D6.14

Scenario Specification, Scenario Management Specification and Scenario and Situation Metrics - Draft

Instrument	Collaborative Project
Call / Topic	H2020-SEC-2016-2017/H2020-SEC-2016-2017-1
Project Title	Multi-Hazard Cooperative Management Tool for Data Exchange, Response Planning and Scenario Building
Project Number	740689
Project Acronym	HEIMDALL
Project Start Date	01/05/2017
Project Duration	42 months
Contributing WP	WP 6
Dissemination Level	PU
Contractual Delivery Date	M12
Actual Delivery Date	30/04/2018

Editor	Monika Friedemann (DLR-DFD)
Contributors	Martin Mühlbauer, Nico Mandery (DLR-DFD), Jordi Vendrell Flotats (PCF), Paolo Campanella, Roberto Rudari, Lauro Rossi, Eva Trasforini (CIMA), Sefik Muhic, Kim Lintrup (FBBR), Andreas Baur-Ahrens, Friedrich Gabel (EKUT), Tomaso de Cola, Benjamin Barth (DLR-KN), Jesper Marcussen (FBBR)

Document History			
Version	Date	Modifications	Source
0.1	30/11/2017	TOC	DLR
0.2	16/04/2018	QA version	DLR
0.3	19/04/2018	QA-reviewed version	FBBR
0.4	20/04/2018	QA-reviewed version	DLR
1.0.F	30/04/2018	Final Version	DLR

Table of Contents

List of Figures

List of Tables

List of Acronyms

AB	Advisory Board
AOI	Area of Interest
API	Application Programming Interface
C&C	Command & Control Centre
CAP	Common Alerting Protocol
CIMA	Centro Internazionale in Monitoraggio Ambientale – Fondazione CIMA
DB	Database
DES	Decision Support Service
DLR	Deutsches Zentrum für Luft- und Raumfahrt e.V.
DLR-DFD	Deutsches Zentrum für Luft- und Raumfahrt e.V.; German Remote Sensing Data Center
DLR-KN	Deutsches Zentrum für Luft- und Raumfahrt e.V.; Institute of Communications and Navigation
EDXL	Emergency Data Exchange Language
EKUT	Eberhard Karls Universität Tübingen
EO	Earth Observation
EUW	End User Workshop
FBBR	Frederiksborg Brand & Redning
FCP	Forward Command Post
FLI	Fireline Intensity
FR	First Responder
FRS	Fire and Rescue Service
GIS	Geographic Information System
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IC	Incident Commander
ISA	Impact Summary
ISAS	Impact Summary Service
JSON	JavaScript Object Notation
OGC	Open Geospatial Consortium
OS	Operating System

PCF	Fundació d'Ecologia del Foc i Gestió d'Incendis Pau Costa Alcubierre
PE	Plan Execution
PF	Plan Formation
REST	Representational State Transfer
ROS	Rate of Spread
RVA	Risk and Vulnerability Assessment
SA	Situation Assessment
SITREP	Situation Reporting Service
SM	Scenario Management
SMAC	Scenario Matching Service
SMES	Scenario Management Service
SOAP	Simple Object Access Protocol
SP	Service Platform
TOC	Table of Contents
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WP	Work Package

Intentionally blank

Executive Summary

This document presents the scenario definitions and data model elaborated in close collaboration with all end-user and technical partners in the HEIMDALL project. In addition, the developed technical solutions to support the practitioners in scenario building and matching are described. The main objective of this document is to provide a technical specification which enables technical contributors and partners to understand how to develop, deploy, configure and use the scenario management component. Therefore, topics include the external and internal architecture design, interfaces, data structures, formats, functionality, methods, configuration and software issues.

The main task contributing to this deliverable is T6.5 – Scenario Management. However, significant contributions were made by tasks T3.1 – Stakeholder Management and T2.4 – Service Concept Specification and System Architecture as disaster scenarios are a major pillar in the work of the end-user organizations and in the overall HEIMDALL system.

Core element for effective situation assessment and plan formation prior and during an incident in HEIMDALL is a scenario. It assembles all related information that has been collected, for instance incident information, simulation results, risk and impact assessment information, decisions made, measures taken and lessons learnt. Based on a scenario practitioners generate situation reports for analysis, reporting and archiving purposes and for sharing them with their colleagues or other agencies. By the use of technical capabilities to generate hypothetical (simulated) scenarios with alternative parameters, decision makers are supported in the identification of the best-possible response plan for the given situation. This response plan can then be connected to the ’real‘ scenario. The HEIMDALL scenario management module and its submodules provide functionality to access and manage scenarios, response plans, lessons learnt, decisions and measures. The scenario matching submodule aims at supporting decision makers with capabilities for the comparison of a situation with historic and fictive scenarios, their evolution, performed response activities and lessons learnt.

1 Introduction

During different multilateral discussions in the HEIMDALL project practitioners have pointed out that there is a need for an improved standardization and sharing of response plans and disaster scenarios, both, across different groups within an organization and to other involved stakeholders. Use cases for sharing of scenario information and response plans include their usage in staff meetings, their distribution to incident commanders at forward command posts (FCPs), to the media and their application in exercises and training.

This document describes efforts performed and results produced so far in the HEIMDALL project in finding and designing technical solutions which support the end users' response planning and scenario building activities, suitable to all involved perspectives. The document focuses on the different scenario data resources as well as the Scenario Management Service (SMES) and Scenario Matching Service (SMAC).

At this early stage of the HEIMDALL project, the focus of this document lies in a first component design with a basic specification of technical details. The HEIMDALL project aims at a collaborative design which is a methodology that involves people who will be affected by new technologies throughout all design phases [1]. Unlike traditional approaches to develop information systems of having a fixed product and/or system idea that does not change or evolve as the development process take place, an agile approach is a response to the need for a flexible and iterative process to be able to consider unexpected changes [2]. In close cooperation with the relevant stakeholders, both the consortium end-user partners and the members of the AB the component design will evolve. For example, intermediary results of system developments will be presented at end user workshops and iterated together with the end-user partners. Deliverable D6.15 due in M38 will present the mature scenario management component design together with a detailed technical specification.

In particular, this document is organised as follows:

- Section 2 specifies the technical requirements for the SM modules.
- Section 3 describes the SM in the context of the overall HEIMDALL system, inputs and outputs and interfaces with other HEIMDALL components.
- Section 4 focusses on the SM building blocks, functionalities and workflows.
- Section 5 presents the detailed scenario, SMES and SMAC specifications and also includes software and database aspects where applicable.
- Finally, section 6 summarizes the work carried out so far and gives an outlook to the work that still needs to be done.

2 Technical Requirements

2.1 Interface Requirements

2.1.1 **Hardware Interfaces**

The SM services shall be accessible via Ethernet. The SM services must support the use of VPN.

2.1.2 **Software Interfaces**

The SM services and components shall be deployed as containers and/or virtual machines (VMs).

2.1.3 **Communication Interfaces**

The SM services shall be designed as RESTful web services allowing clients to communicate with them using common HTTP methods (e.g. GET, PUT) and/or for georeferenced data OGC methods (based on HTTP).

Focused on providing a lightweight and easily consumable output format, the data shall be exchanged using JSON serialization and the formats specified by the respective OGC standards.

2.2 Functional Technical Requirements

2.2.1 **Short-Term Features**

Table 2-1: Technical Requirement TR_Scen_1

Requirement ID:	TR_Scen_1
Related SR(s):	• Sys_Scen_1
Description: The SMES shall allow for creating a scenario.	
Rational: The system must be capable of handling scenarios.	
Stimulus: The user requests the creation of the respective scenario over the user interface (UI).	
Response: Success status message and created scenario resource representation if successful. Otherwise error status message.	
Verification Criterion: The user is able to create a scenario.	
Notes: none	

Table 2-2: Technical Requirement TR_Scen_2

Requirement ID:	TR_Scen_2
Related SR(s):	• Sys_Scen_1

Description: The SMES shall allow for creating a scenario from a potential or real hazard. Hazard information, current weather information and weather forecast shall be able to be used as input parameters for the scenario creation.
Rational: The system must be capable of handling scenarios.
Stimulus: The user specifies the required input parameters over a UI and requests the creation of the respective scenario.
Response: Success status message and created scenario resource representation if successful. Otherwise error status message.
Verification Criterion: The user is able to create a scenario from a set of scenario information.
Notes: none

Table 2-3: Technical Requirement TR_Scen_3

Requirement ID:	TR_Scen_3
Related SR(s):	• Sys_Scen_1
Description: The data structure of a scenario shall be compatible with EDXL-SitRep or CAP.	
Rational: The system must be capable of handling scenarios compatible with standardized or commonly used situation reporting formats.	
Stimulus: The user requests the creation of the respective scenario in a format compatible with EDXL-SitRep or CAP.	
Response: Success status message and created scenario resource representation if successful. Otherwise error status message.	
Verification Criterion: The user is able to create a scenario in a format compatible with EDXL-SitRep or CAP.	
Notes: none	

Table 2-4: Technical Requirement TR_Scen_4

Requirement ID:	TR_Scen_4
Related SR(s):	• Sys_Scen_2
Description: The SMES shall allow for accessing scenarios.	
Rational: The system must be capable of handling scenarios.	

Stimulus: The user selects and opens a scenario over a UI.
Response: Success status message and requested scenario resource representation if successful. Otherwise error status message.
Verification Criterion: The user is able to access the scenario.
Notes: none

Table 2-5: Technical Requirement TR_Scen_5

Requirement ID:	TR_Scen_5
Related SR(s):	• Sys_Scen_3 • Sys_Scen_8
Description: The SMES shall allow for modifying scenario parameters and saving changes to a scenario.	
Rational: The system must allow for simulating different conditions and situation evolutions in order to support the identification of appropriate response strategies and must be capable of handling scenarios.	
Stimulus: The user modifies scenario parameters and saves the scenario over a UI.	
Response: Success status message and modified scenario resource representation if successful. Otherwise error status message.	
Verification Criterion: The user is able to save changes to the scenario. The scenario contains the modified information after saving.	
Notes: none	

Table 2-6: Technical Requirement TR_Scen_8

Requirement ID:	TR_Scen_8
Related SR(s):	• Sys_Scen_6
Description: The SMES shall allow for defining a scenario either as “real” or “simulated” scenario.	
Rational: The system must be capable of distinguishing real situation scenarios from simulated planning scenarios based on a real or fictive hazard.	
Stimulus: The user sets the scenario type over a UI.	
Response: Success status message and modified scenario resource representation if successful. Otherwise error status message.	
Verification Criterion: The user is able to set the scenario type.	

