



*Platform for European Medical Support
During Major Emergencies*

D3.2 Scenario Generator





PULSE
Platform for European Medical Support
during major emergencies

WP3 MODELLING

Deliverable D3.2-Scenario generator

30/11/2015



Project Number:	Project Acronym:	Project Title:
607799	PULSE	Platform for European Medical Support during major emergencies

Title:	Document Version:
D3.2: Scenario generator	v1.2

Contractual Delivery Date:	Actual Delivery Date:	Deliverable Type*-Security**:
30/11/2015 (M18)	30/11/2015	P - PU

*Type: P: Prototype; R: Report; D: Demonstrator; O: Other.

**Security Class: PU: Public; PP: Restricted to other programme participants (including the Commission); RE: Restricted to a group defined by the consortium (including the Commission);

Responsible:	Organisation:	Contributing WP:
Alessandro Borri	UCSC	WP3

CO: Confidential, only for members of the consortium (including the Commission).

Authors (organisation)
<p>Alessandro Borri (UCSC)</p> <p>Andrea De Gaetano (UCSC)</p> <p>Rachele Brancaleoni (UCSC)</p> <p>Daniele Gui (UCSC)</p> <p>Giuseppe Martufi (SES)</p> <p>Carlo Pieralisi (SES)</p> <p>Reinhard Hutter (CESS)</p> <p>Viorel Petcu (OST)</p>

**Abstract:**

The purpose of this document is to support the delivery of the PULSE prototypes related to the generation of the scenarios for the simulation environment. The PULSE basic scenarios of "Stadium-crush incident" and "SARS-like epidemic" are investigated.

The scope of deliverable D3.2 "Scenario generator" covers task T3.4 of PULSE WP3 but is strongly related to the other modeling tasks of WP3, addressed in deliverable D3.1 "Context models".

Based on suggestions made by stakeholders and end-users involved, as well as on the experience gathered in the first part of the project, the scenario generator software modules help build the "first frame" of the evolution of the synthetic simulated environment, on which the PULSE platform will be evaluated by means of exercises and trials.

In the stadium-crush scenario, the focus is on the generation of the incident (number of interested people, lesions reported), based on which the healthcare response will be provided by the system and hospitals involved in the interested area. A library of potential complicating events or variants and their probabilities is maintained, allowing for repeated use of the basic scenario with different run developments. A java interface is built for the selection of some inputs by the users and for the retrieval of some aggregate data.

In the SARS-like scenario, with respect to what was done in D3.1 for a map focused on Italy, other possible areas of diffusion for SARS in Europe are analyzed, with different distributions of population, geographical factors, transportation, and connectivity. Moreover, support is provided for the future generation of maps on arbitrary regions, definable by the user of the PULSE Platform. The results are exported in terms of spreadsheets with a given format. The data are read by the EnSIR model/tool (see D.3.6 and D.4.7) with the aim of defining the initial picture of the epidemics.

Keywords:

Scenario generator, stadium crush, SARS, webservice



D3.2-Scenario generator REVISIONS:

Revision	Date	Description	Author (Organisation)
1.0	13/11/2015	Draft	Alessandro Borri (UCSC)
1.1	20/11/2015	Revised draft	Alessandro Borri (UCSC)
1.2	27/11/2015	Final draft	Alessandro Borri (UCSC)



TABLE OF CONTENTS

1	Glossary	8
2	Introduction	11
2.1	Purpose and scope of the document	11
2.2	Structure of the document	11
3	General architecture	12
3.1	Relation to the overall PULSE platform	12
3.2	Features overview in the context of the other tasks/deliverables	12
3.3	Software delivery	15
4	Stadium crush scenario generation	16
4.1	Description and functionalities	16
4.2	Implementation	18
5	SARS epidemics scenario generation	23
5.1	Description and functionalities	23
5.2	Implementation	28
6	3 rd Party libraries and licenses	32
7	Concluding remarks and future work	33
A.	Attachments	34



List of figures

Figure 1: WP3 tasks in relation to the scenarios	14
Figure 2: WP3 tasks in relation to the deliverables	14
Figure 3: Scheme of the architecture used for the service	15
Figure 4: Java GUI for the Stadium Crush scenario generation.....	19
Figure 5: ScenGen definitions.....	20
Figure 6: Damager definitions	21
Figure 7: ScenGen and Damager request and response messages.....	21
Figure 8: ScenGen and Damager operations.....	22
Figure 9: ScenGen and Damager java methods definitions.....	22
Figure 10: ENSIR prototype client interface	23
Figure 11: Moldova region.....	25
Figure 12: Transilvania region.....	25
Figure 13: Muntenia region	26
Figure 14: Moldova gridded map with population values	27
Figure 15: Transilvania gridded map with population values	28
Figure 16: Muntania gridded map with population values	28
Figure 17: ENSIR function implementation.....	29
Figure 18: ENSIR_core function implementation.....	30
Figure 19: M_pop.csv (Moldova region)	31



List of Tables

Table 1 - Glossary	8
Table 2 - Estimated probability of each lesion in the stadium crush scenario	17
Table 3 - 3rd party libraries and licenses	32

1 Glossary

Table 1 - Glossary

Term	Definition	Notes (examples from PULSE, explanations. ...)
CCS	Casualty Clearance Station	It is located at a safe distance away from the incident, to safely manage casualties delivered from the scene. It serves as a point for secondary triage and for provision of life saving treatments to safely package the casualties for transport to hospital
DoW	Description of Work	The official document, version 2013-10-11, that states PULSE project scope and content
ECDC	European Center of Disease Prevention and Control	
ECM	Event Medical Co-ordinator. The person with the clear task of overall control and coordination of medical/first-aid provision at the event. That person is not a public servant. They are the agent of the organisers and the single point of contact in relation to the event medical plan. That person is also the point of contact for the Regional Authority official agencies dealing with the planning and running of events.	
EHS	EU health system	
Ethical issues	Ethical issues refer to the issues concerning some aspect that raise ethical questions	
Ethics	Ethics is the systematic reflection on right and wrong conduct according to norms and values that we think should be adhered to	
EU	European Union	
Functionality	Any service that a product or a software can do for a user	
IHR	International Health Regulations The International Health Regulations (2005) are legally binding regulations (forming international law) that aim to a) assist countries to work together to save lives and livelihoods endangered by the spread of diseases and other health risks, and b) avoid unnecessary interference with international trade and travel	
Model (see also PULSE Model)	An abstraction of reality with the aims of better understanding it, mostly described in mathematical/ analytical, also sociological or philosophical terms and methodologies	
MoE	Measures of Effectiveness	
MPORG	MultiPlayer Online Role Playing Game are popular for both training and recreational gaming. People typically use an avatar to represent themselves in a virtual world where they can perform tasks in predefined scenarios. Multiple people participate and interact in the same virtual world in parallel. MPORG system are typically accessed via the internet and used by end users in disparate locations.	Within PULSE an MPORG system and environment will be used to train personnel within the stadium crush scenario where individuals will assume the roles of different resource personnel involved in such a scenario.
NGO	Non Governmental Organisation	
OSCE	Organization for Security and Co-operation in Europe	



