Horizon 2020 Societal challenge 5

Climate action, environment, resource

Efficiency and raw materials

D3.8: FINAL REPORT ON DATA FLOW AND ORGANISATION

LEAD AUTHORS: Lydia S. Vamvakeridou-Lyroudia & Xavier Domingo Albin

OTHER AUTHORS: Lluis Echeverria (EURECAT) , Aitor Corchero (EURECAT), Marcel Ortiz (Eurecat), Pelagia Koutsantoni (EPSILON), Barry Evans (UNEXE), Maria Blanco (UPM)

DATE: 31 May 2020

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement NO 689150 SIM4NEXUS

PROJECT	Sustainable Integrated Management FOR the NEXUS of water-land-food-energy-climate for a resource-efficient Europe (SIM4NEXUS)
PROJECT NUMBER	689150
TYPE OF FUNDING	RIA
DELIVERABLE	D3.8: FINAL REPORT ON DATA FLOW AND ORGANISATION
WP NAME/WP NUMBER	Thematic Models and Integration / WP3
TASK	Task 3.1
VERSION	1.0
DISSEMINATION LEVEL	Public
DATE	31/05/2020 (Due date)
LEAD BENEFICIARY	UNEXE
RESPONSIBLE AUTHOR	Lydia S. Vamvakeridou-Lyroudia (UNEXE)
AUTHOR(S)	Xavier Domingo Albin (EURECAT), Lluis Echeverria (EURECAT), Aitor Corchero (EURECAT), Marcel Ortiz (Eurecat), Pelagia Koutsantoni (EPSILON), Barry Evans (UNEXE), Maria Blanco (UPM)
INTERNAL REVIEWER	Floor Brouwer (WUR)

DOCUMENT HISTORY

VERSION	INITIALS/NAME	DATE	COMMENTS-DESCRIPTION OF ACTIONS
1.0	LVL (UNEXE)	21/05/2020	FIRST OUTLINE AND VERSION
1.1	LER (EURECAT)	22/05/2020	REVIEW OF MAIN STRUCTURE
1.2	ACR (EURECAT)	24/05/2020	SEMANTIC REPOSITORY
1.3	MOS (EURECAT)	26/05/2020	KEE
1.4	BE (UNEXE)	29/05/2020	SDM ENGINE
2.0	LER (EURECAT)	29/05/2020	METADATA & FINAL VERSION

Table of Contents

Figures

Tables

Executive summary

This deliverable reports on the data flow between WP3 and WP4 for the development of the System Dynamics Models for each project study in WP3 and the information exchange for the development of the Serious Game in WP4. It is a final report, showing that at the end of the project (M48) the Architecture of the database, the data flow, the metadata ontology, and data management have been developed.

More concretely, it firstly defines the Semantic Repository, which is the storage where all information related to the Nexus (e.g. nexus variables, policy cards, policy objectives, etc), the specific Case Studies, and the Serious Game, among others, is stored in a standardised manner, making use of semantic technologies.

However, not all information is needed to be stored in the Semantic Repository, but is really important for the construction of the System Dynamic Models. These files, the folder structure where they are stored, the way they are shared, and accessed is provided in this report.

Finally, all this information will be put in production in the Serious Game User Interface. For doing that, web services are used to interchange information between the different modules in the SIM4NEXUS architecture. The current version of the architecture, the modules, and the web service are also provided in this report.

Changes with respect to the DoA

Not applicable.

Dissemination and uptake

This report will be released on the project website. The deliverable has been written to support the development of the SIM4NEXUS project and is open to all stakeholders, including the case study leaders and researchers contributing to the case studies.

Short Summary of results (<250 words)

This deliverable reports on the data flow between WP3 and WP4 for the development of the System Dynamics Models for each project study in WP3 and the information exchange for the development of the Serious Game in WP4.. It is a final report, showing that at the end of the project (M48) the Architecture of the database, the data flow, the metadata ontology, and data management have been developed.

Evidence of accomplishment

Submission of report.

Glossary / Acronyms

As the document is being written, terms and glossary will be added here as needed. Before the last version is submitted this list will be re-arranged alphabetically by the lead author.

TERM	EXPLANATION/MEANING
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
CSV	Comma Separated Values
GDP	Gross Domestic Product
GIS	Geographic Information System
JSON	JavaScript Object Notation
KEE	Knowledge Elicitation Engine
OWL	Web Ontology Language
R	Programming language
SDM	System Dynamics Modelling
STELLA	Graphics and development software for SDM
WPS	Web Processing Service
XML	eXtensible Markup Language

1 Introduction

In SIM4NEXUS (WP3) different types of thematic models will be applied for the case studies (T3.3), each with their own data type format, under different scenarios, as detailed in this Deliverable 3.1, and Deliverable 4.2. Also, the outputs from the thematic models, the downscaling (T3.2) and the data coming from freely accessible sources need to be organized in a convenient way for the complexity science models (T3.4 and T3.5) and compatible for WP4, including additional and non-numerical information, as needed. Task 3.1 coordinates all these actions and activities for the smooth execution of all the tasks in WP3 and the compatibility of data flow for smooth cooperation with the other related WP, namely WP4.

This Deliverable is an output related to the work carried out within Task 3.1. It is the final report, at the end of M48, showing that the data flow and structure, for the development of complexity science models and the Serious Game for each Case Study have been defined and developed. It covers work related to both WP3 and WP4, covering activities in both Work Packages and coordinating data share and data flow between them. It also details data types used, and shared folder structure to better understand how the information is accessed and used. Consequently, the partners involved in this report are the two WP Leaders and Co-Leaders: UNEXE and UPM (WP3), EURECAT and EPSILON (WP4).

The work detailed in this document is strongly linked to another Deliverable (D4.2), where the Data Management Plan in full is detailed, together with the definition of existing datasets.

Structure of this document:

- Introduction: This section.
- Semantic Repository: Current version of the Semantic Repository is depicted, entities, relationships, formats, and storage is defined.
- Metadata and other data sources: Datasets and other data sources used for building the SDM are depicted. Also how these datasets are stored, shared, and accessed.
- Architecture and data flow between modules: this section previews the SIM4NEXUS architecture and data flow between modules which gives support to all computation needs in this project.
- Data Management: Refers to Deliverable 4.7, where it is deeply explained.

2 Semantic Repository

The Semantic Repository is designed to store information related to the concepts, properties and restrictions from the Nexus procedures; to improve the data integration of diverse sources and, finally, to support the analytical services to be developed under SIM4NEXUS. This repository, which is currently focused on describing and interrelating the nexus variables with the corresponding policies and game context. Indeed, the semantic repository supports also the harmonization and undestanding of the System Dynamic Models and Thematic Models under a common structure.

The semantic repository has been fully described in the D4.4 entitled as "Semantic Repository". In this deliverable, the semantic repository is formed by two important elements: (i) the semantic model (called SIM4NEXUS ontology) to interrelate and make accessible nexus information; and (ii) data exploration and navigation tool in order to explore nexus information and knowledge.