Notes: none

Table 2-7: Technical Requirement TR_Scen_12

Requirement ID:	TR_Scen_12
Related SR(s):	• Sys_Scen_12
Description: The SMES shall provide means to associate additional information to a scenario such as internal and external data, simulation results or impact summary information.	
Rational: The system must be capable of handling scenarios and all information associated to them.	
Stimulus: The user associates EO products to a scenario over a UI.	
Response: Success status message and respective scenario resource representation which includes reference to associated information if successful. Otherwise error status message.	
Verification Criterion: The user is able to associate additional information to a scenario.	
Notes: none	

Table 2-8: Technical Requirement TR_Scen_15

Requirement ID:	TR_Scen_15
Related SR(s):	• Sys_Scen_15
Description: The SMES shall provide means for accessing EO products associated to a scenario.	
Rational: The system must be capable of providing access to EO products associated to a scenario.	
Stimulus: The user requests a scenario with associated EO products over a UI.	
Response: Success status message and respective scenario resource representation which includes reference to associated EO products if successful. Otherwise error status message.	
Verification Criterion: The user is able to access the associated EO products over the reference.	
Notes: can be either verified by sending a HTTP GET request for a scenario resource representation with references to associated EO products which shall be followed to access associated information (via browser) or in the UI by showing the user associated EO products for that scenario.	

Table 2-9: Technical Requirement TR_Scen_17

Requirement ID:	TR_Scen_17
Related SR(s):	• Sys_Scen_17
Description: The SMES shall allow for creating, storing, accessing, editing and deleting lessons learnt for a scenario.	
Rational: Lessons learnt shall be used to improve the emergency management process.	
Stimulus: The user creates stores and accesses a lesson learnt for a scenario. Afterwards, he modifies it. Finally, he deletes it.	
Response: Success status message and respective scenario resource representation which includes lesson learnt if successful (no lesson learnt if it has been deleted). Otherwise error status message.	
Verification Criterion: The user is able to create, store, access, modify and delete a lesson learnt for a scenario.	
Notes: none	

2.2.2 **Mid-Term Features**

Table 2-10: Technical Requirement TR_Scen_6

Requirement ID:	TR_Scen_6
Related SR(s):	• Sys_Scen_4
Description: The SMES shall be able to save a new scenario from a given scenario (“copy” functionality).	
Rational: The system must be capable of handling scenarios.	
Stimulus: The user saves a copy of a scenario over a UI. The user provides a new unique scenario name.	
Response: Success status message and copied scenario resource representation if successful. Otherwise error status message.	
Verification Criterion: The user is able to save a new scenario from a given scenario.	
Notes: none	

Table 2-11: Technical Requirement TR_Scen_7

Requirement ID:	TR_Scen_7
Related SR(s):	• Sys_Scen_5

Description:
The SMES shall be able to delete scenario.
Rational: The system must be capable of handling scenarios.
Stimulus: The user deletes a scenario over a UI.
Response: Success status message. Otherwise error status message.
Verification Criterion: The user is able to delete a scenario.
Notes: none

Table 2-12: Technical Requirement TR_Scen_9

Requirement ID:	TR_Scen_9
Related SR(s):	• Sys_Scen_7
Description: The SMES shall allow for distinguishing scenarios which address the SA phase from scenarios which address the PF phase.	
Rational: The system must be capable of distinguishing scenarios addressing different phases in the decision making process.	
Stimulus: The user defines a scenario which addresses the SA phase over a UI. Then, the user defines a scenario which addresses the PF phase over a UI.	
Response: Success status message and respective scenario resource representation if successful. Otherwise error status message.	
Verification Criterion: The user is able to set the scenario phase.	
Notes: This can be achieved either by letting the user decide or by letting the system decide by checking whether the respective PF products (e.g. response plans, decisions) have been specified for the scenario or not	

Table 2-13: Technical Requirement TR_Scen_10

Requirement ID:	TR_Scen_10
Related SR(s):	• Sys_Scen_9
Description: The SMES shall provide means for saving a “snapshot” of a scenario and all related information at different points in time. The snapshot shall be tagged with the scenario ID and a snapshot ID in order to correlate different “snapshots” of a scenario.	
Rational: The system must be capable of recording the scenario evolution in terms of scenario status at different points in time for planning and training purposes.	

Stimulus: The user saves a “snapshot” of a scenario over a UI.
Response: Success status message and respective scenario snapshot resource representation if successful. Otherwise error status message.
Verification Criterion: The user is able to save a “snapshot” of a scenario and all related information at different points in time.
Notes: The snapshot ID should be an integer number starting with 0 for each scenario ID. This counter should be increased for each snapshot taken.

Table 2-14: Technical Requirement TR_Scen_11

Requirement ID:	TR_Scen_11
Related SR(s):	• Sys_Scen_11
Description: The SMES shall provide means for accessing all information associated to a scenario.	
Rational: The system must be capable of handling scenarios and providing all information associated to them.	
Stimulus: The user requests a scenario with associated information over a UI.	
Response: Success status message and respective scenario resource representation which includes reference to associated information if successful. Otherwise error status message.	
Verification Criterion: The user is able to access the associated information over the reference.	
Notes: Can be either verified by sending a HTTP GET request for a scenario resource representation with references to associated information which shall be followed to access associated information (via browser) or in the UI by showing the user additional information for that scenario.	

Table 2-15: Technical Requirement TR_Scen_13

Requirement ID:	TR_Scen_13
Related SR(s):	• Sys_Scen_13
Description: The SMES shall provide means for deleting information associated to a scenario.	
Rational: The system must be capable of handling scenarios and all information associated to them.	
Stimulus: The user removes associated information from a scenario over a UI.	
Response: Success status message and respective scenario resource representation which does not include reference to associated information if successful. Otherwise error status	

message.
Verification Criterion: The user is able to delete associated information.
Notes: none

Table 2-16: Technical Requirement TR_Scen_14

Requirement ID:	TR_Scen_14
Related SR(s):	• Sys_Scen_14
Description: The SMES shall provide means for modifying information associated to a scenario.	
Rational: The system must be capable of handling scenarios and all information associated to them.	
Stimulus: The user modifies associated information to a scenario over a UI.	
Response: Success status message and respective scenario resource representation which includes modified reference to associated information if successful. Otherwise error status message.	
Verification Criterion: The user is able to modify associated information.	
Notes: none	

Table 2-17: Technical Requirement TR_Scen_18

Requirement ID:	TR_Scen_18
Related SR(s):	• Sys_Scen_19
Description: The SMES shall allow for creating, storing, accessing, editing and deleting of response plans.	
Rational: The system must be capable of handling response plans.	
Stimulus: The user creates a response plan and accesses it over a UI. Afterwards, he modifies the plan. In a last step he deletes it.	
Response: Success status message and respective response plan (no response plan if it has been deleted). Otherwise error status message.	
Verification Criterion: The response plan can be created, stored, accessed, modified and deleted.	
Notes: none	

Table 2-18: Technical Requirement TR_Scen_19

Requirement ID:	TR_Scen_19
Related SR(s):	• Sys_Scen_20
Description: The SMES shall allow response plans available in the system to be updated with lessons learnt.	
Rational: Lessons learnt shall be used to improve response plans.	
Stimulus: The user modifies a response plan based on information defined in a lesson learnt.	
Response: Success status message and respective response plan. Otherwise error status message.	
Verification Criterion: The response plan can be modified.	
Notes: none	

Table 2-19: Technical Requirement TR_Scen_20

Requirement ID:	TR_Scen_20
Related SR(s):	• Sys_Scen_21
Description: The SMES shall provide lessons learnt in a format that can be shared with other instances of the system. Standard formats shall be used as far as possible.	
Rational: By adopting existing standards where possible data interoperability is increased. In order to share lessons learnt they must be provided in such a format.	
Stimulus: The SP requests a lesson learnt for a specific scenario from the SMES in a format that can be shared with other instances of the system.	
Response: Lesson learnt in a format that can be shared with other instances of the system.	
Verification Criterion: Lesson learnt can be requested for a specific scenario in a format that can be shared with other instances of the system.	
Notes: There might be no standardized format that covers all data of the lessons learnt; not all standard formats can be considered.	

Table 2-20: Technical Requirement TR_Scen_22

Requirement ID:	TR_Scen_22
Related SR(s):	• Sys_Scen_10 • Sys_SADS_8
Description: The SMAC shall offer means to compare input parameters with the available scenarios.	

Comparison shall be based on parameters describing the hazard, current weather, weather forecast and/or estimated consequences and their probability. Parameters shall be configurable. The result of the comparison shall be a list of scenarios similar to the input parameters. The list shall be ordered by similarity. In addition, the system shall provide comparison metadata for each matched scenario including the degree of difference to the input parameters.
Rational: Scenarios shall be comparable to each other and to the real situation for analysis, planning and training.
Stimulus: The user requests scenario matching over the UI. Input parameters and parameters of the available scenarios must be as complete as possible as comparison can only take place for the available parameters. There must be a sufficient amount of scenarios for performing a relevant matching.
Response: A sorted list of similar scenarios is returned if any found, otherwise a “Not found” status message is returned.
Verification Criterion: To a given scenario a correct sorted list of similar scenarios is returned with the degree of difference of each matched scenario to the input parameters which can be displayed in a map or included in situation reports. The input parameters should be alternated in order to test all the different matching functions.
Notes: none

Table 2-21: Technical Requirement TR_Scen_23

Requirement ID:	TR_Scen_23
Related SR(s):	• Sys_SADS_10
Description: The SMAC shall provide assigned response plans, lessons learnt, decisions made and performed measures for every matched scenario if these are available for that scenario.	
Rational: Performed actions in similar scenarios shall be presented as possible actions to be taken for planning and training purposes This functionality is not applicable to real situations as these require a dynamic assessment of possible actions performed by staff on tactical and operational level.	
Stimulus: Automatically during scenario matching if option selected.	
Response: A list of assigned response plans, lessons learnt, decisions made and performed measures for every matched scenario if these are available. Empty element in case none available.	
Verification Criterion: For every matched scenario SMAC returns a list of assigned response plans, lessons learnt, decisions made and performed measures if these are available.	
Notes: none	

2.2.3 **Long-Term Features**

Table 2-22: Technical Requirement TR_Scen_16

Requirement ID:	TR_Scen_16
Related SR(s):	• Sys_Scen_16
Description: The SMES shall provide means for adding information provided by the public to scenarios.	
Rational: Users must be able to connect information provided by the public with scenarios.	
Stimulus: The user associates information provided by the public to a scenario over a UI.	
Response: Success status message and respective scenario resource representation which includes reference to associated information if successful. Otherwise error status message.	
Verification Criterion: The user is able to access the associated information provided by the public over the reference.	
Notes: Can be either verified by sending a HTTP GET request for a scenario resource representation with references to associated information provided by the public which shall be followed to access associated information (via browser) or in the UI by showing the user associated information provided by the public for that scenario.	

