



*Platform for European Medical Support  
During Major Emergencies*

## **D4.1 Decision Support and Validation Tool**





## ***PULSE***

### ***Platform for European Medical Support during major emergencies***

#### **WP4 - Tools**

#### **Deliverable D4.1 - Decision support validation tool**

**30/11/2015**



Contractual Delivery Date:	Actual Delivery Date:	Deliverable Type*-Security**:
30/11/2015	30/11/2015	R - PU
607799	PULSE	Platform for European Medical Support during major emergencies

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

Responsible:	Organisation:	Contributing WP:
Francesco Malmignati	Selex ES	WP4

Authors (organisation)
Francesco Malmignati (SES), Antonio Di Novi (SES), Karl Chadwick (SKY), Paul Kiernan (SKY)

Abstract:
The purpose of this document is to provide a general overview on the PULSE architecture and to describe the functionalities provided by the Decision Support Validation tool (DSVT). A detailed DSVT component architecture is provided together with a description of all the interactions occurring between this component and the other PULSE tools. A separated section is dedicated to the Authentication server that is in charge of managing all the security aspects of the PULSE platform.

Keywords:
Decision, Support, Validation, Security, Service, Preparedness, Response, Simulation, Mobile, Applications.



## Revisions:

Revision	Date	Description	Author (Organisation)
0.1	01/10/15	Update of TOC	Francesco Malmignati (SES)
0.2	15/10/15	Architecture description	Francesco Malmignati (SES)
0.3	20/10/15	Authentication server description	Francesco Malmignati (SES)
0.4	26/10/15	Added contribution regarding DSVT component architecture and DSVT functionalities	Antonio Di Novi (SES)
0.5	30/10/15	Update of the component architecture and definition of the relations with the WP2 use cases	Antonio Di Novi (SES), Francesco Malmignati (SES)
0.6	02/11/15	Integration of the contribution regarding the technologies used	Francesco Malmignati (SES), Antonio Di Novi (SES)
0.7	03/11/15	Internal SES revision	Francesco Malmignati (SES)
1.0	03/11/15	Internal Review version	Francesco Malmignati (SES)
1.1	26/11/15	Addition of Mobile Extensions	Karl Chadwick (Skytek) Paul Kiernan (Skytek)
1.2	30/11/15	Final released version	Francesco Malmignati (SES)



## Table of contents

Revisions: .....	3
List of figures .....	6
List of Tables .....	7
List of acronyms.....	8
1 Executive Summary .....	9
2 Introduction.....	10
2.1 Scope of the Document .....	10
2.2 Structure of the Document.....	10
2.3 Relation with other Deliverables .....	10
3 Pulse platform .....	11
3.1 High level architecture .....	11
4 Tools description .....	14
4.1 Decision Support and Validation tool .....	14
4.1.1 Objective.....	14
4.1.2 Functionalities .....	14
4.1.2.1 Operational overview .....	14
4.1.2.2 Screen sharing .....	18
4.1.2.3 Weak signals classification .....	19
4.1.2.4 Simulation.....	20
4.1.2.5 Recommendation and Alerting system .....	24
4.1.2.6 Post-crisis evaluation .....	26
4.1.2.7 Resource Management.....	27
4.1.2.8 Preparedness support.....	27
4.1.3 Relation with WP2 Use Cases .....	28
4.1.3.1 SARS-like scenario .....	28
4.1.3.2 Stadium scenario.....	30
4.2 Authentication Server .....	31
5 Tools architecture.....	34
5.1 Decision Support and Validation Tool .....	34
5.1.1 DSVT Component Architecture.....	34
5.1.1.1 DSVT Core .....	35
5.1.1.1.1 DSVT Manager.....	35
5.1.1.2 Message Broker .....	37
5.1.1.3 Simulation.....	38
5.1.1.4 Rule Engine .....	39



5.1.2	DSVT relation to overall PULSE architecture .....	40
5.2	Authentication Server .....	40
5.2.1	Authentication Server Component Architecture .....	40
5.2.1.1	Web Container .....	41
5.2.1.2	Directory Server .....	41
6	Components Technologies .....	42
6.1	Decision Support and Validation tool .....	42
6.1.1	List of core technologies .....	42
6.1.2	3 <sup>rd</sup> Party libraries and licenses .....	43
6.2	Authorization Server .....	44
6.2.1	List of core technologies .....	44
6.2.2	3 <sup>rd</sup> Party libraries and licenses .....	44
7	Mobile Extensions .....	45
7.1	The Smartphone App.....	45
7.1.1	Login .....	46
7.1.2	Geolocation .....	47
7.1.3	Network .....	47
7.1.4	Tasks .....	47
7.1.5	Data Record .....	49
7.1.6	Privacy & Security .....	50
7.1.6.1	Identification .....	50
7.1.6.2	Recorded Media .....	50
7.1.7	List of core technologies .....	50
7.1.8	3 <sup>rd</sup> Party libraries and licenses .....	51
7.2	Public Web Forms.....	51
7.2.1	Missing Persons form.....	52
7.2.2	Airline Passenger Information form .....	53
7.2.3	Output.....	54
7.2.4	List of core technologies .....	57
7.2.5	3 <sup>rd</sup> Party libraries and licenses .....	57
	References .....	58



## List of figures

Figure 1: PULSE Architecture – Component Diagram .....	11
Figure 2: High level architecture .....	12
Figure 3: Stadium Operational Overview .....	15
Figure 4: SARS scenario Operational Overview.....	16
Figure 5: Data table .....	17
Figure 6: Data details .....	17
Figure 7: Dashboard with Navigator map .....	18
Figure 8: Screen sharing functionality.....	19
Figure 9: Weak signals classification .....	19
Figure 10: Epidemic evolution .....	21
Figure 11: Simulation results.....	22
Figure 12: Dispatchment simulation results.....	23
Figure 13: Simulation trace log .....	24
Figure 14: Recommendations .....	25
Figure 15: Post crisis evaluation feedback .....	27
Figure 16: Preparedness panel.....	28
Figure 17: Authentication steps.....	32
Figure 18: Access Token validation .....	33
Figure 19: DSVT architecture.....	34
Figure 20: DSVT Manager .....	36
Figure 21: DSVT GUI .....	37
Figure 22: Authentication Server Architecture .....	41
Figure 23: Smartphone App Architecture.....	45
Figure 24: Login.....	46
Figure 25: Tasks.....	48
Figure 26: Triage Reporting .....	49
Figure 27: Missing Person Form .....	52
Figure 28: Airline Passenger Form .....	53
Figure 29: Sample Data Feed .....	55
Figure 30: Basic Data Browser List View .....	56
Figure 31: Data Browser Detail View .....	56



## List of Tables

Table 1 – DSVT 3rd party libraries and licenses .....	43
Table 2 – Authorization Server 3rd party libraries and licenses .....	44
Table 3 – Smartphone App 3rd party libraries and licenses.....	51
Table 4 – Web Forms 3rd party libraries and licenses .....	57





## List of acronyms

Acronym	Definition
AS	Authentication Server
DES	Discrete Event Simulation
DSVT	Decision Support and Validation tool
DoW	Description of Work
ECDC	European Centre of Disease Prevention and Control
ENSIR	ENhanced SIR
EU	European Union
GPS	Global Positioning System
GUI	Graphical User Interface
IAT	Intelligence Analysis Tool
IHR	International Health Regulations
IP	Internet Protocol
JWT	JSON Web Token
LDAP	Lightweight Directory Access Protocol
LMS	Learning Management System
LRS	Learning Record Store
LT	Logistic Tool
MIC	Medical Incident Commander
MoE	Measures of Effectiveness
PCET	Post Crisis Evaluation Tool
PMA	Posto Medico Avanzato
MPORG	Multi-Player Online Roleplaying Game
NGO	Non Governmental Organisation
REST	Representational State Transfer
SA	Smartphone application
SARS	Severe Acute Respiratory Syndrome
SGCT	Surge Capacity Generation Tool
SIR	Susceptible - Infected - Recovered
SOAP	Simple Object Access Protocol
SOP	Standard Operational Procedures
TT	Training Tools



WHO	World Health Organisation
-----	---------------------------

## 1 Executive Summary

This document is a report focused on the development of the *Decision Support and Validation tool* (DSVT), one of most important software components composing the PULSE platform architecture.

One of the main objectives of the PULSE platform is to develop a technical and operational framework that allows the platform's stakeholders (e.g. European or National Authorities) to have access to timely key data, planning and decisions that efficiently help them to manage a major healthcare crisis.

Amongst the PULSE components, the tool that is in charge of providing this "operational picture" functionality is the *Decision Support and Validation tool* (DSVT). The DSVT provides a front-end Graphical User Interface (GUI) that is directly exploited by the platform's stakeholders in order to obtain the necessary information to handle the crisis. The *Decision support and Validation tool* provides a complete set of functionalities that allow the decision makers to efficiently handle the crisis.

The DSVT assumes also an important role inside the architecture. In fact, the component resides in the platform's core and controls the communications among all the PULSE tools.

This document explains also the PULSE platform's security aspects. In particular, part of the report is dedicated to the description of another platform's component called *Authentication Server*. This tool provides a layer of protection that forces users and clients to authenticate through the server before e.g. accessing the network or invoking the web service interfaces exposed by the PULSE tools.



## 2 Introduction

### 2.1 Scope of the Document

This document is a covering document to support the software delivery of the *Decision Support Validation Tool* and mobile extensions.

It summarizes the software component delivery and provides high-level details on the architecture, technologies and underlying libraries on which the software component has been developed.

It provides also the description of the Authentication Server and the relative authentication mechanisms that have been implemented to guarantee the security of the PULSE platform.

Moreover, as an introduction to the WP4 tools, the present document provides also a high-level description of the PULSE platform architecture.

### 2.2 Structure of the Document

This document is structured into the following sections.

- High level description of the PULSE architecture
- Main functionalities of the DSVT component.
- Architecture used within the DSVT component and an overview of where the software component is located within the overall PULSE system architecture.
- Main functionalities of the Authentication Server
- Architecture used within the Authentication Server component
- Technologies used for development of the DSVT and the Authentication Server.
- List of underlying 3<sup>rd</sup> party libraries used by the DSVT and the Authentication Server and license summaries of these components.
- Mobile Extensions, 3<sup>rd</sup> party libraries used by the mobile applications and core technologies.

### 2.3 Relation with other Deliverables

Considering the nature of the DSVT, the work presented in this deliverable is related to most of the WP4 deliverables:

- D4.2 – Intelligence Analysis Tool [1]
- D4.3 – Logistic Tool [2]
- D4.4 – Surge Capacity Generation Tool [3]
- D4.6 – Post Crisis Evaluation Tool [5]
- D4.7 – Event evolution model for biological events [6]

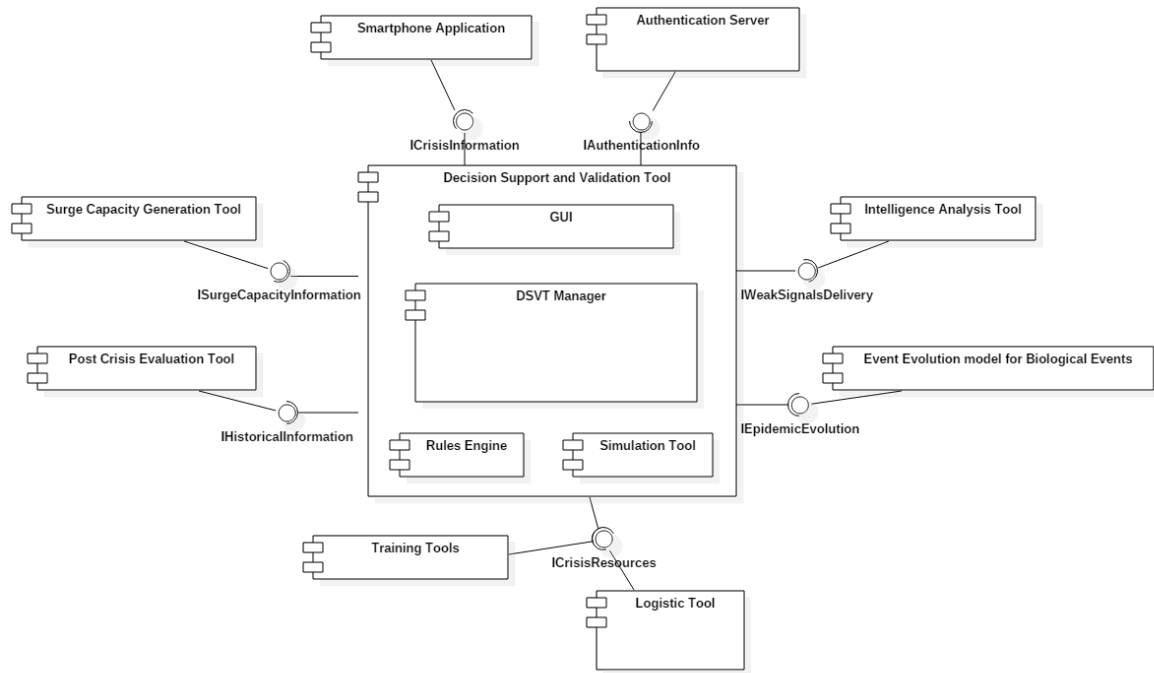
A relation is also present with two WP2 deliverables:

- D2.1 – Requirements specification [9]
- D2.2 – Use Case specification [8]

### 3 Pulse platform

#### 3.1 High level architecture

The architecture of the PULSE platform is composed of several software modules distributed on a service-based architecture.



**Figure 1: PULSE Architecture – Component Diagram**

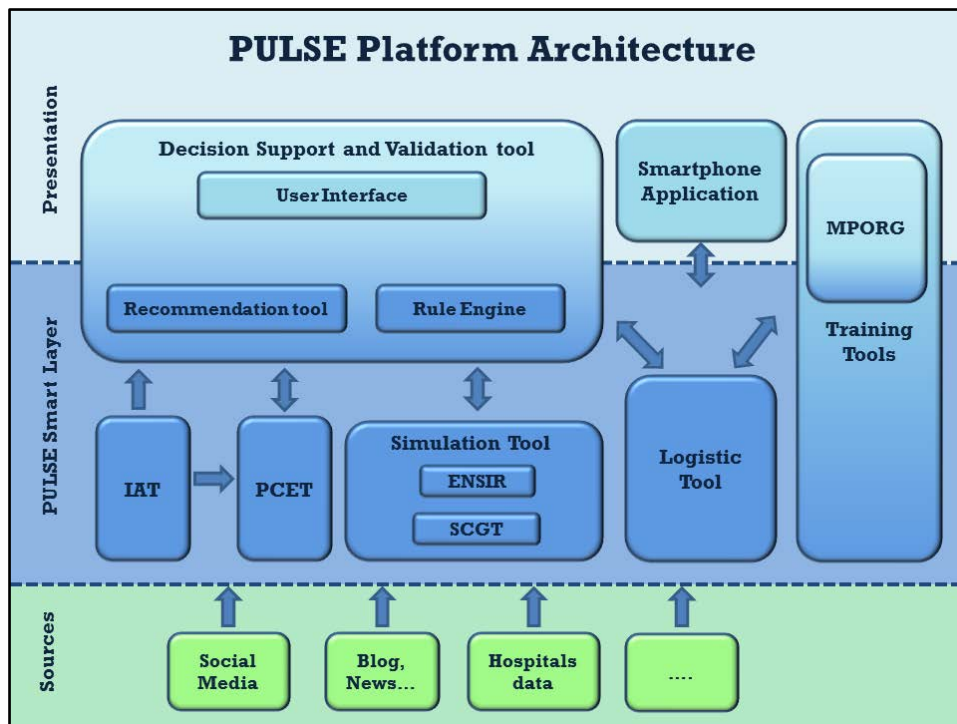
As shown in Figure 1 the core of the architecture is represented by the *Decision Support and Validation tool* (DSVT) that acts as the front-end interface as well as the communication backbone of the platform. All the other PULSE tools can (1) exploit the interfaces provided by the DSVT or (2) provide functionalities to the DSVT itself.