The SIM4NEXUS ontology corresponds to an Ontology Web Language (OWL) model that has been serialised in Turtel notation among others (RDF, NT and JSON-LD). The onology has been published in the following link:

https://seriousgame.sim4nexus.eu/ontology/

The model (see Figure 1) has been updated since the initial version published in D4.4. The final version of the model already available in the link is divided into two main parts mainly: (i) representation of the policies and nexus variables interrelationship; and (ii) the interrelationship between the game knowledge (policy cards, policies, policy objectives, etc) and the corresponding implications in the nexus. Based on these insights, the SIM4Nexus ontology top concept starts with the "Case-Study" that represents the different geographical scenarios in which the different simulations and actions are performed. The "Case-Studies" are divided into different "Regions" that represent small parts or territories of the overall scenario. The "Case-studies" and the "Regions" can contain different assets of infrastructures such as "Energy infrastructure", "Water infrastructure" to link some of the variables with specific context information. Moreover, "Case-studies" also involve several decision-making procedures corresponding to specific organizations and associations (ministries, scientific associations, etc) at different scales (local, municipal, and national). With this mentioned information, context information about the case studies are present and stored in the semantic repository.

According to the serious game, the "Case-studies" are formed by learning goals (game objectives in the case study) and policy goals (decision-making goals that will be reached). Moreover, the policy goals are specified though the called policy objectives that represent the assessment methods to validate it. The policy goals are strictly dependent from the nexus. That means, the policy goals affects nexus variables and also, other nexus components by the decisions tacked under the consolidation of one of the objectives. The instrument to reach the objectives and goals are called "Policies" (e.g Policy Cards in the game). The policies represent the specific actions to be performed under a specific scenario through the modification of specific nexus variables. Therefore, "Case-studies" and "Regions" are represented by nexus variables that continuously are modified through the application of "Policies" under different scenarios.

In the ontology, the nexus variables are represented by "Measurements" to align the information representation to SAREF (and specifically to SAREF4WATR extension). Following the measurement pattern defined in such ontology, each measurement is related to a property (type of variable). These properties are subdivided into water, land use, climate, economic, food and energy properties that represent the variables at different scales and nature.

All the specific information about the variables, case studies and policies are stored in the so-called Semantic Repository. The semantic repository mainly is formed by a triple-store (JENA), a backend service as a REST API to manage the information to be stored in the triple store and, a front-end to explore the data intuitively, making them accessible for the community:

https://seriousgame.sim4nexus.eu/semanticRepository/

Finally, some existing ontologies, related to the Nexus, have been analysed to be involved in the SIM4NEXUS context:

- WatERP ontology, which reflects the water manager's expertise to manage water supply and demand. The novelty of WatERP ontology lies on including man interactions with the natural paths as a mechanism to understand how affect into the water resources management with the objective to match supply with demand, these interactions could range from infrastructures to management decisions.
- WEFNexus ontology, which concerns Water, Energy and Food derived by the European Directives: Article 2 of EU Directive 98/83/EC that defines the water intended for human consumption; Article 2 of EU Directive 2003/30/EC that defines bio-fuels; Article 2 of EU Regulation 178/2002/EC that defines food.
- SAREF ontology which is being converted in a standard to interrelate different domain information. This ontology provides standard patterns to represent measurements interrelated to geographical areas.

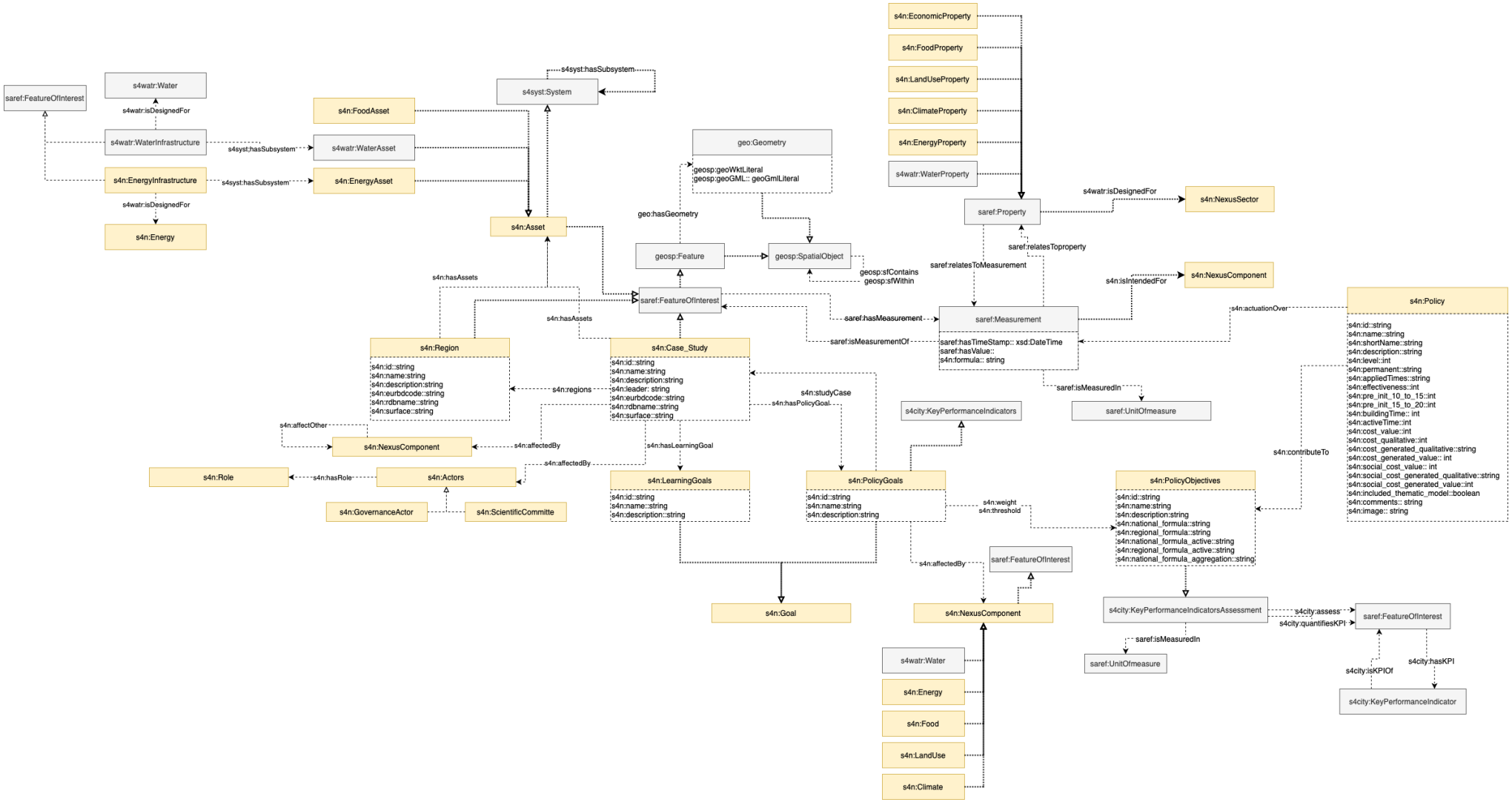

Figure 1. Summary of final version of SIM4NEXUS ontology

3 Metadata and other data sources

Several datasets are used as input for the System Dynamics Models, basically coming from the thematic models. The SIM4NEXUS repository (see D4.7) provides cloud storage and file sharing (among others), and stores the different necessary datasets and other files to build the SDMs. These files can be divided in 4 main categories: i) Model Data; ii) Arbitrary Data; iii) Thematic Data; and iv) Climate Data.

3.1 Case study related data: model, arbitrary, thematic, and climate data

Data Collection aims to serve both internal and public use. For internal project-use a tree-like folder organization was created (Figure 2), supporting easy dataset identification and revisions.

