D4.2 – SECOND REPORT ON PILOT REQUIREMENTS

Grant Agreement	676547
Project Acronym	CoeGSS
Project Title	Centre of Excellence for Global Systems Science
Topic	EINFRA-5-2015
Project website	http://www.coegss-project.eu
Start Date of project	October 1, 2015
Duration	36 months
Deliverable due date	31.01.2017
Actual date of submission	31.01.2017
Dissemination level	Public
Nature	Report
Version	2 (after internal review)
Work Package	4
Lead beneficiary	GCF
Responsible scientist/administrator	Sarah Wolf
Contributor(s)	Margaret Edwards, Steffen Fürst, Andreas Geiges, Luca Rossi, Michele Tizzoni, Enrico Ubaldi
Internal reviewers	Michael Gienger, Sergiy Gogolenko, Cezar Ionescu
Keywords	Requirements, Synthetic information system, Data, Synthetic Population, Agent-based Model, Simulation Analysis and Visualisation
Total number of pages:	33

The CoeGSS ("Centre of Excellence for Global Systems Science") project is funded by the European Union. For more information on the project please see the website http:// http://coegss-project.eu/

Version History

	Name	**Partner**	**Date**
From	Sarah Wolf	GCF	
First Version	for internal review		January, 2017
Second Version	for submission		January, 2017
Reviewed by	Michael Gienger	HLRS	January, 2017
	Sergiy Gogolenko	HLRS	January, 2017
	Cezar Ionescu	Chalmers	January, 2017
Approved by	Coordinator	UP	January, 2017

Abstract

This deliverable is the second report on pilot requirements to the Centre of Excellence for Global Systems Science (CoeGSS), and hence acts as a second version of the living document begun with D4.1.

The three pilot studies – Health Habits, Green Growth, and Global Urbanisation – represent typical applications of Global Systems Science (GSS) and address the tobacco epidemics, the evolution of the global car market, and the two-way relations between transport infrastructure and real-estate pricing, respectively. They develop synthetic information systems (SISs), that is, computer simulation based tools to help explore and understand these global challenges. Main requirements for this work stated in D4.1 have by now been sharpened through the work on the pilots undertaken in the meantime. They concern a common language and training, data collection and pre-processing, synthetic populations, synthetic networks, a GSS-specific agent-based modelling framework, and analysis tools for a synthetic information system.

Interaction between GSS and HPC (high performance computing) communities in the project, in particular in dedicated working groups focusing on elements of a SIS, provide an ongoing exchange about GSS requirements to HPC. This document presents a snapshot of the pilots' requirements at this point in time. Where useful, pilots also look beyond their cases of study with a view on other GSS applications.

Table of Contents

1 About this document

This deliverable is the second report on pilot requirements to the Centre of Excellence for Global Systems Science, and hence acts as a second version of the living document begun with D4.1. Main requirements stated in the first version have by now been sharpened through the work on the pilots undertaken in the meantime. This document does not repeat text from D4.1, but details those points from the first version that were found to be the most pertinent requirements of the pilots. After a brief introduction (Section 2), it presents the pilots' requirements as concerning a common language and training (Section 3), data collection and pre-processing (Section 4), synthetic populations (Section 5), synthetic networks (Section 6), a GSS-specific agent-based modelling framework (Section 7), and analysis tools for a synthetic information system (Section 8). Then Section 9 concludes. Section 10 provides a glossary and abbreviations for easy reference.

2 Introduction

The three pilot studies of CoeGSS – Health Habits, Green Growth, and Global Urbanisation – develop synthetic information systems: computer simulation based tools to help explore and understand global systems. Adressing the cases of the tobacco epidemics, the evolution of the global car market, and the two-way relations between transport infrastructure and real-estate pricing, respectively, pilots are working on synthetic populations (SPs) and agent-based models (ABMs) for simulating and thus exploring the corresponding global systems. As a brief recap, Figure 1 summarises the elements of a SIS for GSS and the steps involved in its development.

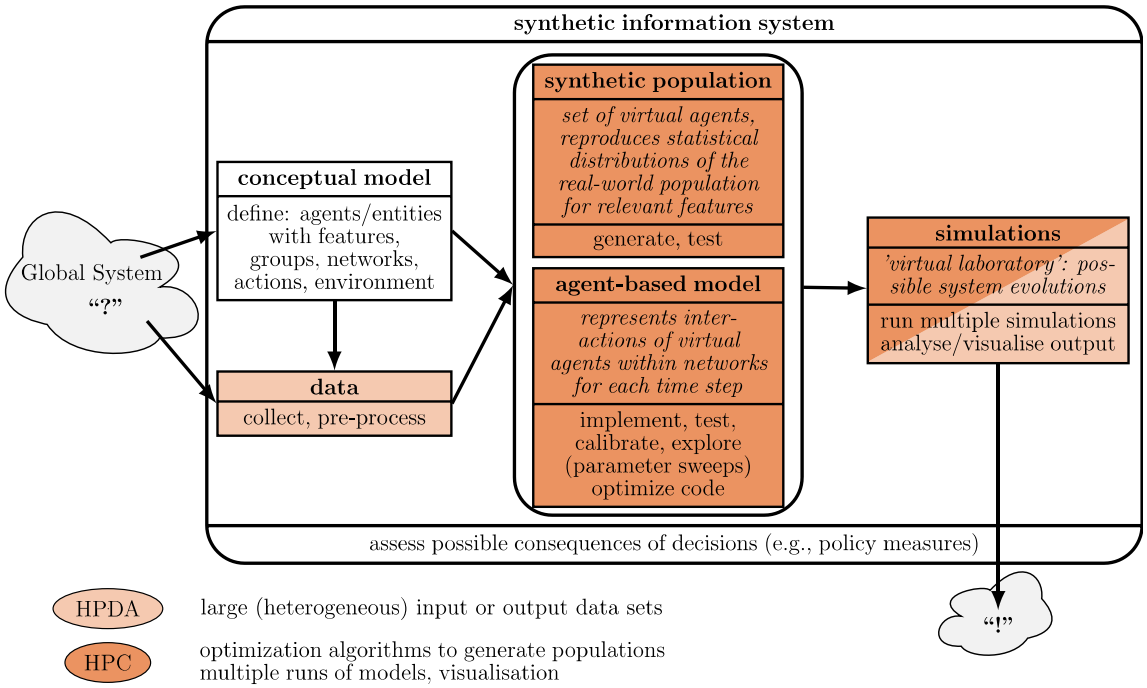

Figure 1: The structure of an HPC-SIS for GSS. Colours indicate where high perfomance computing and high performance data analytics (HPDA) play a role. Boxes show the name of the element, explain its role, if necessary, and list the tasks performed in relation to it.

Several remarks are in order before going into pilot requirements:

First, pilots state requirements specific to developing their pilot SIS. However, given that the development of an HPC-based framework to generate customized synthetic information systems for GSS applications shall be a key contribution of the Centre of Excellence for Global Systems Science, they also keep an eye open to broader GSS needs. Therefore, some requirements are phrased in more general terms with a view to other potential GSS applications.