Phase	A subset of a Scenario. It may be, for instance, identified, for instance, in terms of time (e.g. before the incident) and/or location (e.g. Hospital) and/or type of population involved (e.g. people in "uncertain" status in a SARS like epidemic), and/or purpose (prepare, recover)	Each PULSE Scenario is split in two Phases: Preparedness and Response.
Platform	see <i>PULSE Platform</i>	
Policy	Documents that provide high level guidelines, in terms of actors and responsibilities; they may also specify key phases	The " <i>Decision No 1082/2013/EU of European Parliament and of the Council of 22 October 2013 on serious cross-border threats to health</i> " is an example of Policy
Preparedness phase	Activities that prepare and train responders and ensure that the needed mix of resources are ready to respond in case of adverse event	
PULSE	Platform for European Medical Support during major emergencies	
PULSE Model (see also Model)	A Software routine, based on mathematical models/ algorithms for describing phenomena (e.g. processes, problems,...) and for helping to find solutions. In PULSE project, in order to avoid confusion with the general meaning of the term "Model" (see definition), the term "PULSE Model" is introduced	In the PULSE DoW (Document of Work) WP3 has the purpose to produce Models, meaning the "PULSE Models" here defined
PULSE Platform	PULSE System + PULSE SOP	
PULSE Project	The Project that will specify, design, implement and validate the PULSE Platform	
RCS	Recognised Current Situation	
Requirements	Justified characteristic needs, formulated by users and experts. For IT systems, usually one distinguishes between technical and operational (possibly strategic) requirements	
Response phase	Activities that are triggered by the adverse event, with the purpose to diminish/contain its effects	
RSI	Regolamento Sanitario Internazionale (italian term for IHR)	
SARS	Severe Acute Respiratory Syndrome	
SARS-like	Infectious respiratory disease	
Scenario	Description of an incident in terms of background, occurrence and the course of a incident, including response and other related processes of relevance	In PULSE we consider two Scenarios: SARS-like epidemics and Stadium crush-like incident
SOP	Standard Operational Procedures	SOPs may have different levels of detail: e.g. Policy, Actor/Activity tables, Procedures
Tactical Preparedness sub-phase	Activities that prepare the response to a specific adverse event; the sub-phase starts when the situation that may generate the event is announced and ends when the event happens or the situation is no more in place. Lesson learning after the end of the response phase are included in the Tactical Preparedness sub-phase.	In Stadium Scenario the sub-phase may start when the authorization for the concert is asked.
Tool	Any helping software instrument, including input/output interfaces with users or other Tools or Systems (mostly software). A Tool may use PULSE Models. A software Tool may also be identified with a set of functionalities.	PULSE Platform includes 7 Tools.
Use Case	A sample materialization of a scenario quantitatively described, including hazardous event or attack event lines, organizations involved, response procedures, numbers and classes of victims, responder and health resources etc.	
USMAF	USMAF (Uffici di Sanità Marittima, Aerea e di frontiera), reporting to Ministry of Health in Italy	



WHO	World Health Organization	
WP	Work Package of the PULSE Project	



2 Introduction

2.1 Purpose and scope of the document

This document is a covering document to support the software delivery D3.2 – Scenario generator, providing high level details on the architecture and functionalities on which the software component has been developed.

The scope of deliverable D3.2 covers task T3.4 of PULSE WP3 but is strongly related to the other modeling tasks of WP3, which have been addressed in deliverable D3.1 – Context models.

2.2 Structure of the document

This document is structured into the following sections.

- Section 3 illustrates the general architecture of the scenario generation components, highlighting its role in the PULSE Platform and with respect to the other tasks and deliverables. Moreover, the general client-server architecture for the software delivery is reviewed.
- Section 4 is focused on the stadium crush scenario generation, including a general description, its functionalities and implementation details.
- Section 5 is focused on the SARS-like epidemic scenario generation, including a general description, its functionalities and implementation details.
- Section 6 provides a list of underlying 3rd party libraries used by the software component and license summaries of these components.
- Section 7 addresses concluding remarks and future work.
- Annex A reviews the software attachments of the present deliverable.



3 General architecture

3.1 Relation to the overall PULSE platform

The purpose of this document is to support the delivery of the PULSE prototypes related to the generation of the scenarios for the simulation environment.

The PULSE Platform is showcased by two basic scenarios: a Stadium Crush scenario, representing cases of trauma incidents, and a SARS Scenario, covering the issues related to the epidemics evolution. Such scenarios have been extensively described in WP2 deliverables. The suite of WP3 models and algorithms, *fed with inputs provided by the Scenario Generator*, constitute the basic layer in the simulation engine of the corresponding WP4 tools, which also take into account the validated procedures provided by WP5. Such tools will be integrated in the final year of the project (WP6) to complete the PULSE Decision Support System (DSS).

3.2 Features overview in the context of the other tasks/deliverables

As extensively illustrated in D3.1, the implementation of the models in PULSE follows two distinct approaches for the modelling of individuals in the two following scenarios, which have been extensively described in D2.2 – Use Case Specifications:

- Stadium Crush scenario (trauma incidents): the patient is modelled by means of continuous physiological variables; the state of the system can assume infinitely many values. As a consequence, the evolution of the patient is governed by sets of Ordinary Differential Equations (ODEs) that determine the continuous “trajectory” of the patient in the space defined by the physiological variables.
- SARS scenario (epidemics event): each individual can assume one over 4 possible states: Susceptible, Exposed, Infectious, Recovered. The epidemics evolution is based on a deterministic and spatially distributed mean-value model; the rate at which the population of each terrain cell changes its state within the next time step depends on the state of the neighbors.

In general, the scenario generator software modules help build the “first frame” of the



evolution of the synthetic simulated environment. However, due to the deep difference in modeling the incidents in the two cases, the architecture of the present deliverable covers the generation of the two basic scenarios in different ways:

- In the stadium crush scenario, the focus is on the generation of the incident (number of interested people, lesions reported), based on which the healthcare response will be provided by the system and hospitals involved in the interested area. The generated data constitute an initial dataset for the simulation of models described in T3.1, T3.2 and T3.3 regarding the evolution of physiology of the individuals, of the healthcare and of the facilities involved. A webservice has been implemented, allowing for the generation of the environment parameters and of the individuals affected in a stadium incident, with a number of lesions of different severity. A library of potential complicating events or variants and their probabilities is maintained, allowing for repeated use of the basic scenario with different run developments. A client is developed in Java to test the basic functionalities of the Scenario Generation in a practical case and to retrieve some aggregate data.
- In the SARS like epidemic, the generation of the “initial picture” of the epidemics has been partially addressed in D3.1 as the initial condition of the EnSIR model (see T3.6), in terms of subpopulations of Exposed and Infected, which could be chosen as input parameters. We hereby consider the generation and the construction of the population and the connectivity matrices in some other possible areas of diffusion for SARS in Europe (in addition to the Italian context addressed in D2.2 and D3.1), with different distributions of population, geographical factors, transportation, and connectivity. Support is also given for the construction of maps when the considered area is variable by input (this will be possible in the PULSE integrated platform). On the server-side, the EnSIR model (described in D3.1) has been updated in order to be able to upload different maps. The results are exported in terms of spreadsheets with a given format. The data are read by the EnSIR model/tool (see D.3.6 and D.4.7) with the aim of completing the definition of the initial picture of the epidemics.

Figure1 and Figure 2 highlight the role of the Scenario Generation with respect to the other WP3 tasks, in relation to the two scenarios (Stadium and SARS) and to the two deliverables, respectively.

Finally, it has to be mentioned that the Scenario Generator has a major role in the preparedness phase. In the phase of response to a real crisis, anyway, these modules can be also be of help in generating information integrating with data coming from the field, typically when there is not enough information to feed the models and the tools.