The PULSE platform is specifically composed of the following tools:

- **Decision Support and Validation Tool (DSVT):** as said above, it is front-end interface as well as the communication backbone of the platform. It will be deeply analysed and described in chapter 4.
- **Intelligence Analysis Tool (IAT):** as stated in the DoW, it focuses on weak signal detection to alert decision makers to the occurrence of an unusual biological event. It is described in D4.2 [1].
- **Logistic Tool (LT):** as stated in the DoW, it is used to assess the required stockpiles of any necessary equipment, medications and vaccinations. It is described in D4.3 [2].
- **Surge Capacity Generation Tool (SCGT):** it provides support for the creation of surge capacity in the event of a major health crisis. It is described in D4.4 [3].
- **Training Tools (TT):** as stated in the DoW, these tools include a MPORG

training platform for personnel involved in crisis management and a training learning management system (LMS) tailored for the emergency and health services. They are described in D4.5 [4].

- **Post Crisis Evaluation Tool (PCET):** this tool is in charge of storing and classifying all the resources (including geospatial and time references), events and decisions that have been taken during the crisis. It allows then the creation of an historical crisis report and the definition of lessons learnt. It is described in D4.6 [5].
- **Event evolution model for Biological Events (ENSIR):** this tool is the implementation of a mathematical model of epidemics evolution. It is described in D4.7 [6].
- **Smartphone application (SA):** the Android application can be used to access the PULSE platform. It is described in section 8 here.
- **Authentication Server (AS):** it is the tool that facilitates the authentication of the entities that attempts to access the platform. It is based on the *OAuth2* standard [7] that assure the security protection of all the tools composing the platform. It is described in 4.2.



**Figure 2: High level architecture**

Figure 2 shows the architecture from a multi-layers perspective. The tools have been grouped in three different layers:

- The **Presentation Layer:** composed of the *User Interface module* (part of the *DSVT*), the *Smartphone Application* and the *MPORG's* GUI, represents the graphical user interface of the PULSE platform. It gives the opportunity, to the different consumers (e.g. National Authorities, for more info see D2.2 [8]) to exploit the features provided by the platform.
- The **PULSE Smart Layer:** it is the core of the PULSE platform. It is composed



of all those tools that are able to analyse, store and elaborate the pieces of information coming from the Sources Layer and to provide enriched crisis management functionalities to the upper Presentation level.

- The **Sources Layer**: it is the bottom layer of the platform. It includes all the external services, data sources providing the medical and environmental information.



## 4 Tools description

### 4.1 Decision Support and Validation tool

#### 4.1.1 Objective

One of the main objectives of the PULSE platform is to develop a technical and operational framework that allows the platform's stakeholders (e.g. European or National Authorities) to have access to timely key data, planning and decisions that efficiently help them to manage a major healthcare crisis.

Amongst the PULSE components, the tool that is in charge of providing this “operational picture” functionality is the **Decision Support and Validation tool** (DSVT). The DSVT provides in fact a front-end Graphical User Interface (GUI) that is directly exploited by the platform's stakeholders in order to obtain the necessary information to handle the crisis.

In addition to this, the DSVT assumes also an important role inside the architecture. In fact, the component resides in the platform's core (as seen in Figure 1) and controls the communications among all the PULSE tools.

These functionalities and others are deeply described in section 4.1.2 while the component architecture will be described in section 5.1.

#### 4.1.2 Functionalities

The *Decision support and Validation tool* provides a complete set of functionalities that allow the decision makers to efficiently handle the crisis.

These functionalities can be grouped in two different collections:

- Enhanced operational picture: the DSVT provides an innovative approach to categorize and visualize the information obtained during the crisis. The functionalities in question are: *Screen sharing* (4.1.2.2), *Operational overview* (4.1.2.1) and *Weak signals classification* (0)
- Decision making suggestions: the DSVT is able to automatically create personalized suggestions to the decision makers in charge of the crisis management. The functionalities in question are: *Simulation* (4.1.2.4), *Recommendation and Alerting system* (4.1.2.5) and *Preparedness support* (4.1.2.8).
- Resource Management: the DSVT allows the management (insert, update, delete) of the set of resources available in the platform. The functionality in question is *Resource Management* (5.1.2.6).

##### 4.1.2.1 Operational overview

As said above, the DSVT provides a front-end Graphical User Interface (GUI), which is directly exploited by the platform's stakeholders in order to obtain the information related to the crisis and to have a complete overview on the incident status.

The operational overview screen is principally composed of a dashboard where the user, depending on the scenario (Stadium or SARS-like), can see different data



related to the crisis:

- As depicted in Figure 3, for the Stadium scenario the user is able to see the following information:
  - Hospitals resources
  - Wounded status (including their updated triage code)
  - Rescuers details (including their current locations and tasks)
  - Ambulances Status

Figure 3: Stadium Operational Overview

- As depicted in Figure 4, for the SARS-like scenario the user is able to see the following information:
  - Hospital resources
  - Confirmed cases
  - Probable cases
  - Weak signals



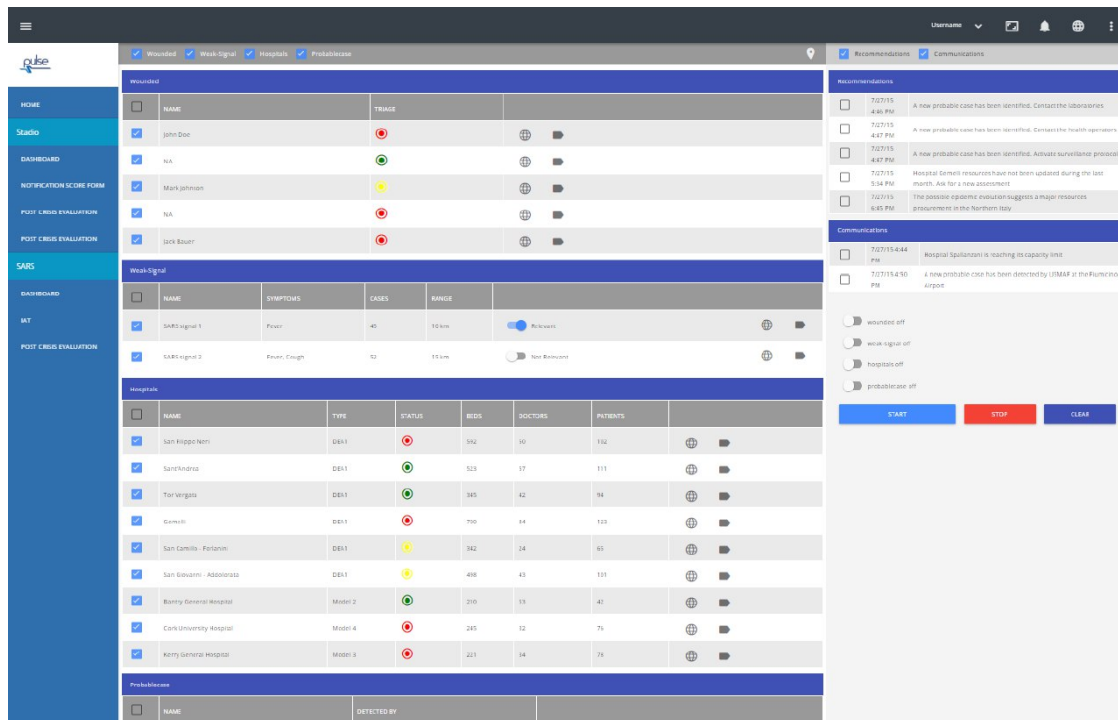


Figure 4: SARS scenario Operational Overview

The dashboard is composed of two main parts: a *Data Table* (4.1.2.1.1) and a *Navigator Map* (4.1.2.1.2).

#### 4.1.2.1.1 Data Tables

The Data Tables sections contain all the information related to the crisis resources. As specified above, we could have different types of resources depending on the referred scenario. Figure 5 shows an example of the Ambulance Data Table where all the ambulances involved into the crisis are shown on different rows. In each row, we could have specific data related to the ambulance. In our example these data are (1) the identification name, (2) the ambulance status (e.g. Free, with a patient), (3) the ambulance category (e.g. Type A, Type B) (4) the current Estimated Time of Arrival (ETA) to the location specified in (5) Direction.









<input checked="" type="checkbox"/> Ambulances <input checked="" type="checkbox"/> Wounded <input checked="" type="checkbox"/> Hospitals						
Ambulance						
<input type="checkbox"/>	NAME	STATUS	CATEGORY	ETA	DIRECTION	
<input checked="" type="checkbox"/>	AM1	Free	Type A	10min	To Gemelli	 
<input checked="" type="checkbox"/>	AM2	With a patient	Type A	6min	To incident location	 
<input checked="" type="checkbox"/>	AM3	Free	Type B	14min	To Sant'Andrea	 
<input checked="" type="checkbox"/>	AM4	Free	Type C	10min	To incident location	 

Figure 5: Data table

In each row two additional features for data visualization are available:

- *Localize on map*: Localize the resource on the map if geo-localization data are available (this functionality is described in the following section 4.1.2.1.2)
- *Details*: enables the user to visualize detailed information of the resource by means of a dialog window (as shown in Figure 6). This dialog contains a representation of the resource on an embedded map.

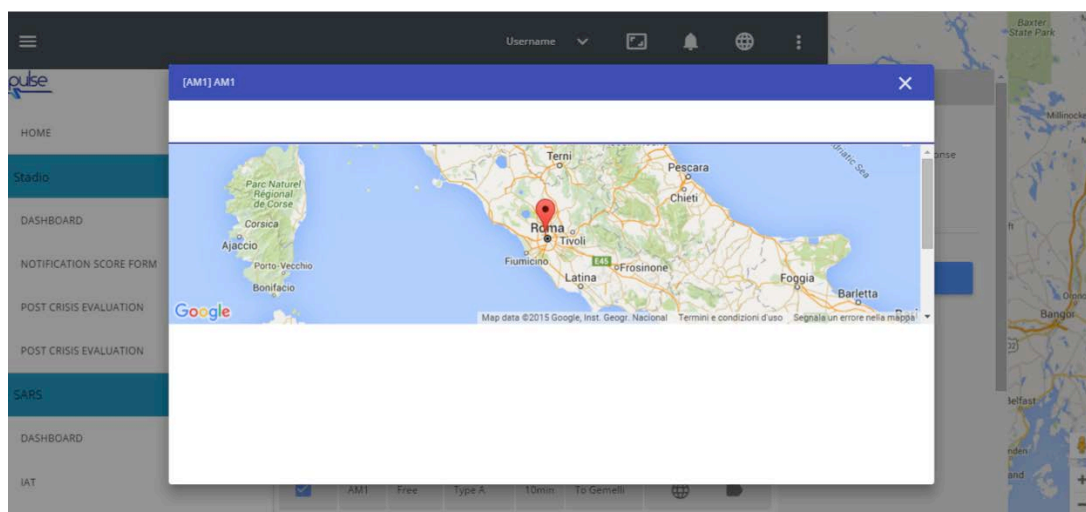


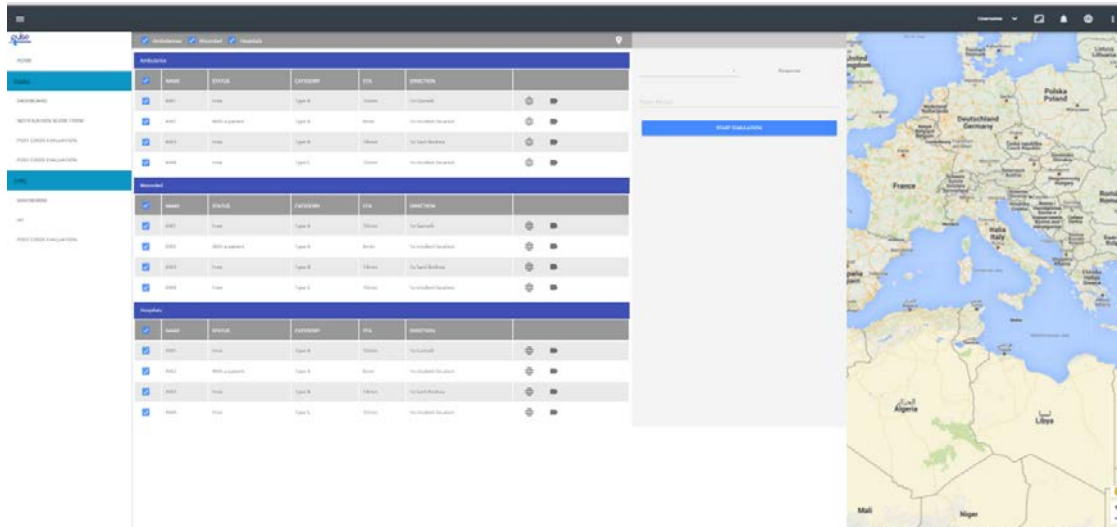
Figure 6: Data details

#### 4.1.2.1.2 Navigator Map

As said in the previous section, a resource can be visualized on a map if it is associated with some specific GPS coordinates.

This map, as shown in Figure 7, is positioned on the right side of the screen. Whenever an item is selected on the Data Table, a marker, located at the resource's GPS coordinates, is visualized on the map. The DSVT will obviously show proper markers depending on the resources' type.

In addition to this, this map can act as a navigation tool for the screen-sharing feature, which will be described in 4.1.2.2.



**Figure 7: Dashboard with Navigator map**

#### 4.1.2.2 Screen sharing

When a severe incident happens, a board composed of directors and crisis manager is usually rapidly organized in order to facilitate a fast resolution of the crisis. Several meetings are organized between the board's member and an intensive collaboration is highly requested. Nowadays, the tools that are used during the emergencies are partially able to provide support in terms of aggregation of different and heterogeneous information but tremendously lack for what regards the aspects of collaboration between the board's members.

