European Commission

JRC TECHNICAL REPORTS

Risk Data Hub software and data architecture

Solutions and tools for Disaster Risk Management

Ríos Díaz, Francisco
Marín Ferrer, Montserrat
Antofie, Tiberiu Eugen
Luoni, Stefano
Faiella, Anna

2019

Contact information
Name:
Address:
Email:
Tel.:

JRC Science Hub
https://ec.europa.eu/jrc

JRCxxxxx

EUR xxxxx xx

PDF	ISBN xxx-xx-xx-xxxxx-x	ISSN xxxx-xxxx	doi:xx.xxxxx/xxxxxx
Print	ISBN xxx-xx-xx-xxxxx-x	ISSN xxxx-xxxx	doi:xx.xxxxx/xxxxxx

Luxembourg: Publications Office of the European Union, 2019

How to cite this report: Author(s), *Title*, EUR (where available), Publisher, Publisher City, Year of Publication, ISBN (where available), doi (where available), PUBSY No.

Authors affiliations:

Tiberiu Eugen Antofie, European Commission, Joint Research Centre (JRC), Ispra, Italy

Stefano Luoni, Unisystems Italy external service provider of European Commission, Joint Research Centre (JRC), Ispra, Italy

Anna Faiella, Trainee at the European Commission, Joint Research Centre (JRC), Ispra, Italy

Francisco Ríos Díaz, European Commission, Joint Research Centre (JRC), Ispra, Italy

Montserrat Marín Ferrer, European Commission, Joint Research Centre (JRC), Ispra, Italy

Contents

Abstract

Risk Data Hub is an initiative of Disaster Risk Management Knowledge Centre (DRMKC) and consists of a publicly available web-GIS platform intended to improve the access and sharing of curated European-wide risk data, tools and methodologies for fostering Disaster Risk Management (DRM) related actions.

The implementation of the concept is made of multiple steps, including the definition of type of analysis to be presented, the design of methodologies to compute data needed, the design of database architecture and software tools and finally the development of the software.

This document will focus on the design of software architecture, starting from a high level analysis of the business needs, going to the explanation of the solutions proposed, considering previous works in the topic of Disaster Risk Management and showing how the existent Loss Database architecture has been extended to fit the requirements of a complex and multi-context application.

Role of the authors

Tiberiu Eugen Antofie, as author, provided methodological example bringing together information associated with the loss and damage data, as well as contributed in structuring the historical event catalogue.

Stefano Luoni, as external consultant and co-author, adapted the database architecture to match the Risk Data Hub needs and developed the Risk Data Hub platform.

Anna Faiella, as trainee, contributed to the review of data and the report.

Francisco Ríos Díaz, as contributor, was responsible for designing the concept of the database architecture behind the Risk Data Hub and helping prepare the report.

Montserrat Marin Ferrer, as the coordinator of the DRMKC projects, overviewed the whole process.

1 Introduction

Despite the great number of projects developed in the context of Disaster Risk Management, there are no widely shared resources to analyse disaster risk data, as every country has its own databases and organisations, with different levels of usage and effectiveness. With Risk Data Hub, European Commission wants to offer a common platform to access innovative tools and methodologies, granting more equity to those who will decide to adopt it.

To understand what this application is about, we can start with a couple of considerations.

Working with risk data means to deal with hazards, in first place: Risk Data Hub has a multi-hazard approach, implementing methodologies to present data about different hazards, both one at once and altogether. While the first datasets introduced are related to natural hazards, also technological hazards and all kind of man-made disasters are to be involved, with the final aim to have a complete map of risk, including bot direct and indirect impacts.

Risk Data Hub is also multi-context, as it can be used to analyse exposures and vulnerabilities, as well as it shows historical events. This means that on a unique platform, the user is allowed to discover most exposed and vulnerable areas for every hazard, verify and compare real impacts, perform statistical analysis, find trends, check eligibility for solidarity fund requests and more.

Risk Data Hub wants to be a "second house" for research results, satisfying the need to make them accessible. This purpose may be clarified by defining input and output for this platform.

Input is granted by scientific partnerships where this platform represent a real added value, as it improves interoperability by connecting different sources and sharing their data.