Table 2-23: Technical Requirement TR_Scen_21

Requirement ID:	TR_Scen_21
Related SR(s):	• Sys_Scen_22
Description: The SMES shall allow for assigning a scenario to the PE phase.	
Rational: The system must be capable of distinguishing scenarios addressing different phases in the decision making process.	
Stimulus: The user defines a scenario which addresses the PE phase over a UI.	
Response: Success status message and respective scenario resource representation if successful. Otherwise error status message.	
Verification Criterion: The user is able to set the PE scenario phase.	
Notes: none	

2.3 Other Requirements

2.3.1 **Short-Term Requirements**

No short-term non-functional requirements have been identified.

2.3.2 **Mid-Term Requirements**

No mid-term non-functional requirements have been identified.

2.3.3 **Long-Term Requirements**

No long-term non-functional requirements have been identified.

3 Reference Architecture

This section describes the scenario management module including the scenario matching module in the context of the overall HEIMDALL architecture. Afterwards, the different inputs and outputs expected from the module are listed. Finally, the section describes the interfaces needed to allow the module communicate with the other modules.

3.1 HEIMDALL Overall Architecture

Figure 3-1 illustrates the HEIMDALL overall architecture highlighting the modules belonging to the SM. The SM as well as its sub-modules is treated as a “black box” within the overall architecture. As can be observed in the diagram the SM modules perform all communication and interactions with the other components of HEIMDALL through the Service Platform (SP). Interface 8 (I8) serves as the general interface.

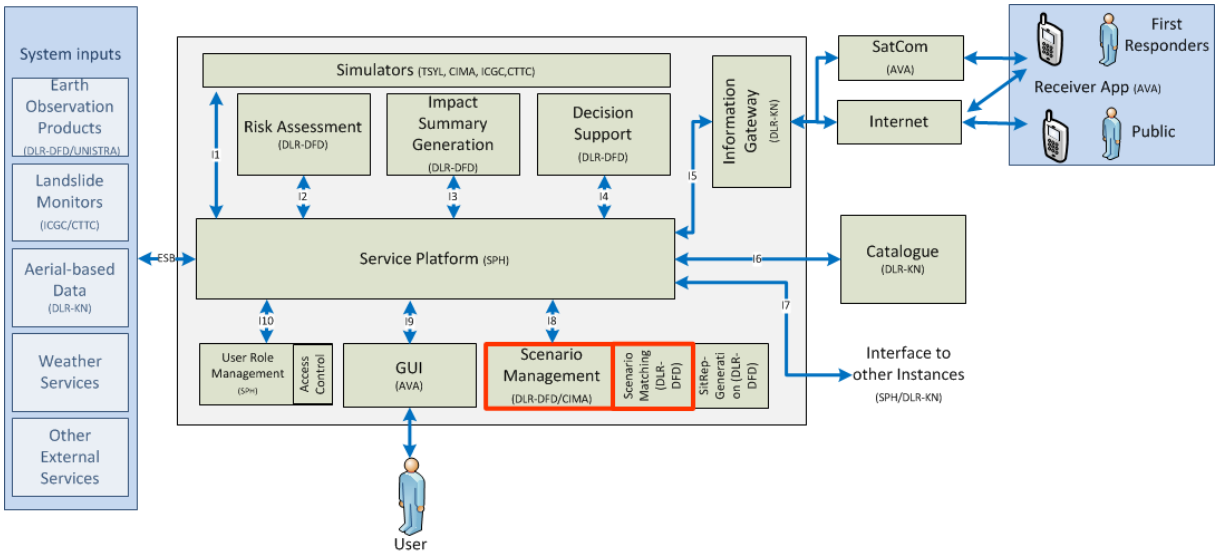

Figure 3-1: HEIMDALL overall architecture

SM encompasses three different services: The SMES, the SMAC and the Situation Report Generation Service (SITREP). This technical specification addresses the SMES and SMAC modules. The SITREP module will be described in detail in D6.7 [3]. Consequently, the remainder of this deliverable leaves out specifics related to the SITREP module.

The SM enables clients (e.g. the UI) to create, store, retrieve, edit and delete scenarios. In addition, a scenario shall be able to be copied, e.g. in order to derive a fictive scenario from a real situation. Scenarios and scenario snapshots are stored in a database called *Scenario Repository*. Clients are able to compare a set of input parameters to scenarios stored in the repository in order to receive a list of best-matching scenarios.

Furthermore, the SM provides functionality to enhance scenarios with associated incident information provided by the different internal and external data sources, simulation results, risk and impact summary information, response plans, decisions made, measures taken and lessons learnt. Users shall be able to access all information associated to a scenario in order to explore them in detail.

At any time during a scenario lifecycle a snapshot can be generated for analysis, reporting and archiving purposes. Based on a sequence of scenario snapshots end users are able to retrace the scenario evolution. Scenario snapshots can be transformed into a standard-based situation report format. This functionality will be provided by the SITREP module.

3.2 Inputs and Outputs

Table 3-1 gives an overview of SM inputs and outputs with a short description respectively. A detailed description of SM inputs, outputs and formats can be found in section 5.1 Scenario Specification.

Table 3-1: Scenario management inputs and outputs

Output	Short description	Inputs needed	Format
Scenario	Scenario object describing a hazard (or incident) and conditions. Additional information can be associated to a scenario by reference	Scenario parameters, weather and environmental conditions, associated information (e.g. internal and external data, simulation results, risk assessment products, impact summary, decision support information, response plans, lessons learnt)	Descriptive JSON-based format Associated information as URI
Scenario snapshot	Instance of a scenario at a specific point in time including all associated information	Scenario	Descriptive JSON-based format Associated information as URI
Response plan	Incident response plan for a specific scenario	Inputs should allow the system the compilation of response plans	Descriptive JSON-based format
Decision	Decision taken by emergency service roles in charge, i.e. C&C, I.C. associated to a scenario	Inputs should allow the system the compilation of decisions	Descriptive JSON-based format
Measure	Countermeasure or prevention measure taken during a scenario such as sandbag location and height, fire break, etc.	Inputs should allow the system the compilation of measures. Inputs should be specified appropriately so that they are useful as an input to simulation runs	Descriptive JSON-based format
Lesson learnt	Lesson learnt associated to a scenario	Inputs should allow the system the compilation of lessons learnt	Descriptive JSON-based format
List of similar scenarios (for instance a list of the representative scenarios)	List of similar scenarios including measures taken in these scenarios, sorted by mismatch Mismatch is a numerical value in order to quantify the difference of a scenario to input parameters	Expert criteria for matching Input parameters or input scenario snapshot	Descriptive JSON-based format Scenarios as URI

3.3 Interfaces with other HEIMDALL components

The SM provides access to data resources and functionality by use of different RESTful web services. Figure 3-1 shows I8 as the major interface connecting the SP with the SM modules (see Table 3-2 for details on I8).

Table 3-2: Interfaces with other components

Interface	**Short description**	**Methods**	**Protocol**
I8	RESTful web service interface	GET, POST, PUT, DELETE	HTTP(S)

The SM modules as web services provide a REST API for accessing, creating, updating and deleting of their data resources. Any input needed by the modules must be attached as a data resource by the client to the request.

4 Module Functionality

4.1 Scenario Management Architecture

The SM consists of (1) the Scenario Repository, (2) the SMES for the management of its data resources including functionality for taking scenario snapshots and (3) the SMAC. Figure 4-1 gives an overview over the SM building blocks and their relation to each other.

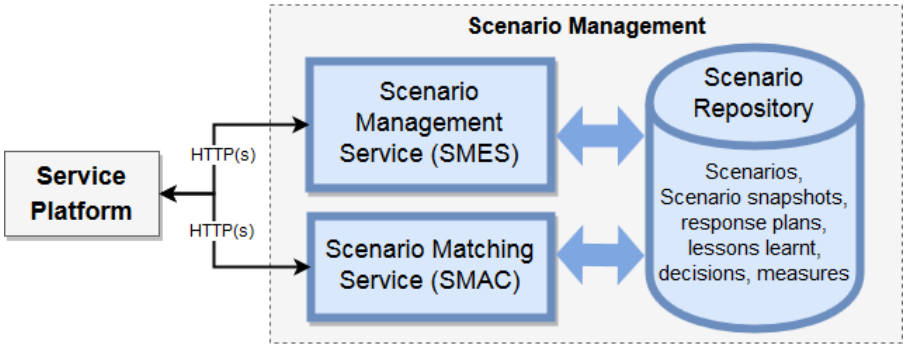

Figure 4-1: Scenario management modules

The Scenario Repository represents the database containing scenarios, scenario snapshots, response plans, lessons learnt, decisions and prevention and mitigation measures. The SMES takes care of the management of these resources and gives access to them over its REST API. In addition, this API provides methods for users to associate additional information to scenarios such as simulation results, response plans, lessons learnt, etc. Furthermore, the snapshot functionality is part of the SMES and accesses directly the Scenario Repository for creating a snapshot instance of a scenario including all associated information.

The SMAC module aims at supporting decision makers with capabilities for the comparison of a situation with historic and fictive scenarios, their evolution, performed response activities and lessons learnt. Information on similar scenarios and associated information on decisions and prevention and mitigation measures taken in these scenarios, their positive or negative evaluation and lessons learnt help decision makers to identify possible measures. The SMAC module accesses the Scenario Repository directly in order to query for respective similar scenarios. Results can be accessed by external clients over the SMAC module's REST API.

4.2 Data Model

The scenario management component manages and gives access to the following data resources: scenarios, scenario snapshots, response plans, lessons learnt, decisions and prevention and mitigation measures. Figure 4-2 illustrates the relations between these data entities.