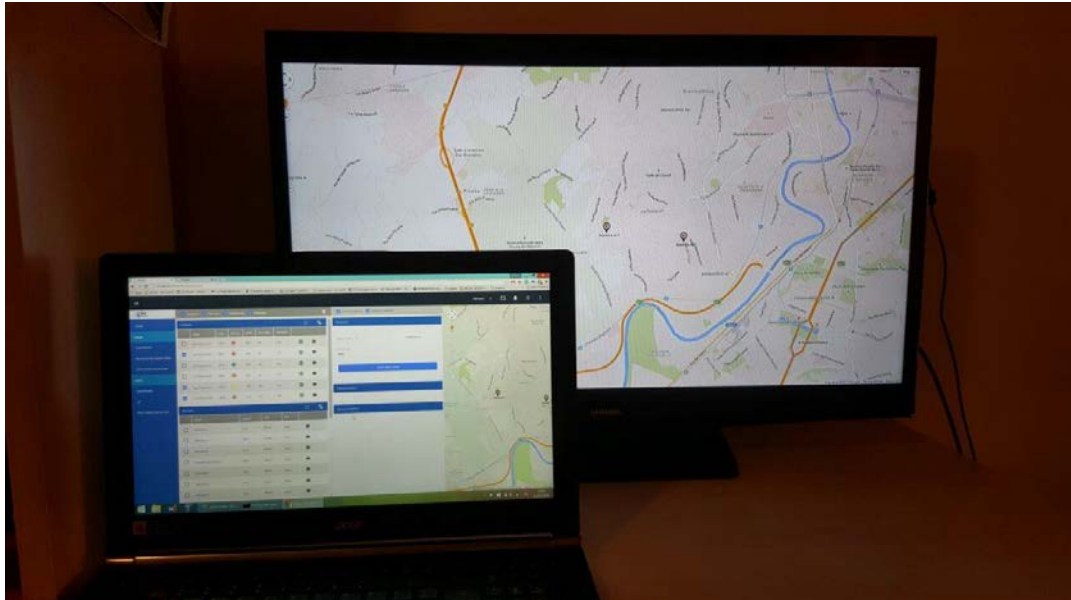
Considering this aspect and trying to fill out this gap, we decided to integrate into the DSVT an enhanced screen sharing functionality able to facilitate the collaboration and, as a consequence, to improve the effectiveness of the board's decisions.

This functionality is based on the fact that during the board's members meeting it could be useful to have one big picture of the current situation over a centralized map and the same map could be visualized and actually managed over all the single user's devices. Moreover, this approach could facilitate the usage of big screen technologies.

The DSVT GUI can be decoupled in two separated applications:

- *Big screen map:*
  - This map is a stand-alone application and can be visualized on a dedicated screen (or if necessary, projected on a wall through a beamer)
  - The map can be shared and managed by the users accessing the platform: as said before, it could be extremely useful to have one big picture of the current situation over a centralized map while the same map can be visualized over the user's device.
  - As mentioned in the previous section 4.1.2.1.1, each item that has been associated with GPS coordinates can be represented on a map.
- *Dashboard:* it is the Operational Overview described in 4.1.2.1 and it is visualized on the user's pc, smartphone or tablet. It allows the user to have the control of their own big map (e.g. move or zoom it, add/remove layers). In addition to this, all the commands sent by the users are synchronized within

the DSVT so that all the users connected to the same DSVT instance will be able to see the same changes.



**Figure 8: Screen sharing functionality**

Figure 8 shows a real example of the screen sharing functionality, where the Big screen Map has been projected on a television monitor while the dashboard is shown on a “private” laptop.

#### 4.1.2.3 Weak signals classification

The PULSE platform, specifically the Intelligence Analysis Tool (see more info in D4.2), is able to automatically generate weak signals by analysing and classifying news articles from specialised official and unofficial medical sites, blogs, online newspapers and clinical records. In our specific SARS-like scenario, the analysis is able to:

- Declare whether the documents are related to the SARS-like topic or not
- Trigger a signal if a certain amount of people suffering from the SARS-like symptoms are concentrated within a certain km range.

The DSVT shows a proper panel where the user is allowed to classify the received weak signals as “relevant” or “not relevant”.

Weak-Signal					
<input type="checkbox"/>	NAME	SYMPTOMS	CASES	RANGE	
<input checked="" type="checkbox"/>	SARS signal 1	Fever	45	10 km	<input checked="" type="checkbox"/> Relevant  
<input checked="" type="checkbox"/>	SARS signal 2	Fever, Cough	52	15 km	<input type="checkbox"/> Not Relevant  

**Figure 9: Weak signals classification**



Figure 9 shows the above-mentioned panel where the user can see the list of the received weak signals. For each weak signal the user can:

- Define it as “relevant” or “not relevant” by simply clicking on a button
- Visualize it on a map (only if the signal is accompanied with geographical coordinates)
- See the signal details so that the description of cause that forced the generation of the signal

#### 4.1.2.4 Simulation

The DSVT provides a specific functionality that help assessing the possible future outcome of the crisis considering the actual status of the e.g. resources, persons, agents and their probable evolution during the course of the crisis. This information can be tremendously useful when decision makers have to find, usually in a really short time, the “most-efficient” way to get out of the emergency and when even the tiniest decision taken in a rushed way can threaten the positive outcome of the crisis.

Obviously, this solution does not guarantee that the scenario will really evolve as depicted, but it can just help understanding the “most probable” course of evolution starting from the available pieces of information.

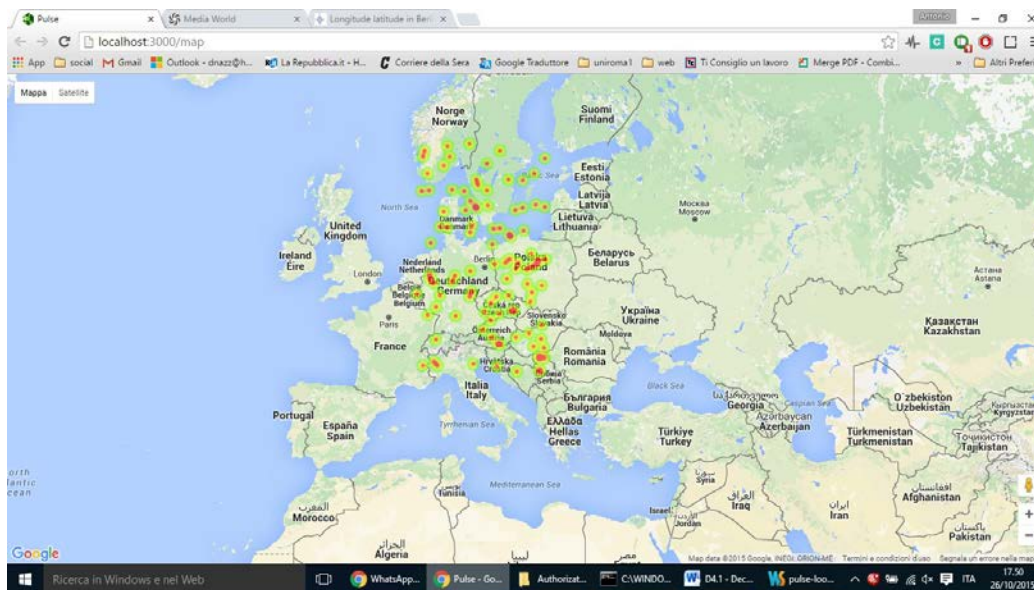
The Simulation functionality is based on the evolution models defined in WP3. As described in D3.X and D3.x, these evolution models treat different aspects of possible evolution paths: they vary from the possible variation of the physiological status of a person to the possible evolution of an epidemic disease. Taking this into account and for the sake of clarity, we will describe the Simulation functionality by splitting it within the two reference scenarios: the SARS-like scenario and the Stadium crush scenario.

##### 4.1.2.4.1 SARS-Like scenario

In the SARS-like scenario, a decision maker could be mostly interested in knowing the possible spread of the disease within the following days/months/years. This information can help him/her to suggest an intervention (e.g. major procurement of resources) in the hospitals located in the zones that according to the simulated scenario will probably suffer from the epidemic evolution.

In this case, we explicitly reference to the Evolution model for biological events (ENSIR) defined in D3.1 and the relative tool defined in D4.7. The goal of this model is the prediction of the spatial-temporal evolution of epidemics. It takes into account different factors, allowing for disease spread with different rates depending on the social and logistic characteristics of the interested area.





**Figure 10: Epidemic evolution**

Figure 10 contains the DSVT interface where the results of the epidemic spreading are shown to the PULSE user. The graphic interface is composed of a Map where the user can see the possible evolution. This map is represented as a Heat Map where, depending on the concentration of probable/suspected cases in a specific zone, the dot on the map assumes different colours. In this case, the red colour means a high concentration of probable and confirmed cases. This kind of map can immediately suggest to the user the possible outcome of the disease.

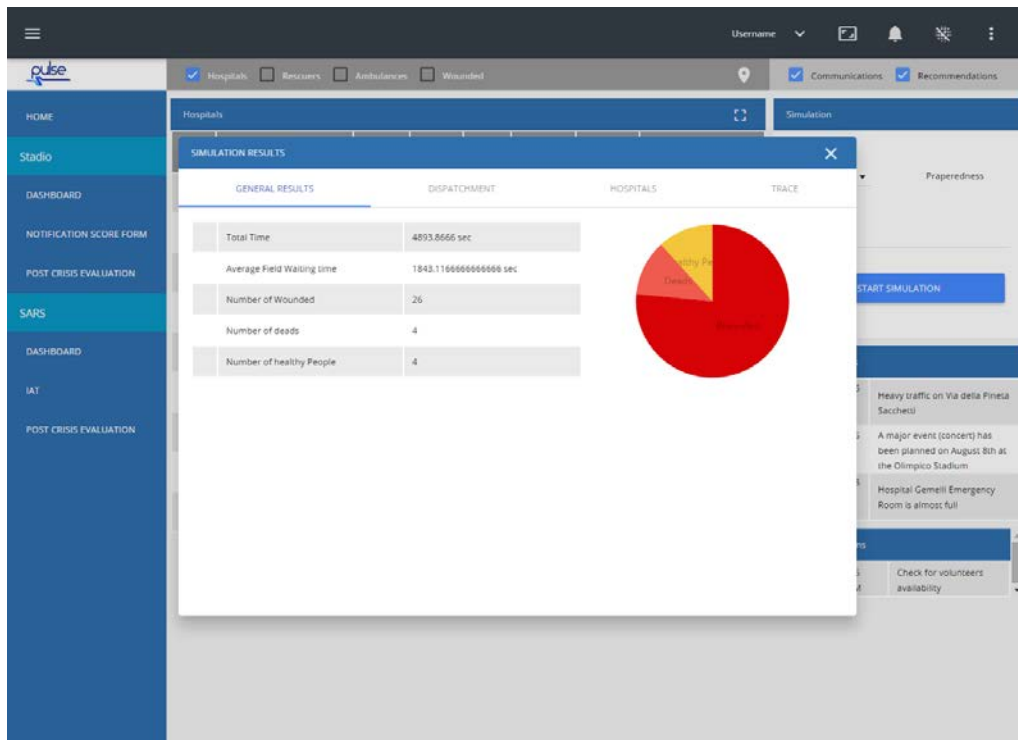
Moreover, the user can customize the initial parameters of the simulation. He/she can in fact select on the DSVT GUI interface:

- If the simulation should consider the number of actual suspected or confirmed cases
- The requested timeframe
- The map bounding box where the simulation should be initialized
- The simulation speed.

#### 4.1.2.4.2 Stadium crush scenario

In the Stadium crush scenario, a decision maker is interested in saving as many lives as possible in the limited amount of time he/she has at his/her disposal. The DSVT in this case tries to help him/her by providing a tool that is able to predict “the most probable” evolution of the incident or, in other words, the description of the possible behaviour of all the actors involved in the scenario (e.g. Wounded, first responders, ambulances, hospitals) according to the available information and the integrated evolution models. Therefore, the decision maker can exploit such kind of information and can probably take a better decision if he/she considers it during the mass casualty incident management.

In this case, we explicitly reference to the *Patient model*, *Health care effect model* and the *Health care facilities model* defined in D3.1. The relative tool is the Surge Capacity Generation tool described in D4.4.



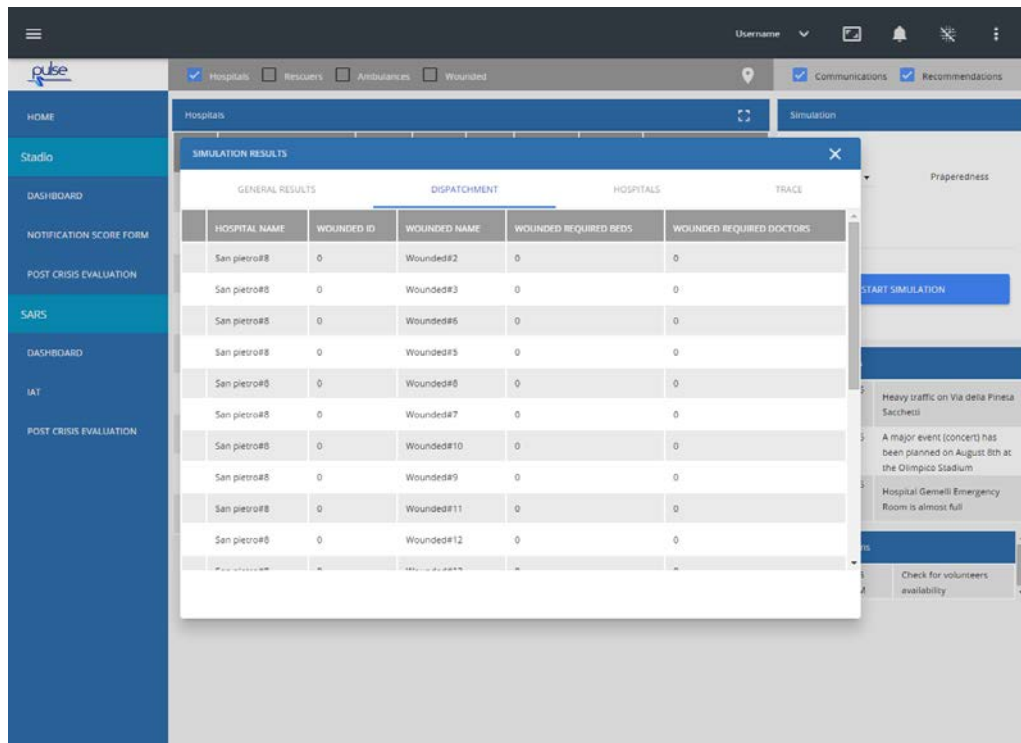
**Figure 11: Simulation results**

Figure 11 shows the general results provided by the DSVT to the PULSE user where the following information are specified:

- The total amount time so that the expected duration time of the crisis according to the available information
- The average time that the wounded waited on the incident field
- The expected number of wounded, dead and healthy people at the end of the crisis.