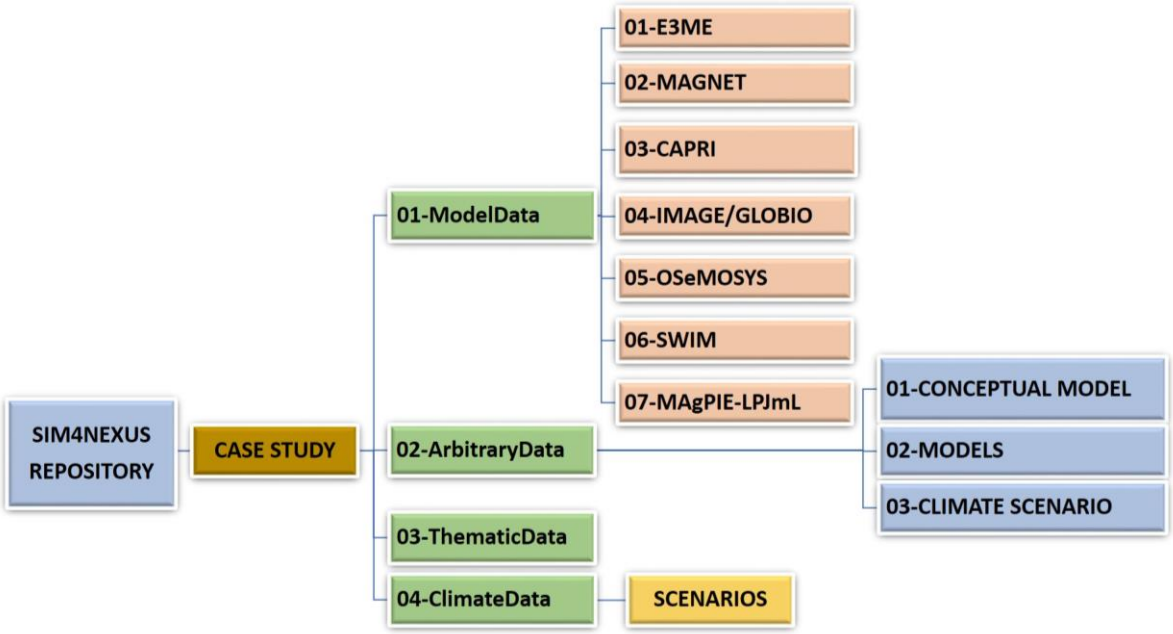

Figure 2 . Structure of the SIM4NEXUS repository

For each of the SIM4NEXUS cases, a specific folder was created in the repository. The folder contains the following sub-folders: 01-ModelData, 02-ArbitraryData, 03-ThematicData, 04-ClimateData (Figure 2).

- The 1st folder provides the outputs from the selected models applied in each case study. The file type of the outputs is either Microsoft Excel Open XML Spreadsheet (.XLSX File Extension) or Commas Separated Values files (.CSV File Extension).
- The sub-folder 02-ArbitraryData contains information about the relevant case study. These include the conceptual model of the case, the concept harmonization process etc.
- The 3rd sub-folder contains the thematic datasets of each case study along with its metadata.
- Finally, the sub-folder 04-ClimateData contains the climate datasets such as precipitation, relative humidity, long-wave downward solar radiation at the ground, long-wave downward solar radiation at the ground, daily maximum air temperature, daily minimum air temperature, and wind speed at 10m height (Figure 3). The datasets in this folder are of a generic file type with .DAT file extension. Each dataset of this type may contain data in binary or text format. A standardised name has been assigned to each dataset in the following format:

Country code_Earth System Model_Simulation Method_Period_Time frequency.dat

The value depends on the variable coded in the individual
file name. These are the standardised variable acronyms
used in climatology:

pr = Precipitation, originally given as m/s or mm/s, converted to mm/d
rhs = Relative humidity in %, 2 m above ground
rlds = Long-wave downward solar radiation at the ground in W/m²
rsds = Short-wave downward solar radiation at the ground in W/m²
tas = Average air temperature 2 m above ground, usually given in K, but converted to °C
tasmax = Daily maximum air temperature
tasmin = Daily minimum air temperature
wind = Wind speed at 10 m height, given in m/s

Figure 3 . Example of metadata information in each subfolder

In this way, all the necessary elements of each dataset such as the way that each dataset has been produced (i.e. model and simulation methods), the addressed area, and the duration and time-frequency, are provided. Each subfolder in the file hosting service contains files with descriptive information about the available datasets.

These datasets are useful for the specific case studies' System Dynamic Models generation. The baselines of the case studies are being developed and may be of interest for the scientific community. In this regard, SIM4NEXUS project will make these baselines available under Open Access. For any other Dataset (scenarios) it has been decided to make available publicly only with the permission of the data providers due to Intellectual Property Rights.

The thematic models that are being applied in each case study are listed below (Table 1)

Table 1. Thematic models / case study

	Case study	Thematic models applied/to be applied
1	Sardinia	E3ME, CAPRI
2	Greece	E3ME, MAGNET, CAPRI, GLOBIO
3	Andalusia	E3ME, MAGNET, CAPRI
4	UK	E3ME, CAPRI
5	Sweden	MAGNET, CAPRI, GLOBIO
6	Netherlands	E3ME, MAGNET, CAPRI
7	Azerbaijan	E3ME, MAGNET, CAPRI, OSEMOSYS
8	Latvia	E3ME, MAGNET, CAPRI
9	Germany - FRANCE	E3ME, CAPRI, SWIM
10	Eastern Germany, Czech Republic and Slovakia	CAPRI, SWIM
11	Europe	E3ME, MAGNET, CAPRI, IMAGE, MAGPIE
12	Global	E3ME, MAGNET, CAPRI, IMAGE, OSeMOSYS, MAGPIE

4 Architecture and data flow between modules

This section describes the SIM4NEXUS Serious Game architecture and how the different modules are interconnected to exchange data. The S4N SG platform is composed of four main elements (Figure 4): i) the Graphic User Interface (GUI), ii) the Knowledge Elicitation Engine (KEE), iii) the S4N Database and iv) the System Dynamic Models Engine (SDM Engine).

4.1 SIM4NEXUS Architecture

The GUI is the visual part of the tool and aims to create a realistic virtual environment where the players can interact with the proposed Case Studies and learn about the complex connections between the nexus elements and the impact of applying different policies.

The KEE is the core of the SG. It acts as a central connector between the other SG components and implements all the Game logic based on the outputs from other WPs such as the SDMs (WP3) or the policy definitions (WP2).

All SIM4NEXUS data, either generated during the project, such as the Learning Goals (T4.1) or the policies (WP2), or by the players during the execution of the Game, are stored in the SIM4NEXUS database. It is divided into two components: i) the Semantic Repository (SR) (D4.4), introduced in section 2, and a relational database. Depending on the source, type and utility of the data, it will be stored in one of these two databases.

Finally, the SDM Engine, a specific key interface which has two main functionalities. First, it is in charge of integrating the SDMs (provided by WP3) to the KEE and, second, it manages their execution to simulate the different Game turns.

