG Data Sharing tool, and directly and automatically integrated into the SLNAE. Further information regarding data created by each WP and shared across other WPs is presented in section 5.3.1

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

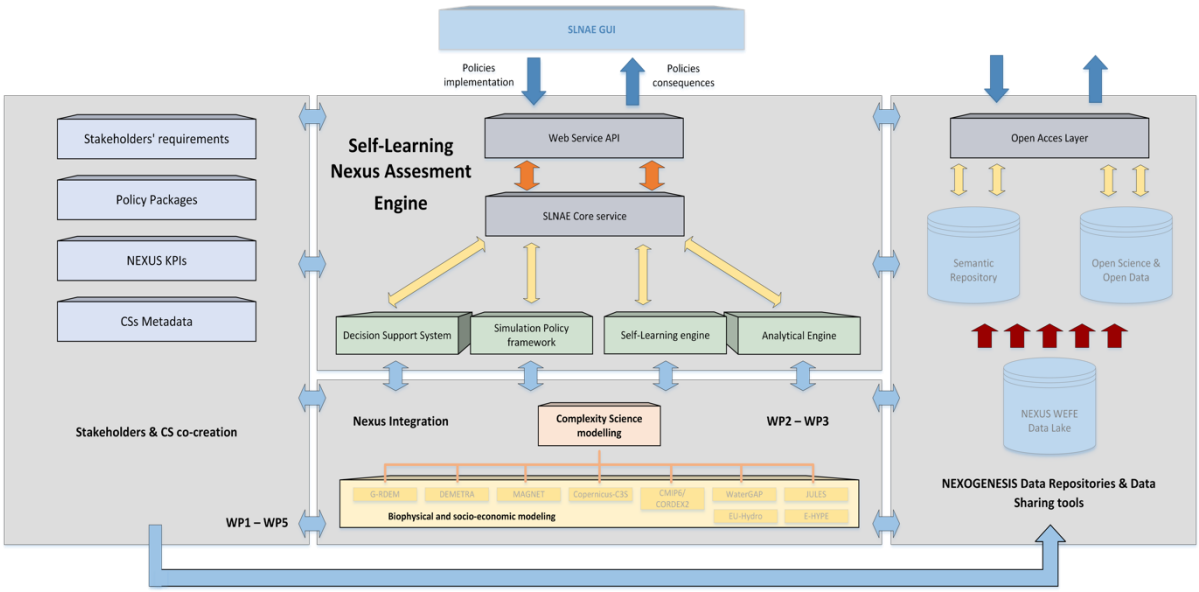

Figure 3. SLNAE platform schema

Below, all the SLNAE components are introduced, and an initial design is proposed. Each decision and functionality proposal is linked to the corresponding requirement from table AII.1.

5.2. SLNAE

The SLNAE final platform is designed as an online tool (NFR-03 to NFR-05) to enable remote access to the users. In this sense, the users will not need additional Hardware resources since a browser is the only requirement to access and use it.

The platform architecture is divided into seven modules, as can be seen in Figure 3, each one with a specific objective and responsibility:

- SLNAE Core service
- Web Service API
- SLNAE database
- SLNAE GUI
- Simulation Policy Framework
- Self-Learning engine
- Decision Support System
- Analytical Engine

Python[1] programming language will be used to develop such modules, and special attention will be put to the solution components testing. Continuous delivery methodologies will enable automated and continuous validation, integration and deployment of new developments.

Three environments are proposed for the development, test, and final deployment of the SLNAE platform:

- Development: First environment where all developments will be tested by the technical team. A series of automatic unit-tests, end-to-end tests and integration tests will be defined to cover different issues such as Global components, Local components, load

[1] https://www.python.org/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

testing, browsers compatibility, screen resolutions, security, resilience, scalability and/or interoperability.

- Staging: Once developments are ready and validated in the DEV environment, they will be deployed into the Staging one. Here, the NXG partners and CSs will be able to access and use the new additions. It will enable a second testing phase, more focused on the context validation and correct functioning of Nexus logics (e.g. policies, goals, etc).
- Production: Final environment where the SLNAE tool will be publicly available.

The SLNAE Coordination & Integration service will manage all these tasks and processes to ensure a high-quality system in terms of availability, capacity, interoperability, performance, reliability, robustness, safety, security, resilience and usability (NFR-01 to NFR-09).

5.2.1. SLNAE Core service

The SLNAE Core service will principally coordinate all SLNAE platform and module communication, and will monitor the status of the services.

It will implement the required mechanisms to persist and load data from the SLNAE Data repositories. In this context, it will trigger and coordinate the automatic processes to integrate other WPs outputs into the system logic. To do so, it will be connected to the Data Sharing Tool and will consume new information regarding policies, goals, targets, indicators, CS information and SDMs. Here, the I18N module will be in charge of enabling text translation based on the user profile (FR-07, English, Greek, Bulgarian, Italian, German, Romanian, Latvian, Lithuanian).

Finally, the SLNAE Core service will also implement an authentication and authorization framework to support the Web Service API module by validating all incoming communications. It will be based on the JSON Web Tokens[2] (JWT) standard, which is an open, industry-standard RFC-7519[3] method for representing claims securely between two parties. This framework will be based on the flask-jwt[4] and flask-jwt-extended[5] Python libraries. A guest log-in mode will be implemented but, although authenticated access will not be mandatory, some advanced functionalities will only be available for registered users.

To simplify all the data flows and platform development, an object-oriented programming paradigm will be integrated. Thus, the following software entities will be modelled and will be available by other platform modules to implement its logic.

- CS: To represent the CS entity.
- Policy Goal: To represent the Policy Goal entity.
- Policy Target: To represent the Policy Target entity.
- Indicator: To represent the indicator entity, including WEFE footprint indicators.
- SDM: To represent the SDM entity.
- Simulation: To represent the simulation entity.
- Users: To represent the user entity.

[2] https://jwt.io/

[3] https://tools.ietf.org/html/rfc7519

[4] https://pythonhosted.org/Flask-JWT/

[5] https://flask-jwt-extended.readthedocs.io/en/stable/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

The expert programming principle will also be integrated, thus each software entity will implement the necessary entity-related methods.
An ORM, implemented with SQLAlchemy[6] Python library, will manage the mapping and operations between SLNAE tool and NXG Data repositories.

5.2.2. Web Service API

The Web Service API provides the communication between the SLNAE GUI and the SLNAE platform or backend. It is designed as a RESTful stateless service that implements a REST (Representational State Transfer) architecture. The client-server communication uses JSON (JavaScript Object Notation) data format over the HTTPS (Hypertext Transfer Protocol Secure) protocol.
The Web Service endpoints listed in

Table 4 will be provided (FR-54).

Table 4. SLNAE Web Service endpoints

Web Service Endpoint	HTTP methods	Requires authenticated JWT	Description	Request parameters
/slnae/case_studies **/slnae/case_studies/id**	GET	No	To obtain CSs' information	CS ID
/slnae/policy_goals **/slnae/policy_goals/id**	GET	No	To obtain default Policy Goals and targets information	Policy Goal ID
/slnae/indicators **/slnae/indicators/id**	GET	No	To obtain default CS and WEFE footprint indicators information	Indicator ID
/slnae/policies **/slnae/policies/id**	GET	No	To obtain policies information	Policy ID
/slnae/simulations **/slnae/simulations/id** **/slnae/simulations/id/run**	GET, POST, DELETE	Yes	To manage simulations	Simulation parameters
/slnae/users	GET, POST	Yes	Manage user's information and profile	User's parameters

[6] https://www.sqlalchemy.org/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

/slnae/login	POST	No	To validate authentication credentials and generate JWT	Email and encrypted password
/slnae/logout	GET	Yes	To sign out from the session	
/slnae/users	GET, POST	Yes	Manage user's information and profile	User's parameters

Additional endpoints will be added to the Web Service interface in case it is needed.
Finally, the Web Service will be deployed in the open-source HTTP Apache Server[7] to provide a secure, efficient and extensible tool that enables HTTP services in sync with the current HTTP standards.

5.2.3. SLNAE database

An internal SQL database will provide persistence functionalities to store all the information related to the SLNAE entities (FR-55). It will be managed by the ORM, which will transparently implement all the CRUD (Create, Read, Update and Delete) methods and will simplify their management.
The PostgreSQL[8] open-source technology will be used for the implementation of this component.

5.2.4. SLNAE GUI

The SLNAE GUI is the front view of the platform and will enable users to interact with the NXG project research and outcomes. It will provide the required mechanisms to run WEFE simulations by applying Nexus policies in each of the five NXG CSs. Furthermore, it will let the user analyse the impacts on the interlinked nexus sectors, in terms of policy goals and WEFE footprint indicators achievement, to finally understand which are the best policy packages in each scenario.
Based on the identified requirements, sixteen views have been identified. The proposed SLNAE GUI views flow is presented in Figure 4.

[7] https://httpd.apache.org/
[8] https://www.postgresql.org/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