Second, pilots cover both potential use cases of GSS modellers wanting to develop a new ABM for a SIS, or wanting to use an existing model (or more generally, a given software tool for implementing a model) in a SIS. In particular, the Health Habits and the Green Growth pilots

are developing new agent-based models and thus require a framework for implementing these in HPC-ready fashion, while the Global Urbanization pilot uses the leading partner's proprietary software, the CoSMo Simulation Suite (http://www.thecosmocompany.com/technology/).

Third, the development of a SIS is an iterative process, involving a first model specification and implementation that are then improved step by step based on analyses of the simulation output generated – usually from many simulation runs due to stochasticity of models that represents uncertainty in the modelled systems. For these improvements, each element in the SIS may need to be revisited. This iterative workflow is reflected in the requirements the pilots pose to the CoE: for example, the framework for ABMs just mentioned above should strike a balance between efficiency and generality. The most efficient possible parallelisation of a single given model is not a priority as long as this model is still subject to change in the iterative process.

Fourth, not all elements of the SIS displayed need to be equally important for all cases, nor are there always clear cuts between all elements. For example, interaction networks between agents will often play an important role. In some cases these are included in the information provided by a synthetic population, in other cases they are part of the ABM (for example, when agents randomly interact with other agents from a somehow defined neighbourhood).

The second to fourth point suggest to view the pilots' requirements as the need for a toolbox from which to choose the relevant pieces in each case, rather than pointing to a fixed workflow that is followed in all cases identically.

Fifth, it needs to be noted here that requirements for some of the tools needed are more elaborated than for others. This is simply due to the fact that pilot work had to start somewhere, and not all elements of the SIS have, as yet, been given equal consideration. In particular, having been a first focal point, requirements for the agent-based modelling framework are by now reasonably clear, while for other elements, such as synthetic populations or networks, helpful discussions are going on in the consortium, that shall enable the pilots to provide more precise requirements throughout the second project year.

Sixth and last, the list of requirements presented in this deliverable is quite long. Prioritization of these is undertaken in dialogues between pilot researchers and people working on tools and methods for GSS on HPC in dedicated working groups within CoeGSS.

3 Common language and training

While in D4.1, GSS and HPC were considered “as yet unrelated universes” (p. 6), in the first project year, interaction between these communities has taken place through which individuals from both communities have familiarised themselves with the “unknown universe” of the other community.

A basic shared vocabulary about synthetic information systems for GSS and the pilots’ workflow has been created over the first project year. Main definitions are collected in the glossary section. However, the **consolidation of a common language** is a remaining requirement. The project is organising an international conference on synthetic populations (https://icspconference.wordpress.com/) in February 2017 in Lucca. This 2-day conference, followed by a 1-day training workshop for consortium members with external experts shall help the consolidation process.

4 Data collection and pre-processing

High quality data sources are needed for building a SIS – in order to correctly generate a synthetic population, define an ABM, and calibrate it. Within a SIS, data is used as an input for SP generation and ABM, as well as for comparing with simulation output. Thus, pre-processing of data needs to make sure that the data is available in the forms (and formats) needed for the simulation and later analysis.

However, so far modellers mainly prepare their own data, thus data pre-processing workflows may differ in the applied tools and assured data quality. In each case, the specific data needed depend on the global system under study, however, there are data that are needed for describing many global systems. Thus, for example, geo-spatial data sets, information on populations, data about urban areas, or census data represents a common layer of information that many models will draw on. It therefore makes sense to reuse many of these data sets and make them available to modellers in the future.

In that way, CoeGSS might have the potential to create a common data pool and offer it, not only to its pilots, but also to future modellers. Since data acquisition and pre-processing is generally very time consuming, a repository of high quality GSS data can be an important part of a value chain of a sustainable centre of excellence. Furthermore, it would enable faster model prototyping and development at lower costs for its users. Ultimately, more available data can lead to more comprehensive models that will need to rely on HPC for simulations.

Section 4.1 specifies requirements for the collection of high quality data and how support tools can improve and speed up the process. Section 4.2 points out the current state of pre-processing for the examples of the three pilot studies. Since modellers will always be free to use any available tool for data processing, the majority of the requirements concern the usability of the tools. With tools and assistance software, CoeGSS has the potential to create a community of GSS modelers that share and exchange data in a common data pool.

4.1 Acquisition of GSS data and data quality

Currently, data is mosty collected manually by modellers. This means, one has to browse government agencies, statistical or research institutes, or, for example, national health services websites and collect data by manually starting the download. Then one needs to format the data, potentially aggregate different sources, track changes, append metadata, evaluate the data license and check the consistency of the data. This makes the process very time consuming.

As stated in D4.1, a **shared pool of data** allows pilots to benefit from each others' data collection activities. As an example, both the Health Habits and the Green Growth pilot make use of the gridded world population computed by the Socioeconomic Data and Applications

Center (SEDAC) [1]. Sharing data, however, requires ensuring data quality in such a repository. This cannot be done completely automatically, however some **automation in the process of reporting on data quality**, e.g., counting the number of NaNs, empty values and choosing the procedure to fill or ignore these data holes might be valuable. In the dedicated data analysis group in CoeGSS, an ongoing discussion evaluates which semi-automated processes will be helpful.

The shared data should also comply to certain rules and follow a shared convention regarding the name of resources, the data format in the storage, format and content of the metadata tab (reporting, for instance, the url of collection, date of access and keywords and description of the dataset), the name of columns, the reported kind of data in the columns, and a report on the data quality. Discussions in the consortium revealed the need for a **standard file format for metadata** like DCAT, which can be implemented in the workflow.

GSS data mostly comes in one of two forms: spatially explicit information can be stored in raster files, while tables are useful both for general datasets, such as data that are specified by country for a global model, and for synthetic populations, with which the population of agents in GSS-ABMs is usually initialized.

Specifically for table data, pilots plan to use **HDF5 tables** instead of the most common solution to use csv-files. Using PyTables (http://www.pytables. org/), it is relatively straightforward to convert csv files to HDF5 tables. Beside the advantage that HDF5 is ”HPC ready”, it also allows to add metadata to the tables, and to bundle several tables into a single file.

The following metadata can be added to an HDF5 table dataset:

Attribute	Type	Optional	Comment
DESCRIPTION	H5T_STRING	yes	
STRUCTNAME	H5T_STRING	yes	if not given, TITLE will be used
FILENAME LICENCE	H5T_STRING	yes	if not given, TITLE will be used
CONTRIBUTORS	H5T_STRING	no	most restricted license is used
DATA_VERSION	H5T_STRING	yes	By convention using the form: Name (e-mail); Name (e-mail)
META_VERSION	H5T_STD_I32LE	no	
COMMENT	H5T_STD_I32LE	no	This is the version number of the Metadata specification
	H5T_STRING	yes	

Once collected, data can be stored either locally on a desktop machine, or remotely, either directly on the cluster machines or via some data aggregation and retrieval service such as CKAN (see ckan.org). The latter is a web-based interface for loading data, visualising them (in raw mode or on maps if data are geo-located), editing rows, or adding data, and downloading datasets via a secured API.