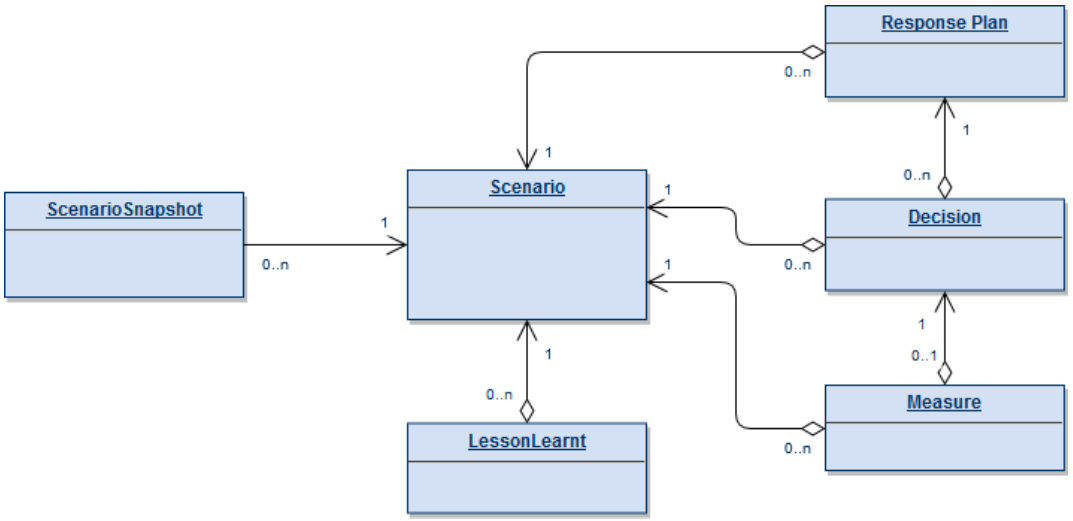

Figure 4-2: Scenario data model

Accordingly, a scenario can be composed of multiple response plans, lessons learnt, decisions and measures and can be connected to multiple scenario snapshots. These snapshots are instances of that scenario (including all related response plans, lessons learnt, etc.). A snapshot extends the scenario by snapshot-specific information such as a snapshot timestamp and description. It is worth noting that the depicted and described relations between scenarios and response plans may be modified in the course of the project. For example, we expect the first scenario management releases and demonstrations to be the trigger for further discussions with the end users. User feedback will be integrated in future versions of the scenario data model and specified in D6.15.

4.3 REST API Conventions

The SM REST APIs communicate by using JSON as the primary data-interchange format. The open-standard format JSON offers support for common datatypes like strings, integer numbers and floats as well as basic data structures in the form of arrays (lists) and objects (hashmaps). In order to design consistent data formats in HEIMDALL, we intend to adhere to common conventions. For example, the JSON API specification provides an elaborated and comprehensive approach for building efficient APIs [4]. By following shared conventions, system developers can increase productivity, take advantage of generalized tooling, and focus on what matters: their application. JSON API covers creating and updating resources as well, not just how to format them. The decision on the most suitable conventions is currently under discussion as the first implementation phase approaches. In the meantime, HEIMDALL system developers are invited to read the JSON API documentation for general and detailed information on how a REST API could work.

4.4 Workflows

This section describes a first set of workflows in order to foster a deeper understanding of how the scenario management component works. In future versions of this document, further workflows will likely be added.

4.4.1 **Event Association Workflow**

In case the user in front of the UI becomes aware of a new event (e.g. new weather information available, new EO products available, new 112 call) the event can be connected to a scenario. Figure 4-3 illustrates the general event-association workflow.

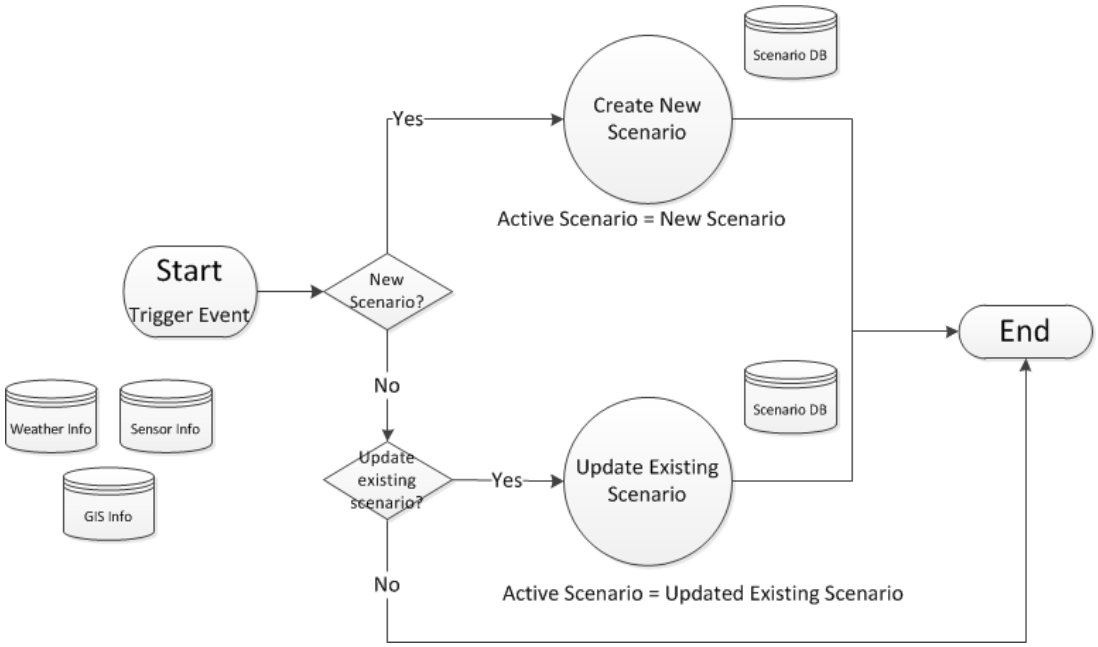

Figure 4-3: Workflow for the association of events to scenarios

The workflow starts with an incoming new piece of information. Then, a decision must be taken on whether the information item belongs to an existing scenario or to a new one. In the former case, the existing scenario must be modified. In the latter case the creation of a new scenario must be requested from the SMES. The diagram does intentionally not imply whether decisions and operations are executed manually by the user or automatically by the system. Our objective is not to keep the user busy with assignment operations, in particular for frequently updated weather data. At the same time, practitioners should be in control of certain assignments according to their individual expert knowledge and profile. Concepts for a user-friendly combination of automatic assignments of information to scenarios and manual decisions are being elaborated. For the first system release, manual assignment is foreseen. This and subsequent releases will give the end users the possibility to work with the system. Based on their feedback, the workflow control concept will be increasingly fine-tuned.

4.4.2 **Scenario Matching Workflow**

Figure 4-4 outlines the scenario matching workflow. The scenario matching process makes use of pre-defined expert criteria and metrics in order to compile a list of similar scenarios. These expert criteria and metrics are customized for the agency in use of the system. In addition, the user is able to modify these criteria if necessary. After criteria selection the scenario matching process takes place.

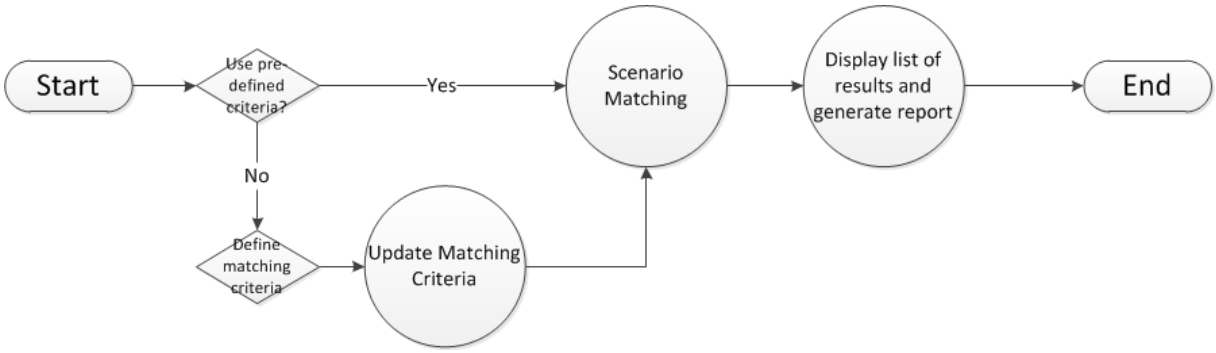

Figure 4-4: Scenario matching workflow

5 Technical Specification

5.1 Scenario Specification

Some of the least effective response plans are mere collections of generic procedures. Generic procedures can be used for events that are unpredictable. However, most emergencies follow a generally known pattern and are composed of elements that have some basis in past disasters of crises. Therefore, plans should instead be firmly based on scenarios of what is likely to happen. There are several justifications for basing plans on scenarios of what is likely to happen [5]:

- If a contingency can be foreseen, it should be;
- Most natural hazards are repetitive and occur in definable geographical areas;
- One of the principle tasks of a response plan is to match urgent needs with available resources. This can only be done efficiently if these needs are foreseen as far as possible;
- The creation of scenarios can help one to explore the consequences of impacts and response actions.

5.1.1 **Scenario Defined**

The Oxford English Dictionary defines ‘scenario’ as “a postulated sequence of (imagined, usually future) events”. Terminology and definitions differ from one country to another, even from one organization to another in the same country. In order to define a baseline set of definitions which would be used by all (end-user and technical) partners throughout the HEIMDALL project the building blocks of a disaster had to be identified. A disaster scenario should consist of a hazard and current conditions in the local area, information about physical impacts, and what measures, resources and forms of organization are needed in order to reduce the consequences.

Therefore, in the HEIMDALL project, we define a scenario as: “*a real or hypothetical situation consisting of a hazard and the current or potential conditions in the local area, information on current or potential physical impacts on people, property, environment and the society as a whole, what response plans, measures and decisions have been taken or are needed to reduce the consequences and lessons learnt tracked*.”

5.1.2 **Scenario Life Cycle**

When building a scenario, the real or hypothetical hazard is taken as a starting point together with the initial conditions in the local area. The scenario is taken forward on the basis of a logical unfolding of what is likely to happen in such a situation. It is usually appropriate to consider the scenario as a form of situation and to define different scenarios for several sets of circumstances, such as daytime and night-time, a working day and a holiday, etc. [5].

For response planners it is interesting to know the impact of a hazard to calculate the resources needed and which organization to involve. Accordingly, the scenario should progress from physical impact and human vulnerability to emergency response. This ensures that emergency resources are not put at risk (e.g. that C&Cs and FCPs are not exposed to flooding or that vital transportation ways remain functional).