In addition to this and given a set of wounded and a set of hospitals, the DSVT is able to suggest the best dispatchment of the wounded to the specified hospitals according to:

- The available hospitals' resources
- The time necessary to reach the different hospitals
- A fair distribution of the wounded amongst the hospitals



**Figure 12: Dispatchment simulation results**

Nowadays when an incident happens, the people in charge usually decide for the dispatchment of the wounded to the hospitals by following the personal expertise and experience. We think that this automatic suggestion generated by the DSVT could tremendously help the decision makers during the emergency management. The DSVT in fact access information that is usually not accessible to the decision makers and it is able to automatically elaborate them by using optimization algorithms that are not currently used in real world situations and that could be executed by only computerized systems. The DSVT is then able to consider the actual resources present in the different hospitals, could provide an almost optimal solution to send all the wounded to the proper hospitals using as many hospital resources as possible and sending them in the minimum amount of time. More information regarding this optimization algorithm can be found in D4.3.



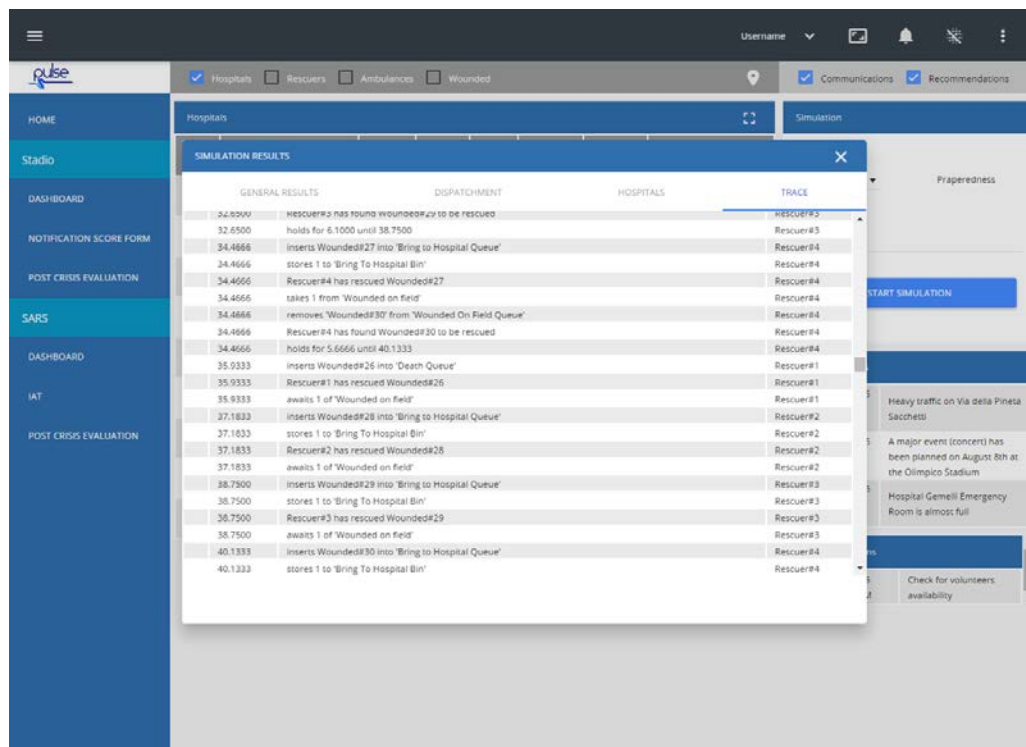


Figure 13: Simulation trace log

The DSVT can also provide a complete trace log of all the steps that have been followed to simulate the crisis. This information can be useful to the PULSE users to understand the goodness of the simulation and to follow the suggested steps. Figure 13 shows an example of trace log.

#### 4.1.2.5 Recommendation and Alerting system

As noted before, the DSVT's objective is to provide a complete support to decision makers that are in charge of a crisis management. An important part of this support is related to the automatic generation of personalized suggestions. These suggestions can help the user to simplify his work during the agitated hours of the crisis where, if not assisted by a tool, it could be easy to lose track of meaningful pieces of information.

One of the DSVT objectives is exactly this, to provide real-time recommendations and alerts generated according to the real status of the crisis.

We could have different recommendations depending on the case: for the Stadium scenario, for example, the DSVT could show predefined procedures (see WP5 for a deeper description of possible procedures) that are suggested to be followed; in the SARS-like scenario, the DSVT could should recommendations issued by international agencies like WHO and ECDC.

WHO		
<input type="checkbox"/>	6/23/15 4:44 PM	WHO calls for urgent action to curb hepatitis -- On World Hepatitis Day (28 July) WHO highlights the urgent need for countries to enhance action to prevent viral hepatitis infection and to ensure that people who have been infected are diagnosed and offered treatment. This year, the Organization is focusing particularly on hepatitis B and C, which together cause approximately 80% of all liver cancer deaths and kill close to 1.4 million people every year.
Communications		
<input type="checkbox"/>	7/27/15 4:44 PM	Hospital Spallanzani is reaching its capacity limit
<input type="checkbox"/>	7/27/15 4:50 PM	A new probable case has been detected by USMAF at the Fiumicino Airport
Recommendations		
<input type="checkbox"/>	7/27/15 4:46 PM	A new probable case has been identified. Contact the laboratories
<input checked="" type="checkbox"/>	7/27/15 4:47 PM	<del>A new probable case has been identified. Contact the health operators</del>
<input type="checkbox"/>	7/27/15 4:47 PM	A new probable case has been identified. Activate surveillance protocol
<input type="checkbox"/>	7/27/15 5:34 PM	Hospital Gemelli resources have not been updated during the last month. Ask for a new assessment
<input type="checkbox"/>	7/27/15 6:45 PM	The possible epidemic evolution suggests a major resources procurement in the Northern Italy

**Figure 14: Recommendations**

Figure 14 shows an example of the panel shown by the DSVT where a list of communications and recommendations for the SARS scenario are listed.

In order to let the user to keep track of the followed recommendations, the DSVT allows to check and then underline all the communications that have been processed by the user.

Moreover, the DSVT provides a simple web interface where the user can easily create and edit the rules for triggering the recommendations and alerts. The DSVT continuously monitor and check predefined environmental conditions and, by following the **if-then-else** logic, **if** these conditions are satisfied, **then** some specific alerts or recommendations are triggered, otherwise (**else**) other specified actions are taken.

Possible examples of environmental conditions for the stadium scenario are the traffic and the weather conditions or the number of wounded on the field. If these conditions vary, different alarms/recommendations can be sent to the user.

A possible example instead of environmental condition for the SARS-like scenario is



the number of resources handled in a specific hospital. As shown in Figure 14 a recommendation could be the suggestion of a new resources assessment in a hospital.

#### 4.1.2.6 Post-crisis evaluation

One of the objective of the PULSE platform is to provide a mean to efficiently evaluate the effectiveness of the decisions taken during the crisis, in order to understand how to improve the choices performed in terms of procedures, quality and quantity of employed resources. The core of this functionality is provided by the PCET as it is the module in charge of storing information about the crisis and recovering and analysing that information with structured queries for functional post crisis evaluations.

The DSVT complements the PCET results by:

- Providing a graphical front-end interface easily exploitable by the PULSE stakeholders to access historical crisis data elaborated by the PCET. These data are then compared by the DSVT to support evaluators in analysing the evolution of emergencies related to the same cause and in identifying both the choices resulted in benefits and those that, on the contrary, have proven to be counterproductive
- Periodically storing into the *PCET* internal repository a sort of 'snapshot' of the crisis correlated to a specified time.

More info regarding the post-crisis evaluation feature can be found in D4.6 [5].

In addition to this, the DSVT provides also a specific panel for a post crisis assessment where it is possible to receive feedbacks from the crisis participating players. These feedbacks are provided in form of a questionnaire as depicted in Figure 15.

Questions	
1. Were the documented procedures followed?	<input type="radio"/> Yes <input type="radio"/> No
2. Were the procedures adequate?	<input type="radio"/> Yes <input type="radio"/> No
3. Did personnel demonstrate a good knowledge of the procedures?	<input type="radio"/> Yes <input type="radio"/> No
4. Were Procedures readily available?	<input type="radio"/> Yes <input type="radio"/> No
5. Were contact details available and up to date?	<input type="radio"/> Yes <input type="radio"/> No
6. Were means of communications operational?	<input type="radio"/> Yes <input type="radio"/> No

Recommendation And Corrective Actions Can Be Included	
1. Based on the exercise today and the tasks identified, list the top 3 areas that need improvement?	<input type="radio"/> Yes <input type="radio"/> No
2. Is there anything you saw in the exercise that the umpire(s) might not have been able to experience, observe, and record?	<input type="radio"/> Yes <input type="radio"/> No
3. Identify the corrective actions that should be taken to address the issues identified in 1. above. For each corrective action, indicate if it is a high, medium, or low priority.	<input type="radio"/> Yes <input type="radio"/> No
4. Describe the corrective actions that relate to your area of responsibility. Who should be assigned responsibility for each corrective action?	<input type="radio"/> Yes <input type="radio"/> No
5. List the applicable equipment, training, policies, plans, and procedures that should be reviewed, revised, or developed. Indicate the priority level for each.	<input type="radio"/> Yes <input type="radio"/> No

**Figure 15: Post crisis evaluation feedback**

#### 4.1.2.7 Resource Management

The DSVT, through the DSVT GUI enables the user to manage the set of platform resources in a graphical way. The platform resources are stored in the Logistic tool and the relation of the DSVT with the Logistic tool will be described in 5.1.1.1.2.

#### 4.1.2.8 Preparedness support

The DSVT plans to provide support both in the preparedness and the response phase of a mass-casualty emergency.

One of the most crucial aspects of the preparedness phase is to correctly estimate the number of resources on the field (e.g. number of ambulances, first responders etc.) necessary to manage an event (e.g. a big concert in a stadium) and to efficiently respond in case of an incident. Currently, different emergency agencies around Europe (like 118 [21] in Italy) calculate this estimate by using specific algorithms (e.g. Maurer [22] in Italy) that are able to generate such results by taking in input some predefined scenario evaluation parameters (e.g. number of foreseen participants, event location, nature of the event, etc.). This task is usually accomplished without the support of a computerized system, therefore adding an additional component of possible human miscalculation.

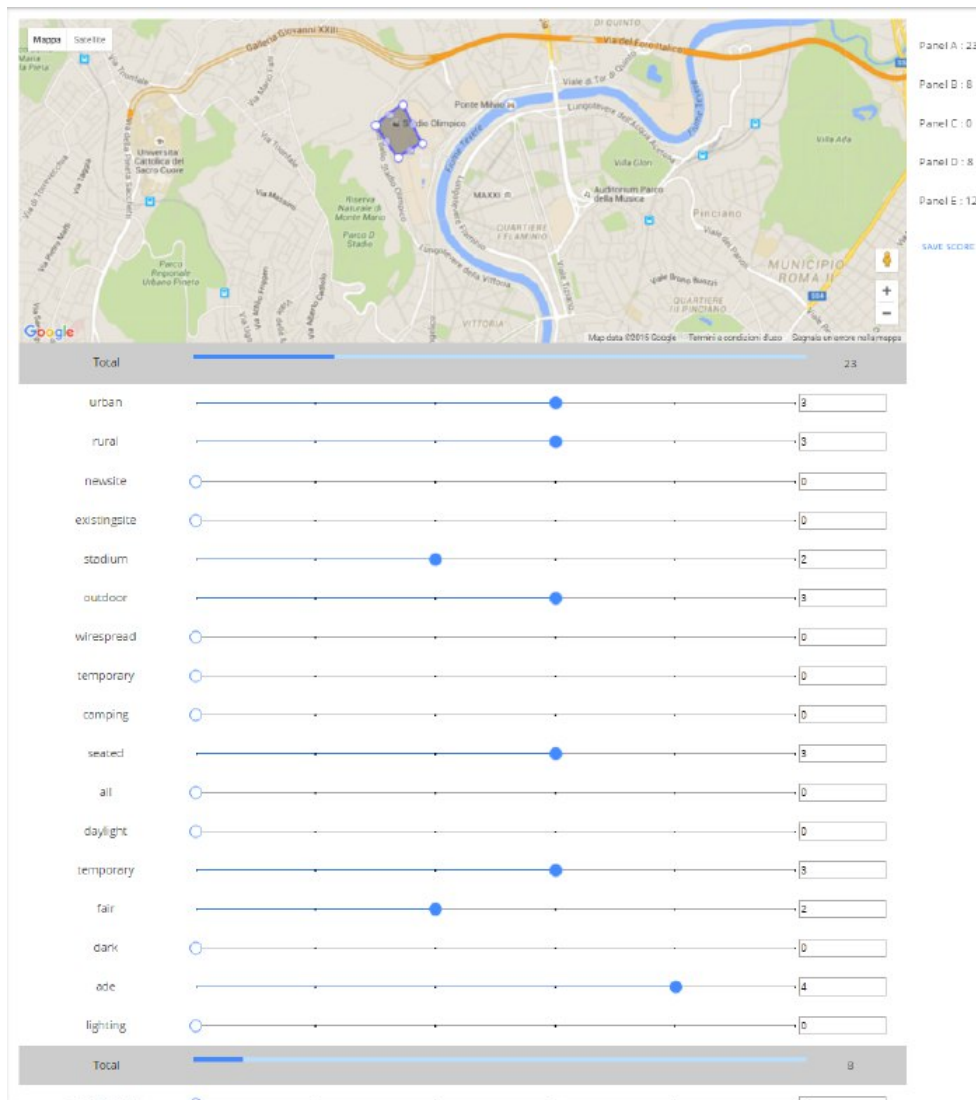


Figure 16: Preparedness panel

As shown in Figure 16, the DSVT tries to fill out this gap, and provides a GUI where the manager in charge of the event planning can insert these scenario evaluation parameters and receive back an automatic estimate of the necessary resources. The current DSVT prototype supports the Maurer algorithm used in Italy and the partially different algorithm used in Ireland.

#### 4.1.3 Relation with WP2 Use Cases

Part of the work carried out in WP2 was focused on the definition of a set of use cases for PULSE that have been detailed in the deliverable D2.2. In the following paragraph, we will try to associate the functionalities described in section 4.1.2 with the use cases where the DSVT is actually involved.

##### 4.1.3.1 SARS-like scenario

###### 1. UC-SARS LIKE – 01 - Weak signal detection and surveillance

In this use case a weak signal is notified to the competent authorities that will take the decisions based on the WHO and ECDC guidelines.