A database system like CKAN is heavily favourable from the view of the Centre of Excellence, however, independent modellers will only use such a database if it provides benefits and is user-friendly.

Thus, from a pilot point of view this requires:

- uploading data into a common data pool does not slow down the workflow
- data sets can be handled as private
- rights and license management is integrated
- the process of data treatment (e.g., changing values, deleting outliers, filling gaps, normalizing data) must be tracked
- an integrated versioning system for data sets would be useful
- effective search functionality and exploration tools for data sets like visualisation are necessary to help modellers find useful data sets
- the number of different tools should be limited, an integration in a unified portal is favourable.

4.2 Data pre-processing for simulations

As mentioned in D4.1, **data pre-processing** is a pilot requirement. As much of it as possible should be done in an **assisted and generic** way. There are several tasks relevant in pre-processing data for simulations.

One of the common needs of the pilots is to **convert geo-referenced data from one representation to another**. Generally, widely used formats for geo-referenced data like **geoJson, ESRI shapefiles or geotiff** are required input formats to the SIS. Operations like **converting a shapefile to a raster or the other way around, geo-referencing of new data sets, data aggregation or dis-aggregation conversion to different resolutions** will be common tasks to fit data to the particular agent-based model structures. For example, a table containing GDP data per country may need to be converted into a raster file, that is, a map, in which each grid cell is attributed the GDP value of the country that the cell belongs to.

These file formats and conversion steps also play a role for the output of simulations. To understand the model output, data is usually aggregated towards a particular output variable of interest. So, one requirement is an **automated and generic way to extract from a given file, and aggregate it according to a given specification**.

Visualisation of geo-referenced data on a map is needed, and the **computation of different aggregations based on provided shape-files accounting for different authority levels** needs to be possible, for example, in order to switch between regional and country level data.

Also, available data may not be given at the targeted resolution (particularly spatially) of the planned simulations, requiring pre-processing either to **aggregate or disaggregate data**.

While data aggregation appears trivial, disaggregating data raises challenges and requires further hypotheses following the chosen approach. Indeed, interpolation can be performed either in a deterministic or a stochastic way and can be based on various possible interpolation functions.

Finally, an easy way to **define and implement data relations** would be beneficial: as an example, the Green Growth pilot could define “green cars” to be electric vehicles only, or electric vehicles and hybrids. Merging tables or geo-referenced files and operations on the data would be required to switch from one case to the other in automatically supported fashion.

Overall, the pre-processing of data needs to establish a common data structure that is automatically accessible by the simulation tools as input. Only in that way, potentially a semi-automated simulation workflow can be achieved in the future to support more efficient model development.

5 Synthetic populations

A synthetic population is a set of virtual agents that reproduces statistical distributions of the real-world population for relevant features. While sometimes the term is used to refer also to a simulation model for the dynamic evolution of the population, CoeGSS pilots refer to the dynamic part as an agent-based model. The synthetic populations are an input to agent-based models: they provide the agents.

5.1 Basic synthetic population storage

Synthetic populations are mostly stored in table form: for each agent in a population, the characteristics of interest are recorded. The functionality of HDF5 makes it a suitable data format for SPs: metadata can be included, and the various tables of agents can be bundled into one file for the whole SP.

5.2 Extension of pre-existing synthetic populations

A notable open source for already computed synthetic populations is the MIDAS project (http://www.epimodels.org/drupal-new/?q=node/112). Synthetic populations for about 80 countries are available, however, due to differing data availability, the quality of the populations differs, and some populations are based on outdated survey data.

Due to this availability and the purpose of re-usability it would be beneficial to be able to **alter and extend existing synthetic populations**. An example is ageing of populations if more recent data is not available. Furthermore, an existing SP could be extended by additional parameters (e.g., the habit of smoking, preferred transportation type, or car type owned, if any). New parameters could be added stochastically, or based on conditional probabilities regarding the original properties, or by other user-defined rules. Thus, general SPs could be tailored for specific model application. The possibility to drop some parameters and extend with others would also make an SP developed for one pilot re-usable in another pilot study.

5.3 Generate SPs according to a given specification

For modelling global systems, it is often required to model large areas of the world or the whole world together. For such cases, the problem of incomplete information becomes unavoidable. The generation of such global synthetic populations therefore needs to include methods that can construct such populations based on incomplete data and reasonable assumptions. At the moment of writing this deliverable, there is ongoing work on the generation of synthetic populations in the methods and tools workpackage (WP3). Also, a working group on synthetic populations, which brings together the pilot workpackage (WP4) with WP3, is looking into SPEW [2] as a tool for generating synthetic populations. The Green

Growth pilot has also initiated a dialogue with members of the group that has developed this tool. Requirements that may go beyond this tool may arise from the work of the synthetic population group.

6 Synthetic networks

Agents in a population relevant for a global system are connected with each other via an agent network, including a contact network (people meeting face-to-face) and social networks (including other means of communication, such as via the internet), see also D4.4. As these networks are relevant for the evolution of the global system, they should be represented in the system of virtual agents that represents the global system on the computer. We refer to the networks between the virtual agents of the synthetic population that determine who interacts or exchanges information with whom in the ABM's simulations as synthetic networks.

Sometimes, network information is included in a synthetic population (e.g., one of the agents' features may be a list of other agents that an agent is connected with). In other cases, a network between agents may be deduced directly from the population, even if not explicitly recorded. E.g., if agents have a home and a workplace, and one wants to consider the network in which all agents are connected to all other agents from their home and their workplace.

If using a given SP that comes without a network between agents or generating a new SP, pilots may have the requirement to **generate a synthetic network between agents of a synthetic population**.

Example, Green Growth pilot: The Green Growth pilot considers a global population of households. These agents partly base their decision which car to buy on information obtained from their social networks. Therefore, the Green Growth pilot team needs to generate an empirically sound social network between households. It shall be learned in interaction with the network experts at IMT Lucca, whether and how far this can become an automatised process in the CoeGSS modeller's toolbox.

Slightly anticipating on the dynamical part of GSS-SISs, it will be mentioned as a requirement for the agent-based modelling framework that the networks between agents must be allowed to evolve in a simulation (see Section 7 below). The evolution of networks over time will have to be defined for each ABM as fit to the particular problem under scrutiny. General **expertise on features of the evolution of agent networks in global systems**, for example based on the empirically observed evolutions of some prototypical systems, would be helpful to the pilots.

Example, Global Urbanisation pilot: The global urbanisation pilot, which studies the two way relationship between transport infrastructure and real estate pricing, requires the representation different kinds of interaction networks within one model, such as:

- Transport network (possibly multi-modal) redefining spatial accessibility, concerning roads (allowing for access with personal cars), bus lines, subway, etc.
- Real estate market, seen as linking every available dwelling unit to a set of possible bidders

- Eventually a social network (possibly multifold: family, friends, neighbourhood, colleagues, etc.) to capture social influence on housing and transport preferences and choices.