During a disaster situation, scenarios can act as a recording of actual events and actions as the situation evolves as well as a means for tracking lessons learnt in the aftermath of a disaster. At any time during a scenario lifecycle a snapshot can be generated for analysis, reporting and archiving purposes. Based on a sequence of scenario snapshots end users are able to retrace the scenario evolution.

In extraordinary incidents, the compatibility of plans and an interoperable situation/scenario report exchange is crucial to ensure that emergency response work can be carried out in

close collaboration of the involved organizations. In that sense, scenarios can act as pools for information and knowledge and the HEIMDALL system should support end users to manage those efficiently and in a collaborative manner. The system should help to feed scenarios with relevant information, to:

- close the gaps in the situational and operational picture as it evolves,
- use them as a source of information for the generation, evaluation and modification of situation reports and response plans;
- and, finally, to disseminate those to the different involved stakeholders.

As mentioned before, we intend to provide situation report generation functionality by transforming a scenario snapshot at a certain time in the scenario lifecycle into a standards-based situation report, e.g. EDXL-CAP or EDXL-SitRep. This functionality will be described in D6.7 [3].

5.1.3 **Scenario Parameters and Associated Information**

According to the definition above, Table 5-1 specifies the scenario parameters identified so far. These parameters, values and parameter types may be subject to change as end-user discussions are ongoing. In particular, user feedback is expected during and after release demonstrations. These iterative changes are intended as they allow for a flexible, practitioner-oriented software development. The final release of this document, D6.15, will contain a final consolidated set of parameters.

Each parameter is described by a short description, possible values and information on parameter type and usage. Usage can address: multiple use; optional use; automatic generation of the value; name of the parameter in a JSON structure;

Table 5-1: Scenario parameters

Parameter	**Short Description**	**Values**	**Type** **Usage**
Type	Scenario type	“real” or “simulated”	String JSON: “attributes.type”
Ongoing	States whether scenario is ongoing or not	True or false	Boolean JSON: “attributes.ongoing”
Identifier	Unique identifier	Number	Integer Automatic generation JSON: “id”
URL	Globally unique identifier	URI	URL Automatic generation when not set JSON: “links.self”
Name	Descriptive name of the scenario	Free text	String JSON: “attributes.name”

Hazard type	Hazard type	“Forest fire” “Flood” “Flash flood” “Landslide”	Enumeration JSON: “attributes.hazardtype”
Hazard time	Start date of the hazard or incident	Timestamp	Date JSON: “attributes.hazardtime”
Hazard location	Geographical location of the hazard. Incidents will be located always with a point and can include a buffer. For simulated scenarios or other considerations a polygon shall be used to define an AOI	The paired values of a point (point location) or the paired values of points defining a polygon that delineates the affected location	Point or polygon JSON: “attributes.hazardlocation”
Hazard behaviour	Different parameters for different hazard types, e.g. forest fire behaviour can be described by FLI, ROS and spotting. Criteria will be defined by the end-user partners after submission of this document	Separate referenced data structure	DB relation JSON: “attributes.hazardbehaviour”
Conditions	Weather and further environmental conditions including forecast	Separate data structure, see Table 5-2	DB relation See Table 5-2 for data structure JSON: “relationships.conditions”
Impact	Effects of the natural events on the considered asset, mainly in terms of loss of functionality	“1”, “2”, “3”, “4”, “5”	Enumeration JSON: “attributes.impact”
Risk level	Risk is a combination of the consequences (impact) of an event (hazard) and the associated likelihood/ probability of its occurrence; Definition based on ISO Guide 73:2009 [6]	“Very low”, “low”, “medium”, “high”, “very high”	Enumeration JSON: “attributes.risklevel”
Casualties	Total number of casualties	Number	Integer JSON: “attributes.casualties”

Injured	Total number of injured	Number	Integer
			JSON: “attributes.injured”
Credibility	Credibility of the scenario. Different horizons of weather forecasts affect the credibility of a scenario. For instance, a scenario generated with a horizon of weather forecast of +9 hours will have less credibility than scenarios generated with +3 or +6 hours of weather forecast	Percentage	Decimal
			JSON: “attributes.credibility”
Snapshots	List of snapshots of the scenario	List of references Scenario snapshot is a separate data structure	DB Relation See Table 5-3 for snapshot structure
			JSON: “relationships.snapshots”
Related scenarios	List of related scenarios. For instance, different scenario options can be referenced.	List of references to other scenarios	DB Relation
			JSON: “relationships.scenarios”
Response Plans	List of related incident response plans.	List of references Response plan is a separate data structure	DB Relation See section 5.1.4 Response Plan Specification for more information
			JSON: “relationships.responseplans”
Lessons learnt	List of related lessons learnt	List of references Lesson learnt is a separate data structure	DB Relation See Table 5-5 for lesson learnt data structure
			JSON: “relationships.lessonslearnt”
Decisions	List of related decisions	List of references Decision is a separate data structure	DB Relation See Table 5-6 for decision data structure
			JSON: “relationships.decisions”
Measures	List of related prevention and mitigation measures	List of references Measure is a separate	DB Relation See Table 5-7 for

		data structure	measure data structure
			JSON: “relationships.measures”

As conditions are described by multiple parameters a separate data structure is foreseen which is shown in Table 5-2.

Table 5-2: Conditions parameters

Parameter	**Short Description**	**Values**	**Type**
			Usage
Temperature	Current and forecasted temperature	Degrees	Integer
			JSON: “attributes.temperature”
Relative humidity	Current and forecasted relative humidity	Percentage	Decimal
			JSON: “attributes.humidity”
Wind Speed	Current and forecasted wind speed	km/h	Integer
			JSON: “attributes.windspeed”
Wind Direction	Current and forecasted wind direction	Degrees	Integer
			JSON: “attributes.winddirection”

It has been agreed by the end-user partners that different hazard types need different horizons of weather forecast. It has been also agreed that the horizon of weather forecast shall be pre-defined as follows:

- Forest fire: +3, +6, +9 hours
- Floods: +3, +6, +9 hours
- Flash floods and landslides to be defined

A scenario snapshot as an instance of a scenario at a specific point in time including all associated information provides the additional information listed in Table 5-3.

Table 5-3: Scenario snapshot parameters

Parameter	**Short Description**	**Values**	**Type**
			Usage
Identifier	Unique identifier	Number	Integer
			Automatic generation JSON: “id”
Snapshot Identifier	Unique identifier for this snapshot	Number starting with 0 for each scenario. This counter is increased for	Integer
			Automatic generation

		each snapshot taken	JSON: “attributes.snapshot-id”
URL	Globally unique identifier Can be used by external modules for referencing this snapshot	URI	URL
			Automatic generation when not set JSON: “links.self”
Time	Date and time this snapshot has been taken	Timestamp	Date
			Automatic generation JSON: “attributes.time”

A scenario snapshot is used by the Impact Summary (ISAS) module to generate a situation report in a standardized situation reporting message format such as EDXL-CAP or EDXL-SitRep.

Information products provided by other HEIMDALL services can be related to a scenario. Table 5-4 lists these products. Association is performed by the user via the UI. Association is implemented by reference which means that the scenario file contains the service URI of each related product. JSON API provides a relationship structure for adding related links which could look like:

```
“relationships”: {
   “eo-products”: {
      “links”: {
         "related": "http://eo.eu/eo-products/34"
      }
   }
}
```

The actual products can be requested from the respective HEIMDALL internal and external data sources and modules by following the link. This implies that only those products can be associated which are provided over a service interface.

It is still to be discussed whether some of these information items can be included as a whole into a scenario and not only by reference. To reduce the number of HTTP requests, it would make sense to allow responses that include related resources along with the requested primary resources. For example, EO-based fire hot spots could be a good candidate for including them directly. Most data resources managed by the SM modules such as response plans, lessons learnt or weather conditions could also be included into the scenario resource (though snapshots of a scenario should be provided in the form of a linked list). For this, JSON API provides an “includes” element. For an example, please refer to section 5.2.1.2 SMES REST API.

Table 5-4: Information that can be associated to a scenario

Parameter	**Short Description**	**Values**	**Type**
			Usage
EO-based products	See D2.12 [7]	URI	URL
			Multiple use possible

			JSON: “relationships.eo-products”
Aerial-based products	See D2.12 [7]	URI	URL
			Multiple use possible JSON: “relationships.aerial-products”
Landslide monitoring products	See D2.12 [7]	URI	URL
			Multiple use possible JSON: “relationships.landslide-products”
Crowdsourced and FR data	See D2.12 [7]	URI	URL
			Multiple use possible JSON: “relationships.crowd-products”
Simulation results	See D2.12 [7]	URI	URL
			JSON: “relationships.simulations”
Risk and Vulnerability Information	See D2.12 [7]	URI	URL
			JSON: “relationships.rva-products”
Impact Summary Information (ISA Information)	See D2.12 [7]	URI	URL
			JSON: “relationships.isa”
Decision Support Information (DES Information)	See D2.12 [7]	URI	URL
			JSON: “relationships.des”

5.1.4 **Response Plan Specification**

The HEIMDALL Scenario Management will provide management capabilities for response plans connected to a scenario. According to the definition in D3.1 [8], a response plan is the formal product that collects and summarizes all information produced during the response planning phase.

Response plans are usually arranged in hierarchies from national down to the local levels, with cascading flows of command, control and information sharing. However, the bedrock of emergency planning is the local authority level, for this is always the theatre of operations, no matter how large the emergency is [9]. In multi-lateral discussions, HEIMDALL end-users partners have confirmed that assumption. The local emergency plan is very much the point of reference for both emergency operations and other types of contingency planning [5].

According to the end-user partners, their organizations manage different types of local response plans. These local response plans are applied and utilized in different phases of the response planning process. End users have agreed that in the HEIMDALL project response plans always refer to 'Incident Response Plans' (also referred to as 'Incident Action Plans'). These response plans are assembled on-the-fly during the emergency response phase and associated to the respective scenario by the user.

The major challenge of response plans in HEIMDALL is how to integrate the different agencies and the civil protection into the whole. In general, all agencies define a similar set of minimum contents that are required in the plans. However, end users have identified information, particularly tactical information that is specific to every local authority. Therefore, it has been agreed by all end-user partners that HEIMDALL shall provide a response plan for each specific authority composed by: (1) a multi-agency common plan, which will contain the overall strategy for the emergency management, (2) the common objectives (for all agencies) and (3) specific response plans for each agency, e.g. a FRS response plan, a police response plan and a medical service response plan. Figure 5-1 shows the HEIMDALL response plan structure.