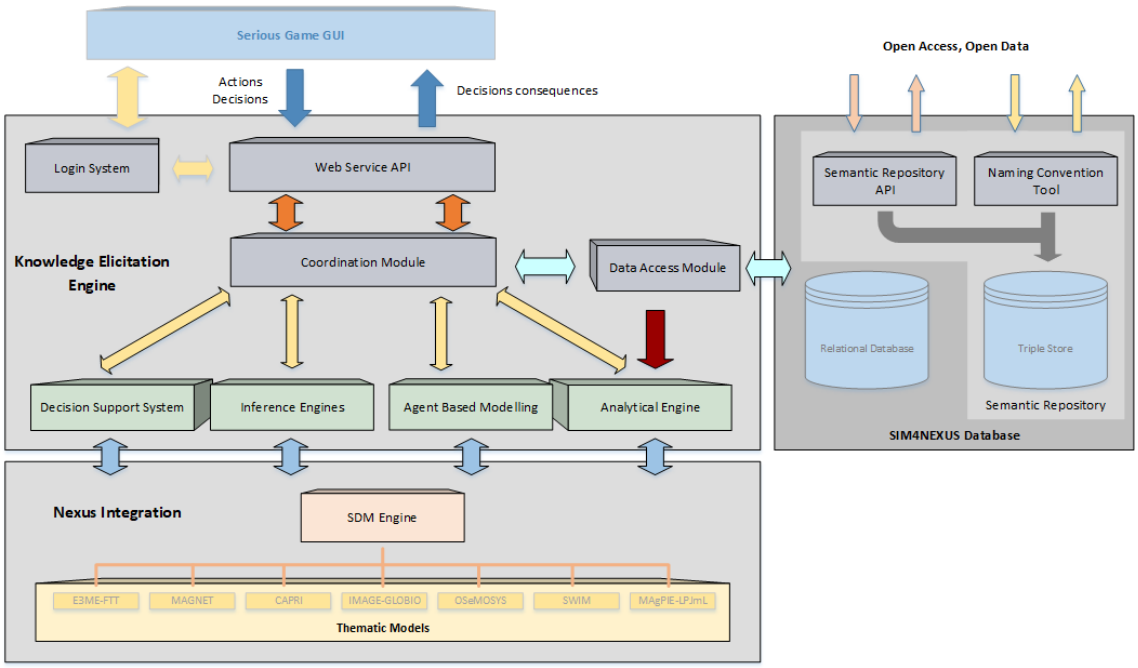

Figure 4. Serious Game tool Architecture.

4.2 Data flow between modules

The KEE (and the full platform) is managed by the Coordination Module, and it also includes the Web Service API, the Data Access Module and the SDM Engine, which are the main modules in charge of the communication between the S4N SG blocs and the different data flows.

4.2.1 Web Service API

The Web Service API provides the communication between the SG UI and the KEE, dealing with all the requests and responses. Its main processes (the key processes of the Game flow) are accessible through a Service Oriented Architecture (SOA), using XML, which implements OGC® Standards [1] for the information exchange. A RESTful approach has been implemented to facilitate the data exchange in secondary Game processes, such as the gathering of the initial case study information shown in the main pages of the Game.

Details of the available web service endpoints and routes are listed in the following sections.

[1] https://www.opengeospatial.org/standards/owc

Table 2: Case study route system analysis.

Location	/kee/case_
Methods	GET, POST
Headers	-
Body	{}
Response	Message

	[200, OK]:
	{
	“policy_id1”: {
	“case_study”: “2”,
	“nexus_sector”: “Water”,
	“name”: “Water savings in the household/commercial …”,
	“type”: “@CaseStudy”,
	“leader”: “Maria Papadopoulou”,
	“summary”: “Water saving in households by establishing …”,
	“affected_nexus_comp”: {“Climate”: {“Food”}, “Water”: {…}, …},
	“learning_goals”: [LearningGoal1, LearningGoal2, …],
	“decision_making_actors”: {“National energy agency”: False, …},
	“policy_goals”: {“Food security”: False, …},
	“indicators”: {“Population”: True, …},
	“eurbdcode”: “ZXY1”,
	“rbdname”: “ZXY1”,
	“surface”: 123213,
	“coordinates”: {},
	“regions”: {1: Region, 2: Region, …}
	}
	}

	Cookies
	-

Description	Every Case Study represents a study area selected for the Sim4Nexus SeriousGame project. Each new area will be treated in a unique way and with its respective values. Case Study parameters are what makes them different from one each other. Operations which can be performed over the location of this system can be 'GET' or 'POST' requests. Thanks to the 'GET' request we can obtain the different Case Studies allocated in the Triple Score DB. Meanwhile, the 'POST' operation allows us to introduce more Case Studies to the already existent set of them.

LEARNING GOALS SYSTEM

Table 3: Learning Goals route system analysis.

Location	/kee/learning_goals
Methods	GET, POST
Headers	-
Body	{}
Response	Message

	[200, OK]:
	{
	"id1": {
	"description": "You will learn how national policies in the domains of water management, renewable power production, and land use affect each other and result in changes in food production, tourism, greenhouse gas emissions, and quality and quantity of water resources."
	},
	"id2": {...},
	"id3": {...}
	}

	Cookies
	-
Description	Learning Goals system provides all the Learning Goals available in the Semantic Repository and defined in T4.1 Learning Goals definition. Method 'POST' is also allowed but it will require to declare a JSON containing one or more LearningGoals in order to be stored as new ones.

POLICY CARDS SYSTEM

Table 4: Policy Cards route system analysis.

Location	/kee/policy_cards
Methods	GET, POST
Headers	-
Body	{}
Response	Message

[200, OK]:

{

“policy_id1”: {

“case_study”: “2”,

“nexus_sector”: “Water”,

“name”: “Water savings in the household/commercial …”,

“short_name”: “Water savings at homes/hotels”,

“description”: “Water saving in households by establishing …”,

“level”: 1,

“permanent”: “Yes ”,

“applied_times”: “Multiple”,

“effectiveness”: 0,

“pre_init_10_to_15”: 0,

“pre_init_15_to_20”: 0.0,

“building_time”: 5,

“active_time”: 30,

“cost_qualitative”: “Medium”,

“cost_value”: 60,

“cost_generated_qualitative”: “Medium”,

“social_cost_qualitative”: “Medium”,

“social_cost_value”: 50,

“social_cost_generated_qualitative”: “Medium positive”,

“social_cost_generated_value”: 90,

“included_in_thematic_model”: “No”,

“model_input_translation”: “Decrease of water demand by the …”,

	"comments": "Column H: From 2020 - Until 2050", "sub_class": "", "image": "WaterSavingSmartTaps.png" }, "policy_id2": {...} }
	Cookies
	-
Description	Policy Cards generic system provides the entire set of data which the KEE manages. PolicyCard endpoint also allows to upload PolicyCards by performing a 'POST' request into it. That request must have a LearningGoals looking like JSON. The SeriousGame uses Policy Cards in order to modify the game state while adding progression and value to a game session.

POLICY GOALS SYSTEM

Table 5: Policy Goals route system analysis.

Location	/kee/policy_goals
Methods	GET, POST
Headers	-
Body	{}
Response	Message

	[200, OK]:
	{
	"id1": {
	"case_study": "2",
	"nexus_sector": "Water",

	"name": "Water savings in the household/commercial …",
	"description": "Water saving in households by establishing …",
	"thresholds": {
	"low": 0.33,
	"medium": 0.66,
	"high": 1.0
	},
	"weights": {
	"O2": 0.5,
	"O3": 0.5
	}
	},
	"id2": {…}
	}

	Cookies
	-
Description	Policy Goals system allow the upload and download of Policy Goals. Policy Goals contain all the information related to the different Goals of a given Study Case. Goals are the last point of the SeriousGame iterations.

POLICY OBJECTIVES SYSTEM

Table 6: Policy Objectives route system analysis.