Looking beyond the current state of the pilots to a broader class of potential GSS applications, any model simulating a city might request different kinds of interaction networks, possibly of different nature and at different scales. To mention just a few examples:

- Multi-fold networks concerning different kinds of edges (example: social influence versus bus lines) and nodes (example: persons versus bus stops), allowing to link heterogeneous sets of nodes (example: inhabitants and dwelling units)
- Networks at different scales (example: transportation network including private and public, this one made of for instance bus lines and subway, a line being itself a linear network)
- Networks open to evolution (examples: the extension or appearing of a bus line, inhabitants moving into a new dwelling, etc.)

Generating such networks from partial data is a potential future requirement relating to these examples. Further, the CoeGSS ABM-framework needs to be able to deal with such networks, as will be discussed in the following section.

7 GSS-specific ABM-framework

To study the evolution of a global system, an agent-based model implements interactions between virtual agents and their environment. Simulation runs of this ABM compute repeated interactions to simulate the overall system evolution.

The implementation of new GSS-ABMs in HPC-ready fashion requires an **agent-based modelling framework**. This is the case for two of the three pilots, Health Habits and Green Growth. As mentioned above, the Global Urbanization pilot is an example of the second potential use case: it uses its own domain-specific development for simulation purposes, porting this simulation tool to HPC platforms available in the project.

During the first year of the project, it became obvious that no available ABM framework is sufficient for attracting larger communities of GSS modellers to use HPC resources. Available software solutions are proprietary to a small user group, are too specific to the particular field of study, or they require deep knowledge of modern software development and parallelisation from the modeller.

As stated in D4.1, the Green Growth pilot chose to begin model implementation using Pandora [3] as this framework. The Health Habits pilot is also implementing a model in Pandora. While this framework provided useful functionalities, in its current state it was not sufficient, for example due to its partitioning of the system into equal rectangles. Useful for the historical simulations without telecommunications that Pandora was originally developed for, in the case of a global map, this partitioning was obviously inefficient (e.g., some processors would be allocated pieces of oceans). Therefore, a GSS-focused HPC agent-based modelling framework is currently under development in CoeGGS. This section describes the pilots' requirements for this framework.

7.1 Meta-requirements

First, the new framework needs to be **reasonably compatible with parts of Pandora that the pilots are using**. That is, once the new framework is ready, pilot models should be easily transferable to it. Some adaptation will probably be necessary, but where Pandora functionality should be kept, this is stated in the requirements below. Until the new framework can be used, Health Habits and Green Growth pilots will continue to use Pandora, renouncing model structures that cannot be implemented, or accepting lower efficiency in the meantime.

Secondly, we aim to sustain an **object-oriented programming style for the model implementation by the user**. This intuitively supports the agent-centred perspective taken in implementing ABMs and will hence make the framework user-friendly for GSS-experts, who are usually no parallel programming experts. The agent-based modeller needs to be able to

stick to common practice for ABMs in that he or she can implement the model dynamic focusing on agents' actions and interactions, both between agents and between an agent and its environment. While the whole ABM may be considered a dynamical system with a state (including the states of all agents and the environment) and a transition function that defines how this state changes from one time step to the next, the modeller usually does not define this overall transition function, but defines single transitions for the system's elements (agents, network between them, environment) and defines in which order these take place. Object-oriented programming fits this style of model definition and implementation.

This is closely related to the goal of **hiding parallel MPI code completely from the user of the framework**. For pilot modellers in particular, but later also for other GSS users of the framework, model development should be independent of a later parallel or sequential execution. For prototyping purposes and small model code changes in the iterative model improvement process, the modeller should also be able to quickly check an implementation by running the model (at least a version reduced in size, for example a country instead of the world) locally on a laptop or similar. Furthermore, an API or code framework in C++ or other languages that does not require writing MPI-code when implementing a model, would enable access of many more potential GSS users. This will allow bringing new applications onto the HPC-canvas and, if a specific model proves successful and useful, it can then be improved towards efficient scalability by services provided by CoeGSS.

Together, these meta-requirements shall overcome the difficulties we found with existing ABM-frameworks for HPC, as they can be summarized in providing a simple, intuitive API and design of the framework.

7.2 Model structure requirements

The following are important features of GSS ABMs together with requirements resulting from these:

- Agents can be of different types with different characteristics/properties and actions/methods. The framework therefore needs to **allow for defining different types of agents and for letting them store a variable number of traits and variables**.
- In global systems, often, agents' locations are a relevant feature. Therefore, the framework must allow **agents to be assigned to a (discrete 2D) location**. The spatial distribution of agents is generally rather imbalanced (unless one looks at rather small spatial scales, for example a homogeneousy populated part of a city). Losses in efficiency, that this causes, should be kept at a reasonably small level, in particular, the framework should provide a better solution than a mere spatial partitioning into geographically equal parts would provide. However, as short distance communication and interaction is also an important feature in global systems, the framework should

offer optimised functions for short distance communication, for example: select a random agent in a radius of 3 cells.

- The behaviour of agents has three underlying layers: individual, social, and environment (for details, see D4.4). Also, agents mostly do not have perfect information about the global system they operate in. Therefore, an agent can only access information from its "neighbourhood" defined via connections with other agents within the social structure of its agent networks, or via the definition of a visible part of the environment it is placed into via its location. Overlaying these two structures generally leads to **complex graph structures that the framework needs to be able to handle**. In particular, examples of structures present in global systems are the following:
    - the social networks need not be restricted by a spatial interaction radius for each agent, as **long-range communications** are present in global systems, for example due to internet-based communication.
    - **hierarchical structures** may be present, as for example in synthetic populations that place agents in households in communities in cities etc., with more interaction the closer agents are in this structure
    - the agent network may contain not only **different types of agents but also different types of connections between them**. For example, in a macro-economic implementation of a market, households, companies and banks may interact and the relations between these can be private, commercial or regulatory. A household may work at a firm or buy goods from a firm, so there may even be various edges of different types between two given agents.
    - information may pertain to large groups of agents in a global system, for example policy and regulation or advertisement may be relevant for the whole population of a country or region. It must thus be possible to **represent entities that influence large groups of agents in the framework**.
- Changes in agent networks and in agents' locations may be important in analysing certain global challenges. The framework therefore needs to be able to **handle dynamically changing networks and mobile agents**.

In view of the above, multiple discussions between pilot modellers and HPC-experts concluded that the **underlying structure of an HPC-ABM framework tailored to global systems must be graph-based**. Only graphs can represent social as well as spatial (including regulatory) layers of interactions within global systems efficiently. Fortunately, a mathematical body around graphs already exists and comprises important solutions for parallel programming with graphs. For example, the partitioning of a complex graph is a known problem and parallel solving routines already exist. However, as seen for example with the different types of nodes and edges required, GSS-specific graphs require extensions of traditional graph approaches.