Output consists of different analysis performed on available data, as the implementation of specific methodologies that should be useful for multiple policies. According to the vision of Risk Data Hub as main resource to access risk data analysis, its usage will enhance coherence across portfolios.

The rest of this document explains how this main concepts are applied by the software.

A more complete explanation of methodologies developed for Risk Data Hub and its relation to policies is included in "Risk Data Hub - web platform to facilitate management of disaster risks" JRC technical report.

2 Main challenges and solutions identified

Before even start any implementation, all concepts, methodologies and business needs in general have to be translated into technical requirements. This chapter contains a simple explanation of challenges and solutions proposed for the development of the software.

2.1 Dealing with uncertainty of data

It's a fact that none of the data available is 100% correct, nor gives us a certain value, because of different reasons.

First of all: availability and accuracy of data on past disaster events are poor, mainly because data collection at local level is not homogeneous and it not shared with higher administrative levels for statistics. This is something confirmed by many research projects published in this topic and explains how it's difficult to collect and present damages and losses.

Then let's take the models: of course they cannot give us certainty, as they are used to predict future events and have a probabilistic approach. There is another problem, though: they are shaped by identifying trends in past events and as data on past events are scarcely available, how much can we trust them?

This challenge, along with solution proposed, represents a main critical concept of which most of next points are the logic consequence.

Solution

Risk Data Hub does not offer early warning support; instead, it presents **pre-event** and **post-event** data, where pre-event data come from models and post-event data come from collections of past events.

Having stated that data always contain uncertainty, RDH tries to give a better overview by comparing data from different sources. The idea is to provide the user of the platform with data from both models and archives of past events, for each event presented. This way, it's easy to spot anomalies (e.g. only one value which differs from others), identify which source is overall more accurate, give at least probabilistic values if no assessed damages are available; on the other hand, having the most possible complete collection of past events, should help better tuning the models and their forecasting capacities.

2.2 Representing together exposures, vulnerabilities and historical events

Risk Data Hub aims to handle different type of datasets, making them available on a unique portal to help end users in many tasks related to Disaster Risk Management.

Basically, data used come from models, or archives of past events. While models tell us what could happen, past events are something already occurred, in a specific date, under specific circumstances, hence the visualization of these two types of data cannot be the same.

Data is also referred to many **natural** hazards (**technological** hazards are not included yet, but a future implementation is foreseen) and every hazards has its own peculiarities; that's why data inputs differ from hazard to hazard.

So the first challenge is actually double and consists of storing different type of data in a single database and presenting them in a way that preserve their specificity in a single user interface.

Solution

The database should include a main entity **Damage Assessment** which, along with the overall flexible design, let the system manage and present data about different type of analysis. A more detailed explanation is included in chapter 4 of this document.

2.3 Harvesting data from multiple sources

After considering uncertainty (see chapter 2.1) and working with such a wide area of interest, it's clear how a single source of data is not enough. Risk Data Hub works with many scientific partners that provide the application with the outputs of their work. These are typically models used for populating the Risk Analysis datasets of RDH, but sometimes archives of past events are included.

While models have a good coverage and are produced on a regular basis by scientists, the collection of loss data is something that is not homogeneous, nor well defined and structured; that is why data availability is poor, especially on a large scale.

Scientific, economic and political issues that causes this poor availability of data are not a concern of this document. Technically speaking, a way identified to get as many data as possible, is to connect with different sources.

Solution

This challenge leads to the development of a dedicated data integration flow for each data source activated. RDH has an ETL layer that is needed to transform data extracted before inserting them into the database.

Important note: at this time, all data used by the system is stored in its own database. This approach may be considered a downside and criticized as duplicates data already existent on external resources. As it will be better explained later in this document, most of the data managed by the system, particularly events, need to be transformed and/or validated and this is not something that can be done on the fly, for different reasons. First of all performance: a complex processing of a whole dataset at every page request does not make sense as would be a waste of system resources and it would cause a dramatic fall of the overall performance. The second reason is about the validation process: many events need to be moderated, as they often contain wrong or incomplete information and this operation should be done one at once. The third reason is about data availability: not every sources expose services, so data have to be massively downloaded before being able to use them.