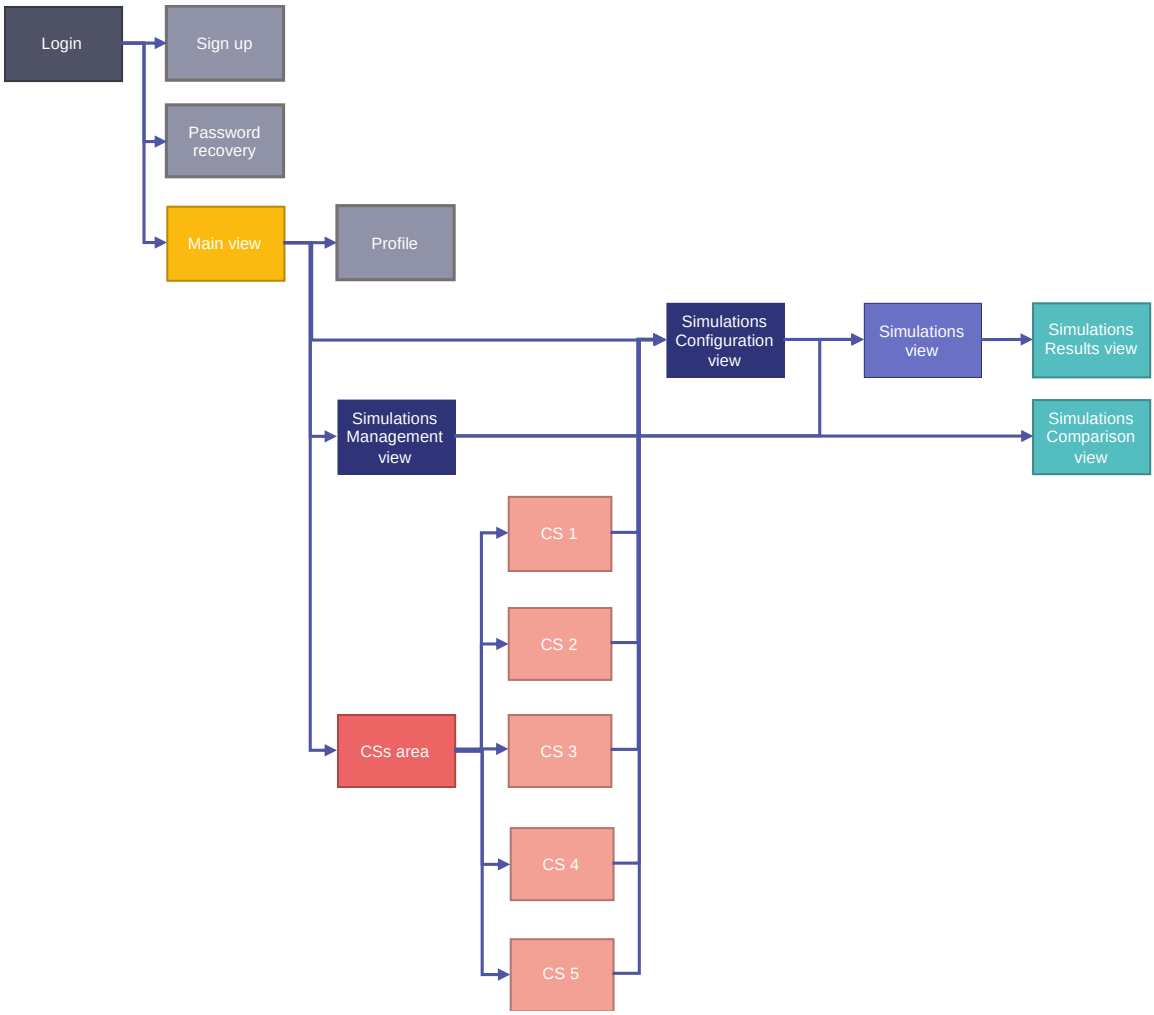

Figure 4. SLNAE GUI views Flow

The SLNAE GUI entry point is the Login page, where the user will be requested to log in (FR-02). In case it is not registered yet, it will be able to access the Signup page to start the registration process (FR-01), or to access the platform as a guest user (FR-04). Furthermore, the Password recovery (FR-05) view is also accessible from the Login page. Also, information regarding the data and cookie policies (FR-09) will be presented to the user on this initial page, and an option to remember the credentials will be included in the login form.

Once logged in, the user will be redirected to the Main view (FR-10). Here, general information about the NGX project will be presented. From this page, four possibilities are available to the user. It can go to the Profile section (FR-06) where, in case it is registered into the system, the user's information is shown and can be modified (username, email). Also, in any case, the user can select the platform language (FR-07).

The second option redirects the user to the CS area (FR-11 and FR-12), where specific information and CSs' characteristics will be displayed.

The third option is the Simulations Management view (FR-20), which will be always accessible independently of the view (FR-14). It will present, in a table structure, all the simulations previously saved by the user, and will have mechanisms to load (FR-22), see the results (FR-21), rename (FR-24), duplicate (FR-25), compare (by selecting more than one entry, FR-26) or

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

delete (FR-23) them. In case the user is not registered, only the simulations created during the current user session will be presented.

Finally, the fourth option (which will be always accessible independently of the view, FR-13) redirects the user to the Simulation Configuration (FR-18) view, a wizard-based page where the user will be able to configure a new simulation. To do so, the user can select the CS and the reference pathway, can activate the uncertainty mode, and can define a set of user-defined goals, targets and WEFE and CSs indicators (FR-19). Once a simulation is configured, the Simulation (FR-27) view, the main SLNAE functionality, will be presented to the user. Here, a dynamic view will enable the user to interact with the WEFE simulation framework. The view will be organized per sections, so the user always has visible the most important nexus information, such as the time horizon, the policies goals level of achievement or the CS map. Moreover, in order to have other information as available as possible, a flexible tabs schema is designed. The following sections are proposed:

- Time horizon (FR-28): it will show the simulation step, from the year 2020 to the year 2050 in a year timestep. It will have functions to move forward or backward (FR-29) in the simulation horizon time in user-defined timesteps (e.g. to advance four years, FR-30). Additionally, the applied policies will be presented (FR-34) so the user can disapply them (FR-33).
- Default policy goals, targets and WEFE footprint indicators (FR-39): will show the level of achievement of the CS predefined policy goals, targets and WEFE footprint indicators. Uncertainty in the computations will be shown in case it is requested.
- User-defined policy goals, targets and WEFE footprint indicators (FR-40): will show the level of achievement of the user-defined policy goals, targets and WEFE footprint indicators. Uncertainty in the computations will be shown in case it is requested.
- CS map: a central view to represent the CS (e.g. regional or transboundary case).
- Available tokens (FR-37): Information about available tokens will be always shown.
- Policies inventory (FR-31): It will present the available policies to be applied. Here, a pop-up function will be used to show their information, and restrictions between them will be also considered, thus the restricted ones will be hidden (FR-36, FR-38). From there, a policy can be selected (FR-32) and configured (FR-35) to be applied at a specific year.
- Indicators inventory: Other available indicators' levels of achievement will be presented.

In parallel, the user can access the SLNAE DSS advice, which will provide support to identify the most optimal policy packages. Advice focused on the maximization of default policy goals (FR-43) will be immediately available, but advice targeting the user-defined policy goals (FR-44) will require time for computation, thus the user will have to trigger it in case it is needed. This second type of advice will present optimal policy packages as they are identified by the DSS engine, and the user will be always aware of the computations.

Once the user considers that a simulation is finalized, the Simulation results view (FR-46) presents a summary of the process. Here, two types of views are available: a basic view (FR-47) and an advanced (FR-48) one. In the basic view, only applied policies and the indicator level of achievement are presented. On the other hand, the advanced view presents all the Nexus data and its evolution during the simulation (FR-50), considering also data uncertainty (FR-49).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

The details of the proposed functionalities and what and how data will be shown will be discussed with SHs and documented in the final version of this deliverable.
Finally, the user will be asked whether the simulation has to be stored (FR-42) or not for future usage (e.g. in a comparison). The way how simulations will be compared in the Comparison view (FR-51) will be also discussed with SHs. At least, comparisons among simulations corresponding to the same CS (FR-52) will be enabled by comparing the default goals, and comparisons among different CS simulations (FR-53) will be enabled by comparing the common WEFE footprint indicators.
The iteration and finalization of SLNAE GUI functionalities and design is the main WP4 target for workshops WS2 and WS3. In WS2, the current design will be presented to the SHs, and first feedback and ideas are expected. On this basis, WP4 will proceed to develop the required mockups to graphically describe the GUI views. Then, in the WS3, these mockups will be presented to the SHs to have a final discussion and feedback. All these decisions will be incorporated into the final version of this deliverable.
HTML, JS and CSS will be used to build the GUI, and the following frameworks are identified to implement the proposed functionalities:

- Angular[9]: Framework for web applications development.
- JWT[10]: Library for representing claims securely between two parties using industry standard RFC 7519.
- D3[11]: a JavaScript library for manipulating documents based on data.
- Leaflet[12]: a JavaScript library for friendly interactive maps.

5.2.5. Self-learning engine and AI algorithms

The self-learning engine is considered the core of the SLNAE platform and one of the most important research, development, and outcome of the NXG project. The self-learning engine will be used as a recommendation tool for decision making in i) policy package definition during the project, and ii) policy package advice in the final online tool.
In the first case, the self-learning engine will be used in the co-creation framework cycle to generate optimal policy packages given a set of policies and policy goals and targets defined by the SHs in each CS. Once generated, the identified policy packages will be evaluated by the SHs and a new learning iteration will be carried out if possible (at least in frontrunner CSs).

[9] https://angular.io/
[10] https://jwt.io/
[11] https://d3js.org/
[12] https://leafletjs.com/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

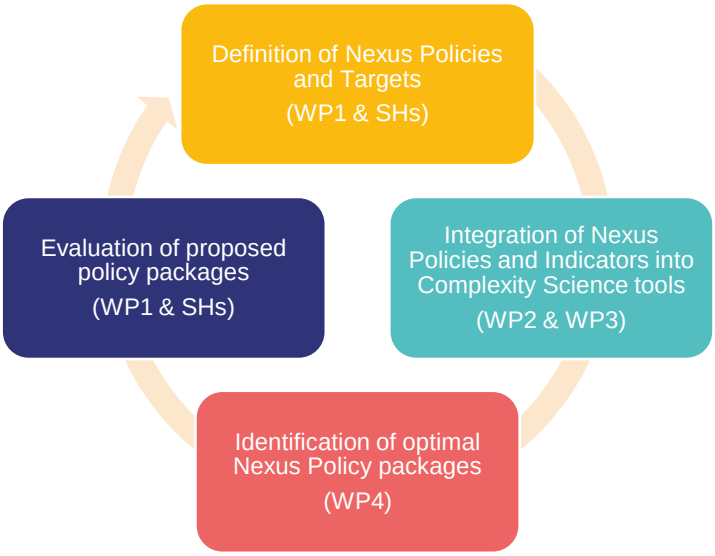

Figure 5. NXG co-creation framework for Nexus Policy packages identification

The second case refers to the online DSS, which will provide policy package advice to the platform users. As it is introduced in the previous section, the users will be provided with two types of advice. The first one will be focused to accomplish CS predefined goals and targets, which will be based on the outputs of the co-creation framework for policy governance.

The policy package identification task can be understood as an optimal control problem, where decisions have to be made along time to select policy packages with the aim to achieve (or maximize) certain metrics, such as policy goals targets or WEFE footprint indicators, at a certain instant of time.

Given the considerable number of policy combinations, the scenario restrictions (e.g. budgets or allowed policy combinations), the huge quantity of instants when a decision can be made, or the underlying complexity and non-linearity of the problem, its solution is non-trivial. Moreover, if data uncertainty is taken into account, the problem becomes non-deterministic, thus it is even more difficult to confront.

Below, the algorithmic methodology proposed to implement the self-learning engine is presented. It is based on the definition and self-training of a set of intelligent AI-powered agents. Once trained, these agents will be able to find the most efficient ways for decision-making in dynamic and uncertain systems based on cost function definitions that represent the nexus concerns and desired policy goals.

We propose Reinforcement Learning (RL) [3], a hybrid family of algorithms between control theory and AI, to deal with the task of decision-making in policy packages application for achieving different targets (goals). Moreover, taking into account the problem complexity, it becomes also necessary to include function approximations able to deal with it. In this scenario, Deep Learning (DL) provides the required power and capabilities to deal with non-linearities and high-dimensional tasks. As a result, the Deep Reinforcement Learning family of algorithms is proposed.

5.2.5.1. MDP formalization

In the control problem scenario, the task is formalized as a Markov Decision Process (MDP). An MDP provides a mathematical framework for modelling decision-making in a sequential, stochastic, and discrete-time environment.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

The formalization consists of a 5-tuple:

- A set of states S
- A set of actions A
- A scalar reward function $R \subset \mathbb{R}$ that maps state-action pairs to scalar values (a reward signal)
- A state transition probability distribution P that defines the dynamics of the MDP; from all states $s \in S$ to all their successors $s' \in S$
- A discounted factor $\gamma \in [0,1]$, allows weighing the importance of future rewards over current ones

Theoretically, in its basic form, the Nexus policy-decision-making problem can be mathematically formalized as a fully observable MDP, where the dynamic transitions are given by the SDMs.