A mathematical definition of a matching ”Agent Graph” structure has been prepared[1]. In short, an agent graph is a directed multi-graph (multiple directed edges between a pair of vertices), where:

- a vertex symbolises an agent (in a broad sense), an edge symbolises a connection between agents
- agents/vertices and connections/edges may have different types
- part of the agent graph is a valuing function; this function ”attaches” the state (that is, the value of all its characteristics at a given time step) of the agent to the vertex
- for each edge and agent type exist separate transition functions
- agent and transition type have to match, otherwise the identity is the only possible transition
- the transition function of an agent depends only on information the agent can observe
- agents and edges may be removed or new ones added at runtime

In view of parallelisation, the agent graph structure also specifies some restrictions:

- an agent cannot change the state of another agent (but can send messages to agents on outgoing edges, in the sense that these agents can receive information from the former)
- agent decisions (transition functions) are based only on the previous state of agents connected via incoming edges (and the state of the agent itself)
- an agent can only be removed from the graph by itself
- an agent can only add or remove edges between vertices which are adjacent to it or between itself and an adjacent vertex
- if an agent is removed, all adjacent edges are removed
- similarly, an agent can only add edges between vertices which are adjacent to it, or add an edge of a different type between itself and an adjacent vertex
- the type of an agent is immutable

While these restrictions should be helpful in reducing the necessary communication in a parallel version, they can be justified by the structure of global systems made up of many agents that are far from omniscient. Agents obtain information from and interact within given networks, their spatial position plays a role, and their decisions can be viewed as rather autonomous.

Finally, since the network may evolve for long-term simulations, **it might be necessary to dynamically repartition a graph within a simulation**. This will only be an essential

[1] Internal document, available upon request.

requirement if otherwise the simulation performance dramatically suffers by imbalances that arise from changes in the networks.

7.3 Input/output requirements

Within the SIS, data transfer into the ABM (initialisation and potentially runtime input data) and out of its simulation runs (output data) is necessary. The ABM framework therefore needs to provide the required **interfaces and connection to other tools** the pilot/GSS modeller uses. Dealing with global systems, **handling big datasets** needs to be possible – for example, the above mentioned synthetic populations of the MIDAS project, that cover about two thirds of the world population, would be roughly a terrabyte. At the current state of the pilots' models, this requirement has, however, not yet been an issue.

7.4 Data import requirements

As mentioned above, GSS data mostly come in two forms: maps and tables. Two main input requirements are therefore the **support of geographic information system (GIS) raster data**, and **import of HDF5 tables**. In particular, when working with a global synthetic population, the framework must allow handling large numbers of agents (potentially several billion) and the related data.

The support of GIS raster files is present in Pandora and needs to be kept, since many data sources have to be spatially explicit in the analysis of global systems.

Concerning the import of HDF5 tables, defining all the necessary parameters for reading such a table needs a lot of boilerplate code, as can be seen in examples from the HDF5 Table tutorial (https://support.hdfgroup.org/ftp/HDF5/current/src/unpacked/hl/examples/ex_table_01.c). All the needed information for this code already exists in the HDF5 file itself, and the additional metadata described in Section 4.1 could be used to add e.g. the description of the fields as comments in the generated struct. One viable approach to handling this boilerplate is to **write a code generator** that is interoperable with the agent-based modelling framework just described. The code generation itself should be part of the model's build process. An advantage of this would be that consistency between the model and the input data can be checked at compile time. For example, a compilation error will occur in the case that the model tries to access a removed or renamed column of a table. A concrete specification of the table import code generator is currently work in progress.

7.5 Output requirements

Concerning raster and HDF5 files, if possible, the output should keep the file format Pandora is using, as this would allow to reuse existing post-processing solutions. Concerning output data for single agents in the model, Pandora produces single files per process. Here,

discussions between the framework developers, pilot researchers and visualisation experts should be the basis for deciding how to best implement output of agent data in the framework. Generally, the modeller needs to be able to **customise which output data to write**. For example, one might work with monthly computations, but want to look at yearly output data. The output data has to be compatible with tools used for simulation analysis and visualisation further down the line.

8 SIS analysis tools

For constructing a complete synthetic information system, several auxiliary software tools are required to complete and support simulation model development. Especially concerning the difficult conceptual model development, the modeller requires extensive support by automated analysis tools. Currently, the pilots' conceptual development process is slower, and hence the time between consecutive model versions is longer, than it would be with those tools in place, due to the need to manually analyse models.

The required tools need to allow organising multiple model runs and analysing these, thus speeding up model development in the future. To fit a broad class of modellers' needs, the auxiliary software needs to meet the following general requirements:

- The software controls the entire simulations workflow.
- It is able to set up multiple model runs with different input parameters and data sources.
- It automatically starts the model application, collects the output data and aggregates to specified levels, if necessary.
- Finally, it performs an analysis that is defined by the modeller in the simulation workflow.
- The software should be platform independent and work on different HPC environments.
- The software framework provides the necessary visualisation for the modeller to assess the model's dynamics, as well as the quality of the model output, and the uncertainty.
- The software tools should offer a well-documented API to allow easy adaptation to new models.

In the following, we describe three stages of model development and usage. By meeting the requirements above, the software should be able to support the modeller in all three stages. Each use case provides more detail on the required functionalities.

8.1 Initial model analysis and development

Firstly, for the initial phase of model development, analysis of the model dynamics and proof of concepts are required. Simple observation of the results of the outcome of a simulation and the aggregated dynamics of all agents need to be analysed. This prototyping phase includes tuning the model related to different aspects:

- Spatial resolution and aggregation (resolution of grid, aggregation for instance over areas or districts)

- Conceptual level of detail (for instance the number of agents, or agent-related parameters and attributes)
- Temporal resolution and aggregation (What is the necessary temporal resolution? Is the evolution itself of interest, or only the final state?)

Indeed, preliminary work has to be done in order to run exploratory versions of the model. Once a particular implementation has been chosen, one performs a set of calibration runs in order to fix the optimal value of one or more parameters of the simulation.

Example, Health Habits pilot: Several preliminary simulations of the model are needed in order to calibrate the infection rate, say β, of the smoking epidemics. Specifically, for each value of β in a given range one has to run several simulations starting from slightly different initial conditions of the system (i.e., the initial smoking prevalence) and see whether the value of β allows for a stable solution of the system dynamics. In other words, one has to check that at a fixed β value, the system always ends up in a given asymptotic state featuring a specific prevalence of the smoking habit. Once this is done, β will be set to the value that produces the best agreement between asymptotic smoking prevalence and empirical data.