Location	/kee/policy_objectives
Methods	GET, POST
Headers	-
Body	{}
Response	Message

	[200, OK]:
	{
	“id1”: {
	“case_study”: “2”,
	“name”: “Water savings in the household/commercial …”,
	“description”: “Water saving in households by establishing …”,
	“national_formula”: “initial = get_first_year(‘National_Demand_SW…”,
	“regional_formula”: “initial = get_first_year(‘RBD_W_GRXX_Demand…”
	},
	“id2”: {…}
	}

	Cookies
	-
Description	Policy Objectives represent a partial goal which needs to be reached. Once the game evolution reaches one of these Policy Objectives computation can be done in order to know the game performance for a specific game session.

Authentication Module

LOGIN SYSTEM

The Login system is part of the Web Service API and provides authentication and authorization to the Game, validating all the KEE incoming communications. It is based on Json Web Tokens [2](JWT), which are an open, industry-standard RFC 7519[3] method for representing claims securely between two parties.

Table 7: Login route system analysis.

Location	/kee/login
Methods	POST
Headers	Content-Type: application/json

Body	{
	"email": "johndoe@example.com",
	"password": "guestw"
	}

Response	Message
	[200, OK]: Successfully logged in [401, Unauthorized]: Invalid login request. Tip: content: application/json, fields: email, password
	Cookies
	[200, OK]: access_token_cookie: eyJ0eXAiOiNiJ9. […] B3PgPqdbZtSoBCyBgI refresh_token_cookie: eyJ0eXAiOiJKVOiJIUzI1NiJ […] H40Lz25Tcu2KzzOg
Description	Kee version of /login ('/kee/login') only allows 'POST' method to be used. The request body must contain the User email and password. As a response, it will

[2] https://jwt.io/

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

	provide the authentication credentials inside response Cookies which are located in response headers. There will be two tokens (“acces_token_cookie” and “refresh_token_cookie”). These two parameters can be used in order to know who is doing the request to the back-end. From now in advance, users will use these parameters in their request headers in order to verify their session and identity. This functionality can be tested using ‘/whoami’ endpoint.

LOGOUT SYSTEM

Table 8: Logout route system analysis.

Location	/kee/logout
Methods	GET
Headers	Cookie: "access_token_cookie=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9. [...] B3PgPqdbZtSoBCyBgI; refresh_token_cookie=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ [...] H40Lz25Tcu2KzzOg; session=.eJwNyNEKgyAUANBfudznGFnTXG [...] C39dYBNaWd0bwGW6sqCuZJahNSA "
Body	{}
Response	Message
	[200, OK]: Successfully logged out
	Cookies
	-

Description	By performing a GET over the logout endpoint the User can sign out from their session and clean the cookies. The browser client will remain as no one has signed in before.

EMAIL SYSTEM

Table 9: Email route system analysis.

Location	/kee/changeEmail
Methods	POST
Headers	Content-Type: application/json Cookie: "access_token_cookie=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9. [...] B3PgPqdbZtSoBCyBgI; refresh_token_cookie=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ [...] H40Lz25Tcu2KzzOg; session=.eJwNyNEKgyAUANBfudznGFnTXG [...] C39dYBNaWd0bwGW6sqCuZJahNSA "

Body	{
	"email": "johnDoe@mail.com"
	}

Response	Message

	[200, OK]
	{
	"email": "johnDoe@mail.com",
	"username": "johnDoe",
	"role_id": 1,
	"confirmed": False,
	"name": "John",
	"location": "Sample City"
	}
	
	[400, Bad Request]

	Bad request, missing "email" field
	Cookies
	-
Description	Thanks to the “ChangeEmail” endpoint, the client is able to change the email of an authenticated User. As a response, it will return the User instance updated with the new email on it.

QUESTIONS SYSTEM

Table 10: Questions route system analysis.

Location	/kee/questions?tag=<tag:str>&caseStudy=<case_study:int>	
Params	Tag: - PRE_GAME - MID_GAME - POST_GAME	CaseStudy: - 2 (Greece) - 3 (Azerbaijan)
Methods	GET	
Headers	Content-Type: application/json	

Body	{
	"title": "Sim4Nexus WS answer schema",
	"type": "object",
	"properties": {
	"question_id": {
	"type": "integer",
	"minimum": 0
	},
	"answer": {
	"type": "string"
	},
	"extra_question_answer": {
	"type": "string"
	},
	"user_id": {
	"type": "string"
	},
	"session_id": {
	"type": "string"
	}
	}
	}	

Response	Message	
	[200, OK] [400, Bad Request]	
	Cookies	
	-	
Description	Questionnaires KEE endpoint provides the questions which relate the case study to the situation in which the user finds himself respect the Game progression.	

ANSWERS SYSTEM

Table 11: Answers route system analysis.

Location	/kee/answers?multiple=<multiple:Boolean>
Methods	POST
Headers	-
Body	{}
Response	Message

	[200, OK]:
	{
	"CaseStudy": “Greece”,
	"Description": “Description”,
	"Id": 3,
	"Questions": [
	{
	"Id": 27,
	"Options": [
	{
	"Id": 21,
	"Value": "Strongly disagree"
	}
	]
	}
	]
	}

	Cookies
	-
Description	Answers KEE endpoint allows the Sim4Nexus SeriousGame to upload the answers related to a question formulary previously downloaded, which can be

	done thanks to the "Questions" endpoint. The “multiple” parameter lets the client answer the questions one by one or all at the same time.

SG Initialization step

This KEE WS endpoint manages the initialization of an S4N SG session, it is accessed by the UI to obtain the case study, initialize the game (year 2020), initial values for each parameter, load case study data, policy cards, policy objectives, policy goals, nexus health, etc.

Table 12: Game Initialization route system analysis.

Location	http://127.0.0.1:5000/kee/wps?service=<service:str>&request=<request:int>&identifier=<identifier:str>&datainputs=<input:str>	
Params	Service: WPS	
	Request: Execute	
	Version: 1.0.0	
	Identifier: sim4nexus_initialization_step	

	Datainputs:
	input={
	SessionID: 9aba-e880aff102,
	ScenarioId:6,
	PlayerID:0,
	Language:en-US,
	SessionDateTime:1/1/2020 00:00:00,
	ReceiveSDMOutputs:1
	}
		- SessionID: str
		- ScenarioId: int
		- **PlayerID**: str (by default is 0, guest)
		- Language: str
		- **SessionDateTime**: Datetime dd/MM/yyyy_hh:mm:ss
		- **ReceiveSDMOutputs**: Boolean (1 yes, 0 no)

Methods	GET	
Headers	Content-Type: application/json	
Body	{}	

Response	Message	

	[200, OK]:
	{
	"Budget": 1000,
	"CaseStudyName": Object <CaseStudy>,
	"CurrentTurnDateTime": "01/01/2020 00:00:00",
	"Interventions": [
	Object <PolicyCard>
	],
	"InterventionsHistory": {},
	"InterventionsHistoryOrder": {},
	"Language": "en-Gb ",
	"NexusHealth": {
	<nexus>_formula: "",
	<nexus>_desc: ""
	},
	"NexusHealthScore": {
	<nexus>: 0
	},
	"PlayerID": 0,
	"PolicyGoals": [
	Object <PolicyGoal>
	],
	"PolicyGoalsScore": {
	"pg_id": 0
	},
	"PolicyObjectives": [
	Object <PolicyObjective>
	],
	"Regions": [
	Object <Region>
		- **PolicyCard** seen in D4.5, 2.1.1. Game Module, Policy Cards System
		