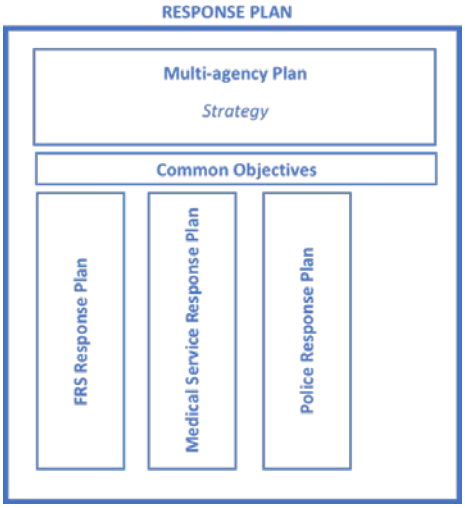

Figure 5-1: HEIMDALL response plan structure (preliminary)

In general, the HEIMDALL response plan consists of common descriptive information items such as a situation summary, working strategy, response objectives and tactics, an organization list, a resources summary, etc. The detailed data structure is currently under development by the end users and is not expected to be finalized before submission. Therefore, the final response plan specification is postponed to the final version of this document, D6.15, due in M38.

The first part of a response plan related to the situation summary and assessment shall be assembled on user request out of scenario information. A scenario snapshot shall be used by the Impact Summary Service (ISAS) module to generate a situation report in a standardized situation reporting message format such as EDXL-CAP or EDXL-SitRep. From the end users' perspective, a situation report is used as a basis for taking decisions on prevention and response actions. A situation report which contains sufficient information on the incident will be used as a basis for the creation of a response plan. Information on strategy, objectives and tactics will be added by the user.

The discussion on the relation between scenario and response plans and multiplicities is ongoing, e.g. whether multiple response plans shall be assigned to a scenario and whether one response plan shall be able to fit to multiple scenarios. For instance, one scenario can be solved by different response plans depending on who is leading / deploying the action. This is translated technically in a scenario only linked to one single response plan which shall be customized to the respective agency which currently uses the system.

5.1.5 **Lesson Learnt Specification**

The user will be able to define lessons learnt for a scenario. A HEIMDALL lesson learnt consists of several data and reference fields describing the lesson. As lessons learnt are still under discussion, the final specification will be provided in D6.15. Table 5-5 lists a first set of agreed parameters.

Table 5-5: Lesson learnt parameters

Parameter	**Short Description**	**Values**	**Type** **Usage**
Identifier	Unique identifier	Number starting with 0	Integer Automatic generation JSON: “id”
URL	Globally unique identifier	URI	URL Automatic generation when not set JSON: “links.self”
Scenario	Reference to scenario this lesson refers to	URI	URL JSON: “relationships.scenario”
Level of Command	Level of command	Strategic, tactical, operational	Enumeration JSON: “attributes.level”
Time	Date and time this lesson learnt has been created	Timestamp	Date JSON: “attributes.time”
Description	Textual information on lessons learnt	Free text	String JSON: “attributes.description”

5.1.6 **Decision Specification**

The user will be able to define decisions made in a scenario. The feature of keeping track of significant decisions has been identified by the end-user partners as very useful. A HEIMDALL decision consists of several data and reference fields describing the decision. As the decisions data structure is still under discussion the final specification will be provided in D6.15. Table 5-6 lists a first set of agreed parameters.

Table 5-6: Decision parameters

Parameter	**Short Description**	**Values**	**Type** **Usage**

Identifier	Unique identifier	Number starting with 0	Integer
			Automatic generation JSON: “id”
URL	Globally unique identifier	URI	URL
			Automatic generation when not set JSON: “links.self”
Scenario	Reference to scenario this decision refers to	URI	URL
			JSON: “relationships.scenario”
Response Plan	Reference to response plan if decision is taken in order to achieve the goals set in the plan	URI	URL
			Optional use JSON: “relationships.responseplan”
Level of Command	Level of command	Strategic, tactical, operational	Enumeration
			JSON: “attributes.level”
Time	Date and Time the decision has been created	Timestamp	Date
			JSON: “attributes.time”
Location	Decisions often refer to a geographical location within the AOI	The paired values of a point (point location) or the paired values of points defining a line or polygon that delineates the location	Point, Line or Polygon
			Optional use JSON: “attributes.location”
Description	Textual description of the decision	Free text	String
			JSON: “attributes.description”

5.1.7 **Measure Specification**

The user will be able to add prevention and response measures taken during a scenario such as a sandbag location and height, a fire break, NoFloods water tube barriers, etc. Tracking of measures taken allows users to evaluate their usefulness post-incident and to use this information for future similar scenarios. A measure can be a tactical means of implementation for a decision. As such it can be connected to that decision and to the corresponding response plan. A HEIMDALL measure consists of several data and reference fields describing the measure. As the measures data structure is still under discussion the final specification will be provided in D6.15. For example, the characteristics of a measure such as height, length, material and stability have to be defined according to what

information is needed to use the measure information at any later stage. The same applies to a field which allows the evaluation of the usefulness of the measure. The definition of an appropriate parameter type is still pending. Table 5-7 lists a first set of agreed parameters.

Table 5-7: Measure parameters

Parameter	**Short Description**	**Values**	**Type** **Usage**
Identifier	Unique identifier	Number starting with 0	Integer Automatic generation JSON: “id”
URL	Globally unique identifier	URI	URL Automatic generation when not set JSON: “links.self”
Scenario	Reference to scenario this measure refers to	URI	URL JSON: “relationships.scenario”
Response Plan	Reference to response plan if measure is taken in order to achieve the goals set in the plan	URI	URL Optional use JSON: “relationships.responseplan”
Decision	Reference to decision if measure serves as implementation means for the decision	URI	URL Optional use JSON: “relationships.decision”
Time	Date and time this measure has been defined	Timestamp	Date JSON: “attributes.time”
Type	Type of the measure	Fire break, sand bag or NoFloods water tube	Enumeration JSON: “attributes.type”
Location	Geographical location of the measure	The paired values of a point (point location) or the paired values of points defining a line or polygon that delineates the measure location	Point, line or polygon JSON: “attributes.location”

5.2 Scenario Management Specification

5.2.1 **Scenario Management Service**

The SMES provides functions for

- Creating, saving, modifying, accessing, deleting scenarios
- Associating information to scenarios (e.g. EO products, crowdsourced and FR data, simulation results, response plans, decisions, prevention and mitigation measures, lessons learnt, etc.), accessing, deleting of associated information
- Saving “snapshots” of a scenario and all related information at different points in time.

In addition, the SMES takes care of the management (i.e. create, save, modify, access, delete) of the following associated information:

- Response plans
- Lessons learnt
- Decisions
- Measures.

The SMES provides access to all of its data resources over dedicated REST APIs over the communication protocol HTTP(S). The tables in the next sections list the functionalities to be implemented in HEIMDALL with a specification of the respective method, (i.e. HTTP(S) GET, POST, PUT, DELETE) and data resource formats (e.g. JSON). Detailed REST APIs are expected to be specified in the second release of this document, D6.15, as the scenario management data resources are still under discussion and development in collaboration with the end-user partners.

5.2.1.1 Functionality related to Scenarios

Table 5-8 lists the functionality related to scenarios.

Table 5-8: Functionality related to scenarios

ID	Functionality	Interface method	Data exchanged and data representations
SM_SC_01	Create scenario This function covers also the functionality ‘copy scenario’	HTTP POST /scenarios	Request body: Scenario resource (JSON) with basic scenario parameters defined in 5.1.3. Automatically generated parameters shall be omitted Response body: Created scenario resource
SM_SC_02	Access scenario This function covers also the functionality ‘access associated information item’ (if item has been associated by reference, service consumers must follow the reference in the scenario resource)	HTTP GET /scenarios/<id>	Request body: nothing Response body: JSON representing the scenario resource
SM_SC_03	Save scenario This function covers also the following	HTTP PUT /scenarios/<id>	Request body: scenario resource (JSON) For function ‘associate

	functionalities: modify scenario parameters, associate information item to scenario		information item': Response plans, decisions, measures, lessons learnt must be provided as additional elements (JSON) in the scenario resource structure; All other information is expected to be included in the scenario resource as a reference (URI) Response body: nothing
SM_SC_04	Delete scenario	HTTP DELETE /scenarios/<id>	Request body: nothing Response body: nothing
SM_SC_05	Save snapshot	HTTP POST /scenarios/snapshots where URI is an extension of the URI of the scenario the snapshot shall be taken from, e.g. http://sm.eu/scenarios/1/snapshots/	Scenario resource (JSON) with all available parameters and associated information as defined in 5.1.3

5.2.1.2 SMES REST API

Like all SM services the SMES REST API communicates using JSON as the primary data-interchange format. As stated before, the format will be based on shared conventions for designing web APIs, e.g. the JSON API conventions [4]. The concrete JSON structure of the scenario resource is subject to frequent change as the different HEIMDALL modules mature. For example, data types may change or end users will require additional values in a list of possible values. Based on common procedures and technologies in web development the SMES API documentation will be built and deployed together with every software release. This way, the API documentation remains always up-to-date. However, the following examples show how client requests and SMES responses could look like.