Another preliminary study needed is sensitivity analysis, which is conducted so as to identify relevant and redundant parameters of the model. In this use case an efficient initial development requires tools to easily set up automatic parameter runs. It would comprise the following functionalities:

- **Set up of multiple batch jobs** (e.g., for exploration of the parameter space)
- **Visual comparison of different model runs**
- **Reporting of log files and extracts of the model output**

This combination of support tools is required to speed up the development process and lead to more complex and larger-scale models. For sensitivity analysis and exploratory model runs, available tools like openmole tool (https://www.openmole.org) can minimise the number of runs and simulations needed in order to characterise the parameter space. The functionality of the SIMENV tool (https://www.pik-potsdam.de/research/transdisciplinary-concepts-and-methods/ tools/simenv) is helpful at this stage.

8.2 Statistical model analysis

Secondly, for more consolidated models, the further development requires more specific simulation runs. A set of many simulations of different configurations of the system where the value of one or more parameters is regularly changed is usually referred to as a parameter sweep. Higher-level investigation includes parameter calibration, history matching and design of experiments.

At this stage, the modeller pursues a specific aim. Formulating a scoring function for this aim, the modeller wants to find the best results according to this scoring function. An example is a

model calibration process, that is, matching simulation output to available data in the best possible way. The scoring function in this case will define the error of the simulation with respect to the given data. This error should be minimised, that is, the parameters to be calibrated should be set in such a way that the simulation comes closest (in terms of the scoring function) to the data. The software should be able to choose those parameters that optimise the score of a simulation.

Due to stochasticity of the models, all of the previous levels of results' analysis are further completed by calculating statistical indicators over a sufficient number of replications for a given initial state and set of parameter values. That is, stochastic analysis for given parameters and initial conditions is needed to assess the randomness of the model.

In summary, GSS-modellers at this stage would require following functionalities:

- automatic start of multiple model runs with different input parameters and data sources
- tools to provide sampling techniques for minimal number of model runs (e.g., by Latin Hybercube Sampling, Marcov Chain Monte Carlo or other sampling techniques)
- directed sampling to optimize a given user-defined score function
- visualisation tools of multiple model runs and uncertainty (e.g., probability distributions, uncertainty, sensitivities, regression, correlation, scatter plots)
- data analysis of the vast amount of agent data will require Big Data tools, e.g., clustering of similar agents behaviours or properties

8.3 Stakeholder decision support

Thirdly, for fully developed models, results analysis can aim at clarifying the model's behaviour by finding high-level insights about the system. This behaviour is usually of interest for the stakeholder who is interested in answering the initial research question. As examples, this stage includes the following types of analysis:

- Answering pre-existing questions at the level of the simulation
- Calculating high-level indicators over a set of simulations: study of model dynamics, resiliency
- Identifying regularities at higher / non explicitly simulated levels: emergence
- Decision support by analyzing the model output for different policy or regulatory decisions.

This highest level of analysis of GSS simulation models may benefit from HPC and high performance data analytics facilities allowing to perform larger (more systematic) sets of statistical analyses for instance to identify high-level patterns (emergence of regularities) or try to identify high-level surrogate laws (as the perfect gas equation summarises all the particle movements).

Thus, for this final policy-advising phase, the following tools are required:

- Scenario and uncertainty analysis
- Solving optimisation problems in parallel for user defined score functions
- Visualisation of model runs, statistical outcomes of multiple model runs (e.g., uncertainty margins, joint probability distributions)
- Decision support by real-time visualisation

8.4 Visualisation of simulation output

The requirements regarding the visualisation of simulation output are basically unchanged since D4.1 and D3.2. The need to visualise statistical outcomes of (multiple) runs is already picked up in Sections 8 and 8.3. In the following, we present some mockups to flesh out some of the requests presented in those documents.

Thereby it is important to note that the visualisation tools should work on the visualisation nodes at HLRS and PSNC as the generated output of a simulation can be large (the current version of the still rather simple Green Growth pilot model already produces 80 GB per simulation), so that a transfer of the results to other locations before starting the visualisation seems impractical.

8.5 Maps

One of the requirements mentioned in D3.2 was the possibility to **display the different rasters of a simulation run, with the possibility to zoom into regions**. Additionally it should be possible to **compare different simulation runs**. Figure 2 shows an exemplary (minimal) user interface for this purpose.

In this example, simulation outputs of the Green Growth pilot of two runs with different values for one parameter are compared[2]. As many datasets have outliers, it is important that an upper limit for the colour bar can be set. Therefore three different methods are useful:

- Give a fixed upper limit
- Use a quantile of the datasets
- Use a factor of the maximal value

To be able to visually compare output for different time steps in a simulation, it is necessary that in the quantile and factor case, optionally the quantile or maximal value of a certain time step can be fixed and used also for other steps.

[2] As the figure is only a mockup, there is nothing meaningful to detect in the presented results.

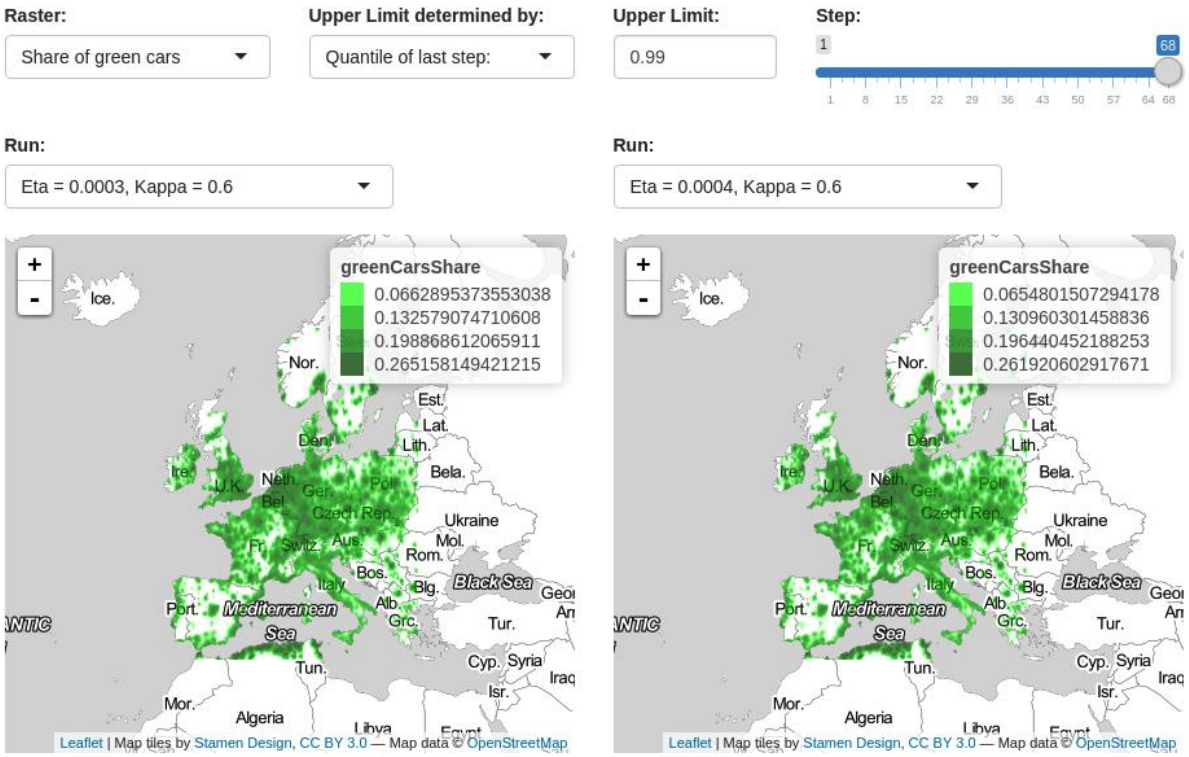

Figure 2: Visual comparison of rasters on a world map.