		- **<nexus>** references the components:
		o Water
		o Energy
		o Climate
		o Food
		o Forest
		
		- **PolicyGoal** seen in D4.5, 2.1.1. Game Module, Policy Goals System
		
		- **PolicyObjective** seen in D4.5, 2.1.1. Game Module, Policy Objectives System

<table>
<tr><td></td><td>
<pre>
  ],
  "SDMOutputs2020": [[]],
  "SDMVars": [
    "Climate__GHGb",
    ...
  ],
  "SessionDateTime": "22/4/2020 13:21:53",
  "SessionID": "4506-fc4e-a30050d3342e",
  "SocialAcceptance": 1000,
  "StateVars": {
    "Water__wd_Irr": [],
    ...
  },
  "Stocks": {
    "Climate__GHGb": [],
    ...
  },
  "TotalScore": 1.5410385470252457
}
</pre>
</td><td></td></tr>
<tr><td></td><td>Cookies</td><td></td></tr>
<tr><td></td><td>lang: <en,es,aze,...>
gameId: A000x001
access_token_cookie: eyJ0eXAiOiNiJ9. [...] B3PgPqdbZtSoBCyBgI</td><td></td></tr>
<tr><td>Description</td><td colspan="2">By sending this GET request, the KEE starts the Game initialization process. This process consists in preparing the CaseStudy context in order to start playing. The main action is to initialize and create the year 2020 context, this one involves the process of loading all the PolicyGoals, PolicyObjectives, PolicyCards, variables and regions for a specific CaseStudy given as a dictionary element “ScenarioId” inside URL parameter “datainputs”. Once the context is loaded, it is sent to the User Interface (object Response) to interact with it.

Referring to cookies:

- lang: It is used to keep tracking of the language in which the user wants to see the interface.
- gameId: Different param as the “sessionId”. A SessionId involves a user and a game in a specific time. Meanwhile, a gameId is the reference of a game in the database.
- access_token_cookie: It is used to retrieve a User that is logged into the application and attach the game to it to track their evolution.</td></tr>
</table>

SG Simulation step

This KEE WS endpoint manages the simulation of the S4N SG session turn, it is accessed by the UI to run each time step based on the current Game status and provide the next one.

Table 13: Game Simulation route system analysis.

<table>
<tr><td>Location</td><td colspan="2">http://127.0.0.1:5000/kee/wps?service=<service:str>&request=<request:int>&identifier=<identifier:str>&datainputs=<input:str></td></tr>
<tr><td rowspan="5">Params</td><td colspan="2">Service:
WPS</td></tr>
<tr><td colspan="2">Request:
Execute</td></tr>
<tr><td colspan="2">Version:
1.0.0</td></tr>
<tr><td colspan="2">Identifier:
sim4nexus_initialization_step</td></tr>
<tr><td>Datainputs:
input={
SessionID:caf7-4588-9aba-eaff102,
ScenarioId: 6,
PlayerID: 0,
Language: en-US,
SessionDateTime:1/1/2020 00:00:00,
ReceiveSDMOutputs: 1,
CurrentTurnNumber: 1,
PolicyGoals: [],
Budget: 1000,
SocialAcceptance: 1000,
Interventions: {
New: [],
History: {},
Order:[]
},
Stocks: {},
StateVars: {}
}</td><td>- SessionID: str
- ScenarioId: int
- PlayerID: str (by default is 0, guest)
- Language: str
- SessionDateTime: Datetime dd/MM/yyyy_hh:mm:ss
- ReceiveSDMOutputs: Boolean (1 yes, 0 no)
- CurrentTurnNumber: int
- PolicyGoals: [object<PolicyGoal>]
- Budget: int
- SocialAcceptance: int
- Interventions : Stores objects <PolicyCard>. User can only add PolicyCards into “New”. KEE will handle the rest.
- Stocks: Key-Value structure, stores the evolution of Stock variables.</td></tr>
</table>

<table>
<tr><td></td><td></td><td>- StateVars: Key-Value structure, stores the evolution of StateVars variables.</td></tr>
<tr><td>Methods</td><td colspan="2">GET</td></tr>
<tr><td>Headers</td><td colspan="2">Content-Type: application/json</td></tr>
<tr><td>Body</td><td colspan="2">{}</td></tr>
<tr><td rowspan="2">Response</td><td colspan="2">Message</td></tr>
<tr><td>[200, OK]:
{
Budget: 10028000
CurrentTurnDateTime: "01/01/2030 00:00:00"
CurrentTurnNumber: 2
InterventionsHistory: {
"PolicyCardID": {
"National": [<year:int>]
}
},
InterventionsHistoryOrder: [
{
"Id": "2",
"Region": "National",
"Year": 2030
}
],
"NexusHealthScore": {
<nexus>: 0
},
"PolicyGoalsScore": {
"pg_id": 0
},
"Regions": {
Object <Region>
},
"SDMOutputs2020": [[]],
SessionDateTime: "22/4/2020 14:46:40",</td><td>- <nexus> references the components:
o Water
o Energy
o Climate
o Food
o Forest</td></tr>
</table>

	SessionID: "45c2-4900be8f3b39",
	SocialAcceptance: 10028000,
	"StateVars": {
	"Water__wd_Irr": [],
	...
	},
	"Stocks": {
	"Climate__GHGb": [],
	...
	},
	TotalScore: 1.5541678719103018
	}	

	Cookies	
	lang: <en,es,aze,...> gameId: A000x001 access_token_cookie: eyJ0eXAiOiNiJ9. [...] B3PgPqdbZtSoBCyBgI	
Description	As can be seen, the Response for the Game Simulation System is quite similar to the one seen previously for the Game Initialization System. That is because both systems share a trivial similarity. Game Simulation System can be defined as a Game Initialization System which is capable of loading the CaseStudy context only taking the data sent by the User Interface and their local files. Following this definition, this system will take the modified context by the User, will interpret it, simulate the variations involved in the next 5 years, and return a new CaseStudy context (object Response) to the User interface. As the game was already initialized, this Response will not be as long as the one seen in the Game Initialization System. Referring to cookies: - **lang**: It is used to keep tracking of the language in which the user wants to see the interface. - **gameId**: Different param as the "sessionId". A SessionId involves a user and a game in a specific time. Meanwhile, a gameId is the reference of a game in the database. - access_token_cookie: It is used to retrieve a User that is logged into the application and attach the game to it to track their evolution.	

4.2.2 Data Access Module

The Serious Game data are persisted in the S4N databases, defined and implemented under T4.3 ‘Setting-up the project database and metadata ontology’, which are deployed along with the KEE in the S4N servers.

There are two kinds of databases which store the data based on their type, source and utility.

All that information that has to be public and open data is available in the Semantic Repository (SR) and can be accessed via the KEE, the Semantic Repository API or the Naming Convention Tool. For example, the Case Study information, Learning Goals, Policy Cards and Policy Goals are stored in the SR.

On the other hand, other data that don’t fill these requirements is persisted in a relational database. Data related to the user's management or results from the analytical engine are stored here.

The Data Access Module acts as a bridge between the KEE and the S4N databases to simplify and isolate the data exchange between them.

Apart from that, it is initially used to fill the databases with the raw information provided by the different Case Studies.

4.2.2.1 KEE Data Access Interface

The following UML diagram (Figure 5) introduces how the data models work referring to the Sim4Nexus Relational Database.