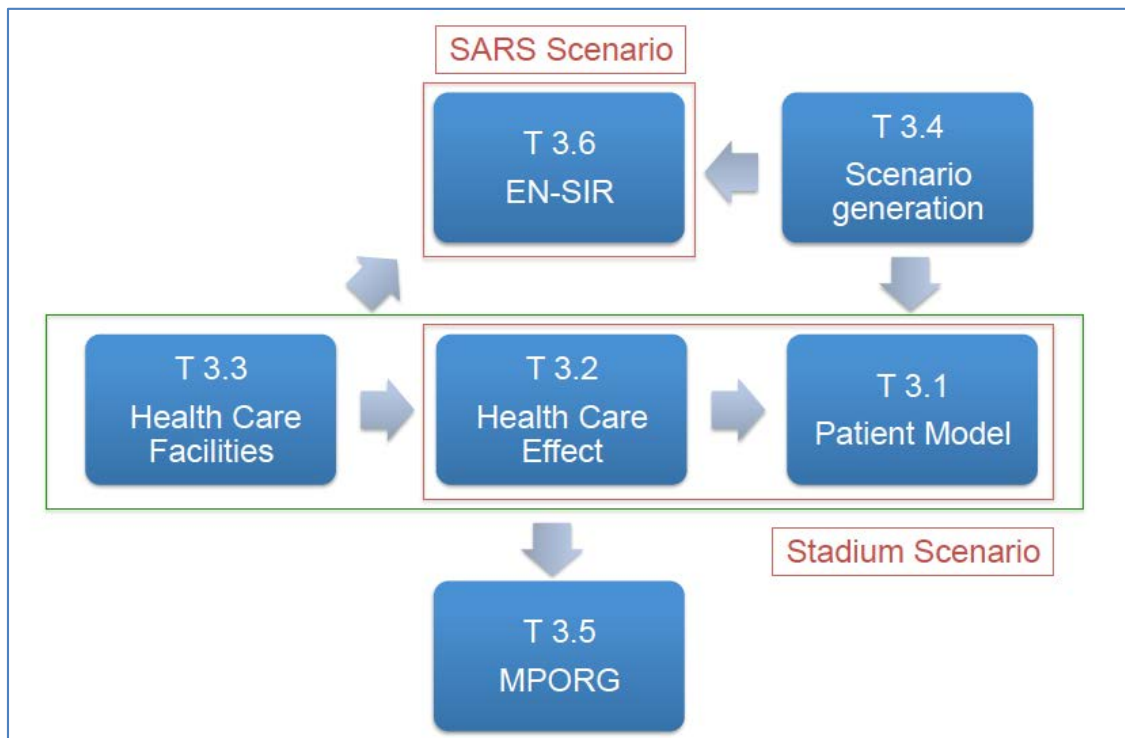


Figure1: WP3 tasks in relation to the scenarios

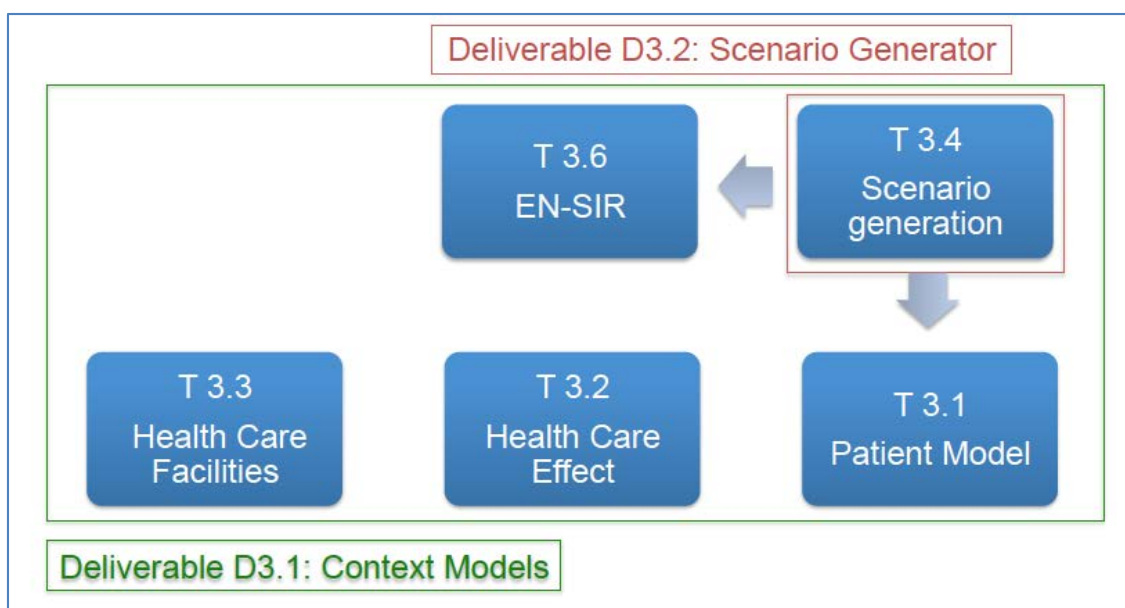


Figure 2: WP3 tasks in relation to the deliverables



3.3 Software delivery

The WP3 models and the scenario generator are implemented by means of webservices on the LAMP (Linux-Apache-MySQL-PHP) server located at the Biomathematics Laboratory of CNR-IASI at the UCSC Gemelli hospital, and constitute the core of the software prototypes.

The architecture at the basis of a webservice is formed by a *server* and one or more *clients*. The client knows the functionalities offered by a server by means of the *Web Services Description Language* file (wsdl), which is located at the web address: <http://biomat1.iasi.cnr.it/webservices/pulse/M18/webservice.wsdl>.

The webservices are implemented in php language, and communicate with the clients according to the SOAP protocol, which provides a basic messaging framework for web services. In such a way, the client is able to send requests even in another programming language (i.e. Java) with respect to the server, though respecting the SOAP specifications. Both the requests and the response are exchanged in XML language (Figure 3).

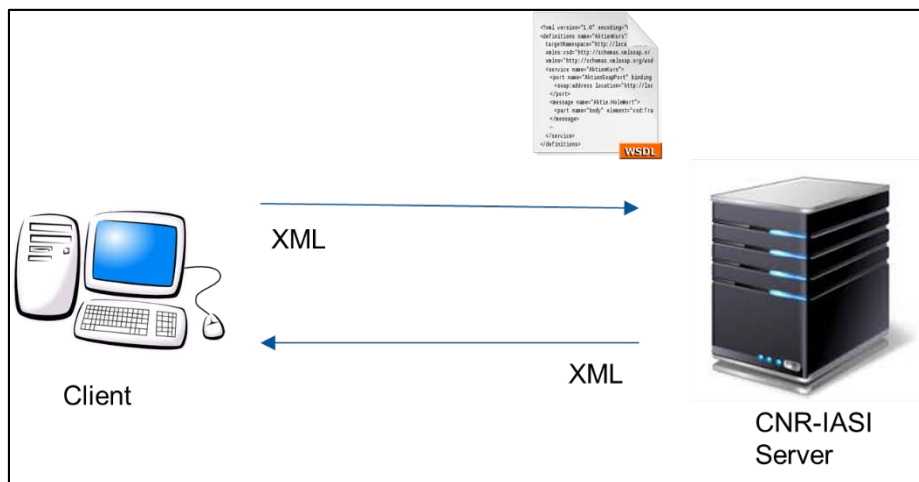


Figure 3: Scheme of the architecture used for the service



4 Stadium crush scenario generation

4.1 Description and functionalities

From an input/output perspective, the scenario generation for the stadium crush can be conceptually considered as a function taking some scenarios as inputs and returning a desired output.

INPUT

1. Number of bystanders `num_bystanders`, natural number;
2. Event type `event_type`, value in `event_set` (natural number);
3. Event size `event_size` in $[0,1]$: 0=no affected, 1=all the bystanders are affected;
4. Event longitude `event_long` [deg];
5. Event latitude `event_lat` [deg];
6. Event dimension `event_dim`[m];
7. Event minimum latency `event_min_latency` [h];

OUTPUT

1. A set `affected_population` (of cardinality `num_affected`, natural number), whose elements are “virtual patients” characterized by individual longitude, latitude, latency, health status and lesions (with different levels of severity).

Some comments are provided to illustrate the meaning of the input quantities and the behavior of the module:

- The parameter `num_bystanders` represents the number of people in the interested area.
- The parameter `event_type` (fixed in the case of stadium crush) is a degree of freedom that (in future implementations) will allow discriminating among different possible sub-scenarios.
- The parameter `event_size` affects the (average) percentage of the affected people with respect to the number of bystanders.
- The parameter `event_dim` is the dimension (typically the radius) of the area of the event, in which the population of the affected people is generated.
- The event has geographic coordinates (`event_lat`,`event_long`).



- The parameter `event_min_latency` is a lower bound on the time needed from the occurrence of the event to the beginning of the health care delivery.

The module returns a population `affected_population`, composed of `num_affected` individuals (patients). Each affected individual is characterized by:

- Individual spatial coordinates in terms of horizontal/vertical displacements [m] with respect to the event coordinates.
- An individual latency representing the time (from the event) at which the affected person is detected in the field and can be taken care of by the Health Care System.
- A health status represented in terms of values and rates of variations of 10 (normalized) physiological variables (0 is the minimum or deadly and 1 is the maximum or perfectly healthy value), based on the ABCDE paradigm:
 - A1: airway patency (intact, at risk, partially obstructed, or completely obstructed);
 - B1: respiratory rate and drive;
 - B2: tidal volume and mechanics;
 - B3: oxygen saturation and transport;
 - C1: heart pump function;
 - C2: circulation filling and resistances;
 - D1: central nervous System Function (Glasgow Coma Scale, GCS);
 - D2: seizures;
 - D3: cholinergic activity;
 - E1: exposure, hypothermia, burns.
- A set of lesions of different gravity (in a scale from 0 to 1) in a subset of 6 areas (reported in Table 2 with the corresponding estimated probabilities of occurrence in the stadium scenario).