8.6 Parameter exploration

Two other requirements from D3.2 were the **visualisation of a large set of parameters** and the **presentation of aggregated results as time series**.

The mockup in Figure 3 shows an ensemble of 60 simulation runs, where 20 parameter combinations were run with three different random seeds. The upper part of the figure displays a parallel coordinates plot, the lower part the time series plot of the 60 runs for the aggregate share of green cars in Italy.

The plots are interactively connected, when some points in one plot are selected, the corresponding data of the other plot(s) will be highlighted. In the example given here, the two simulation runs with the highest share of green cars in the last year are selected in the time series plot, and the parallel coordinates plot shows the parameter combinations that lead to these trajectories.

It is useful that multiple selections can be done and combined with “and” or “or”. For example, this would allow selecting a specific run using the parallel coordinates plot.

Another useful plot type would be the scatter plot, for example, to visualise the state of (a subset) of the agents.

As mentioned in D4.1, GGobi (http://www.ggobi.org/) is an open source visualisation program with a lot of features in this direction. Maybe it is possible to reuse some of the work already done for this tool in the visualisation task in WP3.

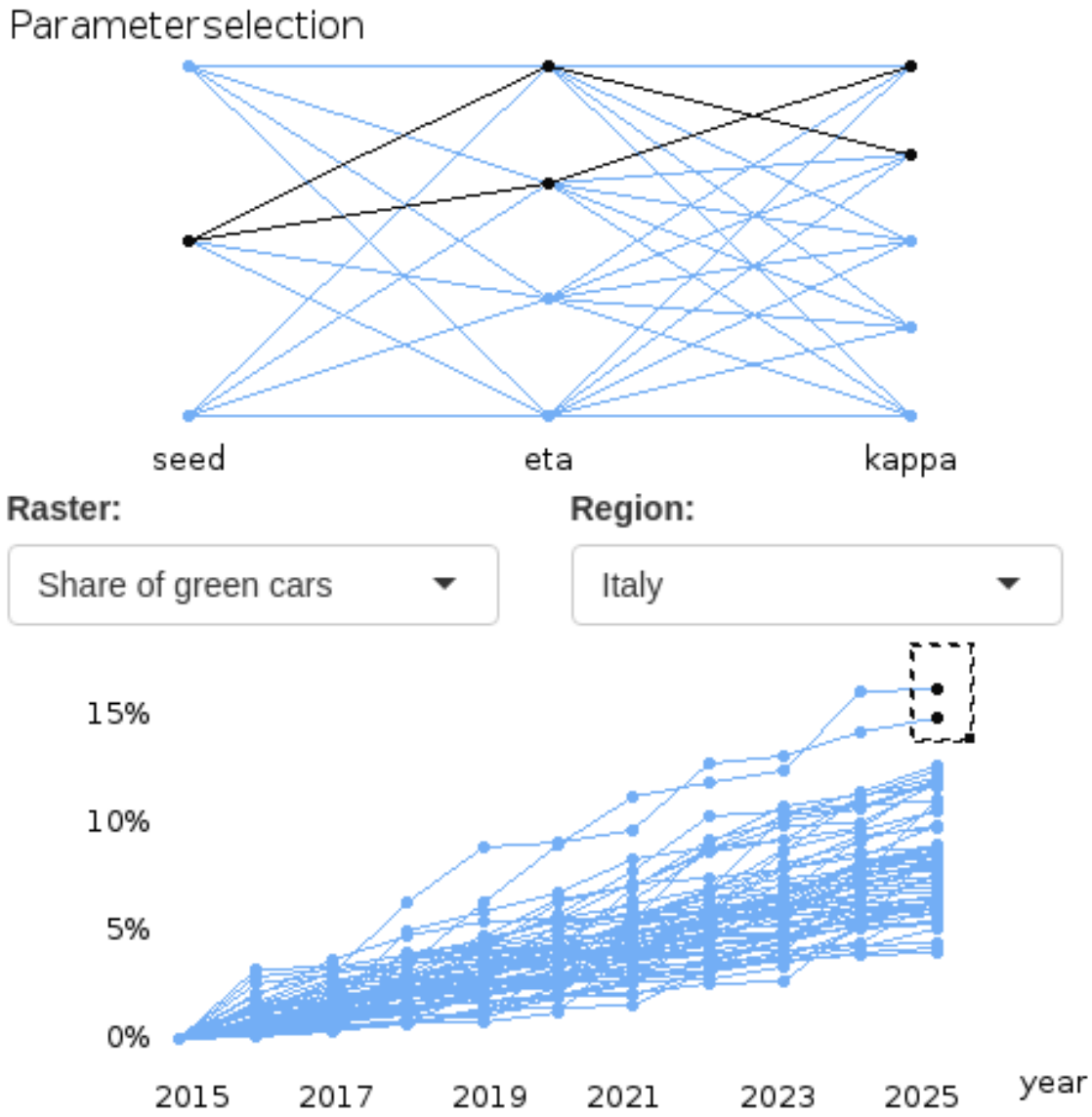

Figure 3: Time series with interactive run selection.

9 Conclusions and outlook

To summarise, the pilot requirements are listed again below for a short overview. As far as explanations are needed, they can be found in the respective sections above. As mentioned throughout the document, the consortium is discussing several of the points described here in dedicated working groups. The interaction between GSS and HPC communities in the project thus provides an ongoing exchange about GSS requirements to HPC. This document presents a snapshot of the pilots' requirements at this point in time, in some cases with a view also to other GSS applications in the HPC world. The ongoing discussions will in turn further sharpen the requirements, and decisions taken on some points may influence related requirements in the future. A third version of this living document shall be presented in deliverable D4.3 in about a year's time. Smaller documents with specific requirements for certain elements of GSS SISs will be exchanged in the consortium as deemed useful in the meantime.

One further necessary step in the pilots' SIS development is stakeholder interaction, both for obtaining further insights on the systems under study from practitioners dealing with these systems on a day-to-day basis and for feedback on which questions about the systems stakeholders are most interested in. With such feedback, the SISs can be further developed into the most meaningful directions. We have not addressed this point above, as it does not directly pose technical requirements from the pilots to the centre. As an outlook on pilot work, however, it should be mentioned that pilot researchers are embedded in networks of people and organisations, each around their specific field of expertise that can provide such feedback – which will be crucial in ensuring sustainability of the business model of CoeGSS. Roadmaps for each pilot that include a plan for stakeholder engagement to solicit feedback are being defined at the time of writing of this deliverable.