- The MDP state must be correctly designed and must contain all the required information from the SDMs (the nexus WEFE) to ensure that the DSS models can provide consistent recommendations by identifying the necessities in each situation. In each CS, it will be discussed with experts to identify which are the key parameters. Additionally, other information must be considered, such as the year, the policies state, or the available tokens.
- The action space defines the decisions to be taken in the environment. The decision-making is based on discrete actions defined by the available policy packages in a given year (a combination of policies).
- The design of the reward function is one of the most complex parts of the MDP definition since it must guide the agent to the right path. The reward function will be based on the selected policy targets and WEFE indicators to achieve the policy goals. Usually, there will be more than one indicator to consider given a multi-objective problem. More than one expression has to be taken into account to accurately define the reward function. This implies balancing the relative importance of each concept and multiple indicators will be aggregated to a general one.

Given an MDP, the random variables s' and r depend on the preceding state and action. That is, the probability of those variables occurring at the same time t is conditioned on the previous state $s \in S$ and the action $a \in A$ taken, as defined in the following equation:

$$p(s', r|s, a) \doteq \Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}$$

for all $s', s \in S, r \in R$ and $a \in A$. In this case, the Markov property is achieved. Recall that the objective of an MDP is to maximize the expected discounted cumulative reward in a trajectory. A sequence of transitions, such as $\{S_0, A_0, S_1, R_1, A_1, S_2, R_2, A_2, S_3, R_3, \dots, S_{T-1}, A_{T-1}, S_T, R_T\}$, defines a trajectory through the MDP. The equation below defines the expected reward in an episodic MDP.

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + \cdots = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1} = R_{t+1} + \gamma G_{t+1}$$

where T corresponds to the last time-step of the episode. Timestep T leads to terminal states.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

5.2.5.2. Reinforcement Learning

RL is a computational approach in the ML (AI) field that provides support in MDP systems by trial-and-error learning guided by the expected cumulative future reward. The RL model's aim is to learn what action to do in a given state with the final goal of maximizing the numerical reward function [3]. An RL model is a function that maps states to actions based on experience. Solving an RL task means finding a policy π that maximizes the reward signal over the long run. Most RL algorithms involve estimating value functions that estimate how good it is for the agent to be in a given state, or how good it is to perform a given action in a given state.

In the learning process, the decision-maker is called the agent, and the thing it interacts with is the environment. These elements interact continuously through an iterative process where the agent selects actions, and the environment responds to these actions with reward signals and presents new situations to the agent. During this training process, the agent continuously improves a policy to maximize the expected future reward over time through its choice of actions. The ultimate goal of the agent is to learn a policy π that maps states to actions with the aim of generating the highest cumulative reward through the agent-environment interaction. The iterative agent-environment interaction process is shown in Figure 6.

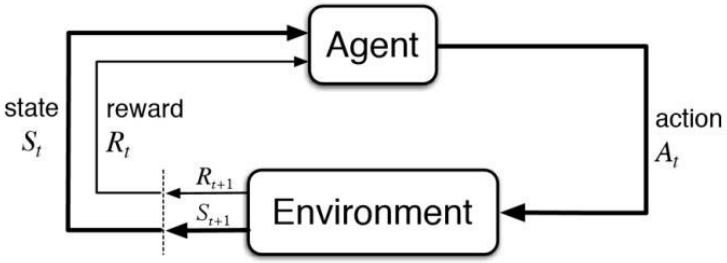

Figure 6. Reinforcement Learning interaction flow

The self-training process is described as follows:

For all defined iterations, at each timestep t:

1. The agent observes the current state of the environment S_t
2. The agent chooses an action A_t in the given state S_t
3. This action and the system dynamics cause a transition between states in the environment. A reward signal R_{t+1} based on the consequences of the action taken and the stochasticity of the MDP is provided to the agent. Moreover, the next discrete-time state, S_{t+1}, is also given to the agent in order to provide the current situation of the system.
4. The agent updates the policy π based on the previous interaction with the environment, denoted by the transition tuple $< S_t, A_t, S_{t+1}, R_{t+1} >$
5. The current state S_t from the environment is updated with the new one S_{t+1}
6. Go back to step 1. The process starts again until the number of iterations is completed

Standard RL maximizes the expected sum of rewards as presented below:

$$\sum_t \mathbb{E}_{(S_t,A_t)\sim\rho_\pi}[R(S_t,A_t)]$$

In the previous equation ρ_π denotes the transitions following policy π.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Dynamic Programming [4] and Monte Carlo [5] methods are considered basic forms of RL, since both provide (restricted and limited) solutions to the MDP problem. One of the most popular tabular RL algorithms is Q-Learning [6], a temporal-difference [7] RL methodology. The classical Q-Learning algorithm learns a state-action value function by trial-and-error. The state-action pair values are updated on every new iteration. These values represent the expected future reward from being on each state-action pair in the environment and are used to know which decision has the higher expectation.

During the training phase, unknown states are visited to allow exploration and approximate the values in the so-called Q-table. There is an exploration and exploitation dilemma during the training phases. While the RL agent is learning expected value and making decisions in the environment based on that, the agent must also explore other alternatives to correct any deviation in the Q-function. Therefore, during data collection in agent-environment interactions, the agent will balance the selection of random action with the selection of the best-known action.

One of the major challenges in RL is the right reward function definition that must guide the agent in the right learning direction. An example of a simple reward function is the following:

$$R(S) = \begin{cases} 0, & S\ not\ terminal \\ 0, & S\ terminal\ and\ goals\ are\ not\ achieved \\ 1, & S\ terminal\ and\ goals\ are\ achieved \end{cases}$$

This reward function is sparse since for all states that are not terminal it is 0, and it is different than 0 when all goals are achieved in a terminal state. Therefore, there are two main problems in the reward function definition: reward sparsity and credit assignment problems. This second issue is related to the difficulty to identify which actions, in an MDP trajectory, guide the system to the best or poor rewards.

Q-Learning, such as other tabular RL methods, have some limitations:

- The agent needs a lot of interactions with the environment to approximate well the value function and learn an acceptable policy π. If SDMs are slow, the learning will be also slow. Thus, python SDMs must be optimized to give transitions in the lowest time possible.
- The action space can be very complex to manage for classical RL. If the number of available combinations of policies at each step is very big, learning all state-action values in a tabular form becomes very challenging and even unfeasible.
- Similarly, a high-dimensional or continuous state space is not feasible to consider in tabular algorithms.

In its classical tabular approach, RL is not able to deal with high-dimensional and non-linear problems due to the curse of dimensionality, which is the exponential growth of states and actions when the problem has a high-dimensional or continuous state and/or action spaces. In order to overcome these limitations, RL methodologies have been recently combined with DL techniques to generate the so-called Deep Reinforcement Learning (DRL) algorithms family. Here, Deep Neural Networks (NN) represent the agent's policy and are used as a function approximator of the complexity behind the state and action spaces.

5.2.5.3. Deep Reinforcement Learning

DRL is the result of combining Deep Neural Networks (DNN – DL) together with RL where the tabular functions are substituted by DNN. In these types of algorithms, information is not

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

explicitly stored for every single state or every state-action pair. What it is done here, is a more compact representation that generalizes across states or states and actions. It reduces the resources needed to store all the information, computation, and experience needed to converge to a reasonable solution in larger problems, even when the state space is continuous. For an n-dimensional state space and a discrete action space containing m actions:

- A neural network that defines the state-action value function is a function from $R^{n\,x\,m}$ to R
- A neural network that defines the policy function is a function from R^n to R^m

There are three main types of DRL algorithms, as in RL:

- Value-based: the algorithm learns the value function V or Q, or the advantage function A.
- Policy-based: the algorithm learns the policy function directly.
- Actor-critic**:** Actor–critic algorithms learn both, policies, and value functions. The actor is the parametric policy, and the critic learns value approximations in order to criticize the behaviour of the actor in such a way that the actor can more efficiently learn. Actor-critic algorithms allow high-dimensional and continuous action spaces, unlike value-based algorithms.

Value-based methods

Under the value-based category, Deep Q-Learning (DQN) is the first implementation of the Q-Learning algorithm to DL. In this case, instead of learning a value per each state-action pair as it is done in Q-Learning, the DNN has as input the state and outputs a probability per each action from the action space. Therefore, the DNN learns a Q-value probability per each action given a state.

RL is known to be unstable, or divergent, when a nonlinear function approximator, such as a neural network, is used to represent the state-action value function [8]. This instability comes from i) the correlations present in the sequence of observations, ii) the fact that small updates to Q may significantly change the policy and the data distribution and, iii) the correlations between Q and the target values $\gamma\, max_{a_{t+1}} Q_\pi(s_{t+1}, a_{t+1})$.

To face these issues, two techniques have been developed. The first one, called experience replay [9], uses a random sample of prior actions instead of the most recent action to proceed. This removes correlations in the observation sequence and smooths changes in the data distribution. In the second, iterative updates adjust Q towards target values that are only periodically updated, further reducing correlations with the target [10].

GoRiLa [11] (General Reinforcement Learning Architecture) adds parallelization to the DQN algorithms, thus achieving a massively distributed version of it. DQRN [12] (Deep Recurrent Q-Network), introduces the capabilities of Recurrent Neural Network architectures, through the addition LSTM (Long Short Term Memory) DNN layers, allowing the agent the ability to remember a bigger picture of the environment.

The max operator in standard Q-learning and DQN uses the same values both to select and to evaluate an action. This makes it more likely to select overestimated values, resulting in overoptimistic value estimates. To prevent this, the selection from the evaluation can be decoupled. In Double Q-learning [13] (Double DQN), two value functions are learned by

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

assigning experiences randomly to update one of the two value functions, resulting in two sets of weights, w and w'. For each update, one set of weights is used to determine the greedy policy and the other to determine its value. The idea of Double Q-learning is to reduce overestimations and although not fully decoupled, the target network in the DQN architecture provides a natural candidate for the second value function, without having to introduce additional networks [14].
With experience stored in a replay memory, it becomes possible to break the temporal correlations by mixing more and less recent experiences for the updates, and rare experiences will be used for more than just a single update. However, an RL agent can learn more effectively from some transitions than from others. Therefore, prioritizing which transitions are replayed can make experience replay more efficient and effective than if all transitions are replayed uniformly. Prioritized experience replay [15] is built on top of DDQN further improving the state-of-the-art.
A Dueling network architecture [16] explicitly separates the representation of state values and (state-dependent) action advantages as it consists of two streams that represent the state value function and the advantage function, while sharing a common convolutional feature learning module.

$$Q(s,a) = V(s) + A(s,a)$$

This dueling network should be understood as a single network where two streams are combined via a special aggregating layer to produce an estimate of the state-action value function, thus replacing the popular single-stream state-action value network in other existing algorithms such as DQN. The advantage function A shows how advantageous taking action a is relative to the others at a given state s. This change is helpful, because sometimes it is unnecessary to know the exact value of each action, so just learning the state-value function can be enough in some cases.
When we have to deal with continuous action spaces, an obvious approach to adapting DRL methods such as DQN to continuous domains is to simply discretize the action space. However, this has many limitations, most notably the curse of dimensionality: the number of actions increases exponentially with the number of degrees of freedom. In the continuous control domain, where actions are continuous and often high-dimensional, we argue that the existing control benchmarks fail to provide a comprehensive set of challenging problems. The situation is even worse for tasks that require fine control of actions as they require a correspondingly finer-grained discretization, leading to an explosion of the number of discrete actions. Such large action spaces are difficult to explore efficiently, and thus successfully training DQN-like networks in this context is likely intractable. Additionally, naive discretization of action spaces needlessly throws away information about the structure of the action domain, which may be essential for solving many problems [17][18].