Table 2-Estimated probability of each lesion in the stadium crush scenario

Scenario/lesions probability	Head/Neck	Face	Chest	Abdomen	Extremities	External
Stadium crush	20%	10%	80%	30%	40%	20%

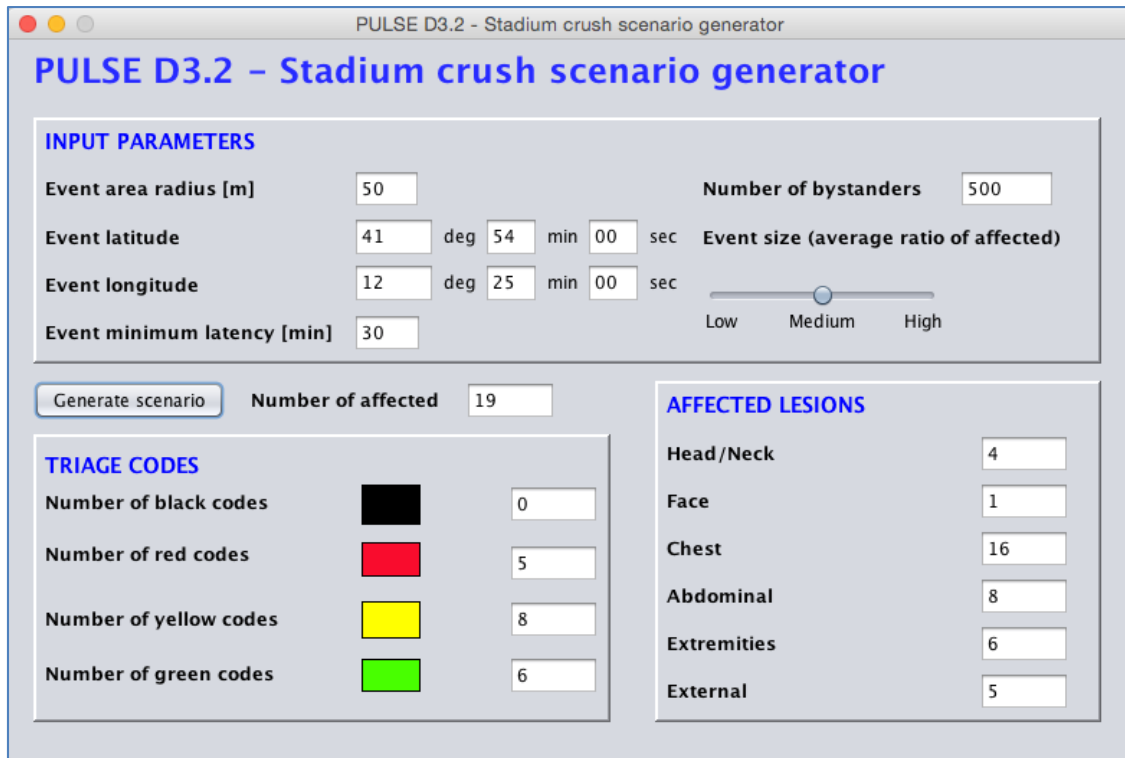


An important feature of the module is that it involves randomization: given the same set of input parameters, different runs of the algorithm generate different scenarios in terms of number of affected and their geographic position, physiological states, number of lesions occurred, etc.

4.2 Implementation

On the clientside, a synthetic Graphical User Interface (GUI) has been designed for testing the connection to the webservices and to perform basic tests on a subset of functionalities for the scenario generation. The GUI has been implemented in the Java programming language by means of the Swing components in Oracle's Netbeans IDE, according to the functionalities offered by the CNR-IASI Scenario generator webservice (ScenGen), formalized in the wsdl file at the web address: <http://biomat1.iasi.cnr.it/webservices/pulse/M18/webservice.wsdl>. The wsdl contains the following definition of complex types, elements, messages and operations for ScenGen.

In Figure 4, a screenshot of the Java GUI is shown. The user can insert the geographic coordinates of the event (positive values for North latitude and West longitudes) and a radial dimension (in metres), its size (in a scale low/medium/high, corresponding to different average ratios of affected), the number of bystanders and the minimum latency (in minutes). After calling the webservices, the GUI returns the number of affected people and reports some aggregate data about the lesions reported and the triage codes of the affected persons. The triage is performed by means of a separate webservice (StatScoring), mentioned also in D3.1 under the name "Simple Triage", which is able to return a triage code given the physiological state of an affected individual.



PULSE D3.2 - Stadium crush scenario generator

INPUT PARAMETERS

Event area radius [m] Number of bystanders

Event latitude deg min sec Event size (average ratio of affected)

Event longitude deg min sec

Event minimum latency [min] Low Medium High

Number of affected

TRIAGE CODES

Number of black codes

Number of red codes

Number of yellow codes

Number of green codes

AFFECTED LESIONS

Head/Neck

Face

Chest

Abdominal

Extremities

External

Figure 4: Java GUI for the Stadium Crush scenario generation

On the server side, a software subcomponent has been designed to support ScenGen and is called Damager. Although Damager could be invoked independently of ScenGen (hence it is exposed as a separate service), in the context of ScenGen it is called as an internal function, taking care of generating the physiological states of each individual of the affected population, given a set of lesions.

In the wsdl file, the keywords “Parameters” and “Results” are used to define the input and output of the server function, matching (in number and type) the input and output described in Section 4.1. The complex data types are highlighted in Figure 5. Analogous definitions are given for Damager and shown in Figure 6.



```
<xsd:element name="ScenGenParameters" type="tns:ScenGenParameters"/>
<xsd:element name="ScenGenResults" type="tns:ListOfGeoLocalizedPatientsWithLatencyAndLesions"/>
[...]
<complexType name="ScenGenParameters">
  <sequence>
    <element name="num_bystanders" type="xsd:integer" maxOccurs="1"/>
    <element name="event_type" type="xsd:integer" maxOccurs="1"/>
    <element name="event_size" type="xsd:double" maxOccurs="1"/>
    <element name="event_longitude" type="xsd:double" maxOccurs="1"/>
    <element name="event_latitude" type="xsd:double" maxOccurs="1"/>
    <element name="event_dimension" type="xsd:double" maxOccurs="1"/>
    <element name="event_min_latency" type="xsd:double" maxOccurs="1"/>
  </sequence>
</complexType>
<complexType name="ListOfGeoLocalizedPatientsWithLatencyAndLesions">
  <sequence>
    <element name="geo_localized_patient_with_latency_and_lesions" type="tns:GeoLocalizedPatientWithLatencyAndLesions" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="GeoLocalizedPatientWithLatencyAndLesions">
  <sequence>
    <element name="health_status" type="tns:PatientStatus" maxOccurs="1"/>
    <element name="longitude" type="xsd:double" maxOccurs="1"/>
    <element name="latitude" type="xsd:double" maxOccurs="1"/>
    <element name="latency" type="xsd:double" maxOccurs="1"/>
    <element name="lesions" type="tns:Lesions" maxOccurs="1"/>
  </sequence>
</complexType>
<complexType name="PatientStatus">
  <sequence>
    <element name="x_A" type="xsd:double" maxOccurs="1"/>
    <element name="x_B1" type="xsd:double" maxOccurs="1"/>
    <element name="x_B2" type="xsd:double" maxOccurs="1"/>
    <element name="x_B3" type="xsd:double" maxOccurs="1"/>
    [...]
    <element name="v_D3" type="xsd:double" maxOccurs="1"/>
    <element name="v_E" type="xsd:double" maxOccurs="1"/>
  </sequence>
</complexType>
<complexType name="Lesions">
  <sequence>
    <element name="lesion" type="xsd:double" maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

Figure 5: ScenGen definitions