9.1 Pilot requirements to CoeGSS in short

- Common language and training: consolidation of a common language
- Data:
    - shared pool of data
    - some automation in the process of reporting on data quality
    - standard file format for metadata: HDF5 tables, UTF8
    - pre-process data in an assited and generic way
    - convert geo-referenced data from one representation to another
    - geo-reference new data sets
    - aggregate and dis-aggregate data
    - automated and generic way to extract data from a given file, and aggregate it according to a given specification

    - visualise geo-tagged data on a map
    - compute different aggregations based on provided shape-files accounting for different authority levels
    - define and implement data relations in an easy way
- Synthetic populations:
    - alter and extend existing synthetic populations, e.g., by adding parameters
    - generate synthetic populations
- Synthetic networks:
    - generate a synthetic network between agents of an SP
    - expertise on features of the evolution of agent networks in global systems
- Agent-based modelling framework
    - keep the framework reasonably compatible with parts of Pandora that the pilots are using
    - object-oriented programing style for the model implementation by the user
    - hiding parallel MPI code completely from the end-user of the framework
    - graph-based structure (directed multigraph):
        - a vertex symbolises an agent
        - agents/vertices and edges have different types
        - agents have a state and a (discrete 2D) location, visibility of parts of other vertices can be defined
        - for each edge and agent type exist separate transition functions
        - agent and transition type have to match, otherwise the identity is the only possible transition
        - the transition function of an agent depends only on information the agent can observe
        - agents and edges may be removed or new ones added at runtime
    - handle complex graph structures resulting from
        - long-range communications,
        - hierarchical structures,
        - entities that influence large groups of agents
    - handle dynamically changing networks and mobile agents
    - keep losses in efficiency due to imbalances in spatial distribution of agents small
    - offer optimised functions for short distance communication
    - interfaces and connections to analysis tools
    - handle big datasets and potentially very large numbers of agents
    - support geographic information system (GIS) raster data
    - import of HDF5 tables
    - keep output data functionality of Pandora for rasters and HDF5 files

- let the user customise which output data to write

- SIS analysis tools
  - allow for organising multiple model runs and analysing these – the software controls the entire simulation workflow
  - the software should be platform-independent and work on different HPC environments
  - the software tools should offer a well documented API to allow easy adaptation to new models
  - set up multiple model runs with different input parameters and data sources
    - set up of multiple batch jobs (e.g., for exploration of the parameter space)
    - sampling techniques for minimal number of model runs (e.g., by Latin Hybercube Sampling or other sampling techniques)
  - automatically start the model application, collect the output data and aggregate to specified levels, if necessary
    - reporting of log files and extracts of the model output
  - perform an analysis that is defined by the modeller in the simulation workflow
    - directed sampling to optimise a user-defined score function
    - scenario and uncertainty analysis
  - visualisation tools
    - multiple model runs and uncertainty (e.g., uncertainty margins, joint probability distributions), statistical outcomes of multiple model runs
    - maps: possibility to zoom in, customise colour bar of maps (upper limit, quantile or factor; fix for comparison of several maps)
    - interactive visualisation of sets of runs for different parameters
    - decision support by real-time visualisation

10 Glossary and abbreviations

ABM Agent-based model: a model that implements interactions between agents and an environment in order to observe the modelled system's behaviour as resulting from these interactions in simulation runs of the model.

API application programming interface

Communication, long/short range: as mentioned, agents in global systems often are assigned to locations on some underlying map. They may obtain information from other agents in the system, or supply other agents with it. The terms long and short range communication designate whether the two agents that communicate in this way are far away from each other in terms of their locations, respectively, close to each other.

GDP gross domestic product

Geo-referenced data: data that is related to a system of geographic coordinates

GSS Global Systems Science

HPC High Performance Computing

HPDA High Performance Data Analytics

MPI message passing interface

NaN "not a number" is a numeric data type value that represents an undefined or unrepresentable value; here, in particular, missing values in data sets

SIS Synthetic information system: a tool for simulating potential evolutions of a given real-world system under study, comprising a synthetic population, an agent-based model, and tools for analysing and visualising simulation runs of the ABM initialized with the SP.

SP Synthetic population: a set of virtual agents that reproduces statistical distributions of the real-world population for relevant features. While sometimes the term is used to refer also to a simulation model for the dynamic evolution of the population, CoeGSS pilots refer to the dynamic part as an agent-based model.

Synthetic network: We refer to the networks between the virtual agents of the synthetic population that determine who interacts with whom in the ABM's simulations as synthetic networks.

Transition / transition function: considering the evolution of a system, one often looks at changes from one time step to the next. In dynamical systems, these changes are described by a transition function that is applied to the state of the system (describing all relevant features of the system at that moment in time) and provides the next state of the system. We use the term transition also for the changes that occur from one step to the next to given elements of the system.

Valuing function: considering agents represented by vertices in a graph, in which the edges are connections between them, we need a means to record the agents' state (values for all their characteristics at a given point in time). We refer to the function that associates to each agent vertex its state as a valueing function because it attaches values.

11 References

[1] Socioeconomic Data and Applications Center (SEDAC). Gridded Population of the World, v3. http://sedac.ciesin.columbia.edu/data/collection/gpw-v3/maps/services, 2015.

[2] Shannon Gallagher, Lee Richardson, Samuel L. Ventura, and William F. Eddy. Spew: Synthetic populations and ecosystems of the world. (arXiv:1701.02383), 2017.

[3] Xavier Rubio-Campillo. Pandora: A Versatile Agent-Based Modelling Platform for Social Simulation. In SIMUL 2014: The Sixth International Conference on Advances in System Simulation, 2014.

D3.2: CoeGSS Deliverable D3.2; FIRST SPECIFICATION OF NEW METHODS, TOOLS AND MECHANISMS PROPOSED FOR THE SUPPORT OF THE APPLICATION USER AND PROGRAMMER; delivered to the European Commission; Patrik Jansson (editor), Marcin Lawenda, Eva Richter, Uwe Woessner, Cezar Ionescu, Michał Pałka, Ralf Schneider, Devdatt Dubhashi, Michele Tizzoni, Daniela Paolotti, Steffen Fürst, Margaret Edwards

D4.1: CoeGSS Deliverable D4.1; FIRST REPORT ON PILOT REQUIREMENTS; delivered to the European Commission; Sarah Wolf (editor), Daniela Paolotti, Michele Tizzoni, Margaret Edwards, Steffen Fürst, Andreas Geiges, Alfred Ireland, Franziska Schütze, Gesine Steudle

D4.4: CoeGSS Deliverable D4.4, FIRST STATUS REPORT OF THE PILOTS; delivered to the European Commission; Sarah Wolf (editor), Marion Dreyer, Margaret Edwards, Steffen Fürst, Andreas Geiges, Jörg Hilpert, Jette von Postel, Fabio Saracco, Michele Tizzoni, Enrico Ubaldi