Starting from the core class of the diagram we get the User class. That class enables the action among the rest of the database models.

Regarding the first relation, Users can interact with questionnaires not only because they provide information about the Questions included among their sections but also because these Questions contain certain Options. These elements will help the User to perform an Answer related to this Question. The generation of these Answers will be through a formulary. Once this formulary is sent or skipped, the User will be inside the general SeriousGame flux.

Introducing SeriousGame flux, the action starts once the User has selected the game region and the game mode (MainPage and GamePage, respectively). At this point, the User is going to fall into the creation of a new Game instance. Regarding this instance, exists and is ready, Users start interacting with Policies as they think it would be the best for their region state. The selection of Policies leaf into UserDecisions over the Game timeline. Every single one of these UserDecisions interacts with the general Game progression by generating a GameStep. The accumulation of GameSteps will carry the game into their ending state (the year 2050).

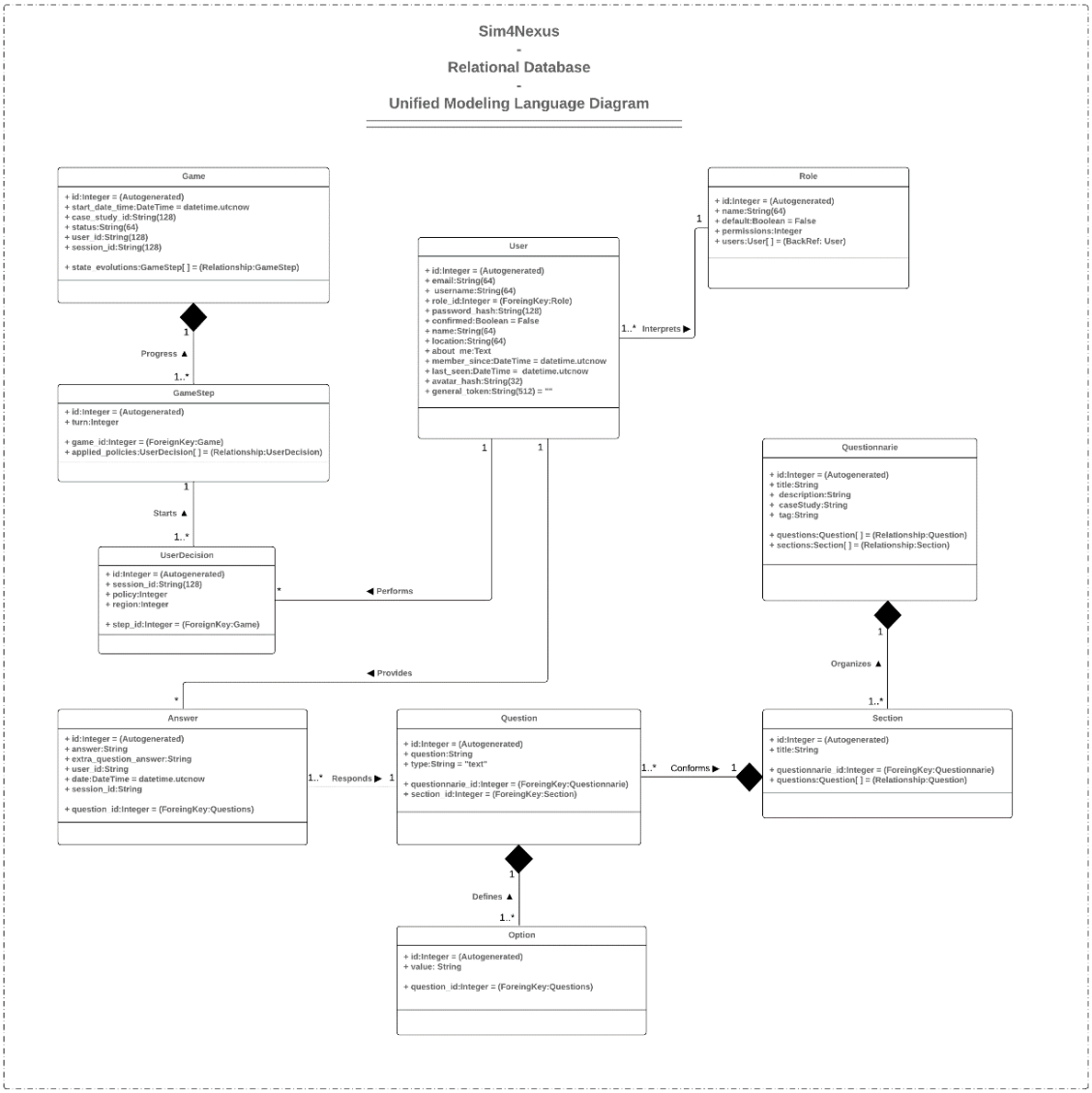

Figure 5. Sim4Nexus Relational Database Unified Modelling Language diagram (UML)

Table 12. Database general routes analysis

Method	Route	DB Model implicated	Data Source
POST	/kee/register	User	Relational DB
GET	/kee/confirm/<token>	User (confirmed)	Relational DB
GET	/kee/login	User (session)	Relational DB
GET	/kee/logout	User (session)	Relational DB
POST	/kee/changeEmail	User (email)	Relational DB
GET	/kee/questions	Question	Relational DB
POST	/kee/answers	Answers	Relational DB
GET/POST	/kee/policy_cards	PolicyCard	Semantic Repository
GET/POST	/kee/policy_goals	PolicyGoal	Semantic Repository
GET/POST	/kee/policy_objectives	PolicyObjective	Semantic Repository
GET/POST	/kee/case_studies	CaseStudy	Semantic Repository
GET/POST	/kee/learning_goals	LearningGoal	Semantic Repository

4.2.2.2 Interfacing the KEE through the Semantic Repository

The Knowledge Elicitation Engine is also connected to the Semantic Repository. The connection between these modules is performed through the Data Access Module and the SR REST API, which offers to create and consume information.

The root path of the REST API permit to create entities (instances of the ontology) and query them in order to retrieve required information from a specific class or properties related to a class (see Table 13).

Table 13. Class REST API

Http Route	Operation	Description	Request	Response	Errors
/{:entity}	GET	Obtain all information regarding an entity	--	JSONLD	200, 401
/	POST	Create an entity into the semantic repository	JSONLD	JSONLD	200, 401
/{:entity}	PUT	Update the information of certain entity in the semantic repository	JSONLD	--	200, 401
/{:entity}	DELETE	Remove an entity for the semantic repository backend.	--	--	204, 403
/class/all	GET	Get all classes from the semantic Repository	--	JSONLD	200, 401
/class/facet	GET	Get all facets from the repository	--	JSONLD	200, 401
/class/{:class}	GET	Get all instances for a certain class	--	JSONLD	200, 401
/class/facet/{:facet}	GET	Get the instances list of a given facet (property)	--	JSONLD	200, 401
/class/{:class}/{:facet}	GET	Get all facets from the instances of a class	--	JSONLD	200, 401

In reference to the data model used to represent the information, the instances have an open data model and for this reason, the REST API cannot bind the information to a specific data model. In case of the ontology classes and facets, the followed data model is represented in the following tables.