```
<xsd:element name="damagerParameters" type="tns:DamagerParameters"/>
<xsd:element name="damagerResults" type="tns:DamagerResults"/>
[...]
<complexType name="DamagerParameters">
  <sequence>
    <element name="event_type" type="xsd:integer" maxOccurs="1"/>
    <element name="patients_lesions" type="tns:ListOfLesions" maxOccurs="1"/>
  </sequence>
</complexType>
<complexType name="DamagerResults">
  <sequence>
    <element name="pop_status" type="tns:PopulationStatus" maxOccurs="1"/>
  </sequence>
</complexType>
<complexType name="ListOfLesions">
  <sequence>
    <element name="pat_lesions" type="tns:Lesions" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="PopulationStatus">
  <sequence>
    <element name="patient_status" type="tns:PatientStatus" maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

Figure 6: Damager definitions

The request and response messages exploit Parameters (inputs) and Results (output) respectively:

```
<message name="ScenGenRequest">
  <part name="scengenpar" element="tns:ScenGenParameters"/>
</message>
<message name="ScenGenResponse">
  <part name="scengenres" element="tns:ScenGenResults"/>
</message>
<message name="damagerRequest">
  <part name="damagerpar" element="tns:damagerParameters"/>
</message>
<message name="damagerResponse">
  <part name="damagerres" element="tns:damagerResults"/>
</message>
```

Figure 7: ScenGen and Damager request and response messages

The operations “ScenGen” and “Damager” match the names of the corresponding php function files on the webserver:



```
<operation name="ScenGen">
<soap:operation soapAction="ScenGen"/>
<input>
<soap:body use="literal"/>
</input>
<output>
<soap:body use="literal"/>
</output>
</operation>
<operation name="Damager">
<soap:operation soapAction="Damager"/>
<input>
<soap:body use="literal"/>
</input>
<output>
<soap:body use="literal"/>
</output>
</operation>
```

Figure 8: ScenGen and Damager operations

The java methods “scenGen” and “damager” are defined in the client code to invoke the webservices. All the classes are created in agreement with the types defined in the wsdl.

```
private static ListOfGeoLocalizedPatientsWithLatencyAndLesions
scenGen(ScenGenParameters scenenpar)
{
    PULSEWebServices service = new PULSEWebServices();
    PULSEModelsPortType port = service.getPULSEModelsPort();
    return port.scenGen(scenenpar);
}

private static DamagerResults damager(DamagerParameters
damagerpar)
{
    PULSEWebServices service = new PULSEWebServices();
    PULSEModelsPortType port = service.getPULSEModelsPort();
    return port.damager(damagerpar);
}
```

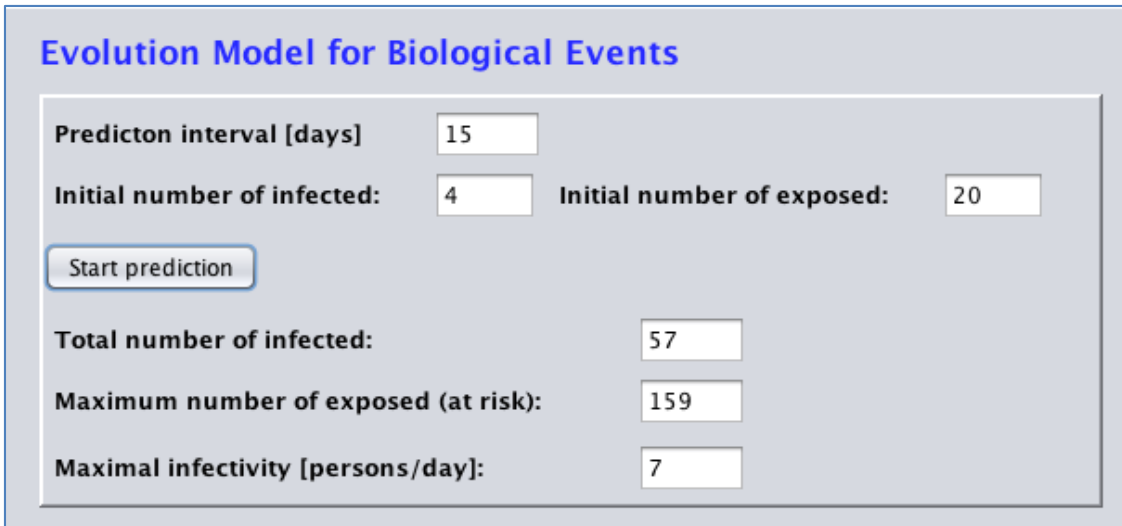
Figure 9: ScenGen and Damager java methods definitions

5 SARS epidemics scenario generation

5.1 Description and functionalities

The Scenario Generator for the SARS scenario helps define the initial inputs and parameters for the ENSIR model (see D3.1 and T.3.6 for further details). We briefly recall that ENSIR predicts the expected evolution of an epidemic in terms of susceptible, exposed, infected and recovered subpopulations.

The ENSIR model prototype delivered in D3.1 already allows inserting a value for the initial count of the subpopulations of exposed and infected:



Evolution Model for Biological Events

Prediction interval [days]	15	
Initial number of infected:	4	Initial number of exposed:
		20
<input type="button" value="Start prediction"/>		
Total number of infected:	57	
Maximum number of exposed (at risk):	159	
Maximal infectivity [persons/day]:	7	

Figure 10: ENSIR prototype client interface

ENSIR performs the prediction of the spatial-temporal evolution of an epidemic, taking into account different factors, allowing for disease spread with different rates depending on the geographic, social and logistic characteristics of the interested area. The main geographic and social factors taken into account are:

- the number/density of population in the interested area;
- the 'natural' connectivity of population, which may depend on the geography of the area;
- the connectivity by means of transportation, daily flights, etc.

The area of interest is partitioned according to a grid of dimensions [num_rows x num_columns], for a total number of cells equal to $\text{num_cells} = \text{num_rows} * \text{num_columns}$.



Three matrices are allocated to manage the information regarding population and connectivity in the ENSIR model:

1. **M_pop**: Initial Population Matrix, with dimension [num_rows x num_columns], where each entry is an integer value counting the population of the corresponding cell.
2. **M_conn**: Rate of 'natural' connectivity between cells, with dimension [num_cells x num_cells], where each entry is a nonnegative real number; $M_conn(i,j)$ is a function of the physical distance between the centres of cell i and cell j and of the populations of cells i and j . $M_conn(i,i)$ is set equal to a strictly positive number to take into account the internal infectivity (within the same cell).
3. **M_vol**: Connectivity Matrix by the daily transportations (flights or high-speed train connections) between cells, dimension [num_cells x num_cells], where each entry is a nonnegative integer.

In the current implementation, M_pop and M_vol can be independently chosen, while M_conn is computed on the basis of M_pop and on the physical distance among cells. When M_vol is not available, the connectivity is estimated on the basis of the population matrix M_pop , assuming that “fast” transportations are more frequent between cells of higher population density.

M_conn and M_vol both contribute to computing the overall connectivity matrix, whose entries affect the rate of spatial-temporal transmission of the epidemics between the corresponding cells.

The ENSIR version illustrated in D3.1 “by default” operated on a fixed scenario (including an area centered on northern and central Italy). In the current version, a script of the scenario generation is incorporated in the service to take into account different maps. In addition to the Italian map, three additional regions in Central Europe are considered. Figure 11, Figure 12 and Figure 13 show the maps of the Moldova, Transilvania and Muntenia regions respectively, with their geographic coordinates.