Policy-based methods

Value-based methods work with discrete and finite action spaces, thus, it is possible to calculate the maximum value over all possible actions in each state. If a high-dimensional action space is considered, this maximum value over all possible actions may become computationally expensive, or even impossible in continuous action spaces. Moreover, sometimes value functions give too much information for the task of selecting an optimal policy and it makes

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

harder the training process, since it is unnecessary to compute the exact value for each state or each pair action-state.

Policy-based methods can learn easily stochastic policies, thus, do not need to force exploration with some probability to ensure optimality, as it is done in value-based methods, given that the exploration is embedded in the learned function which ultimately converges to a deterministic policy. However, in order to improve learning strategies in policy gradient methods, much research focused on exploration is being carried out. In particular, [19] proposed two exploration techniques to address the limitations of gradient methods and [20] introduced a state-dependent exploration function that, during an episode, returns the same action for any given state resulting in less variance per episode and faster convergence.

REINFORCE is a Monte-Carlo variant of policy gradient methods, thus since one full trajectory must be completed to construct a sample space, it is an off-policy method. The agent collects a trajectory of an episode using its policy and uses its results to update the policy. This method suffers from high variance and low convergence even though it gives unbiased estimates. To improve performance, a widely used variation of REINFORCE is to subtract a baseline to reduce the variance of the gradient estimation. For instance, the advantage of an action in a given state, while keeping the bias unchanged. It is similar to what is done in dueling networks from the value-based methods.

Updating the weights of a neural network repeatedly for a batch, pushes the policy function far away from its initial estimation. To limit this issue, Trust Region Policy Optimization [22] (TRPO) methods update the policy function, but do not allow it to change much from the previous policy, by introducing a constraint for it. Given that TRPO is relatively complicated, Proximal Policy Optimization [23] (PPO) simplifies it by using a clipped surrogate objective while retaining similar performance and using multiple epochs of stochastic gradient ascent to perform each policy update. These modified methods have the stability and reliability of trust-region methods but are much simpler to implement. The most common implementation of PPO is via the Actor-Critic Model introduced below.

Actor-critic methods

As the critic network learns which states are better or worse, the actor uses this information to teach the agent to seek out good states and avoid bad states.

The Advantage Actor Critic [24] (A2C) algorithm uses a state-dependent baseline which is the expected advantage of an action in a given state reducing the variance of the gradient

$$A(s,a) = Q(\mathrm{s},a) - V(s)$$

Therefore, the critic will have to approximate two different functions: $Q(\mathrm{s},a)$ and $V(s)$. This algorithm is not very different in essence from REINFORCE, since it follows similar steps: sample transitions, compute the return and update the policy. However, in this method episodes do not need to be finite since A2C relies on the n-step updating approach, where the critic has to be learned in parallel. The actor and critic are stored in a global network and multiple instances of the environment are created in different parallel threads, representing the different actor-learners. All learners sample an episode starting with the actor and critic weights from the global networks and the global networks merge the gradients computed by each learner updating the parameters of the policy and critic networks. The learners continue training on new episodes with the updated global weights until convergence. A2C is a synchronous and

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

deterministic version of Asynchronous Advantage Actor-Critic [25] (A3C), where learners are fully independent, thus they only communicate through asynchronous updating of the global networks. Hence, A3C is designed to work very efficiently for parallel training. However, the fact that in A3C each learner talks to the global parameters independently may lead to learners playing with different versions of the network weights and, therefore, the aggregated update would not be optimal. Aversely, A2C ensures the same starting weights for all learnings by waiting for all the parallel actors to finish their work before updating the global parameters.
These previous methods are modelling the policy function as a probability distribution, however, there are other methods that consider and calculate the policy as a deterministic decision such as Deterministic Policy Gradient [26] (DPG). In the stochastic case, the policy gradient integrates over both state and action spaces, whereas in the deterministic case it only integrates over the state space. As a result, computing the stochastic policy gradient may require more samples, especially if the action space has many dimensions. Deep Deterministic Policy Gradient [27] (DDPG) is an actor-critic algorithm that combines DPG and DQN.
Soft Actor Critic [28] (SAC) incorporates the entropy measure of the policy into the reward to encourage exploration. It is an off-policy actor-critic algorithm following the maximum entropy RL framework. It optimizes a stochastic policy trained to maximize a trade-off between expected return and entropy, which is a measure of randomness in the policy. Increasing entropy results in more exploration, which can accelerate learning later on. It can also prevent the policy from prematurely converging to a bad local optimum. A precedent work is Soft Q-Learning [29].

5.2.5.4. Challenges of DRL application in the NXG context

The application of the aforementioned mechanisms is not trivial, and many research challenges raise in the WEFE decision-policy-making context. During the project, in the co-creation cyclic framework for policy governance, DRL agents will be trained based on a policies set and targets defined by the SHs. As a result, apart from the generated optimal policy packages, a set of trained agents will be available.
Later, these agents will be used in the SLNAE DSS for real-time advice focusing on the default CSs' targets, those targets used in its previous training. The second type of advice, that focused on the achievement of user-defined targets, will be more complex and time-consuming to be provided, since it requires a new online computation (i.e. an agent training).
Below, the identified challenges and the proposed solutions are presented:

- High-dimensional and non-linear tasks: Application of DRL algorithms.
- SDMs slow sample rate: complex SDMs lead to slow simulation cycle time, thus ending with slow learning when it is used in the RL interaction flow. To avoid it, efficient code libraries or even parallel programming will be used when translating SDMs from the Stella modelling framework.
- Complex action-space due to many policy combinations (policy packages): Policy Gradient or Actor-Critic DRL techniques will be used.
- Real-time optimization and advice for user-defined targets advice: taking advantage of the available pre-trained agents, Transfer Learning techniques [30] will be used to generate a base knowledge. With it, the required time in the agent's learning stage will be considerably reduced. The selected base agents will be those that have been trained with similar targets.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

- Multiobjective problem: Many WEFE policy-making scenarios can be proposed and analysed, which may result in different policy packages generation. For example, in transboundary CSs decisions can be made in a collaborative manner, thus taking into account the objectives of all entities at once. The opposite way is to independently consider the implicated entities and analyze which are the impact of upstream decisions in downstream cases. Also, other analyses can be run by separating the WEFE sectors, where different entities focus separately on their objectives. To do so, multi-agent systems are proposed, and collaborative vs no collaborative scenarios will be investigated.
- Reward sparsity and credit assignment problems: research on these issues will be carried out during the project.

5.2.5.5. Other algorithms, mechanisms and system benchmark

The objective of the AI part of the SLNAE tool is to get good models to support the decision-making by recommending favourable policy packages based on defined goals. This is a combinatorial optimization problem. The function to be maximized is the aggregation of indicators that will be co-defined to assess the policy goals achievement. Based on this function, it will be sought which policies are advised to consider and at what moment in time in the defined time horizon.

In combinatorial optimization, the optimal solution must be identified from a large set of possible solutions that cannot usually be calculated one by one, since it would have a too high computational cost. Mathematical optimization such as linear programming and metaheuristics have been widely used to solve combinatorial optimization problems. In very complex problems, mathematical optimization methods can take too long to obtain optimal solutions. Therefore, in these situations, metaheuristic methods are commonly applied since they can obtain pretty good solutions in a reasonable time. However, metaheuristic algorithms can get stuck in local optimums and cannot guarantee an optimal result. Moreover, algorithms must be executed taking some time every time a new recommendation is needed in a given configuration. These may difficult the exploration of different policy package alternatives through an MDP. Classical optimization solutions are considered offline, require complete and previous knowledge of the environment dynamics, and are not always able to react to unexpected changes and handle uncertainties favourably. Thus, to better control complex and changing systems under uncertainties, more adaptive control is needed.

To overcome this last drawback, RL models will be trained on all predefined MDP to facilitate the end-user interaction with the support models of the tool. RL represents a set of solutions that do not previously need to know any information about the system dynamics, in contrast with other traditional control and optimization techniques, and give immediate real-time answers to any faced situation. The adaptive and immediacy nature of RL methods offers great potential to be used as a decision support system or to directly manage, in an autonomous manner, decision processes since RL can generalize to unseen situations. As has been explained before, the RL models are trained offline based on a reward signal that guides the learning by telling the decision model how good the decisions have been taken in previous states or situations. The RL methodology is proposed given that the solution space is so big that all

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

solutions cannot be tested to find the one that better fits the end-user aims. Therefore, an exhaustive search is not an option. In the case of non-predefined MDP, i.e., user-defined targets, RL models will be trained online given the configuration of the simulations to give recommendations on policy packages in the different defined steps from the time horizon.
How do we know that the RL recommendations are favourable given the policy goals defined by the users? Model validation will be done in two ways:

- Domain expert validation: RL recommendations will be evaluated by domain experts and SHs in the co-creation framework workflow, in order to assess whether the recommendations are aligned with the requirements and needs of each CS and provide added value to the potential users of the SLNAE tool.
- Benchmark with metaheuristics: Different scenarios will be executed with other control and optimization techniques in order to compare the results with the RL recommendations. These benchmarks will assess the predictive quality of the control models. Metaheuristic models such as Particle Swarm Optimization [31], Genetic algorithms [32], or Simulated Annealing [33] combined with local search algorithms will be evaluated. Even though metaheuristics are not real-time and adaptable system solutions as RL provides, traditional optimization algorithms can yield good sequential solutions. Moreover, these searching techniques will be considered to support the RL decision-making models. Different options will be examined:
    - RL vs metaheuristics (e.g., RL models on predefined MDP and metaheuristics on user-defined MDP)
    - Hybrid recommendation system
- Benchmark with other algorithms: For instance, the Tree Parzen Estimator [34], which is a sequential model-based optimization algorithm that uses Bayesian reasoning.

5.2.6. DSS

The SLNAE DSS will provide policy package advice to the platform users. Depending on the required recommendations, it will access the pre-trained agents, or it will trigger a new training session as explained in the previous section.
In the first case, real-time advice for default CS goals achievement is assured since the agents are already available. In the second case, a new training stage will be launched, and newly identified optimal policy packages will be continuously presented to the user. Furthermore, these new agents will be persisted and added to the already available set of agents to be used in future scenarios, thus reducing the next training times.