For example, the following request fetches a scenario (functionality SM_SC_02):

```
GET /scenarios/51 HTTP/1.1
Accept: application/vnd.api+json
```

The request could return:

```
HTTP/1.1 200 OK
Content-Type: application/vnd.api+json
{
   “data”: {
      “type”: “scenarios”,
      “id”: “51”,
      “attributes”: {
         “type”: “real”,
         “ongoing”: “true”,
         “name”: “Fire in Solsona”,
```

```
            “hazard_type”: “Forest fire”,
            “hazard_time”: “…”,
            …
        },
        “links”: {
            “self”: “http://sm.eu/scenarios/51”
        },
        "relationships": {
            “eo_products”: {
                "links": {
                    "related": "http://eo.eu/eo_products/34"
                }
            }
            "lessons_learnt": [12, 13]
        }
    },
    “included”: {
        “lessons_learnt”: [
            {
                “id”: “12”,
                “attributes”: {
                    “level_of_command”: “Strategic”,
                    “description”: “this is a lesson learnt”
            }, {
                “id”: “13”,
                “attributes”: {
                    “level_of_command”: “Tactical”,
                    “description”: “another lesson learnt”
            }
        ]
    }
}
```

For example, the following request creates a scenario (functionality SM_SC_01):

```
POST /scenarios HTTP/1.1
Content-Type: application/vnd.api+json
Accept: application/vnd.api+json
{
    “data”: {
```

```
        “type”: “scenarios”,
        “attributes”: {
            “type”: “simulated”,
            “ongoing”: “false”,
            “name”: “Alternative fire in Solsona”,
            “hazard_type”: “Forest fire”,
            “hazard_time”: “…”,
            …
        }
    }
}
```

The response could look like:

```
HTTP/1.1 201 Created
Location: http://sm.eu/scenarios/52
Content-Type: application/vnd.api+json
{
    “data”: {
        “type”: “scenarios”,
        “id”: “52”,
        “attributes”: {
            “type”: “simulated”,
            “ongoing”: “false”,
            “name”: “Alternative fire in Solsona”,
            “hazard_type”: “Forest fire”,
            “hazard_time”: “…”,
            …
        },
        “links”: {
            “self”: “http://sm.eu/scenarios/52”
        }
    }
}
```

5.2.1.3 Functionality related to Response Plans

The HEIMDALL Scenario Management provides access to response plans. These response plans are intended to be associated by the user of the system to specific scenarios and shall serve as a basis for a deeper analysis of the scenario, for response planning and training. For the time being, it is not intended to let the HEIMDALL system propose, assess, select and activate specific response plans, individual actions within a response plan and corresponding objectives and tactics. Table 5-9 lists the functionality which shall be implemented for response plans.

Table 5-9: Functionality related to response plans

ID	Functionality	Interface method	Data exchanged and data representations
SM_RP_01	Create response plan	HTTP POST /responseplans	Request body: Response plan resource (JSON) with basic parameters defined in 5.1.4. Automatically generated parameters shall be omitted Response body: Created response plan resource
SM_RP_02	Access response plan	HTTP GET /responseplans/<id>	Request body: nothing Response body: JSON representing the response plan resource
SM_RP_03	Save response plan This function covers also the functionality 'modify response plan'	HTTP PUT /responseplans/<id>	Request body: Modified response plan resource (JSON) (URI identifies response plan)
SM_RP_04	Delete response plan	HTTP DELETE /responseplans/<id>	Request body: nothing Response body: nothing

As mentioned before in section 5.1.3 Scenario Parameters and Associated Information, the REST API will provide a convention for associating related SM information products (e.g. response plans) to scenarios. Accordingly, product data could be included directly in the scenario resource (i.e. using "includes" element). One way is to add the response plan data to the scenario resource and perform a scenario modification operation (functionality SM_SC_03). However, the more user-friendly way for the client is to create a response plan (functionality SM_RP_01) with a relationship to its parent scenario ("relationships.scenario" element). The scenario would be automatically connected to its new child. The parent-child relationship would become imminent on the client-side with a new request for the scenario.

For example, the following request creates a response plan for a scenario (functionality SM_RP_01):

```
POST /responseplans HTTP/1.1
Content-Type: application/vnd.api+json
Accept: application/vnd.api+json

{
   “data”: {
      “type”: “responseplans”,
      “relationships”: {
         “scenario”: {
            "data": {
               “type”: "scenario",
               “id”: “52”
            }
```

```
            }
        }
    }
}
```

5.2.1.4 Functionality related to Lessons learnt

The user shall be able to define lessons learnt for a scenario. Table 5-10 lists the functionality which shall be implemented for lessons learnt.

Table 5-10: Functionality related to lessons learnt

ID	Functionality	Interface method	Data exchanged and data representations
SM_LL_01	Create lesson learnt	HTTP POST /lessonslearnt	Request body: Lesson learnt resource (JSON) with basic parameters defined in 5.1.5. Automatically generated parameters shall be omitted Response body: Created lesson learnt resource
SM_ LL_02	Access lesson learnt	HTTP GET /lessonslearnt/<id>	Request body: nothing Response body: JSON representing the lesson learnt resource
SM_ LL_03	Save lesson learnt This function covers also the functionality 'modify lesson learnt'	HTTP PUT /lessonslearnt/<id>	Request body: JSON representing the lesson learnt resource Response body: nothing
SM_ LL_04	Delete lesson learnt	HTTP DELETE /lessonslearnt/<id>	Request body: nothing Response body: nothing

5.2.1.5 Functionality related to Decisions

A decision taken by emergency service roles in charge (i.e. C&C, IC) can be created by the user and associated to a scenario. In the long term decision paths are intended to be formalized by use of a georeferenced format. Table 5-11 lists the functionality which shall be implemented for decisions.

Table 5-11: Functionality related to decisions

ID	Functionality	Interface method	Data exchanged and data representations
SM_DC_01	Create decision	HTTP POST /decisions	Request body: Decision resource (JSON) with basic parameters defined in 5.1.6. Automatically generated

			parameters shall be omitted Response body: Created decision resource
SM_ DC_02	Access decision	HTTP GET /decisions/<id>	Request body: nothing Response body: JSON representing the decision resource
SM_ DC_03	Save decision This function covers also the functionality 'modify decision'	HTTP PUT /decisions/<id>	Request body: JSON representing the modified decision resource Response body: nothing
SM_ DC_04	Delete decision	HTTP DELETE /decisions/<id>	Request body: nothing Response body: nothing

5.2.1.6 Functionality related to Measures

Prevention measures and countermeasures taken during a scenario such as a sandbag location and height, fire break, etc. can be defined for a scenario. Table 5-12 lists the functionality which shall be implemented for measures.

Table 5-12: Functionality related to measures

ID	Functionality	Interface method	Data exchanged and data representations
SM_MS_01	Create measure	HTTP POST /measures	Request body: Measure resource (JSON) with basic parameters defined in 5.1.7. Automatically generated parameters shall be omitted Response body: Created measure resource
SM_ MS_02	Access measure	HTTP GET /measures /<id>	Request body: nothing Response body: JSON representing the measure resource
SM_ MS_03	Save measure This function covers also the functionality 'modify measure'	HTTP PUT /measures/<id>	Request body: JSON representing the measure resource Response body: nothing
SM_ MS_04	Delete measure	HTTP DELETE /measures/<id>	Request body: nothing Response body: nothing

5.2.2 **Scenario Repository**

The detailed data model for the Scenario Repository is shown in Figure 5-2.

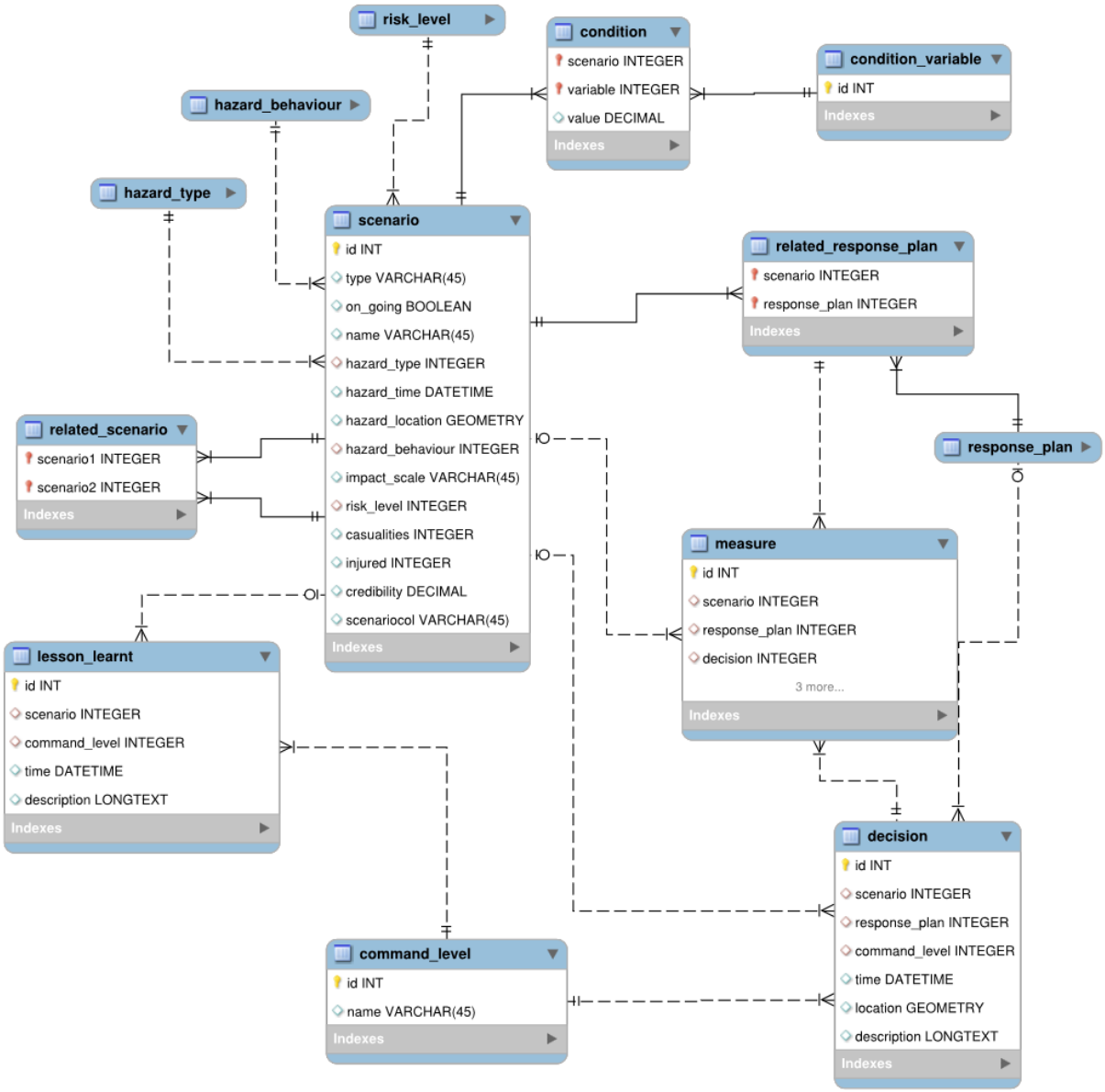

Figure 5-2: Detailed Scenario Repository data model

Regarding response plans it has been decided by the end-user partners that HEIMDALL shall not store pre-defined response plans. Instead, incident response plans will be created on-the-fly for scenarios. These incident response plans will be handled in the SMES as a separate data structure stored in the HEIMDALL Scenario Repository.