The DSVT allows a the user to classify the received weak signals as “relevant” or “not relevant” (functionality described in 4.1.2.3), maintains an history of all the received signals and sends proper suggestions and guidelines to the user if the number of received signals in a specific zone overcomes some predefined thresholds (functionality described in 4.1.2.5).

2. *UC-SARS LIKE – 02 - An airplane is landing in Italy. A probable case is now identified*

In this use case one person suffering from the usual SARS symptoms has been detected by the Malpensa Airport medical centre (USMAF) and the PULSE system gives recommendations on the procedure that needs to be followed.

The DSVT in this case provides the possibility to manage the information regarding the new suspected case through the Operational Overview functionality (described in 4.1.2.1) and sends proper suggestions and alerts to the user according to the person’s status (functionality described in 4.1.2.5)

3. *UC- SARS LIKE – 03 - A ship is arriving in Italy. A passenger has been identified as probable case*

In this case one person suffering from the usual SARS symptoms has been detected by the master of a ship, or the ship’s surgeon if one is carried, and the PULSE system gives recommendations on the procedure that needs to be followed.

As said in the point 2, the DSVT provides the possibility to manage the information regarding the new suspected case through the dashboard described in 4.1.2.1 and sends proper suggestions and alerts to the user according to the person’s status (functionality described in 4.1.2.5)

4. *UC-SARS LIKE – 04 - Identification of a new probable case in a community*

In this case one person coming from a community is suffering of the usual SARS symptoms. The PULSE system gives recommendations on the procedure that needs to be followed. There are already assessed cases of SARS in Italy.

As said in the point 2 and 3, the DSVT provides the possibility to manage the information regarding the new suspected case through the dashboard described in 4.1.2.1 and sends proper suggestions and alerts to the user according to the person’s status (functionality described in 4.1.2.5)

5. *UC-SARS LIKE – 05 - Assessment of the available medical resources during the pandemic phase*

In this case, the National Authority declares SARS as a pandemic disease and requires information on the availability of medical resources from health facilities.

The DSVT is able to show the status of all the hospitals involved in the crisis. In particular, the DSVT shows the status of the number of resources (e.g. rooms, ventilators) available in the structure in that specific moment (functionality described in 4.1.2.1) and allows the user to visualize the epidemic simulation results (described in 4.1.2.4.1). It also suggests re-assessing the resources of all the hospitals residing in a risky zone (functionality described in 4.1.2.5).

6. *UC-SARS LIKE – 06 - ECDC recommendations*

In this use case an assessment of the epidemic evolution is performed during ECDC periodic meetings.

The DSVT, considering the actual disease information and the possible



epidemic spread, creates and suggests guidelines that should be delivered to specific national agencies/institutes (functionality described in 4.1.2.5).

7. *UC-SARS LIKE – 07 - National Authority periodic assessment*

In this use case an assessment of the epidemic evolution is performed during national periodic meetings.

The DSVT is able to show the actual crisis situation including the probable and confirmed cases' health status and location (described in 4.1.2.1) and sends proper suggestions and alerts to the user according to the epidemic's status (functionality described in 4.1.2.5)

8. *UC-SARS LIKE – 08 - Post emergency learning at national level*

In this case, the National Authority evaluates how the country responded to the epidemic.

The DSVT allows recovering the crisis information for functional post crisis evaluations (described in 4.1.2.6).

9. *UC-SARS LIKE – 09 - Post emergency learning at WHO level*

In this case, The WHO evaluates how the epidemic has been handled and compares its evolution with past crisis.

The DSVT allows recovering the crisis information for functional post crisis evaluations and allows the user to graphically compare the results of different crisis (described in 4.1.2.6).

#### 4.1.3.2 Stadium scenario

1. *UC-STADIUM CRUSH – 01 - Scoring System in the Event Medical and Other Plan Preparation Phase*

In this case, a crowd event is planned that requires specific medical plans to be prepared and submitted to a regional authority for permission and to provide the regional authority with a means of accessing the risk likely for a specific event.

The DSVT provides the possibility to obtain the number of resources that should be used in order to efficiently manage the planned event (described in 4.1.2.8).

2. *UC-STADIUM CRUSH – 03 - User wishes to mobilise additional resources from Public, Private, Voluntary and Response Assets from other member states. Via surge capacity tool.*

In this case, the user asks for a mobilisation of additional response resources.

The DSVT provides the possibility to see the actual status of the first responders on the incident field (see 4.1.2.1). The interaction with the first responders is handled by the Smartphone app (see D4.8).

3. *UC-STADIUM CRUSH – 04 - Hospital Surge Capacity and Bed Management*

In this case, summarised information is requested in order to support on-site co-ordinators and commanders.

The DSVT is able to show the status of all the hospitals involved in the crisis. In particular, the DSVT shows the status of the number of resources available in the structure in that specific moment (functionality described in 4.1.2.1)

4. *UC-STADIUM CRUSH – 05 - Triage in Casualty Clearing Station [CCS] and links to PULSE proposals on ePCR.*

In this case, summarised information is requested in order to support hospital controller's regional authorities and crisis management teams in regard to hospital admissions.

The DSVT provides the information related to the crisis and allows having a



complete overview on the incident status (described in 4.1.2.1).

5. *UC-STADIUM CRUSH – 06 - Input critical data for the RCS on Site and from other relevant off-site sources*

In this case, summarised information is requested in order to support hospital controller's regional authorities and crisis management teams in regard to hospital admissions.

The DSVT provides the information related to the crisis and allows having a complete overview on the incident status (described in 4.1.2.1).

6. *UC-STADIUM CRUSH – 07 - Post-Event, Post Exercise Evaluation Tool to identify lessons to be learned.*

In this case, the objective is to ensure feedback from participating players and that the final report is completed promptly, as it is the principal means of ensuring that lessons identified can be used to provide material from which the exercise or incident can be evaluated.

The DSVT provides a panel containing a set of questions that the persons that have been involved in the crisis management should fill out.

7. *UC-STADIUM CRUSH – 08 - Casualty Bureau Operation searchable data base created for specific multi casualty incident.*

In this case, the objective is to create a central contact point for the matching of information available on casualties with requests from all those seeking or providing information about persons involved in the incident.

The DSVT provides a panel where the user can insert the information regarding missing persons. The DSVT is able to return a possible match with the persons that are stored into the PULSE system repository (Logistic tool).

## 4.2 Authentication Server

The PULSE platform's security aspects are handled by a dedicated tool called **Authentication Server**. This tool provides a layer of protection that forces users and clients to authenticate through the server before e.g. accessing the network or invoking the web service interfaces exposed by the PULSE tools.

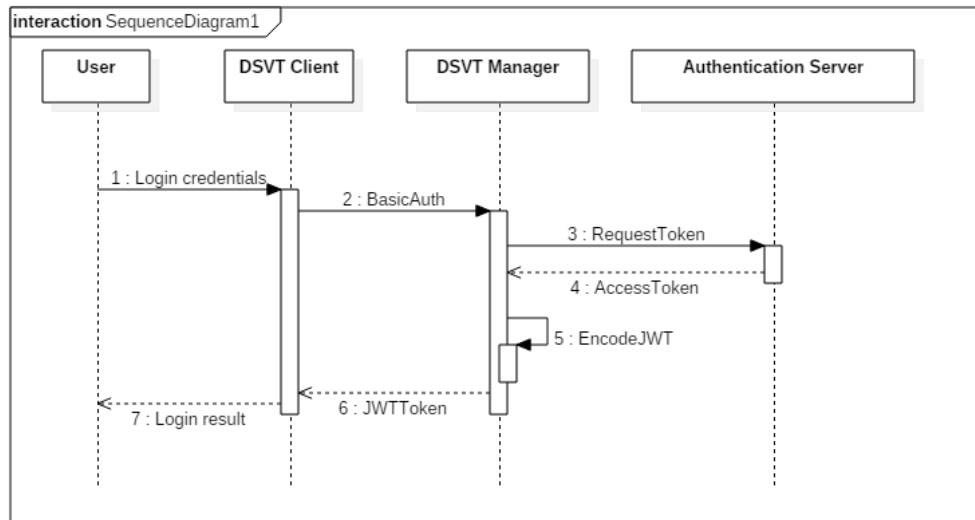
This component is based on the well-known open authentication protocol *OAuth2* [7], which provides different security mechanisms:

1. Only authorized client (e.g. PCs, Smartphone) are allowed to access the platform. This is accomplished by keeping record of the clients' credentials information inside the Authentication Server. In the PULSE context, this means that only users that are accessing through registered PCs and/or Smartphones are authorized to access the platform.
  - a. The Authentication Server generates the client credentials (composed of two different strings called *ClientId* and *ClientSecret*) after the request of a PULSE administrator. These credentials can then be manually inserted to the client configuration file.
2. A user provides its logging credentials by using a registered client (e.g. PC with DSVT in execution). The Authorization Server validates these credentials and returns back to the client an *AccessToken*. This *OAuth2* mechanism is called *Resource Owner Password Credentials Grant*.

Figure 17 shows the steps that should be performed in order to authenticate a user to the PULSE platform. In this example, we depict the case where a user logs-in to the PULSE platform by using a DSVT Client (e.g. by using the DSVT GUI described in 5.1.1.1.2 or the Smartphone application described in section 7.1). The DSVT Client connects to the DSVT Manager by exploiting a simple

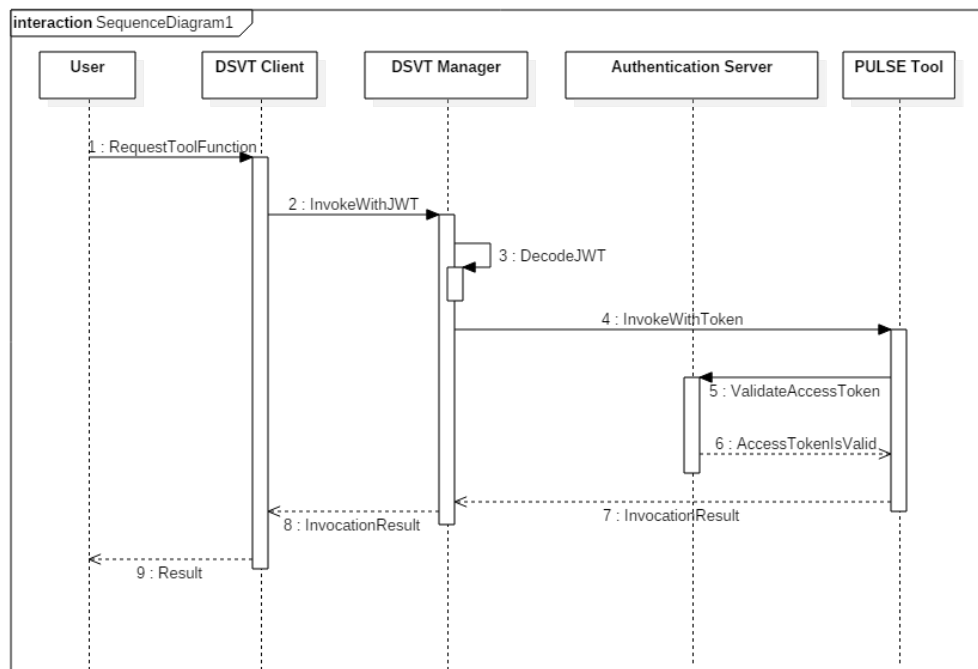


BasicAuthentication mechanism (RFC 2617 [23]). The DSVT Manager then sends the provided credentials to the Authentication Server through the above-mentioned OAuth2-based *Resource Owner Password Credentials Grant* mechanisms. The received *AccessToken* is then encoded as a JWT [24] token in order to guarantee the management of the user's session.



**Figure 17: Authentication steps**

3. The *AccessToken* is used internally in the PULSE platform to guarantee that the client (that is acting on behalf of the user) is authorized to invoke the PULSE tools. All the PULSE tools allow the invocation of their exposed interfaces only if the invoking client provides a valid *AccessToken*. Each tool contacts the Authorization Server in order to verify whether the *AccessToken* is valid or not. If so, the client is authorized to invoke the exposed tool service interface.



**Figure 18: Access Token validation**

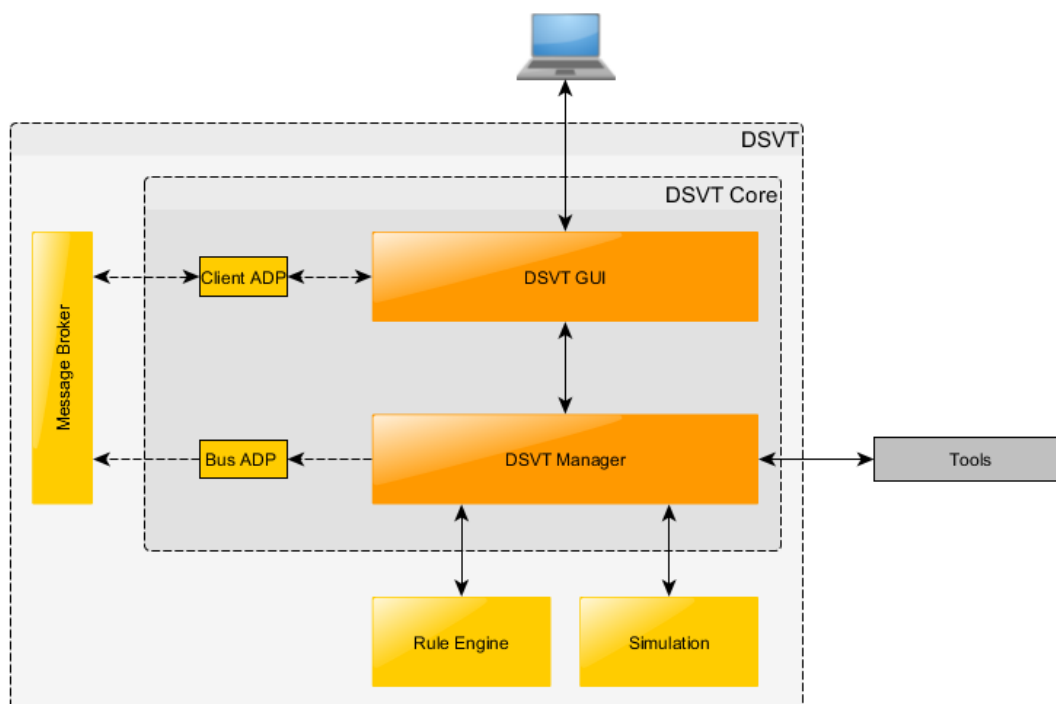
Figure 18 shows the steps that are performed in order to guarantee that only authenticated users are authorized to invoke the functionalities provided by the PULSE Tools. The DSVT Manager will internally decode the JWTToken in order to obtain the *AccessToken* that has to be sent to the PULSE tool.