5.2.7. Simulation Policy Framework

In WP3, the CSs' SDMs are developed using Stella software, a visual programming language for system dynamics modelling introduced in 1985 by Barry Richmond, which enables the definition of stock and flow variables, converters, connectors and other components. Here, SDMs embed also policy effects and low-level nexus indicators.
Due to its format, it cannot be directly integrated nor executed by the SLNAE platform, thus it has to be previously translated into a more convenient programming language. To do so, the Simulation Policy Framework service develops a translation process that takes the SDMs in Stella format as an input, and integrates them into the SLNAE. This procedure is based on the

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

methodology implemented in the H2020 Sim4Nexus project, and proposes various improvements based on the lessons learned there.
Once a new SDM is available, the following automatic flow is triggered:

- Variable names are validated against the NGX Variables Inventory, and later translated to Python nomenclature.
- Constant data is extracted and loaded into a specific and isolated data structure.
- Time series data is extracted and loaded into a specific and isolated data structure.
- Equations are extracted and loaded into a specific and isolated structure.
- Initial stocks are extracted and loaded into a specific and isolated data structure.
- Based on the previous data structures, the Python SDM is defined.
- The Python SDM is executed from the first year till 2050 to check its correctness.
- The outputs are validated monthly against a validation data set.
- The Python SDM is executed from the first year till 2050 with some applied policies to check its correctness.
- The outputs are validated monthly against a validation data set.

Once ready, the SDMs are included in the SLNAE system to be used by other platform components, such as the self-learning engine.
Additionally, the Simulation Policy Framework also integrates policy, goals, targets and WEFE indicators information to control its logic. These data are available in the NXG Data Sharing Tools.

5.2.8. Analytical Engine

The analytical engine will support other technical SLNAE components in the analytical computation tasks. For example, it will enable access to Hardware resources such as CPU or GPU, and will incorporate all the required Python libraries for the DRL agents' training (Tensorflow[13] or Pytorch[14]).
Additionally, it will also incorporate other data analytics pipelines in order to process the NGX data.

5.3. Nexogenesis Data sharing tools

Smooth and dynamic communication between WPs is crucial for the success of the NXG project. In this context, several complex and high dependent cross-WP data pipelines have been identified (Figure 8). This issue enforces not only the initial requisite of having to define a specific and well-defined plan to manage it, e.g. the NXG co-creation framework, but also the necessity of digital services able to act as a bridge between them.
In WP4, task T4.2 is aimed to implement a data-sharing platform able to fill this gap and provide the required support to final users, considering both internal NXG project requirements and external users' necessities. In this scenario, it has been required to identify the roles of data providers and data consumers among the project WPs. Those WPs that, among its objectives, have to generate data to be used by other WPs in the project data pipeline are considered data

[13] https://www.tensorflow.org/
[14] https://pytorch.org/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

providers. Extensively, in a similar way, those WPs that require and use other WPs' outcomes in order to accomplish their tasks are considered data consumers, Finally, in this schema, a WP can be at the same time both data consumer and data provider, as is the case of the WP3. Figure 7 presents this classification, and further details are introduced in the next section.
Additionally, task T4.2 also comprises all those services to harmonize the NXG data and make it available to the general public. In the previous roles classification, the general public is considered a special case of a data consumer.

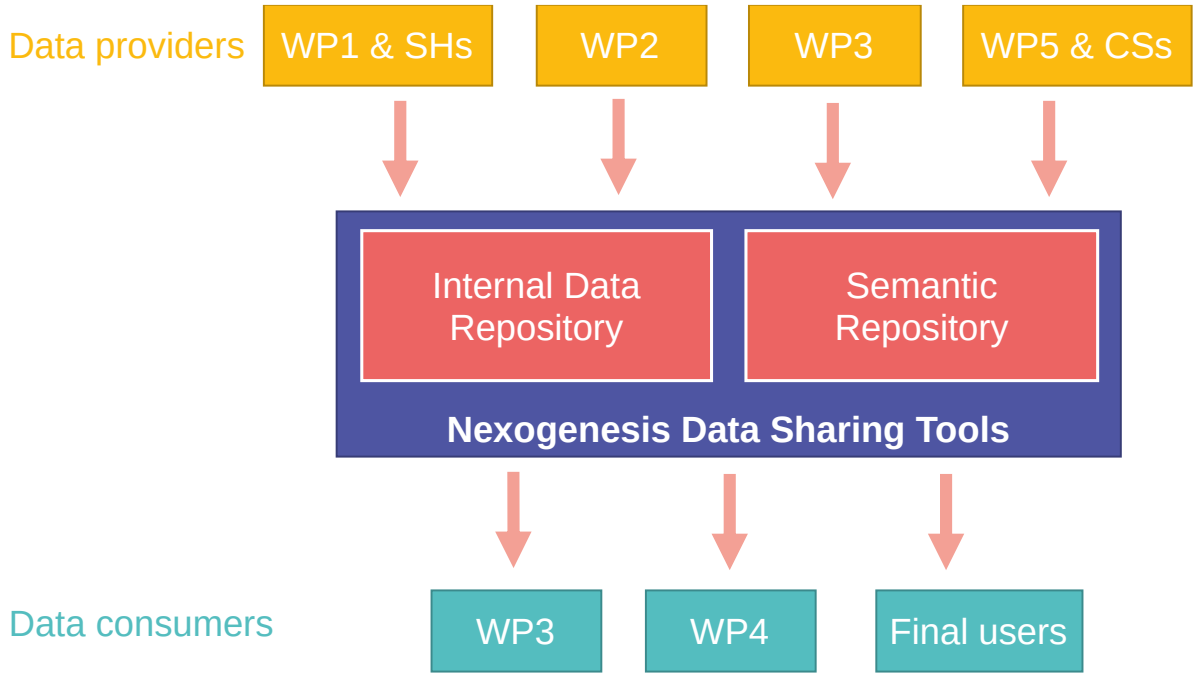

Figure 7. Nexogenesis data pipeline

Depending on the final use of the data, these are available via a specific tool in the NXG Data Sharing Platform. In a first stage, data is used internally by the WPs with the final objective to develop the nexus complexity models (the SDMs) and ultimately the SLNAE service. To this goal, the NXG Internal Data Repository (IDR) is implemented. Second, complying with the Open Data policy and the DMP, data is harmonized and published (when it is possible) in an online Semantic Repository for public open access.
The following sections describe the designs of such data-sharing tools based on the identified requirements.

5.3.1. Internal Data Repository

During the NXG project, many datasets are, and will be, created and updated. This process is being directly monitored by the DMP, but will be also supported by the NXG Internal Data Repository. In this case, the IDR will act as a digital platform to enable efficient cross-WP data sharing among data providers and data consumers.
Given the elevated number of datasets that have to be shared among WPs, this data exchange may become difficult to coordinate. Thus in order to avoid point-to-point (WP-to-WP)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

communication, lessons learned from Sim4Nexus project (where similar requirements were not correctly addressed) demonstrated the utmost importance of this kind of tools.

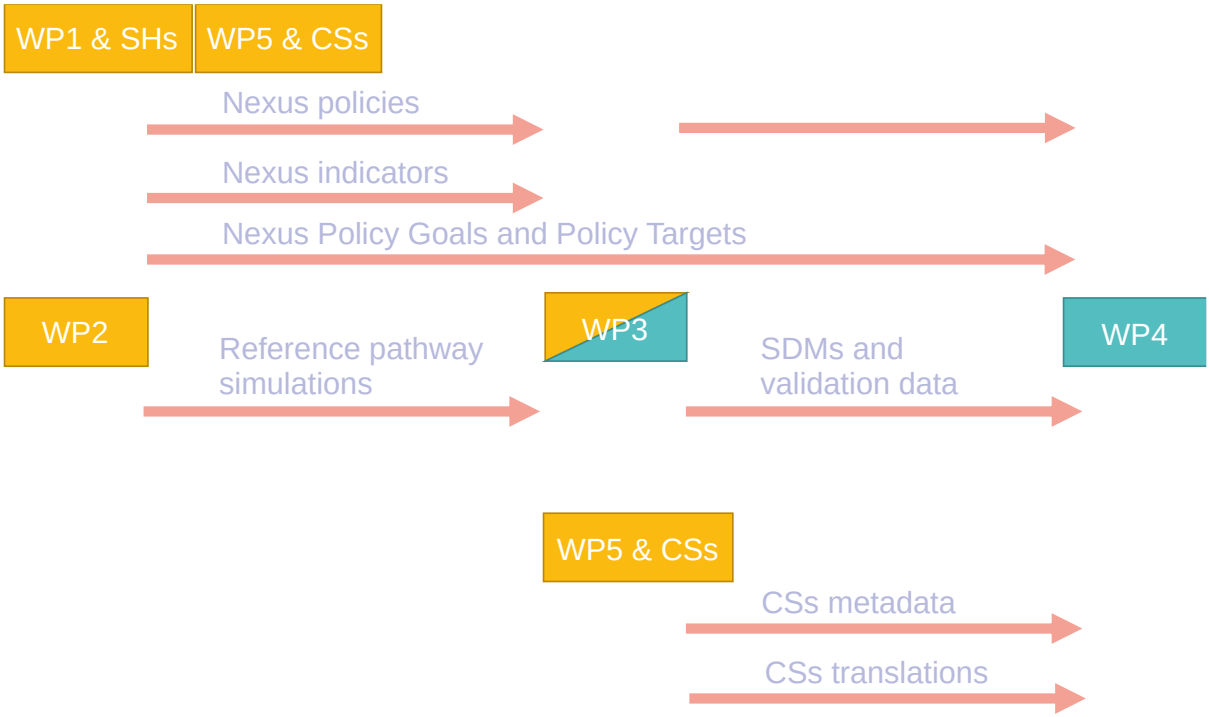

Figure 8. NXG cross-WP data pipelines in the Internal Data Repository

From the proposed use cases, the data pipeline represented in Figure 8 is extracted. This pipeline represents the data flows and data dependencies between WPs in the NXG project. WP1 & SHs, WP2 and WP5 & CSs have the role of data providers, WP4 is a data consumer, and WP3 has both roles at the same time. Details regarding data flow, data and data formats are presented in Table 5.