Having said that, using external services to access particular layers or features on the fly is something that is convenient and will be certainly integrated for specific data sources.

RHD main scientific partners are worth of a mention (see Reference section for details):

- EFAS
- EFFIS
- EDO
- GHSL
- Copernicus
- GDACS
- EMM
- EM-DAT

- HANZE database

2.4 Identification and classification of events

Identifying an event is not obvious: actually it's still a matter of discussion in the scientific community and it's not homogeneous among different hazards. From the RDH point of view, the problem is not about trusting a specific source of data, but since the system extracts events from multiple sources, specific criteria for identifying events are necessary to avoid duplications.

Solution

The logic used by RDH is as follows:

- An event is identified by **Hazard**, **Date**, **Country** and, optionally, by a smaller administrative unit. This means that, for example, a single meteorological event which covers an area shared by 2 countries will generate exactly 2 events in the system, while more events could be identified on the same date and country if the causes are different. For events harvested from external sources with no clear information on the cause, only one events would be generated per Hazard, Date and Country.
- **Event** is a macro entity that may include multiple **phenomena**. This means that while an event can be associated for example to a whole country and it can last several days (or weeks), there are single phenomena that map the event to a more specific location and date, such as single burned areas of a vast forest fire.

2.5 Unique coding of events

Every data source uses its own way of assigning a code to events; furthermore, events coming from various sources may overlap, hence a new code has to be assigned to keep a consistent archive.

Solution

In RDH, the code composition is implemented as follows:

[Hazard] (Code of 2 characters)

+ [Country] (ISO2 of country)

+ [Begin Date] (in YYYYMMDD format)

+ [Glide Number] (4 digits serial number)

An example would be:

Figure 1. Example of event code

When imported into the system, every event has a status equals to "draft" and it needs moderation to be published. Only when the event is approved, the RDH code is generated; this allows to be consistent with the sequence of glide numbers of published events.

2.6 Country corners and user privileges

Risk Data Hub publishes European wide datasets, but the whole system is designed to work also at national or regional level. This is a fundamental part of the concept of RDH: data should always be linked to administrative divisions and should be collected at local level.

While the JRC is able to produce and/or find data with a good coverage of all Europe but quite generic, local institutions likely have access to more detailed data and should be able to use them on the RDH platform for their own disaster risk management purposes.

A "country corner" works like a separate instance of Risk Data Hub, as it implements the same methodologies in a different hierarchy of locations. A single institutional user who is responsible for its country will upload data and choose whether or not to share this data with other users or groups.

Solution

The logic proposed is quite simple and it's based on 2 main points:

- A user belongs to one or multiple groups and each group has some basic permissions
- Each dataset in the system has a unique owner that can set visibility and permissions for it

Let's report a couple of examples to clarify this logic a bit.

Example 1: the group of administrators of the Austrian country corner has privileges for managing all datasets assigned to the Austria Region. A user who belongs to this group uploads 2 layers, decide for the first one to grant View rights with all groups and for the second one all rights only to the group "Austria_Administrators". After this, the first layer will be visible to every user (but not editable), even if not registered to the site (because it belongs to Anonymous group); the second layer will be visible and editable only by users in group "Austria_Administrators".

Example 2: a country corner administrator uploads data for a new Damage Assessment and choose to grant Edit right to the administrators group and only View rights to the group of non admin user of the country corner of reference. After this, a non-logged user, or a user of another country corner will not see anything of that Damage Assessment.

2.7 Scalability and performance

Since RDH is expected to store and manage large amounts of data, scalability is a matter to be addressed to keep the application healthy and responsive.

This document is not a technical guide, nor a list of design patterns in Python or any other language. Here we want just to state that performance is something taken into consideration and about this there are some practices or tools already use, as well as others to be applied in the near future.