## 5 Tools architecture

### 5.1 Decision Support and Validation Tool

#### 5.1.1 DSVT Component Architecture

The main purpose of the *Decision Support and Validation Tool* (DSVT) is to provide the front-end interface of the PULSE platform and to assist the users, with enhanced supporting functionalities, during a major incident.



**Figure 19: DSVT architecture**

The DSVT act also as the backbone communication layer of the platform. All the interactions between the PULSE components are in fact handled by the DSVT that efficiently dispatch the messages from one tool to another.

The Figure 19 shows the logical architecture of the DSVT tool where each element represents a functional component:

- **DSVT Core:** it is the tool's main module. It is itself composed of two sub-modules:
  - **DSVT Manager** acts as the backbone communication layer that includes the main working operations of the tool. The DSVT Manager provides an external interface to the other tools of the platform (the relations with other tools will be described 5.1.1.1.1) and an internal common interface for the integration with the other DSVT sub-components.
  - **DSVT GUI** provides the Graphical User Interface of the tool



- **Message Broker:** it is a middleware layer. It handles the asynchronous communication between the *DSVT Manager* and the *DSVT GUI* in order to achieve a real time notification system.
- **Simulation:** standalone modules that provides the functionality described in 4.1.2.4.
- **Rules Engine:** standalone modules that provides the functionalities described in 4.1.2.5.

#### 5.1.1.1 DSVT Core

As mentioned in 5.1.1, the **DSVT Core** contains the main components of the tool.

In the following paragraphs, we will deeply describe their internal structure and features:

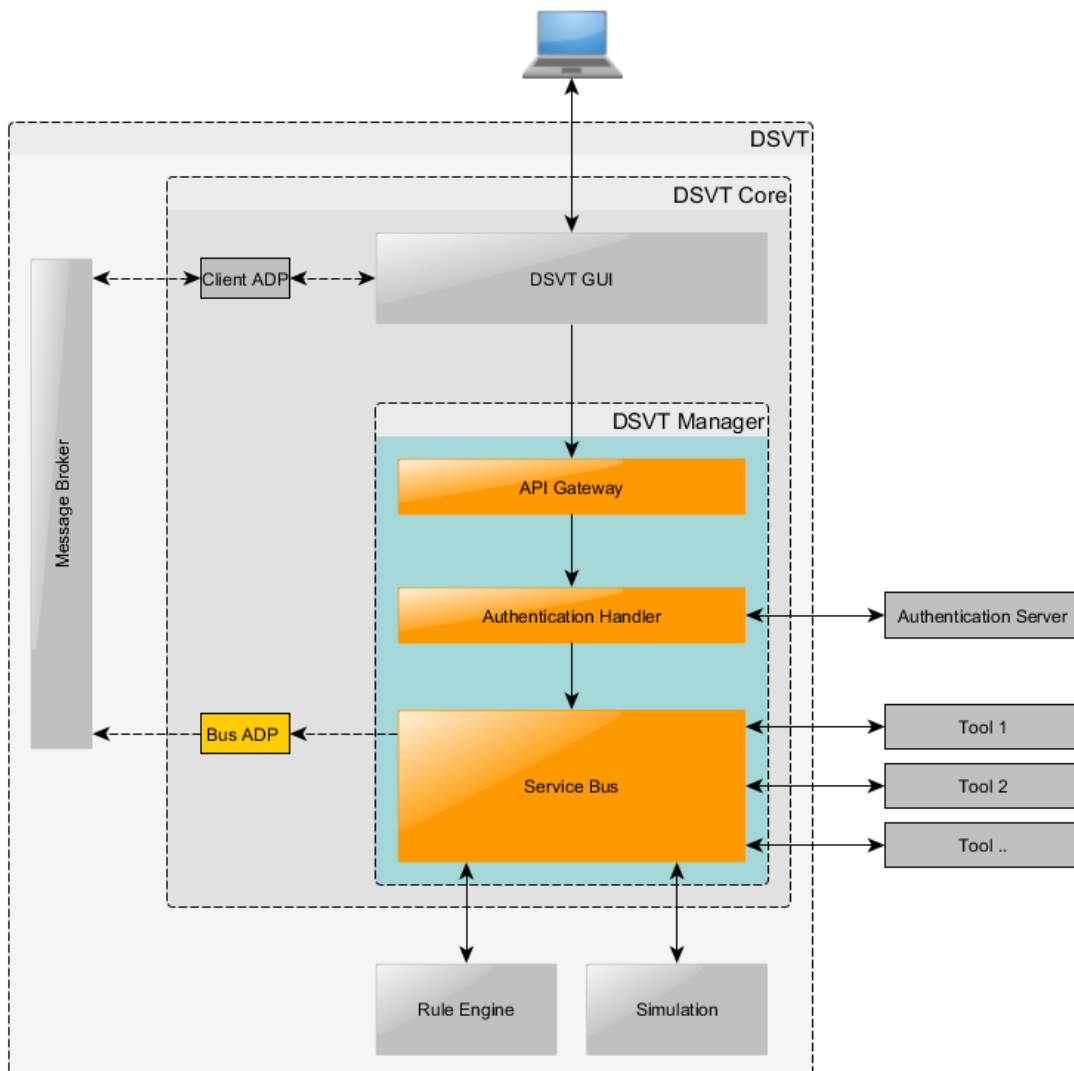
##### 5.1.1.1.1 DSVT Manager

This module can be intended as the backbone communication layer and since the general architecture of the platform has been defined following the SOA approach, it can be considered the “Service Bus” used by the PULSE tools to communicate.

The DSVT Manager takes advantage of this architectural feature and is able to:

- Group the tool functionality under a *single software interface*
- Group several and heterogeneous data sources and provide such sources as a single repository
- Hide the internal modularity of the platform
- Make transparent to the user the internal mapping of the tool

Because of this, the DSVT Manager provides an API that makes available the internal functionalities of the tool.



**Figure 20: DSVT Manager**

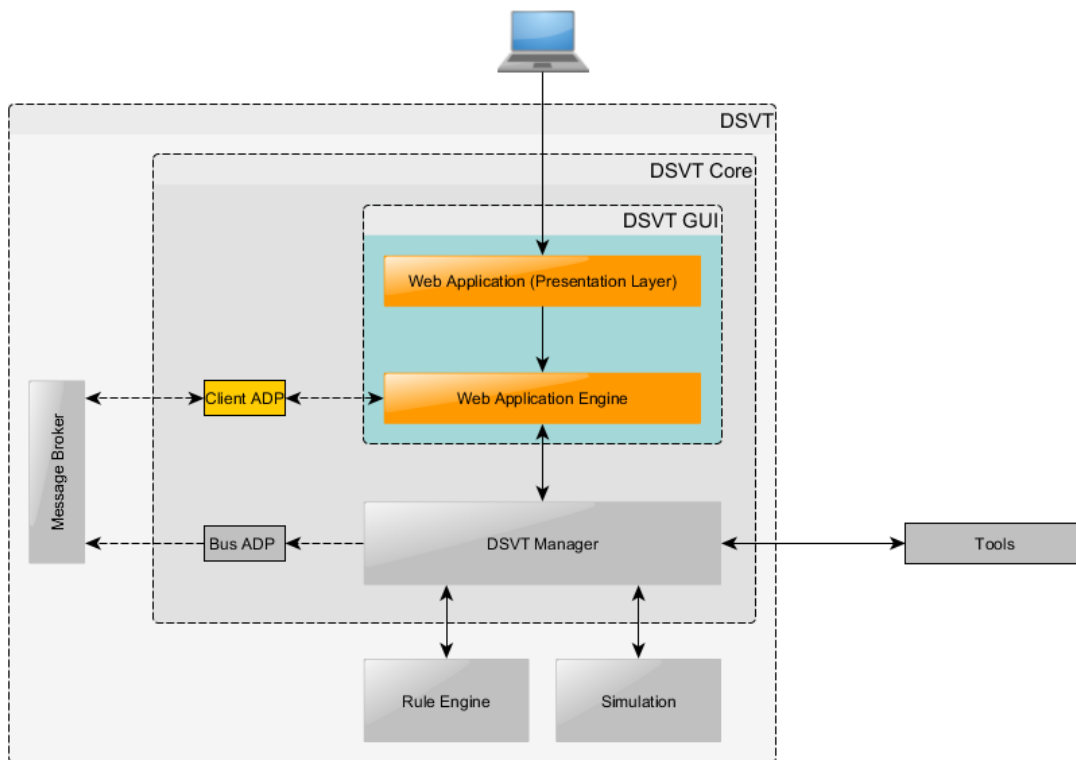
The block diagram represented in Figure 20 shows the functional elements, which constitute the DSVT Manager:

- **API Gateway:** provides an HTTP interface to Clients. Each request is processed here and is routed the *Authentication Handler*
- **Authentication Handler:**
  - Act as a proxy gateway handling the incoming requests;
  - Manage the User authentication against the Authentication server (more info about the Authentication process can be found in section 5.2);
  - Manage sessions after user authentication;
  - Manages the request of access token from the Authentication Server and return to the user an access token
  - Validates the access token against the authentication server and routes the request to *Service Bus* or, if necessary, rejects the request

- **Service Bus:** it is the aggregation layer; it provides a unique software interface and manages the aggregation of the platform tools.

#### 5.1.1.1.2 DSVT GUI

The module provides the Graphical User Interface of the PULSE platform. The DSVT GUI is based on single-page-application (SPA) architecture. It has been developed following the MVVM (Model View ViewModel) pattern that makes possible to separate the User Interface from the business logic of the application.



**Figure 21: DSVT GUI**

The **DSVT GUI** makes available to the user (in graphical form) the functionalities described in 4.1.2, this module communicates with the:

- DSVT Manager API Interfaces (as described in 5.1.1.1.1 it is the API exposed by the DSVT Manager5.1.1.1.1) over *HTTP*.
- DSVT Message Broker over *WebSocket*.

#### 5.1.1.2 Message Broker

It is a message oriented middleware module that manages the publish/subscribe feature and asynchronous messaging system inside the DSVT.

It translates messages from the formal messaging protocol of the sender to the formal messaging protocol of the receiver.

In the context of the DSVT the elements involved in the communication with the Message Broker are the **DSVT GUI** through the **Client ADP** and the **DSVT Manager**



through the **Bus ADP**.

- **Client ADP:** it is a client module used by the DSVT GUI to:
  - Subscribe to events dispatched from the Message Broker
  - Trigger asynchronous actions over the GUI
  - Translate the messages in a GUI-oriented standard.
- **Bus ADP:** it is a client used by the DSVT Manager to publish/subscribe events dispatched by the Message Broker.

#### 5.1.1.3 Simulation

As said in section 4.1.2.4, the DSVT allows the simulation of an underway event like an incident in a stadium or an epidemic disease.

The internal DSVT module, called *Simulation*, is the component in charge of performing this simulation task. It behaves in two different ways depending on the scenario the DSVT is referring to.

##### 5.1.1.3.1 Stadium crush scenario

The Simulation module exposes a RESTful interface that takes as input the status of the actors that are involved in the incident. These actors are:

- The ambulances with their GPS coordinates and the number and type of medical resources at disposal,
- The first responders with their GPS coordinates and the number and type of medical resources at disposal,
- The wounded with their GPS coordinates and triage code,
- The hospitals with their GPS coordinates and the number and type of medical resources at disposal

Once the Simulation module obtains all the actors data, it starts a discrete event simulation (DES) of the event. In this kind of simulation, all system state changes are supposed to happen at discrete points in time and between such events, the system state is assumed to remain constant.

During the simulation, predefined models, that describe the interactions steps that are supposed to be taken by the actor during the emergency, lead all the actors' actions. For example:

- The first responder model expects that:
  - A first responder takes care of one of the wounded on the field (in a customizable amount of time) and does a triage on him.
  - Once finished, the first responder automatically reaches another wounded on the field until all the wounded have been assisted.
- The wounded model expects that:
  - A first responder does a triage code on the wounded in a customizable amount of time.
- The ambulance model expects that:
  - An ambulance takes one of the wounded on the field that have been notified by a first responder with a red code.
  - The best hospital (according to the time, location and resources requirements) is assigned to the ambulance. This is accomplished by invoking the Optimization functionality provided by the Logistic tool. This component and its internal optimization algorithm are described in D4.3 [2].



- An ambulance reaches a hospital in a specific amount of time. This time is calculated by using the Google Maps Web Services API [25] and providing the GPS coordinates of the incident location as the start of the route, and the GPS coordinates of the hospital as the end of the route. In particular, the time in question is the “traffic time” provided by the Google API.
- An ambulance reaches the incident location in a specific amount of time. This time is calculated by using the Google Maps Web Services API and providing the GPS coordinates of the hospital as the start of the route, and the GPS coordinates of the incident location as the end of the route.
- The hospital model expects that:
  - Once a red-code wounded reaches the hospital a surgery starts in a customizable amount of time.
  - The surgery lasts a customizable (random or depending on the type of the injury) amount of time.

The simulation stops as soon as all the wounded have been accepted in a hospital and a surgery has been performed.

Once finished, the Simulation module returns to the DSVT Manager the simulation results that are then forwarded to the DSVT GUI.

#### *5.1.1.3.2 SARS-like scenario*

The simulation module obtains the information regarding the possible spread of the disease by invoking the ENSIR tool (as described in D4.7 [6]). The ENSIR tool exposed a SOAP web service interface takes as input the number and the location (GPS coordinates) of the probable and confirmed cases.

The information returned by the ENSIR tool is forwarded to the DSVT Manager and then to the DSVT GUI where is shown on a satellite map.

#### *5.1.1.4 Rule Engine*

The DSVT contains internally also a component called Rule Engine. This component implements the functionality described in section 4.1.2.5 and it is able to store and execute rules following the **if-then-else** paradigm.

It exposes a RESTful interface to allow an external component to insert a new rule into the system. In the PULSE platform, a user defines a new rule through the DSVT GUI. Then, the DSVT Manager receives the rule from the GUI and automatically delivers it to the Rule Engine through the REST API.

Once a rule is received, the Rule Engine stores it into its internal database.

All the rules are continuously monitored by the Rule Engine and whenever a rule's condition is satisfied, the related event (defined in the **then** or **else** clauses) is automatically triggered. In order to verify the rules' conditions, the Rule Engine needs to get access to a set of the contextual crisis information. This is accomplished by the DSVT Manager that periodically create a sort of “snapshots” of the crisis and send them to the Rule Engine. More information regarding the nature and the content of a “snapshot” can be found in D4.6 [5].





### 5.1.2 DSVT relation to overall PULSE architecture

As said in the sections above, the DSVT is the core component of the PULSE platform. All the PULSE tools interfaces with it in order to communicate with another tool of the platform.