Table 5.Internal Data Repository data, data formats, data flows, and description

Data	Data format	Data provider	Data consumers	Description
Nexus Policies	Excel file per CS	WP1 & SHs and WP5 & CSs	WP3 & WP4	To be used by WP3 when including policies in the SDMs, and by WP4 to build SLNAE and to show policies information in the UI
Policy Goals, Policy Targets, Indicators, etc	Excel file per CS	WP1 & SHs and WP5 & CSs	WP3 & WP4	To be used by WP3 to compute low-level indicators in the SDMs, and by WP4 to build SLNAE and to show information in the UI

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Biophysical and socio-economic modelling (Reference Pathways) simulations	Excel file per CS and RP	WP2	WP3	To be used by WP3 to develop the SDMs. A specific folder structure will be used to organize these datasets.
SDMs	.stmx per CS and RP	WP3	WP4	To be used by WP4 for the SDM translation and integration to the SLNAE
SDMs execution outputs	Excel file per CS, RP and specific policy package	WP3	WP4	To be used by WP4 to validate an SDM translation
WEFE footprint indicators	Excel file per CS	WP1 & SHs, WP3 and WP5 & CSs	WP4	To be used by WP3 to compute low-level indicators in the SDMs, and by WP4 to build SLNAE and to show information in the UI
Case Study metadata	Excel file per CS	WP5 & CSs	WP4	To be used by WP4 to show CS info in the UI
Translations	Excel file per CS	WP5 & CSs	WP4	To be used by WP4 to show CS info in different languages

The technology proposed to implement the NXG IDR is Microsoft OneDrive[15]. The main restriction that guided this design decision is the fact that, generally, the project partners that have to use this tool have no IT knowledge. In this scenario, advanced SQL or NoSQL repositories are not allowed, due to the skills required to work with them. Furthermore, the superior functionalities that these technologies provide have not been identified among the IDR requirements (DST-1 to DST-9).

The NXG OneDrive space will be structured by folders, being the top-level group of folders organized per CSs. The second level, in each CS folder, will have one folder per data type identified in Table 5. The data types *SDMs* and *SDMs execution outputs* are an exception, and will be put in the same folder due to their inter-dependence. Each data provider will be in charge to manage their own data in each sub-space. The data files will follow a versioning convention, starting in version 0.1, and increasing the minor version (0.x) until a big change is added to the data. In this case, the major version (x.0) will be increased. Minor versions will be used when minimal modifications or issues corrections are applied, and changes in major versions will be devoted to representing important improvements. When a new data file is uploaded, the old file will be moved into a folder named *_old*, which will store all the previous versions of that file. Additionally, at the same time, an history.xls file (available at folder level) will be updated by adding a new entry for that new version, and a description regarding the included modifications and the name of the person who did the upload. This history file will be used to have a global

[15] https://www.microsoft.com/es-es/microsoft-365/onedrive/online-cloud-storage

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

idea and keep track of the data modifications. All these rules are considered the NXG IDR protocol.

In those cases where the data format is excel, the structure and content on the file will be agreed upon among data providers and data consumers. The names of each file will be also standardized.

Based on this structure, the IDR will be deployed and hosted in the EURECAT digital premises. EURECAT, the T4.2 leader, will take the role of the system manager, and will administrate the repository. Mainly, three actions are required in this aspect:

1. Administrate the access to project members.
2. Ensure the correct application of the IDR protocol.
3. Solve any possible issues.

Regarding the access topic, a document (e.g. excel file) will be created and maintained to keep track of enabled permissions. By default, Microsoft OneDrive provides security via a log-in service, and access can be enabled by adding accounts to the shared space.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

6. Conclusions

The present document establishes the basis for the SLNAE design, which will be iterated during the first half of the project to finally consider the SHs necessities. Mainly, this feedback will be obtained during the second and third CSs workshops (2nd year of the project), where a deeper discussion on these topics will be proposed, since the first WS was presented as a project introduction.

In this context, some technical details have to be decided, such as the GUI functionalities, how data will be shown, what kind of data will be required to the users in the registration process, the adequate DSS response time, the minimum simulation stepsize, etc.

These pending decisions do not affect the project planning, since they are principally related to T4.5 Self-learning nexus assessment engine, which starts at M24. Other tasks, such T4.2 Data Sharing Tools, T4.3 Modelling of potential WEFE nexus impacts and stakeholder's response, or T4.3. Reinforcement Learning engine, have the required information to start with their responsibilities.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Future Work/Next Steps

The second version of this deliverable will be presented in M24, and will consider the SHs views and feedback on the SLNAE design.

During this time, WP4 will collaborate in workshops WS2 and WS3 with the aim to elaborate on and discuss with SHs the subjects presented in this deliverable to finally design and develop a useful and valuable tool for them.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

References

[1] Ford, Andrew, and Frederick Andrew Ford. Modeling the environment: an introduction to system dynamics models of environmental systems. Island press, 1999.

[2] "IEEE Standard Glossary of Software Engineering Terminology," in IEEE Std 610.12-1990 , vol., no., pp.1-84, 31 Dec. 1990, doi: 10.1109/IEEESTD.1990.101064

[3] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press

[4] Bellman, Richard. "Dynamic programming." Science 153.3731 (1966): 34-37

[5] Singh, Satinder P., and Richard S. Sutton. "Reinforcement learning with replacing eligibility traces." Machine learning 22.1 (1996): 123-158

[6] Watkins, Christopher JCH, and Peter Dayan. "Q-learning." Machine learning 8.3 (1992): 279-292

[7] Sutton, Richard S. "Learning to predict by the methods of temporal differences." Machine learning 3.1 (1988): 9-44

[8] Tsitsiklis, J. N., & Van Roy, B. (1997). Analysis of temporal-diffference learning with function approximation. In Advances in neural information processing systems (pp. 1075-1081)

[9] O'Neill, J., Pleydell-Bouverie, B., Dupret, D., & Csicsvari, J. (2010). Play it again: reactivation of waking experience and memory. Trends in neurosciences, 33(5), 220-229

[10] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533, 2015

[11] Nair, Arun, et al. "Massively parallel methods for deep reinforcement learning." arXiv preprint arXiv:1507.04296 (2015)

[12] Hausknecht, M., & Stone, P. (2015, September). Deep recurrent q-learning for partially observable mdps. In 2015 AAAI Fall Symposium Series

[13] Wang, Ziyu, et al. "Dueling network architectures for deep reinforcement learning." International conference on machine learning. PMLR, 2016

[14] Van Hasselt, H., Guez, A., & Silver, D. Deep reinforcement learning with double q-learning. In Thirtieth AAAI conference on artificial intelligence, 2016

[15] Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. arXiv preprint arXiv:1511.05952

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

[16] Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., & Freitas, N. (2016, June). Dueling network architectures for deep reinforcement learning. In International conference on machine learning (pp. 1995-2003). PMLR

[17] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... & Wierstra, D. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971, 2015

[18] Duan, Y., Chen, X., Houthooft, R., Schulman, J., & Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In International Conference on Machine Learning (pp. 1329-1338), 2016

[19] Larma, M. L., Petersen, B. K., Kim, S. K., Santiago, C. P., Glatt, R., Mundhenk, T. N., ... & Faissol, D. M. (2021). Improving exploration in policy gradient search: Application to symbolic optimization. arXiv preprint arXiv:2107.09158

[20] Rückstieß, T., Felder, M., & Schmidhuber, J. (2008, September). State-dependent exploration for policy gradient methods. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 234-249). Springer, Berlin, Heidelberg

[21] Willianms, R. J. (1988). Toward a theory of reinforcement-learning connectionist systems. Technical Report NU-CCS-88-3, Northeastern University.

[22] Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P. (2015, June). Trust region policy optimization. In International conference on machine learning (pp. 1889-1897). PMLR.

[23] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347

[24] Sewak, M. (2019). Actor-critic models and the A3C. In Deep Reinforcement Learning (pp. 141-152). Springer, Singapore.

[25] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., ... & Kavukcuoglu, K. (2016, June). Asynchronous methods for deep reinforcement learning. In International conference on machine learning (pp. 1928-1937). PMLR

[26] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. (2014, January). Deterministic policy gradient algorithms. In International conference on machine learning (pp. 387-395). PMLR

[27] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, arXiv preprint arXiv:1509.02971

[28] Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018, July). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In International conference on machine learning (pp. 1861-1870). PMLR

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

[29] Haarnoja, T., Tang, H., Abbeel, P., & Levine, S. (2017, July). Reinforcement learning with deep energy-based policies. In International Conference on Machine Learning (pp. 1352-1361). PMLR.

[30] Weiss, Karl, Taghi M. Khoshgoftaar, and DingDing Wang. "A survey of transfer learning." Journal of Big data 3.1 (2016): 1-40

[31] Clerc, M. (2010). Particle swarm optimization (Vol. 93). John Wiley & Sons

[32] Kramer, O. (2017). Genetic algorithms. In Genetic algorithm essentials (pp. 11-19). Springer, Cham

[33] Dowsland, K. A., & Thompson, J. (2012). Simulated annealing. Handbook of natural computing, 1623-1655

[34] Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. Advances in neural information processing systems, 24

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Annexes

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Annex I

Table AI.1. Use Cases

ID	Description	Component	Goal	System users	Main course (steps)	Potential requirements
UI-1	The user is able to see a landing page explaining the NXG project and methodologies (e.g., the origin of reference pathways, and some initial data)	User interface - General	Let people know about the NXG tool and the project itself	User (All)	1. Open nxg tool website 2. Browse the landing page 3. Read the NXG project information (common information for all CSs)	
UI-2	The user shall get information about the CSs: objectives, future scenarios, available policies, policy packages, goals, targets and indicators	User interface - General	Let people know about CSs in the project	User (All)	1. Open nxg tool website 2. Browse the landing page 3. Read the CSs information 4. The user can obtain information about the elements for a simulation: objectives, policies, policy packages, goals, targets, and indicators	Automatic integration of CSs information
UI-3	The user can select which CS wants to see in more detail	User interface - General	Investigate the particularities of a Case study	User (All)	1. The user enter into the tool 2. The user selects one of the available case studies (from a map)	Automatic integration of CSs information

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

					3. A new section is opened with data of the CS	
UI-4	The user is able to sign up and log in in the tool	User interface - General	Let people save results, see detailed analysis and results per simulations and other future features	User (All)	1. Open nxg tool website 2. Browse the landing page searching for register button 3. Fill the form and submit 4. Send confirmation email	GDPR compliance
UI-5	The user is able to manage its profile, settings and another user functionalities (i.e., recover the password, language, etc)	User interface - General	Let the users is able to manage its profile	User (All)	1. Open nxg tool website 3. Go to login section 3.1. Click on recover password section 3.2. Start process to recover the password: send email/type email & new password 4. Go to profile section 4.1. Change language	GDPR compliance
UI-6	The user wants to start a new simulation process	User interface - Simulations Management	Let the users run new simulations	User (All)	1. The user selects the 'NEW simulation' function 2. The simulation's configuration section is presented (UI-11)	

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

UI-7	The user wants to access the saved simulations	User interface - Simulations Management	Let the users access the saved simulations	User (All)	1. The user is logged in 2. The user access to the simulation's management section 3. The saved simulations are presented	API and Database
UI-8	The user wants to duplicate/rename/remove saved simulations in a table format	User interface - Simulations Management	Let the users manage the saved simulations	User (All)	1. The user is logged in 2. The user access the simulation's management section 3. The saved simulations are presented 4. The user can duplicate, edit, rename, and remove saved simulations	API and Database
UI-9	The user wants to see the results of saved simulations	User interface - Simulations Management	Let the users access the saved results	User (All)	1. The user is logged in 2. The user access the simulation's management section 3. The saved simulations are presented 4. The user can select one of the saved simulations 5. The results of the selected simulations are presented	API and Database