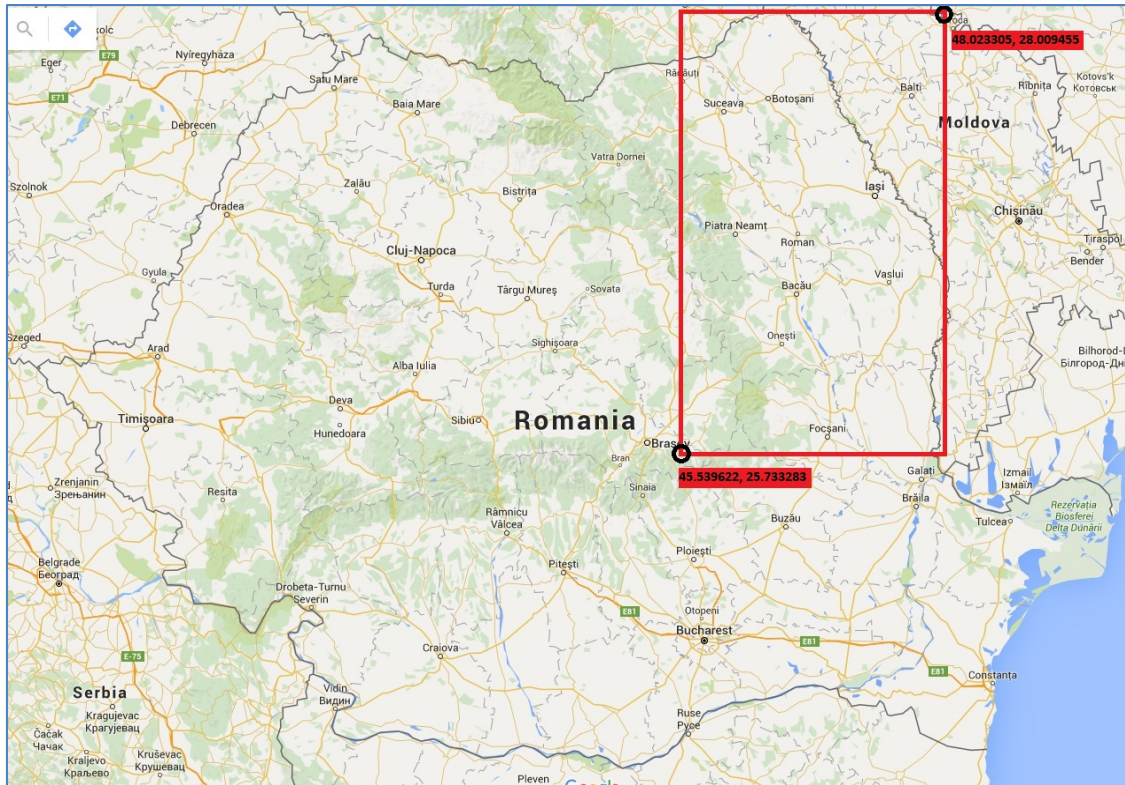


Figure 11: Moldova region

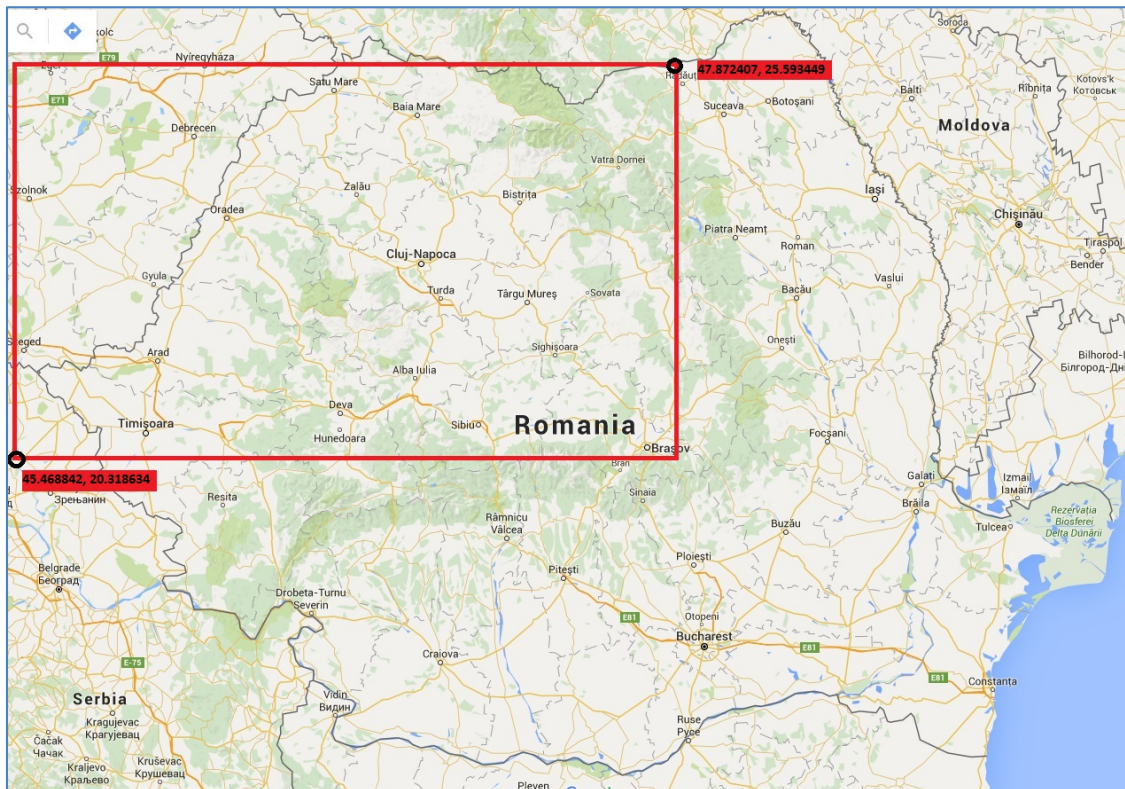


Figure 12: Transilvania region

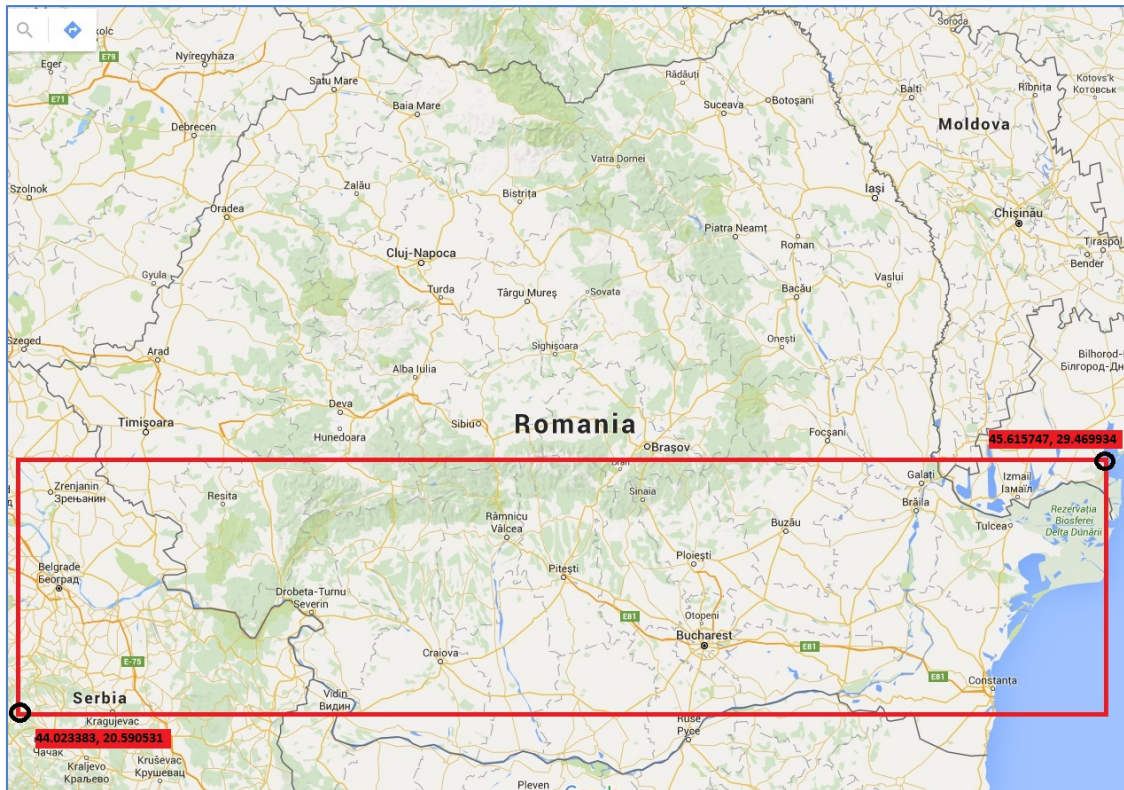


Figure 13: Muntenia region

Figure 14, Figure 15 and Figure 16 show the maps of the same regions, covered by grids with different numbers of rows and columns. On top of each cell, an estimated value of the population is provided. The resulting tables constitute the M_pop matrices for the three regions considered.