Solution

- Database indexes
- Optimizations of queries
- Use of GeoWebCache: this is a tool that come with GeoServer and caches tiles generated by WMS calls. Tiles can be both cached after a call to WMS service, or by a bulk seeding process
- Caching of Django Views: Django integrates a configurable caching system for its views, so multiple page requests would consume resources only once after the cache expiration
- "Reselect" tool for React: the client application keeps data retrieved from the backend API in its own internal "store" and would make a new call to the API only if data is not already into it; this saves both bandwidth and system resources

To be done:

- Deploy of Geoserver on a dedicated machine
- Use of NoSQL database: when data stored starts to exceed a certain amount, old fashioned relational databases start to suffer a degradation of their performance. The use of a NoSQL database should solve this problem, but at this point the technology selection process is not completed, as there are several constraints to be considered about Geonode and Geoserver

3 Technologies used

This chapter is about technology selection and architectural design of RDH application.

3.1 Previous works

Having a look to works already done in the same topics surely useful to identify tools that have proven to work well and best practises using them.

3.1.1 GeoSAFE

GeoSAFE is a web platform that provides the ability to run InaSAFE analyses online. InaSAFE is free software that produces realistic natural hazard impact scenarios for better planning, preparedness and response activities. It provides a simple but rigorous way to combine data from scientists, local governments and communities to provide insights into the likely impacts of future disaster events.

Initiative of Government of Mozambique and the World Bank. Based on Geonode (and GeoServer).

3.1.2 Rasor

RASOR is developing a platform to perform multi-hazard risk analysis for the full cycle of disaster management, including targeted support to critical infrastructure monitoring. A scenario-driven query system simulates future scenarios based on existing or assumed conditions and compares them with historical scenarios. Initially available over five case study areas, RASOR will ultimately offer global services to support in-depth risk assessment and full-cycle risk management.

Developed by CIMA research foundation. Uses Geonode as layer catalog.

3.1.3 ThinkHazard!

ThinkHazard! provides a general view of the hazards, for a given location, that should be considered in project design and implementation to promote disaster and climate resilience. The tool highlights the likelihood of different natural hazards affecting project areas (very low, low, medium and high), provides guidance on how to reduce the impact of these hazards, and where to find more information. The hazard levels provided are based on published hazard data, provided by a range of private, academic and public organizations.

Developed by GFDRR. Uses GeoServer.

3.1.4 Afghanistan Disaster Risk

A public platform for creating, sharing and accessing geospatial data and maps for decision-making about disaster risk. It includes tow modules: one for risk analysis and one for cost/benefit analysis.

Developed by GFDRR. Based on Geonode (and GeoServer).

3.2 Overall architecture

After collecting and analysing the main requirements of the platform to be developed, it was time to choose the technologies and tools to be used. Some of these choices were anticipated by the previous chapter and they were the result of checking previous works in this fields, as they pointed out that significant projects were based on Geonode and Geoserver.

The system architecture as a whole is quite articulate and makes use of several tools to perform all operations needed. Basically, the project is built with Django (Python web framework), using Geonode as dependency, PostGIS as database backend and a client application developed with ReactJS.

What is Geonode?

GeoNode is a web-based application and platform for developing geospatial information systems (GIS) and for deploying spatial data infrastructures (SDI). It can be integrated with third-party Django apps and implements a framework for OGC-compliant web services.

What is GeoServer?

GeoServer is an open source server for sharing geospatial data. Designed for interoperability, it publishes data from any major spatial data source using open standards. GeoServer is an OGC compliant implementation of a number of open standards such as Web Feature Service (WFS), Web Map Service (WMS), and Web Coverage Service (WCS).