5.2.3 **Software**

As stated before, the software design will be based on REST web services implemented in Python. The software architecture is represented in the schema shown in Figure 5-3.

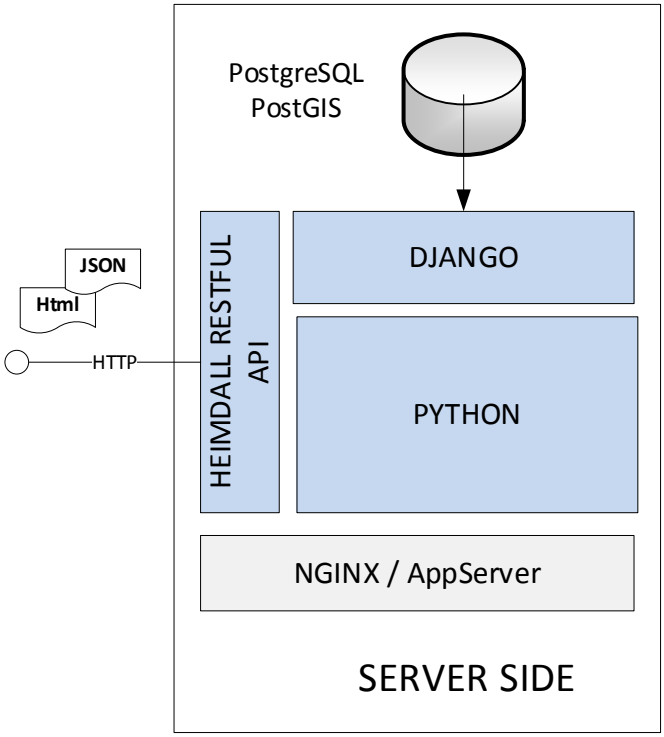

Figure 5-3: Scenario management software architecture

On the server side, on an open-source Linux OS, NGINX or another application server can be used as web server.

The server component acts as a service provider, exposing the platform API to the HEIMDALL system clients.

The system API is based on a RESTful web services layer. The acronym REST stands for REpresentational State Transfer and is a software architectural style. It is not properly a concrete system, neither an established standard. REST is a set of guidelines aiming at the realization of a system architecture. It specifies features, such as the uniform interface. If these features are applied to a web service, they imply good properties (i.e., scalability) on the Web, improving the performance. Data and functionality can be accessed via the Uniform Resource Identifiers (URIs), which on the Web are implemented as links. The main advantage of adopting the URI scheme is that it is already well-defined and no new implementations are required.

A concrete implementation of a REST web service follows four basic design principles [10]:

- Use HTTP methods explicitly;
- Be stateless;
- Expose directory structure-like URIs;
- Transfer XML, JSON, or both.

In comparison with SOAP, REST is easier to use for the most parts and is more flexible. It has the following advantages when compared to SOAP [11]:

- No expensive tools required to interact with the web service
- Smaller learning curve
- Efficient (SOAP uses XML for all messages, REST can use smaller message formats)
- Fast (no extensive processing required)
- Closer to other web technologies in design philosophy

The JSON format is the preferred output but most of the services will support also an XML response type. From a client point of view all the services will be accessible via REST.

The programming language is Python using the Django framework [12]. Django is a free and open-source web application framework, written in Python, which follows the model–view–controller architectural pattern. When the template of Django Views is configured as a JSON or XML document the server acts as a RESTful service.

The database management system is PostgreSQL with PostGIS [13]. PostgreSQL has more than 15 years of active development and a proven architecture that has earned it a strong reputation for reliability, data integrity and correctness. PostGIS is an open source software program that adds support for geographic objects to the PostgreSQL object-relational database. PostGIS follows the Simple Features for SQL specification from the Open Geospatial Consortium (OGC).

5.3 Scenario Matching Specification

The HEIMDALL system can assist end users to assess different alternative scenarios for a situation and assess their potential evolution and consequences using scenario matching capabilities, simulation tools and impact assessment functions. With the scenario matching functionality, users can find similar situations in a database and look for the prevention and response measures taken and their outcomes, to evaluate suitable working strategies.

A major objective of the first project phase has been to define together with the end-user partners scenarios and response plans in HEIMDALL and how the SM can support the management of these entities. The scenario matching functionality will build upon these definitions and developments. Accordingly, this section describes a first set of concepts and considerations. The SMAC module will be specified in detail in the second version of this document, D6.15.

5.3.1 **Scenario Matching Criteria and Metrics**

As a basis for multi-criteria pattern matching, analysis and comparison of scenarios, suitable matching criteria must be identified and formalized along with applicable multi-hazard and risk-based comparison metrics.

For the time being, the hazard type has been identified by the end users as a major matching criterion. Discussions are ongoing on the relevance and importance of further criteria such as weather conditions in the AOI, hazard behavior, impact scale, etc. The Fire and Rescue Service of Catalonia stated that the most valuable feature for them in scenario matching is the comparison of scenarios based on the “Fire Types” concept. For example, wind-driven wild fires are characterized by a different behavior than topographic fires, thus cannot be managed in the same way [14]. A ‘weight’ parameter will be determined for every matching criterion in coordination with the end users. This weight emphasizes the importance of each individual matching criterion.

Comparison metrics will build upon suitable similarity definitions for the determined matching criteria under consideration of criteria weights. Relations between scenarios by application of time and space rule-based algorithms and geospatial analysis will be integrated. Matching criteria and metrics will be configurable and visible to the end users in order to foster matching process customization and optimization.

5.3.2 **Ethical Considerations**

The scenario matching approach must consider ethical acceptability issues which may arise. The acceptability approach focuses on how responsibility can be assigned when the scenario matching service can also provide a so called ‘best matching scenario’. In this case, this specific presentation of information might not support a decision and could rather be seen as a release from taking any. Therefore, the ‘best matching scenario’ should be based on configurable, clear and agreed matching criteria and with an associated visual tool to identify the grounds for similarities. In this way, although not being a final and comprehensive solution, the consortium found a potential answer.

5.3.3 **Scenario Matching Service**

In order to perform a multi-criteria selection of scenarios most similar to a situation a distance measure will be defined which represents the distance of a given data set to a scenario in such a way, that the scenario corresponds exactly to the real situation when the distance is zero. We will call this distance measure the ‘mismatch’. The mismatch will be derived from the individual mismatch measures for each type of applied matching criterion under consideration of the respective criterion weight. In formal terms the total mismatch of a scenario to a given situation will be the sum of weighted (w_i) mismatches (m_i) of the individual matching criteria (i):

$$Mismatch_{total} = \sum w_i \cdot m_i$$

For every scenario matching run the SMAC will return a list of similar scenarios ordered by the respective mismatch per scenario. Scenario matching outputs will be provided in a common descriptive format (e.g. JSON) over the SMAC REST API.

6 Conclusion

This document provided the first design of the SM component and its sub-modules. On its basis the involved technical partners will be able to develop the first releases of the component. We also expect that software development of other HEIMDALL components connecting to scenario management can take place simultaneously and integration of the SM in the overall HEIMDALL system will be facilitated.

The component design is based on concepts developed in close collaboration with end users from different countries and disciplines, which namely are Fire and Rescue Services (FRS), Police, Medical Services (MS), Civil Protection (CP) and Command and Control Centres (C&C). Requirements and solutions evolve through collaborative design and the HEIMDALL agile information system development considers the needs of end users and other stakeholders. Intermediary system releases based on this design document and on the case studies developed by the end users will be demonstrated in real-environment conditions, for example during the End User Workshop 2 (EUW2). This will give the end users the possibility to reflect on current solutions, to validate these and to identify problems. User feedback will be acquired after demonstrations. This information will lead to re-iterations of technical requirements and the component design. The follow-up deliverable D6.15 due in M38 will reflect these refinements.

7 References

[1] Petersen, K., Büscher, M., Kuhnert, M., Schneider, S., & Pottebaum, J. (2015). Designing with Users: Co-Design for Innovation in Emergency Technologies. In: L. Palen, M. Buscher, T. Comes, & A. Hughes (Eds.), In: ISCRAM 2015 Conference Proceedings – 12th International Conference on Information Systems for Crisis Response and Management. Kristiansand, Norway: University of Agder (UiA).

[2] Kroener, I., Watson, H., & Muraszkiewicz, J. (2017). Agility in crisis management information systems requires an iterative and flexible approach to assessing ethical, legal and social issues. In: Proceedings of the 14th International Conference on Information Systems for Crisis Response And Management, 247–255. Albi, France.

[3] Friedemann, M. et al. (2018). HEIMDALL D6.7: Situation Assessment and Reporting Specification and Implementation Report – Draft, in production

[4] JSON API (2015). A specification for building APIs in JSON. Available at http://jsonapi.org/format/ [last accessed in April 2018]

[5] Alexander, D. (2009). Principles of emergency planning: standardisation, integration and sustainability. In: Building safer communities: Risk governance, spatial planning and responses to natural hazards, edited by: Fra Palco, U., NATO science for peace and security series, IOS Press, Amsterdam, 162-174

[6] ISO Guide 73:2009 (2008). Risk Management – Vocabulary. Available at: https://www.iso.org/obp/ui/#iso:std:iso:guide:73:ed-1:en [last accessed in April 2018]

[7] Mulero Chaves, J. et al. (2018). HEIMDALL D2.12: HEIMDALL System Architecture

[8] Vendrell Flotats, J. et al. (2017). HEIMDALL D3.1: Case Studies – Issue 1

[9] Aloyo, O. O. (2006). Emergency preparedness is all local. In: Journal of Emergency Management, 4(1), 11-12

[10] Rodriguez, A. (2008). RESTful Web services: The basics. In: IBM developers Work, available at: https://www.ibm.com/developerworks/library/ws-restful/index.html [last accessed in April 2018]

[11] SOAP vs REST 101: Understand The Differences. Available at: https://www.soapui.org/resources/api-testing/article/soap-vs-rest-api.html [last accessed in April 2018]

[12] The Django web framework project. Available at: https://www.djangoproject.com/ [last accessed in April 2018]

[13] PostGIS, Spatial and Geographic objects for PostgreSQL. Available at: http://postgis.net/ [last accessed in April 2018]

[14] Lecina-Diaz, J., Alvarez, A., & Retana, J. (2014). Extreme Fire Severity Patterns in Topographic, Convective and Wind-Driven Historical Wildfires of Mediterranean Pine Forests. In: PLoS ONE, 9(1), e85127, available at: http://doi.org/10.1371/journal.pone.0085127 [last accessed in April 2018]