Table 14. Ontology Class Data Model

Ontology Class Data Model		
Variable Name	Type	Description

Id	String	Id of the entity that could correspond to an URI or curie
Uri	String	Uri of the entity
Label	String	Specific name of the class
instancesCount	Integer	Number of instances the class have
curie	string	Curie of the entity

Table 15. Ontology Facets Data Model

Ontology Class Data Model		
Variable Name	Type	Description
Id	String	Id of the property that could correspond to an URI or curie
Uri	String	URI of the property
Label	String	Name of the property
Range	String	Range values of the property that could be object (if corresponds to an object property) or data type (if corresponds to a data property)
Context	String	Context of the property (URI root)
instances	Integer	Number of instances that uses the property
Timeused	Integer	Number of uses of the property

The error codes that have been used in the REST API is showed in Table 16.

Table 16. Error codes used in the Semantic Repository

REST API Error Codes		
Error Code	Name	Description
200	OK	The request has succeeded. The information returned is the expected ones according to the API.
204	No Content	The server has fulfilled the request but does not need to return an entity-body.

401	Unauthorised	The request requires user authentication.
403	Forbidden	The request was a legal request, but the server is refusing to respond to it.

4.2.3 SDM Engine

This section of the document details how the SDMs, implemented in WP3, are mixed with the policies, implemented in WP2, to enrich their functionalities and how are finally integrated to the KEE to simulate the game flow through the SDM Engine.

In WP3, the SDMs are built using a complex and specific modelling software, called Stella, which is able to represent the models in a readable format where, basically, all data and equations are listed in a logic way.

Due to its format, it cannot be directly executed by the KEE, thus it has to be previously translated to Python and integrated to the game architecture through the SDM Engine, which will manage their translation and execution.

In the translation process, the engine takes as inputs the base SDM (implemented in Stella software), the Policy Cards (stored in the SR) and other Case Study specific metadata and builds a complex logic structure for each CS that enables the corresponding simulation of each Game turn, including the selected policies (user's decisions), which modify the SDM behaviour and consequently the Game status.

This process is managed by the 'Conversion script' which define and automatic flow that goes through different steps, each one in charge of a specific task in the translation logic:

1. Variable names are translated to Python nomenclature.
2. Constant data is extracted and loaded into a specific and isolated data structure.
3. Time series data is extracted and loaded into a specific and isolated data structure.
4. Equations are extracted and loaded into a specific and isolated structure.
5. Initial stocks are extracted and loaded into a specific and isolated data structure.
6. Based on the previous data structures, the Python SDM is defined.
7. Policies are included in the SDM code.
8. The Python SDM is executed from the first year till 2050 to check its correctness.
    a. The outputs are validated monthly against a validation data set.
9. The Python SDM is executed in chunks of 5 years from 2020 till 2050 to check its correctness.
    a. The outputs are validated monthly against a validation data set.

Figure 6 shows the flow of steps 1 to 7 in relation to the conversion process. This process is carried out in two parts. The Baseline Python Model is the model that reflects that which was created in STELLA. This model is executed from the first year till 2050 whereby the Python model outputs are validated against the STELLA outputs. Once validated the Policy Builder process is implemented to incorporate the Policy Card information to ready the Python model for use within the SG. The model is then executed again though this time in 5-year intervals from it's initiation date to 2050 to and validated again against the STELLA model outputs. Finally, when this process is successfully finished, the Python SDM is added to the SDM Engine to support a new Case Study and any request of execution from the KEE is redirected to this environment.

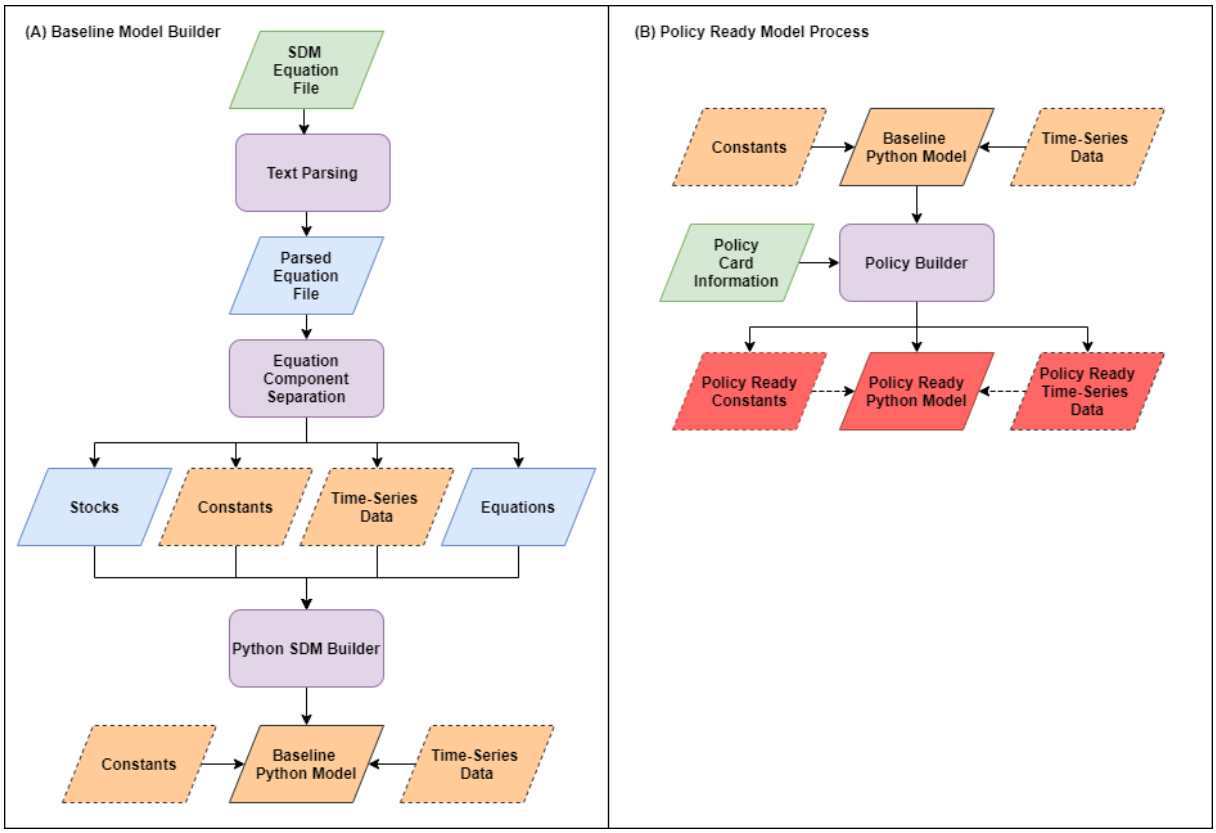

Figure 6. SDM conversion process flow chart

During a SG session, when the Game is initialized, the initialization step is executed and the initial stocks and policies are loaded to simulate (through the SDM Engine) the year 2020 (game starting point). During the rest of the SG session, the simulation step is used to compute the following turns, where the previous game status and the current user's decisions (selected policies) are sent through the UI request to the KEE and the SDM Engine uses them as inputs to obtain the next turn values, which are sent back to the UI to be presented to the player.

5 Data Management overall- Conclusions

For more information regarding the Data Management, please refer to the latest version of D4.7 Data Management Report, a final document which aim is to consider the many aspects of data management, data and metadata generation, data preservation- maintenance- and analysis, whilst ensuring that data is well managed at present and prepared for preservation in the future.

This Deliverable has tried to avoid duplications with regard to Deliverable 4.7. It focuses explicitly on Data Flow between WP3 and WP4, as well as Ontologies, reflecting the progress achieved until M48.