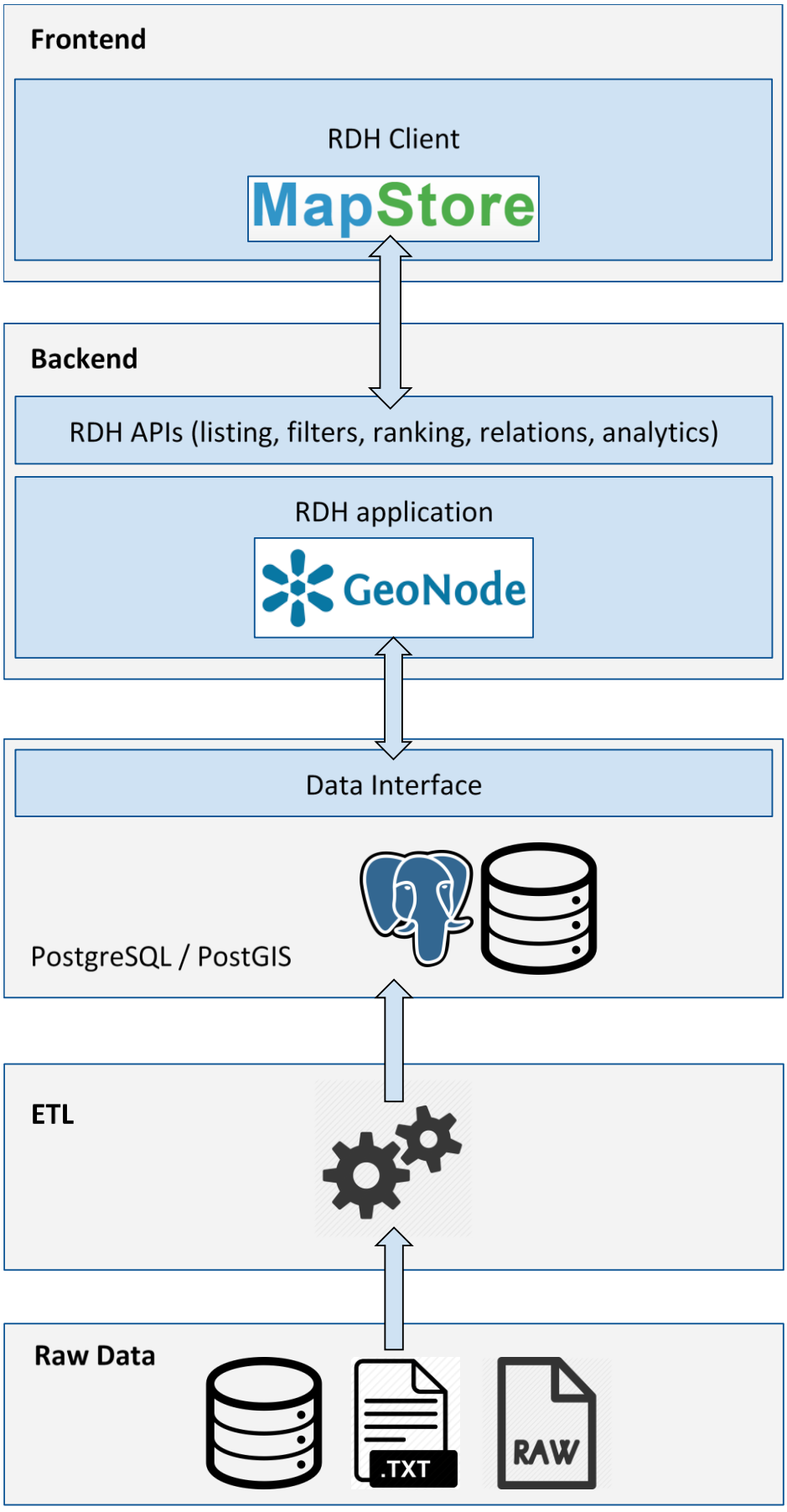

Figure 2. RDH software architecture

Let's a have a more detailed look on the single pieces of architecture.

3.3 Data harvesting and ETL

Data is harvested from multiple heterogeneous sources and loaded into RDH database by ad hoc Python scripts. Relevant operations involved by data ingestion process are:

- Definition and scheduling of importing jobs
- DBs health check
- Grouping data into multiple layers
- Pre-calculate relevant statistics
- Normalize taxonomies
- Check and cast geometry fields
- Create style for different types of layers and geometries
- Import GeoServer layers in Geonode
- Populate keywords and categories from DB view attributes
- Populate title and description fields
- Define Geofence rules

3.4 Data Interface

The basic operations performed by RDH application against PostGIS database are:

- Data extraction and pre-processing (pg/plsql + Python code)
- Spatial queries to extract spatial relations between datasets
- Extract administrative division boundaries

The basic operations performed by RDH application against GeoServer are:

- OGC/WMS service calls to view layers on map *
- (E)CQL to filter layers and contents on map
- SLD for styling multiple geometries and geometry types
- SLD filters for styling contents
- Geofence rules to restrict access to layers and services
- GeoWebCache for tile caching

* Specific layers are created in GeoServer by SQL Views and are used to extract and filter data to show on map.