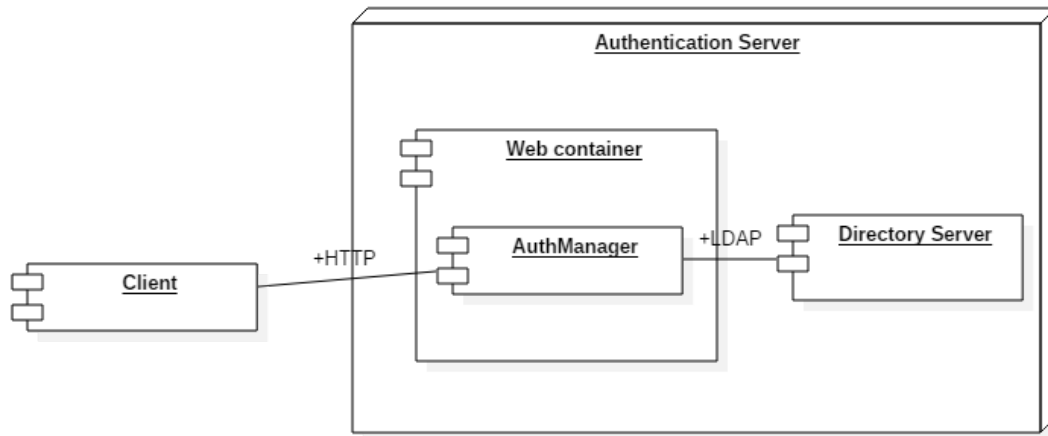
In the following, you can find a recap of all the foreseen interactions between the DSVT and the other platform's tools:

- DSVT – Smartphone app: In this case, the DSVT exposes an external REST API that can be invoked by the Smartphone app to send or retrieve updated information regarding the crisis resources.
- DSVT – Intelligence Analysis Tool: The IAT invokes a REST API provided by the DSVT whenever a new weak signal should be notified.
- DSVT – Logistic tool: The DSVT invokes the REST API provided by the Logistic tool to obtain or update the current status of the crisis resources
- DSVT – Surge Capacity Generation Tool: The DSVT invokes the SOAP interface provided by the SCGT during the Simulation process (see 4.1.2.4 and 5.1.1.3) in order to obtain the possible variation of the medical staff during the emergency.
- DSVT – ENSIR: The DSVT invokes the SOAP interface provided by the ENSIR tool during the Simulation process (see 4.1.2.4 and 5.1.1.3) in order to obtain the possible evolution of the epidemic.
- DSVT – PCET: The DSVT invokes the PCET REST API to send a periodic snapshot of the crisis and to retrieve the historical crisis information during the post crisis evaluation (see 4.1.2.6 and D4.6).

## 5.2 Authentication Server

### 5.2.1 Authentication Server Component Architecture

As described in chapter 4.2, the **Authentication Server** is in charge of providing a OAuth2-based layer of protection that forces users and clients to authenticate through the server before e.g. accessing the network or invoking the web service interfaces exposed by the PULSE tools.



**Figure 22: Authentication Server Architecture**

As shown in Figure 22 the architecture of the **Authentication Server** is rather simple. It is composed of two main modules:

- a *Web container*
- a *Directory server*

#### 5.2.1.1 Web Container

The *Web container* is the recipient where another module called *AuthManager* is actually deployed. The *AuthManager* is the module in charge of providing the HTTP RESTful service exposing the **Authentication Server**'s public interface.

The functionalities provided by the RESTful service are basically three:

1. Log-in of the user accessing the platform
  - a. The **Authentication Server** validates the provided user credentials and returns an *AccessToken*
2. Log-out of the user leaving the platform
  - a. The **Authentication Server** deletes the user's *AccessToken*
3. Validation of the *AccessToken* assigned to a user
  - a. The **Authentication Server** checks whether the provided *AccessToken* matches the internally stored *AccessToken* assigned to user during the Log-in phase.

#### 5.2.1.2 Directory Server

The *Directory Server* is the repository for storing and managing the users information. The *AuthManager* connects to the *Directory Server* through the Lightweight Directory Access Protocol (LDAP), which is a de-facto standard protocol for accessing, and maintaining distributed directory information services over an Internet Protocol (IP) network.

The *Directory Server* used in the PULSE platform stores the following information:

1. Username and ciphered password of the users
2. Users' role
3. Users' access tokens
4. Clients' information (clientId and clientSecret)



## 6 Components Technologies

### 6.1 Decision Support and Validation tool

#### 6.1.1 List of core technologies

As said in 5.1.1, the DSVT is composed of three main modules and sub-modules: DSVT Core, Simulation and Rule Engine.

The core technologies selected for the implementation of these components are:

##### DSVT Core

- *DSVT Manager*
  - o *API Gateway – Service Bus:*
    - **Loopback:** [26] it is an open source Node.js framework (build on top of Express), optimized for building APIs. It provides instruments to connect multiple data sources, defining business logic as simple node.js models on top of existing data and services.
  - o *Authentication Handler:*
    - **JWT:** [27] is a JSON-based open standard for to representing claims between web applications. The claims are encoded in JSON format.
- *Bus ADP*
  - o **Ampqlib:** [18] it is a library for making AMPQ [28] 0.9.1 clients for node.js (it is related to the Bus ADP described in 5.1.1.2). (AMPQ is an open standard application layer protocol for message oriented middleware)
- *DSVT GUI*
  - o **AngularJS:** [29] it is an open-source Javascript web application framework used to build mainly Single Page Application or generic Web application. It allows the use of HTML (CSS) as a template language. It is based on MVVM architecture and provides many features to build powerful dynamic web application (AJAX).
  - o **Angular Material:** [30] it is a graphical library for AngularJS. The library is an implementation of Material Design. Provides a set of predefined set of UI (User Interface) Components.
- *Client ADP*
  - o **StompJS:** [19] it is a Javascript library to receive AMPQ messages over WebSocket. (STOMP [31] is a simple text-orientated messaging protocol. It defines an interoperable wire format so that any of the available STOMP clients can communicate with any STOMP message broker)
- *Message Broker*
  - o **RabbitMQ:** [32] it is a message broker (an implementation of AMPQ), an intermediary for messaging that enables sending, receiving and dispatching of messages. It grants reliability, flexibility, and high performance. It can be used to implement publish/subscribe, asynchronous messaging, work queue.
- *Simulation*



- **DesmoJ**: [20] it is a discrete event simulation library. DESMO-J stands for Discrete-Event Simulation Modelling in Java. The library supports both event-oriented and process-oriented representation of the real world. Its usage is described in section 5.1.1.3.1.
- Rule Engine
  - **EasyRule**: [33] it is a simple and lightweight Java rules engine that provides features to describe and match rules making use of POJO based development with annotation programming model, it enables to create complex rules and to dynamically change the rule configuration. Its usage is described in section 5.1.1.3.2.
  - **Hibernate ORM**: [34] it is an object-relational mapping framework for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database.
  - **Jersey**: [11] it is an open source framework for developing RESTful Web Services in Java that provides support for JAX-RS APIs and serves as a JAX-RS (JSR 311 [12] and JSR 339 [13]) reference implementation
  - **Jetty**: [10] it is an HTTP server and Servlet container capable of serving static and dynamic content either from a standalone or embedded instantiations.

### 6.1.2 3<sup>rd</sup> Party libraries and licenses

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

**Table 1 – DSVT 3rd party libraries and licenses**

Product	Version	Vendor	License
Loopback	2.22.0	StrongLoop	MIT License
AngularJs	1.4.7	Google Inc.	MIT License
RabbitMQ	3.5.6	Pivotal Software Inc.	Mozilla Public License v1.1
Ampqplib	0.4.0	Michael Bridgen	MIT License
Stompjs	1.6.3	Jeff Mesnil	Apache License v2.0
DesmoJ	2.4.2	University of Hamburg	Apache License v2.0
EasyRule	2.1.0	Mahmoud Ben Hassine	MIT License
Hibernate ORM	5.0.2	RedHat	LGPL V2.1
Jersey	9.2.10	Eclipse	Apache 2.0
Jetty	2.17	Oracle Corporation	Dual License: - CDDL 1.1



			- GPL 2
--	--	--	---------

## 6.2 Authorization Server

### 6.2.1 List of core technologies

As said in 4.2, the Authorization Server is composed of two main modules: a *Web container* and a *Directory server*.

The core technologies selected for the implementation of these components are:

- Web Container:
  - **Jetty**: [10] it is an HTTP server and Servlet container capable of serving static and dynamic content either from a standalone or embedded instantiations.
  - **Jersey**: it is an open source framework for developing RESTful Web Services in Java that provides support for JAX-RS APIs and serves as a JAX-RS (JSR 311 [12] and JSR 339 [13]) reference implementation.
  - **Apache Oltu**: it is an OAuth protocol implementation in Java [14].
- Directory server:
  - **ApacheDS**: it is an extensible and embeddable directory server entirely written in Java [15].

### 6.2.2 3<sup>rd</sup> Party libraries and licenses

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

**Table 2 – Authorization Server 3rd party libraries and licenses**

Product	Version	Vendor		License	
<b>Jetty</b>	9.2.10	Eclipse		Apache 2.0	
<b>Jersey</b>	2.17	Oracle Corporation		Dual License: - CDDL 1.1 - GPL 2	
<b>Apache Oltu</b>	1.0	The Software Foundation	Apache	Apache v2.0	License
<b>ApacheDS</b>	2.0	The Software Foundation	Apache	Apache v2.0	License
<b>Apache Directory LDAP API</b>	1.0	The Software Foundation	Apache	Apache v2.0	License



## 7 Mobile Extensions

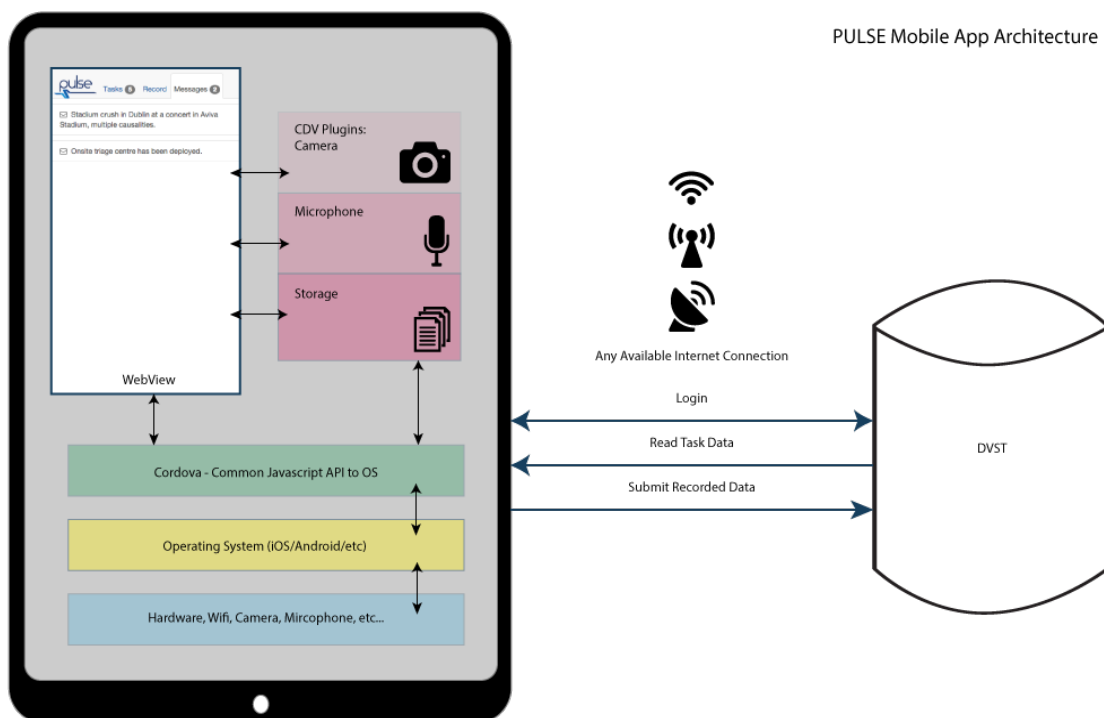
The Pulse Mobile Applications are extensions to the main the main Pulse Server front end. While the DVST provides the main GUI for managing the system, the mobile applications provide supplemental functions in 2 main ways:

1. As a smartphone app for volunteers to install & interact with the system, and
2. As mobile friendly web pages to allow members of the public to submit data into the system.

### 7.1 The Smartphone App

The Smartphone App allows approved volunteers to log into the PULSE system and use the advanced hardware features of the phone to collect and submit data.

As depicted in Figure 23, the app uses the Apache Cordova Framework; this allows for the flexibility of a HTML5 & JavaScript web application, with the power & hardware access of a native application.



**Figure 23: Smartphone App Architecture**

For a purely native mobile application, one would need to develop separate codebases for iOS (Objective-C), Android (Java), and any other platforms to be added. This would take longer, lead to inconsistencies between platforms, and involve replication of code for each OS to be supported.

The Cordova platform provides a common API layer on top of the operating system and underlying hardware (using plugins for Network access, Microphone & Camera access, Storage, barcode scanning, etc.). This allows for the core application to be



developed once then compiled for different platforms thus saving time, efficiency and maintaining consistency across devices.

**Platforms Supported:**

- iOS v7.0 or higher
- Android v4.2 or higher

**7.1.1 Login**

A mockup of a mobile app's login screen. At the top left is the 'pulse' logo. Below it, a light blue box contains the text: 'Enter you name to identify yourself on the system with this device.' (Note the typo 'you' in the original). Underneath is the label 'Name:' followed by a text input field with the placeholder 'Your name' and a small asterisk icon on the right. Below the input field is a toggle switch labeled 'Remember me on this device', which is currently turned off. At the bottom right is a blue button with the text 'Login' in white.

**Figure 24: Login**

As depicted in Figure 24, on starting the App, the user supplies their username & password to log into the PULSE server. (On subsequent runs, they may skip this step by selecting the 'Remember me' option on the login screen, which will save their credentials locally and automatically send the login request on start-up.)





Volunteers will be registered in the system to gain access to the Pulse server. They will log in using the OAuth authentication standard, which will authorise that user to have full access to the app and to identify their reports.

### 7.1.2 Geolocation

As soon as the user starts the application, the current longitude & latitude of the device are continually checked & updated. When the user is active these coordinates are sent back to the server so that the control room may keep track of who is available on the ground and where they are.

Each time the user creates a record the current coordinates are updated again, then saved with that record.

On login, the app updates the user details with their current location:

```
PUT [PulseServer]/users/[ID]/coordinates  
  
53.3383271, -6.2504998
```

### 7.1.3 Network

The app uses whatever network connectivity is available to the device. This can be the cell phone network, local wireless LAN, VPN connections, WiMax hotspots, etc...

Network selection is left to the device and so in cases where the cell phone network is too busy & unreliable, the device may switch to the Wi-Fi when that becomes available; or the user may have their device connected to a virtual private network with a guaranteed quality of service if one has been set up for the emergency.