As shown in the following section, the procedure can be repeated for an arbitrary number of maps and the automatic computation of the matrices for the scenario generation (by means of existing available web services) will be addressed in a future version of the system.

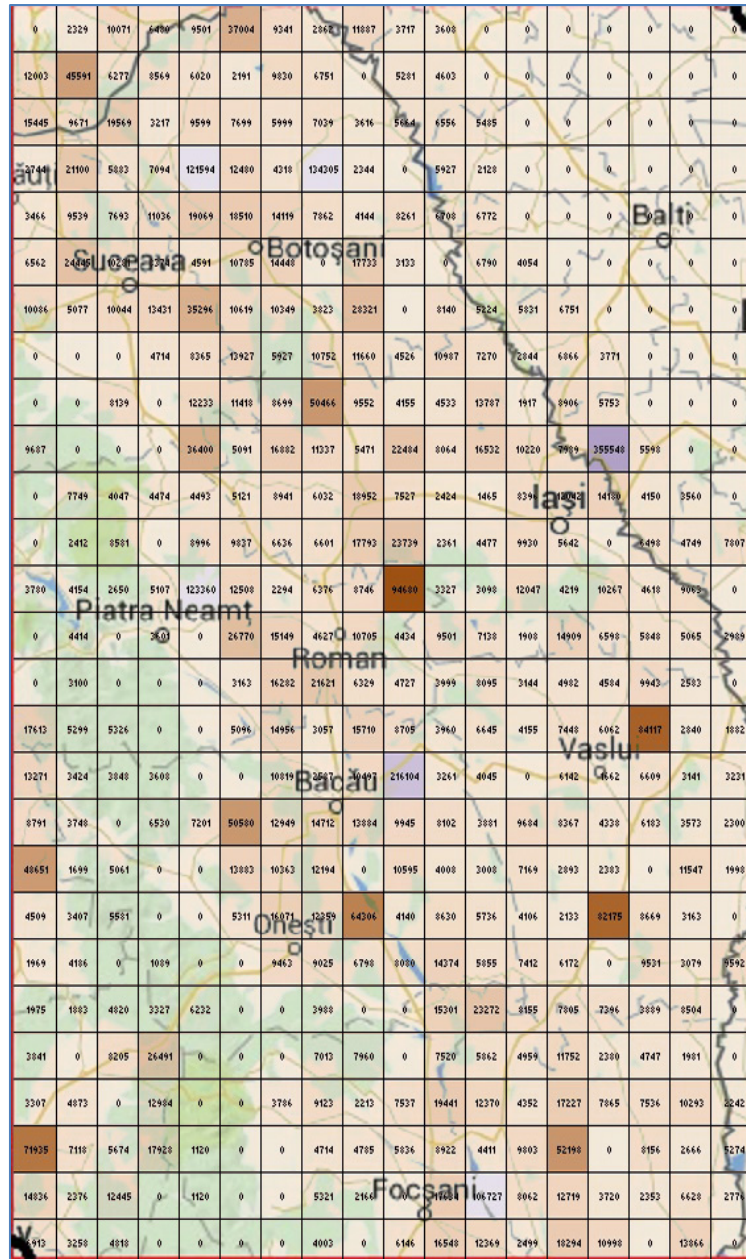


Figure 14: Moldova gridded map with population values

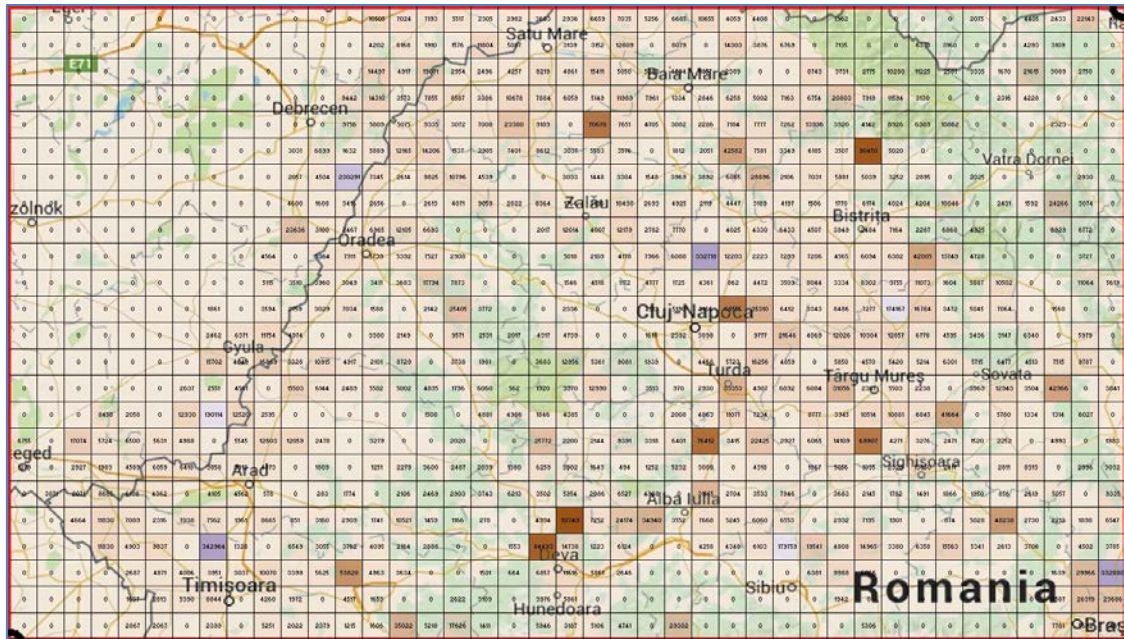


Figure 15: Transilvania gridded map with population values

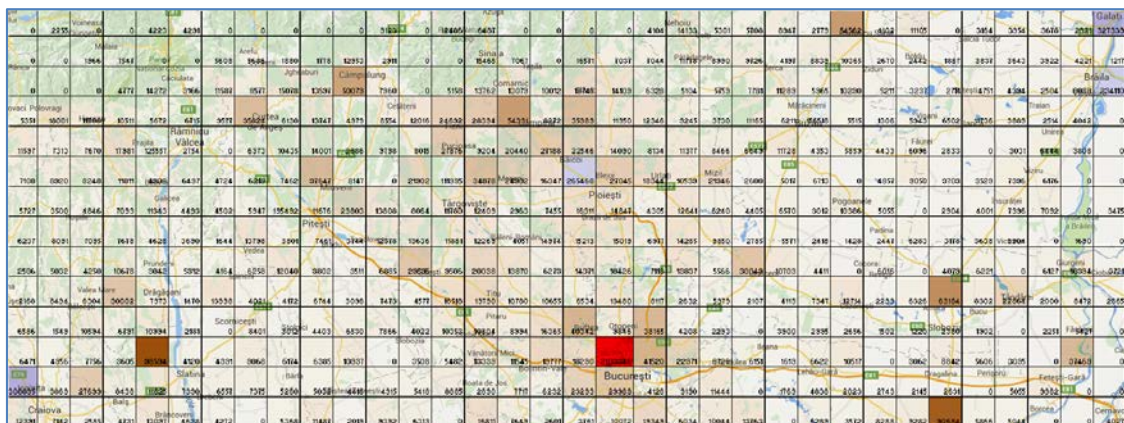


Figure 16: Muntania gridded map with population values

5.2 Implementation

Different to the case of Stadium Crush, the scenario generation for the SARS scenario has not been implemented as a separate service; instead, the implementation of the ENSIR model delivered in D3.1 has been updated to integrate the input from the user and variable maps. In the following, the main modifications to the ENSIR architecture with respect to the formal description given in D3.1 are described. A complete description of the ENSIR webservice is out of the scope of the



present deliverable (see D3.1 and D4.7 for further details).