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

UI-10	The user wants to load a saved simulation	User interface - Simulations Management	Let the users rerun (and possibly modify) a saved simulation	User (All)	1. The user is logged in 2. The user access the simulation's management section 3. The saved simulations are presented 4. The user can select one of the saved simulations 5. The simulation is loaded	API and Database
UI-11	The user shall configure a new simulation in an easy user interface	User interface - Simulations Configuration	To configure all possible parameres to run a simulation: Select a CS, select a reference pathway, level of uncertainty, select/define targets, select indicators, etc	User (All)	(The user is in the Configuration section) 1. The user selects the CS 2. The user selects the reference pathway 3. The user selects the level of uncertainty 4. The user selects/defines targets and indicators (different time horizon on the achievement of targets can be defined) 5. The users clicks "next" to go to the Simulation section	Wizard process Database or API
UI-12	The user shall be able to select and apply policy packages	User interface - Simulation	Simulate the effects of policies in the nexus. Get Level of policy	User (All)	(The user is in the Simulation section) 1. The user has the option to	API and polices logic system

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

	through the simulation time horizon.		integration (conflicts-synergies) – Policy impacts		select multiple policies from different nexus sectors (WEFE) and to apply them at a given time step	
UI-13	The user can navigate forward and backward through the simulation time horizon in a dynamic timestep size	User interface - Simulation	Change the implementation of policy packages based on current results	User (All)	(The user is in the Simulation section) 1. The user selects a policy package to be applied (UI-12) and a specific timestep size to advance in the simulation time horizon 2. The simulation is run 3. The user decides whether to continue o go back into the simulation 4. This cycle is repeated until the user decides to finish the simulation	API and polices logic system
UI-14	The user can personalize a policy when it is applied in the simulation	User interface - Simulation	Configure the policies to simulate with start and end parameters, with a range of application, and some additional decision variables.	User (All)	(The user is in a Simulation section) 1. The user configure the policies (budgets allocated, aims, etc if possible) 2. The user configures when to start applying them 3. The user configures the	API and polices logic system

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

					duration of the policy (if possible) 4. Other configurations may be needed per policy	
UI-15	The user shall be able to see the information of the configuration section, the applied policy packages, the nexus variables and the level of achievement of both default and configured nexus targets along with the simulation time horizon and available tokens	User interface - Simulation	To have the reference about the configuration stage and the current simulation stage	User (All)	(The user is in a Simulation section) 1. The user can visualize the configuration details: CS, reference pathway, uncertainty, indicators, ... 2. The user can visualize the selection of policy packages, targets achievement, etc	UI design
UI-16	The user shall be able to access to policy package recommendations based on both default and configured nexus targets	User interface - Simulation	Get some support in the decision of selecting policies	User (All)	(The user is in a Simulation section) 1. The user opens a selct button an choose the DSS features to apply in the simulation 2. expected options: entire policy packages or fine-	DSS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

					tuning of selected by user policy packages.	
U-17	The user shall be able to save a simulation at any moment	User interface - Simulation	Save simulation state	User (All)	(The user is in the Simulation section) 1. The user saves the current simulation	API and Database
UI-18	The user shall be able to select the way to visualize the results from a set of possibilities	User interface - Simulation results	Select the best way to assess NEXUS	User (All)	(The user is in the Results section) 1. The user is able to see a menu with different options to present the results and evaluations 2. The user is able to select from the menu the way to see the simulation results (simple view, detailed view)	UI design
UI-19	Simple view: The user wants to see results of applying a set of policies in a case study along the years	User interface - Simulation results	Simulate the effects of policies in the nexus, calculate indicators and targets, and output variables for a CS and reference pathway	User (All)	(The user is in the Results section) 1. The user is able to see in graphics the effects of the policies in the nexus sectors 2. The user is able to see in	UI design

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

					a table the indicator results and the targets	
UI-20	Detailed view: The user wants to visualize the uncertainty in the results	User interface - Simulation results	Uncertainty must be considered in the viualization part	User (All)	(The user is in the Results section) 1. The user sees the results 2. The user selects the depth technical analysis: The user sees the results with uncertainty visualization	UI design
UI-21	Detailed view: NEXUS impacts of the application of policy packages and SDM variables	User interface - Simulation results	Obtain a depth numerical analysis in NEXUS sectors and SDM variables simulations	User (All)	(The user is in the Results section) 1. The user sees the results 2. The user selects the depth technical analysis: how the application of a policy in a nexus sector affects another nexus sectors is presented	UI design

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

UI-22	The user wants to compare the results of different simulations in the same CS	User interface - Simulations comparison	Compare simulations in the same CS using the same framework	User (All)	(The user is in the Simulations Management section) 1. The user is able to select the saved simulations that wants to compare from the same CS, same Reference Pathway, and same default policy goals 2. The user is able to see the comparisons with the targets and indicators achievement and applied policy packages	API, Database and UI design
UI-23	The user wants to compare the results of different simulations between different CSs	User interface - Simulations comparison	Compare simulations between different CSs using the same framework	User (All)	(The user is in the Simulations Management section) 1. The user is able to select the saved simulations that wants to compare from different CS, same Reference Pathway, WEFE footprint indicators, and common policy goals 2. The user is able to see the comparisons with the	API, Database and UI design

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

					targets and indicators achievement and applied policy packages	
DSS-1	The user wants to get recommendations on policy packages based on default targets	DSS functionalities - SLNAE	Get policy recommendation according to defined configurations: reference pathway, time steps, number of years, uncertainty, indicators and default targets. And let users see how the 'best' policy option(s) change depending on which targets are prioritised (selected)	User (All)	(The user is in the Simulation section) 1. The user configures stepsize and horizon time for the simulation (**default** targets and indicators already defined in the configuration section) 2. The user asks for recommendation on specific steps during the policy packages selection (can get 1 or more recommendations) 3. The user gets a recommendation based on policies	AI service

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

DSS-2	The user wants to get recommendations on policy packages based on defined by user targets	DSS functionalities - SLNAE	Get policy recommendation according to defined configurations: reference pathway, time steps, number of years, uncertainty, indicators and user-defined targets. And let users see how the 'best' policy option(s) change depending on which targets are prioritised (selected)	User (All)	(The user is in the Simulation section) 1. The user configures stepsize and horizon time for the simulation (**user-defined** targets and indicators already defined in the configuration section) 2. The user asks for recommendation on specific steps during the policy packages selection (can get 1 or more recommendations) 3. The user gets a recommendation based on policies	AI service
DSS-3	Identification of relevant policy packages with the support of an AI in an iterative co-creation process	DSS functionalities - co-creation process	Identification of revelant policy packages for their evaluation with SHs under the co-creation framework	WP1 (Policy modellers) & WP4 (SLNAE) & WP5 (CSs) & SHs	(Iterative co-creation process) 1. Definition of Nexus Policies and Targets (WP1 & WP5 & SHs) 2. Integration of Nexus Policies and Indicators into Complexity Science tools (WP2 & WP3)	Co-creation framework Automatic SDMs integration Automatic Policies logic integration Automatic

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

					3. DSS for the Identification of optimal Nexus Policy packages (WP4) 4. Evaluation of proposed policy packages (WP1 & SHs) 5. Back to 1	Nexus goals, targets and indicators integration AI service
DST-1	the user wants to add/update Policy data for a specific CS	Data Sharing Tool	Last policy data is available for the SDMs and SLNAE development	WP1 (Policy modellers) & WP5 (CSs)	1. The user adds a version number to the new file following stablished protocol 2. The user logs in into the Data repository and save the data (in excel format) in the corresponding folder 3. The user edit the summary file and add information about the new data 4. A notification is sent to required people	A data sharing protocol and platform

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

DST-2	the user wants to add/update Policy Goals, Policy Targets and Indicators data for a specific CS	Data Sharing Tool	Last Policy Goals, Policy Targets and Indicators data is available for the SDMs and SLNAE development	WP1 (Policy modellers) & WP5 (CSs)	1. The user adds a version number to the new file following stablished protocol 2. The user enter into the Data repository and save the data (in excel format) in the corresponding folder 3. The user edit the summary file and add information about the new data 4. A notification is sent to required people	A data sharing protocol and platform
DST-3	the user wants to add/update predictions/simulated biophysical and socio-economic modelling outputs for a specific CS and scenario/reference pathway	Data Sharing Tool	Last simulation data is available for the SDMs construction	WP2 (Environment simulators)	1. The user adds a version number to the new file following stablished protocol 2. The user enter into the Data repository and save the data (in ascii/binary formats) in the corresponding folder 3. The user edit the summary file and add information about the new	A data sharing protocol and platform

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

					data 4. A notification is sent to required people	
DST-4	the user wants to access to data to build an SDM for a specific CS and scenario/reference pathway	Data Sharing Tool	Access to new data (either new version of Policies, Policy Goals or simulated reference pathways) to work on a specific SDM	WP3 (SDMs)	1.1 DST-1 has been successfully completed 1.2 Or DST-2 has been successfully completed 1.3 Or DST-3 has been successfully completed 2. The user enters the Data Repository in the corresponding folder and downloads the data	A data sharing protocol and platform
DST-5	the user wants to add/update an SDM for a specific CS and scenario/reference pathway	Data Sharing Tool	Last SDMs versions are available for translation in SLNAE	WP3 (SDMs)	1. The user adds a version number to the new file following stablished protocol 2. The user enter into the Data repository and save the SDM (in .stmx format) and the required validation data (in excel format) in the	A data sharing protocol and platform

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

					corresponding folder 3. The user edit the summary file and add information about the new data 4. A notification is sent to required people	
DST-6	The user wants to add/update WEFE footprint indicators	Data Sharing Tool	Last WEFE footprint indicators data is available for the SDMs and SLNAE development	WP3 (SDMs)	1. The user adds a version number to the new file following stablished protocol 2. The user enter into the Data repository and save the data (in excel format) in the corresponding folder 3. The user edit the summary file and add information about the new data 4. A notification is sent to required people	A data sharing protocol and platform

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

DST-7	the user wants to access to data to work on SLNAE	Data Sharing Tool	Access to new data (either new versions of Policies, Policy Goals, SDMs or WEFE footprint indicators, CSs info, etc) to work on the SLNAE	WP4 (SLNAE)	1.1 DST-1 has been successfully completed 1.2 Or DST-2 has been successfully completed 1.3 Or DST-5 has been successfully completed 1.3 Or DST-6 has been successfully completed 2. The user enters the Data Repository in the corresponding folder and downloads the data	A data sharing protocol and platform
DST-8	WP5 team wants to add/update CS metadata	Data Sharing Tool	Last CS metadata is available for the SLNAE development	WP5 (CSs)	1. The user adds a version number to the new file following stablished protocol 2. The user enter into the Data repository and save the data (in excel format) in the corresponding folder 3. The user edit the summary file and add information about the new data 4. A notification is sent to required people	A data sharing protocol and platform