In addition to this, network interruption handling is built into the application, as all data recorded is stored locally on the device and transmitted to the server when possible. If the network becomes unavailable for a time, the user may continue to record data at the scene. When the network becomes available again, each record will be transmitted back to the server with regular re-try attempts as needed until receiving a success message back from the server.

This interruption handling also handles cases of application or power interruption, so that if any data has been recorded but then device battery dies or the application is stopped & restarted, etc., then on resuming the app any unsaved data will be transmitted to the server.

### 7.1.4 Tasks

On starting the app, the volunteers will receive a list of available Tasks. An example task could be "Perform triage of personnel at the north gate of the stadium." Then fetches the latest task list from the server by invoking the following URL:

```
GET [PulseServer]/tasks/findAll
```



The screenshot shows the Pulse application interface. At the top, there is a header with the Pulse logo, a 'Tasks' tab with a count of 5, and links for 'Record' and 'Messages' with a count of 2. Below the header, there is a list of five tasks, each preceded by a star icon. Each task has a blue 'Respond' button and a red 'Reject' button.

Task Description	Respond	Reject
★ Perform triage of personnel at the north gate of the stadium	✓ Respond	✗ Reject
★ Support first responders at the on-site triage centre	✓ Respond	✗ Reject
★ Collect medical supplies delivered from St James hospital	✓ Respond	✗ Reject
★ Review code status of all treated personnel	✓ Respond	✗ Reject
★ Coordinate with on-site police chief on incident victims identification.	✓ Respond	✗ Reject

Figure 25: Tasks

An EMT user would receive this and click the 'Respond' option, and this notifies the server that this user is going to respond to that task. On the DVST, the user is then associated with this task, marked as active and their location tracked.

(A non-medical private user could select the 'reject' option to clear that task from their list.)

```
PUT [PulseServer]/task/987/user
123
```

The current task list is rendered on screen (see Figure 25).

The User selects the task they're going to perform and are presented with a data capture screen.

NOTE: The form & fields depend on the type of task selected and this need to be predefined in the system.



### 7.1.5 Data Record

For example, the user selects a triage task and is presented with the screen in Figure 26.

**Figure 26: Triage Reporting**

In the use case, the EMT will place a QR label bracelet on the patient, the scan this with the mobile camera. The 'Scan QR' field is then filled in with this value. They may take a photo of the patient using the device camera, a preview of which will appear beside the camera button.

The EMT may also take notes, typed into the text area at the bottom, or by using the microphone as a Dictaphone recording their additional notes orally, or a combination of both.

They select the risk/severity level for the patient, and the current coordinates are updated and recorded with the triage session.

With the data collected, the user presses the 'Save' button and the record is stored in a local database. If the network connection is available, the app tries to submit this record to the PULSE server. (If no network is available, the record remains saved on the device until we can try again later. These records are persisted in the app storage and remain available after app and device restarts.)

Records are saved as:

```
POST [PulseServer]/record/
{
  "risk": "high",
  "barcode": "ABC123",
```



```
"image": "data:image/jpeg;base64, Lz1qLzRBQ...",  
"audio": "data:audio/mp3;base64, kjxRiUI...",  
"notes": "Test Notes",  
"task": 987,  
"type": "triage",  
"coordinates": "53.3383608, -6.2503854",  
"user": 123,  
"device": "MYID0.06238989788107574"  
}
```

After receiving a HTTP 200/OK message from the server, that record is saved and deleted from the device storage.

Different tasks will require different reporting screens and different data to be sent back.

A simple case could be "Collect medical supplies delivered from St James hospital" requiring just an Accept/In Progress/Done status, and optional notes.

A more detailed form could be for recording the Airline passenger details in the SARS scenario.

As types of tasks are defined, the associated data & forms can be added for reporting.

#### 7.1.6 Privacy & Security

Smartphone applications on iOS and Android run in "**sandboxes**" which separate & protect them from other applications on the same system. This prevents other being able to access any user details or recorded information from the Pulse app.

This is built into mobile applications by default and provides the core level of security and privacy for the application data. [35][36]

For all communication with the server, **HTTPS** is used. This encrypts all traffic between the app and the Pulse server. HTTPS is the standard for securing web connections, to prevent 'sniffing' of data and to ensure the server/device at either end of the conversation are who they say they are. [37]

##### 7.1.6.1 Identification

The OAuth 2.0 standard is used for identifying and authorising pre-approved users from the server, and allowing them access to the system. [7]

##### 7.1.6.2 Recorded Media

With sandboxing, all recorded data is only available to the Pulse app. This includes photos, audio files, etc. When any audio/video/photo data is recorded, the raw data is kept within the application. No jpg files are saved to the photo gallery, no audio files saved to the cloud, etc.

#### 7.1.7 List of core technologies

As noted, the App is built as a single page JavaScript & HTML5 & CSS webapp, this



collection of technologies gives a separation of structure, behaviour & presentation.

The Cordova wrapper gives platform independence for the code with a native installation and hardware access.

The core technologies selected for the implementation of these components are:

- HTML 5: The standard mark-up language of the web, for structure [41]
- JavaScript: The standard scripting language for the web, for behaviour [42]
- CSS: Cascading Style Sheets, the standard styling language of the web, for presentation [43]
- Cordova: Apache Cordova is a set of device APIs that allow a mobile app developer to access native device functions from JavaScript [38].

### 7.1.8 3<sup>rd</sup> Party libraries and licenses

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

**Table 3 – Smartphone App 3rd party libraries and licenses**

Product	Version	Vendor	License
Cordova	5.0	Apache	Apache License Version 2.0
jQuery	2.1	jQuery Foundation	MIT License
Bootstrap	3.3	Bootstrap	MIT License

## 7.2 Public Web Forms

The public web forms are used for more generalised data gathering from the public, without the need for authentication or installing a dedicated app.

There are 2 web forms available currently (Missing Persons & Airline Passenger details), and the system is designed to easily accommodate the addition of further forms without significant extra work.



### 7.2.1 Missing Persons form

**Missing Persons Form (MISPER)**

Surname  Forename

Date of Birth (or approximate age)

Sex ☐ Male ☐ Female

Maiden/Other Name  Nationality

Height

Current Address (and home address if different)

Telephone Contact No.

English Understood? ☐ Yes ☐ No

(If no, please state native language)

Ethnic Appearance ☐ White European ☐ Dark European ☐ Afro-Caribbean ☐ Asian ☐ Oriental ☐ Arab ☐ Unknown

Believed to be with

☐ Additional MISPER form(s) completed

**Figure 27: Missing Person Form**

The missing persons form (see Figure 27) is based on the Garda Síochána MISPER form CB2.

In the case of an emergency (Stadium Crush for example), the URL for the form is distributed to the public (via Twitter, Facebook, etc...).

Any members of the public who may be concerned about someone missing, can fill out the form, upload a photo, and submit the detail to the Pulse system.

On submission, the user then receives a case tracking ID to reference the details in any follow up communication.

```
POST [PulseServer]/record/misper/  
{
```



```
"surname": "Chadwick",
"forename": "Karl",
"dob": "1978-02-24",
"sex": "male",
"photo": "data:image/jpeg;base64, Lz1qLzRBQ...",
  . . .
}

Returns:
{
  "status": "submitted",
  "code": "MP357"
}
```

Recorded data is available to view by a Pulse admin, and as a JSON feed into other systems for matching of data.

## 7.2.2 Airline Passenger Information form

**pulse Airline Passenger Form (AIRPASS)**

Surname  Forename

Seat No.

Date of Birth  Sex ☐ Male ☐ Female

Maiden/Other Name  Nationality

Height

Current Address (and home address if different)

Telephone Contact No.

**Figure 28: Airline Passenger Form**

Similarly, the URL for the Airline Passengers form (see Figure 28) is distributed to all passengers on a flight with a potential incident (SARS case).

Passengers fill out the form with contact details, seat number, etc., and these are submitted to the server:





```
POST [PulseServer]/record/airpass/  
{  
  "surname": "Chadwick",  
  "forename": "Karl",  
  "seat": "1A",  
  . . .  
}
```

```
Returns:  
{  
  "status": "submitted",  
  "code": "AP123"  
}
```

Again, recorded details are made available to view by a Pulse admin, and as a JSON feed to export to another system for analysis & processing.

### 7.2.3 Output

In the following Figures, three possible output of the web forms are listed. Figure 29 shows a sample data feed represented as JSON file while Figure 30 and Figure 31 show the data browser list view and the relative details panel.



```
[
  - {
    - data: {
      vehicle_make: "",
      misper_reasons: "",
      address: "",
      vehicle_model: "",
      vehicle_type: "",
      vehicle_colour: "",
      forename: "Karl",
      nationality: "",
      phone: "",
      surname: "Chadwick",
      dob: "",
      other_name: "",
      believed_to_be_with: "",
      other_language: "",
      vehicle_reg: "",
      vehicle_additional: "",
      height: ""
    },
    created: "2015-11-26",
    modified: "2015-11-26",
    id: 28,
    type: "misper",
    revision: 1
  },
  - {
    - data: {
      vehicle_make: "",
      misper_reasons: "",
      address: "",
      vehicle_model: "",
      vehicle_type: "",
      vehicle_colour: "",
      forename: "yduy",
      nationality: "",
      phone: "",
      surname: "jfdhgfdhgdeUYDduytdUT",
      dob: "jytd",
      other_name: "j",
      believed_to_be_with: "",
      other_language: "",
      vehicle_reg: "",
      vehicle_additional: "",
      height: ""
    },
    created: "2015-11-23",
    modified: "2015-11-23",
    id: 13,
    type: "misper",
    revision: 1
  }
]
```

Figure 29: Sample Data Feed



**pulse Data Browser**

airline basic misper triage

Show 10 entries

Search:

ID	Time	Coordinates	Device	Notes	Passenger Name	Risk	Seat Number
21	2015-11-23	53.3383305,-6.2504743	MYID0.3409196485299617	Test	Test	high	12a

Showing 1 to 1 of 1 entries

Previous 1 Next

Figure 30: Basic Data Browser List View

**pulse Data Browser**

airline basic misper

Show 10 entries


ID	Time	Audio
27	2015-11-23	
26	2015-11-23	
25	2015-11-23	
22	2015-11-23	
20	2015-11-23	
17	2015-11-23	
16	2015-11-23	
9	2015-11-23	

Showing 1 to 8 of 8 entries

**Details**

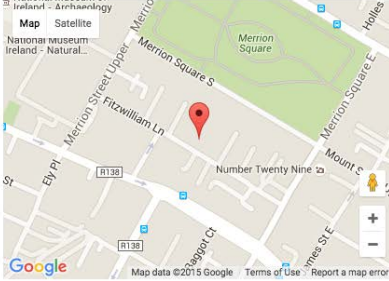
ID 25

Modified 2015-11-23


Image 

Notes Hdhdjif

Task t2

Coordinates 

Risk high

Audio 

Type triage

Barcode 5011054112200

User Karl

Device MYID0.6054535722360015

Close

Figure 31: Data Browser Detail View



#### 7.2.4 List of core technologies

The webforms are simply HTML5, CSS & JavaScript; they're similar to the web component of the Smartphone app, just without the Native Cordova component in this case.

The core technologies selected for the implementation of these components are:

- HTML 5: The standard mark-up language of the web, for structure [41]
- JavaScript: The standard scripting language for the web, for behaviour [42]
- CSS: Cascading Style Sheets, the standard styling language of the we, for presentation [43]

#### 7.2.5 3<sup>rd</sup> Party libraries and licenses

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

**Table 4 – Web Forms 3rd party libraries and licenses**

Product	Version	Vendor	License
jQuery	2.1	jQuery Foundation	MIT License
Bootstrap	3.3	Bootstrap	MIT License



## References

- [1] PULSE Project Deliverable – D4.2 IAT tool
- [2] PULSE Project Deliverable – D4.3 Logistic tool
- [3] PULSE Project Deliverable – D4.4 Surge capacity tool
- [4] PULSE Project Deliverable – D4.5 Training tools
- [5] PULSE Project Deliverable – D4.6 Post crisis evaluation tool
- [6] PULSE Project Deliverable – D4.7 Event evaluation for biological event
- [7] OAuth2 spec, <http://oauth.net/2/>
- [8] PULSE Project Deliverable – D2.2 Use case specification
- [9] PULSE Project Deliverable – D2.1 Requirements specification
- [10] Jetty, <http://www.eclipse.org/jetty/>
- [11] Jersey, <https://jersey.java.net/>
- [12] JSR 311 JAX-RS: Java™ API for RESTful Web Services, version 1.1, <https://jsr311.java.net/nonav/releases/1.1/spec/spec.html>
- [13] JSR 339 The Java™ API for RESTful Web Services, version 2.9, <https://jcp.org/aboutJava/communityprocess/final/jsr339/index.html>
- [14] Apache Oltu, <https://oltu.apache.org/>
- [15] ApacheDS, <https://directory.apache.org/apacheds/>
- [16] RFC 2617, HTTP Authentication: Basic and Digest Access Authentication, <https://tools.ietf.org/html/rfc2617>
- [17] RFC 7519, JSON Web Token (JWT), <https://tools.ietf.org/html/rfc7519>
- [18] AMPQ lib, <http://www.squaremobius.net/amqp.node/>
- [19] StompJS, <http://www.jmesnil.net/stomp-websocket/doc>
- [20] DesmoJ, <http://desmoj.sourceforge.net/>
- [21] 118, <http://www.ares118.it/>
- [22] Maurer algorithm, [http://www.usl3.toscana.it/allegati/norma\\_manifestazioni.pdf](http://www.usl3.toscana.it/allegati/norma_manifestazioni.pdf)
- [23] RFC 2617, <https://www.ietf.org/rfc/rfc2617.txt>
- [24] JWT, <http://jwt.io>
- [25] Google Maps Web Services API, <https://developers.google.com/maps/web-services/overview>
- [26] Loopback, <http://loopback.io/>
- [27] JWT, <http://jwt.io/>
- [28] AMPQ, <https://www.amqp.org/>
- [29] Angularjs, <https://angularjs.org/>
- [30] Angular Material, <https://material.angularjs.org/latest/>
- [31] STOMP, <https://stomp.github.io/>
- [32] RabbitMQ, <https://www.rabbitmq.com/>



- [33] EasyRule, <http://www.easyrules.org/>
- [34] Hibernate ORM, <http://hibernate.org/orm/>
- [35] Apple iOS Sandboxing, <https://developer.apple.com/app-sandboxing/>
- [36] Android Sandboxing, <https://source.android.com/devices/tech/security/>
- [37] HTTPS, <http://searchsoftwarequality.techtarget.com/definition/HTTPS>
- [38] Cordova, <https://cordova.apache.org/>
- [39] jQuery, <http://jquery.com/>
- [40] Bootstrap, <http://getbootstrap.com/>
- [41] HTML5, <http://www.w3.org/TR/html5/>
- [42] Javascript, <https://javascript.spec.whatwg.org/>
- [43] CSS, <http://www.w3.org/TR/CSS2/>