Figure 17 illustrates the structure of the function EnSIR.php. The data retrieved from the input mask in Figure 10 (in particular the variables “exposeds_data” and “infectives_data”) are used to build the sparse data structures “E_start” and “I_start”, containing the subpopulation of initial exposed and infected for each cell of the grid. Then, the utility function ensir_core.php is called.

```
<?php
function EnSIR($xml_input) {
    require_once 'ensir_core.php';

    /*****
     * READ PARAMETERS FROM XML
     *****/

    $asset_vec = $xml_input->assets->asset;
    $T_fin = $xml_input->t_fin;
    $delta_T = $xml_input->delta_t;
    $event_type = $xml_input->event_type;
    $exposeds_data = $xml_input->exposeds_at_start->exposeds_data;
    $infectives_data = $xml_input->infectives_at_start->infectives_data;

    $E_start = array();
    for ($i=0; $i<count($exposeds_data); $i++) {
        $E_start[$i][0] = $exposeds_data->row;
        $E_start[$i][1] = $exposeds_data->col;
        $E_start[$i][2] = max(0,$exposeds_data->exposeds);}
    $I_start = array();
    for ($i=0; $i<count($infectives_data); $i++) {
        $I_start[$i][0] = $infectives_data->row;
        $I_start[$i][1] = $infectives_data->col;
        $I_start[$i][2] = max(0,$infectives_data->infectives);}

    $res=
    ensir_core($E_start,$I_start,$asset_vec,$T_fin,$delta_T,$event_type);
    [...]
?>
```

Figure 17: ENSIR function implementation



Figure 18 describes the structure of the file `ensir_core.php`. A utility function `readDataAndHeaders.php` has been incorporated to read data from csv (comma-separated values) files. This allows building the matrices `M_pop` and `M_vol` to complete the scenario generation. An example of `M_pop.csv` file is given in Figure 19 for the Moldova region.

After the definition of the population/connectivity maps and the initial conditions for the subpopulations, the implementation of the dynamic epidemics model follows the description given in D3.1.

```
<?php
function
ensir_core($E_start,$I_start,$asset_vec,$T_fin,$delta_T,$event_type)
{
    $all_data = readDataAndHeaders('data_ensir/M_pop.csv');
    $M_pop = $all_data[2];

    $all_data = readDataAndHeaders('data_ensir/M_vol.csv');
    $M_vol = $all_data[2];

    [...] // EnSIR model (see D3.1)
}
```

Figure 18: ENSIR_core function implementation

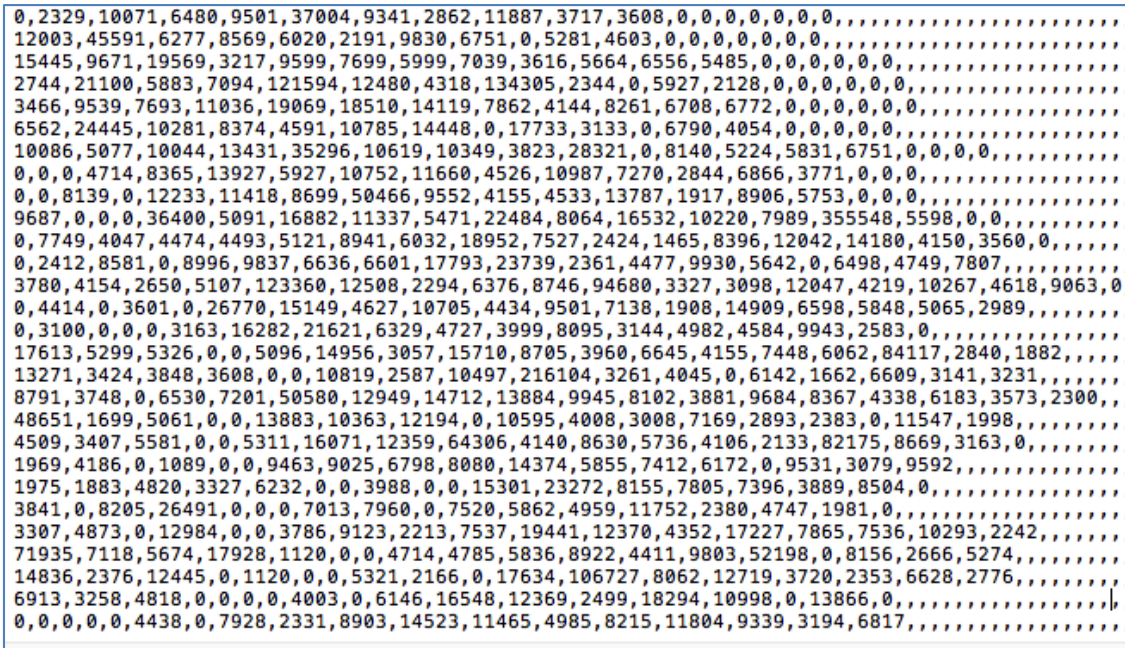


Figure 19: M_pop.csv (Moldova region)

A further extension of the current service will allow consideration of maps that are chosen by input and coded (in real-time) into csv files in the format readable by the files `ensir_core.php` and `readDataAndHeaders.php`. A number of web sites and services are available for the retrieval of information about population and transportation data. Two examples (for population data and for flight data, respectively) are the following:

- <http://www.wolframalpha.com/input/?i=53.3478%C2%B0+N%2C+6.2597%C2%B0+W> (the link retrieves data of the Dublin region, based on the OpenStreetMap engine).
- <http://openflights.org/data.html> (Open Flights).



6 3rd Party libraries and licenses

Below is a list of third party libraries/frameworks used and the licenses under which they are distributed.

Table 3 - 3rd party libraries and licenses

Product	Version	Vendor	License
Java SE Development Kit	8	Oracle	Oracle Binary Code License Agreement
Beans Binding	1.2.1	Java.net	GNU LESSER GENERAL PUBLIC LICENSE
Swing Layout Extensions	1.0.4	Java.net	GNU LESSER GENERAL PUBLIC LICENSE
Linux	3.2	Free Software Foundation	GNU GENERAL PUBLIC LICENSE
Apache	2.4	Apache Software Foundation	Apache License
PHP	5	None (free software)	PHP License



7 Concluding remarks and future work

The prototypes that are the object of the present deliverable cover the set of functionalities involving the scenario generation in the two cases of Stadium-crush incident and SARS-like epidemic, which have been described extensively in the WP2 deliverables.

The Scenario Generator provides correct inputs for the execution of the model functions (see D3.1 “Context Models”), constituting also the basic modeling layer of the prototype WP4 deliverables on Tools. We mention that the current implementation of the model webservices includes refinements (with respect to the one described in D3.1) to take into account input from the concurrent deliverables.

In the final year of the project (M19-M30), the processes of integration and validation/trials in WP6-WP7 will provide further occasion for implementation refinement and parameter tuning, as a consequence of the lessons learnt and of the application of the PULSE platform in realistic conditions for the considered scenarios.



A. Attachments

The software attachments of the present document are delivered as an archive “PULSE_D32”, including two folders:

- PULSE_D32_stadium: it includes the Java Archive “PULSE_D32_stadium.jar” for the test of the PULSE Scenario Generation webservice (exposed at the address <http://biomat1.iasi.cnr.it/webservices/pulse/M18>). The prototype is executable on any computer with a Java Virtual Machine (JVM) installed on it, by typing

```
java -jar "PULSE_D32_stadium.jar"
```

from the command line, after entering the folder containing the java archive. Internet connection is required for the client to allow the access to the web services and the correct retrieval of the results. In absence of Internet connectivity, the client reports that the results are not available (N/A).

- PULSE_D32_SARS: it includes three maps that are read as an input by the EnSIR webservice EnSIR.php for the definition of the initial picture of the epidemics. The numeric parts of the maps are provided in tables included in.csv (comma-separated values) files, which are readable by almost all spreadsheets and database management systems. The .jpg corresponding georeferenced maps are also enclosed.