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

DST-9	WP5 team wants to add/update text translations	Data Sharing Tool	Last translations are available for the SLNAE development	WP5 (CSs)	1. The user adds a version number to the new file following stablished protocol 2. The user enter into the Data repository and save the data (in excel format) in the corresponding folder 3. The user edit the summary file and add information about the new data 4. A notification is sent to required people	A data sharing protocol and platform
DST-10	An external user wants to access to the NXG open data	Data Sharing Tool	An external user access to Open and Harmonized data generated during the NXG project through the Semantic Repository	User (All)	1. The user access to the Semantic Repository 2. The user obtains the required information	Sematic Repository

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

Annex II

Table AII.1. Requirements

ID	Use case	Component	Description	Details	Prerequisites	To be iterated
FR-01	UI-4	User interface - General	The system allows the user's registration (sign in)	At least, username, email and password information will be required		
FR-02	UI-4	User interface - General	The system allows the user to log in			
FR-03	UI-4	User interface - General	The system allows the user to log out			
FR-04	UI-4	User interface - General	The system allows the user to log in in a guest mode	No sign in is required		
FR-05	UI-4	User interface - General	The system allows the user's password recovery			
FR-06	UI-5	User interface - General	The system allows the user's profile management	At least, username and email information		
FR-07	UI-5	User interface - General	The system allows the user to change the language	English, Greek, Bulgarian, Italian, German, Romanian, Serbian, Latvian, Lithuanian	A native translator from English to each CS is required	*
FR-08	all	SLNAE-Backend	The system models the user software entity	The user entity contains a username, email, registration date and encrypted password		*
FR-09	UI-4	User interface - General	The system shows information about cookies and GDPR			
FR-10	UI-1	User interface - General	The system has a main view	General project information is shown		
FR-11	UI-2	User interface - General	The system has a CSs main view	General CSs information is shown on a map		

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

FR-12	UI-2	User interface - General	The system has a CS specific view	5 CS specific pages provide detailed information: - CS's characteristics - Available policies - Nexus goals and targets		
FR-13	UI-6, UI-all	User interface - General	The system has a shortcut to access the simulation configuration view to start a new simulation			
FR-14	UI-7, UI-all	User interface - General	The system has a shortcut to access the simulation management view			
FR-15	UI-all	User interface - General	The system has a shortcut to access the simulation view			
FR-16	UI-all	User interface - General	The system has a shortcut to access the simulation results view			
FR-17	UI-all	User interface - General	The system has a shortcut to access the simulation comparison view			
FR-18	UI-11	User interface - Simulations Configuration	The system has a view to configure a simulation	Static parameters are presented to the user to configure the simulation. At least, the following parameters are included: - The CS - The reference pathway - Simulation goals, targets and WEFE or CS specific indicators (to be defined by the user) - Uncertainty mode		*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

FR-19	UI-11	User interface - Simulations Configuration	The simulation configuration view allows the user to select and configure simulation goals and targets	Specific simulation goals can be defined, and new targets can be configured. The time step to achieve these targets can also be set		*
FR-20	UI-7 to UI-10	User interface - Simulations Management	The system has a view to manage saved simulations	Saved simulations are presented in a table mode.	the user is logged in	
FR-21	UI-9	User interface - Simulations Management	The simulation management view allows the user to see the results of a simulation			
FR-22	UI-10	User interface - Simulations Management	The simulation management view allows the user to load a simulation			
FR-23	UI-8	User interface - Simulations Management	The simulation management view allows the user to remove a simulation			
FR-24	UI-8	User interface - Simulations Management	The simulation management view allows the user to rename a simulation			
FR-25	UI-8	User interface - Simulations Management	The simulation management view allows the user to duplicate a simulation	By default, the new simulation has the source simulation name plus '_copy'		
FR-26	UI-22, UI-23	User interface - Simulations Management	The simulation management view allows the selection of multiple simulations for comparison	No more than five simulations can be compared at the same time		

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

FR-27	UI-12 to UI-17	User interface - Simulation	The system has a view to run a simulation			
FR-28	UI-12, UI-13	User interface - Simulation	The simulation view shows the simulation time horizon	from 2020 till 2050		
FR-29	UI-13	User interface - Simulation	The simulation view allows the user to go forward and backward through the simulation time horizon			
FR-30	UI-12	User interface - Simulation	The simulation view allows the user to select a specific timestep to be simulated	The minimum time step size is 1 year		*
FR-31	UI-12	User interface - Simulation	The simulation view shows available policies to be applied.	Policies information is shown (e.g. building time, duration, required tokens)		*
FR-32	UI-12	User interface - Simulation	The simulation view allows the user to apply policies	Policies can be applied at specific time step, and a simulation button triggers the execution of the configured scenario		
FR-33	UI-12	User interface - Simulation	The simulation view allows the user to remove applied policies			
FR-34	UI-12, UI-15	User interface - Simulation	The simulation view shows the applied policies			
FR-35	UI-14	User interface - Simulation	The simulation view allows the user to configure policies	In those cases when the policy can be configured		
FR-36	UI-12	User interface - Simulation	The simulation view limits the application of policies when required	When a policy can't be applied due to restrictions with other		

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

				already selected policies, they should be hidden.		
FR-37	UI-15	User interface - Simulation	The simulation view shows available tokens to apply policies	The application of a policy has associated an economic and social costs expressed in tokens		*
FR-38	UI-12	User interface - Simulation	The simulation view limits the application of policies when enough tokens are not available	When a policy can't be applied due to token limitations, they should be hidden		*
FR-39	UI-15	User interface - Simulation	The simulation view shows the level of achievement of default Nexus Goals, targets and WEFE footprint indicators	A set of default metrics per CS (WEFE footprint indicators are the same across CSs)		
FR-40	UI-15	User interface - Simulation	The simulation view shows the level of achievement of user-defined Nexus Goals, targets and WEFE footprint indicators	A set of user-defined metrics per simulation		*
FR-41	UI-15	User interface - Simulation	The simulation view shows how Nexus variables evolve during simulation			
FR-42	UI-17	User interface - Simulation	The simulation view allows the user to save a simulation	A simulation name must be given, and another simulation metadata is also persisted, such as the creation and modification date. Simulation configuration, selected policy packages and targets achievement level is saved	the user is logged in	
FR-43	UI-16, DSS-1	User interface - Simulation	The simulation view shows policy packages recommendations aimed to	Recommendations generation time should be low (<30s)		

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

			achieve default goals, targets and WEFE footprint indicators			
FR-44	UI-16, DSS-2	User interface - Simulation	The simulation view shows policy packages recommendations aimed to achieve user-defined goals, targets and WEFE footprint indicators			*
FR-45	UI-6 to UI-23	SLNAE-Backend	The system models the simulation software entity	The simulation entity contains a simulation name, creation and modification date time, the configuration static parameters, the applied policies, and targets achievement level		
FR-46	UI-18 to UI-21	User interface - Simulation results	The system has a view to show the simulation results	The user can choose among two specific view modes to analyse simulation results		
FR-47	UI-19	User interface - Simulation results	The simulation results view has a basic mode	Mode to be used by non-expert users where basic Nexus information is shown		*
FR-48	UI-20, UI-21	User interface - Simulation results	The simulation results view has an advanced mode	Mode to be used by expert users where a complete view of Nexus information is shown		*
FR-49	UI-20, UI-21	User interface - Simulation results	The advanced mode of the simulation results view can show uncertainty in simulations			*
FR-50	UI-20, UI-21	User interface - Simulation results	The advanced mode of the simulation results view shows a deeper detail on Nexus variables			*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

FR-51	UI-22, UI-23	User interface - Simulation comparison	The system has a view to allow the simulations comparison	Applied policies are shown for each simulation		
FR-52	UI-22	User interface - Simulation comparison	The simulations comparison view allows to compare simulations ran for a given CS	The level of achievement of default Goals and Targets is compared. Other common indicators are also shown in the comparison		*
FR-53	UI-23	User interface - Simulation comparison	The simulations comparison view allows to compare simulations ran for any CS	The level of achievement of WEFE footprint indicators		*
FR-54	all	SLNAE-Backend	The system offers a REST API to manage the software entities	CRUD methods are offered for each of the identified entities (e.g. simulations)		
FR-55	all	SLNAE-Backend	The system persists the software entities in a database	CRUD methods are offered for each of the identified entities (e.g. simulations)		
FR-56	UI-12 to UI-17	SLNAE-Backend	The system implements the polices logic	Building time, active time, required tokens, generated tokens, permanent		
FR-57	DST-1, DST-2	SLNAE-Backend	The system automatically integrates policies when new versions are available			
FR-58	DST-5, DST-7	SLNAE-Backend	The system automatically integrates SDMs when new versions are available	Stella language is translated to python.		
FR-59	DST-2, DST-6, DST-7	SLNAE-Backend	The system automatically integrates Nexus goals, targets and WEFE and CSs indicators when new versions are available			

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

FR-60	DST-7, DST-8	SLNAE-Backend	The system automatically integrates CSs information			
FR-61	DST-10	Data Sharing Tool	The system automatically harmonize data	Following adequate ontologies		
FR-62	DST-10	Data Sharing Tool	The system automatically publishes data in an open infrastructure	In the Semantic Repository		
FR-63	DSS-3	DSS	The system identifies optimal policy packages given a predefined set of indicators to be achieved	In background		
FR-64	DST-1, DST-7		The system allows the user to upload/download data to define Policies	In excel format for a specific CS		
FR-65	DST-2, DST-7		The system allows the user to upload/download data to define goals, targets and indicators	In excel format for a specific CS		
FR-66	DST-3, DST-4		The system allows the user to upload/download biophysical and socio-economic modelling outputs	In ascii/binary formats for a specific CS and reference pathway		
FR-67	DST-5, DST-7		The system allows the user to upload/download SDMs	In Stella (.stmx) format for a specific CS and reference pathway		
FR-68	DST-6, DST-7		The system allows the user to upload/download data to define WEFE footprint indicators	In excel format for a specific CS		
NFR-01	all	all	Usability. The interaction by the user with the system must be simple and easy to use.			

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

NFR-02	all	all	Usability: Simple navigation structure, with the buttons always in the same place and with the possibility of going back or to the main menu.			
NFR-03	all	all	Usability: The UI will follow the principles of adaptive/responsive web design.			
NFR-04	all	all	Usability: The UI must be compatible with all major modern browsers (Firefox, Chrome, Edge/IE).			
NFR-05	all	all	Security: Standard security capabilities in browsers, this includes the ability to manage cookies and the use of HTTPs.			
NFR-06	all	all	Security: The system must implement all the necessary security components to ensure the privacy and confidentiality of users and the protection of the processed data.			
NFR-07	all	all	Reliability: The system must be consistent, with good error control and management.			
NFR-08	all	all	Reliability: The system must be a high availability service ("High availability"), ensuring that the platform can continue to function at a lower level in the event of failures of some of its components.			

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881

NFR-09	all	all	Scalability: The service must be scalable to provide acceptable response time			

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101003881