3.5 Backend

Geonode is mainly used for uploading and managing vector and raster layers. Its models and APIs are used as well for:

- Enrich original data with metadata and additional informations (keywords and categories)
- Support frontend functionalities
- Publish a CSW catalogue of the layers
- Consume Geoserver APIs for management commands
- Proxy WMS requests under ACLs

Inventory, Analysis and Loss data are loaded into a dedicated database that will be described later in this document.

3.6 Frontend

The frontend is based on Mapstore framework for web mapping and it uses some of its core components along with custom components to build the User Interface. It is a single page application developed with React JS and Leaflet maps.

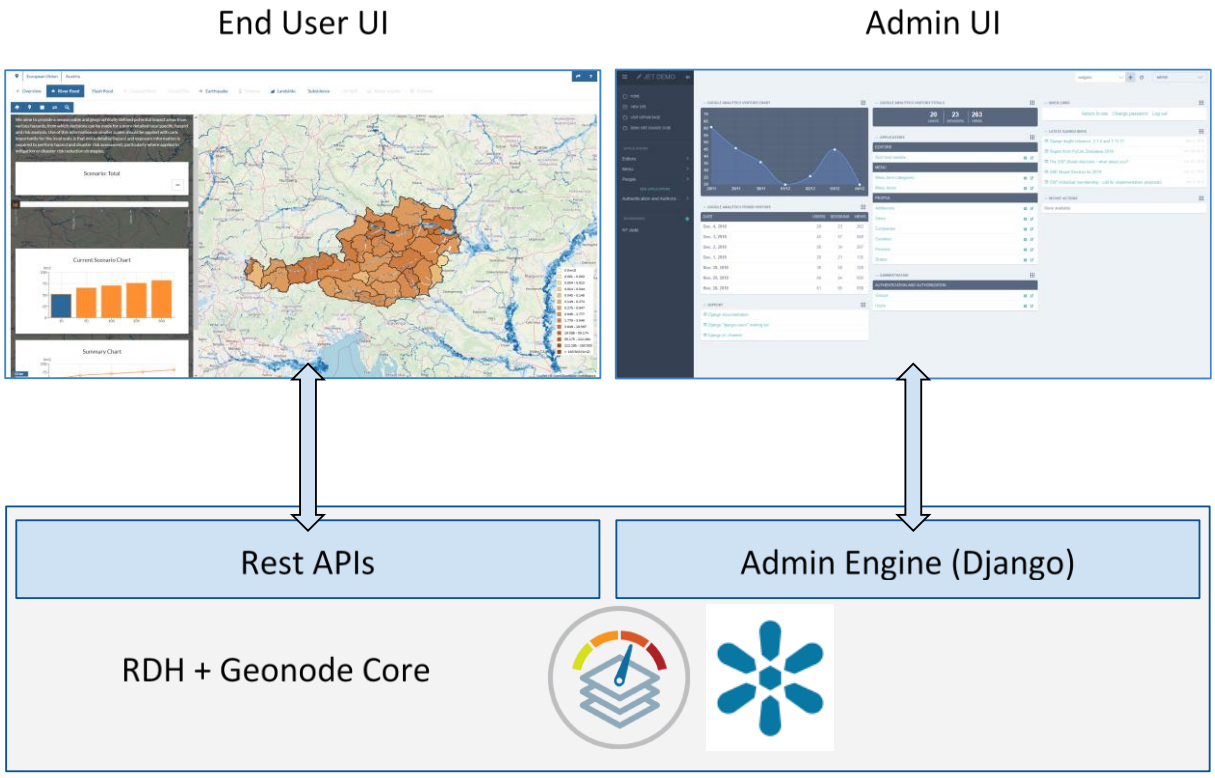

Figure 3. Interaction of User Interfaces with backend

4 Database architecture

4.1 Evolution of Loss Database architecture

The implementation of the base concept of Risk Data Hub required storing data for different purposes, such as Risk Analysis, Inventory of Assets and Damage Assessments.

The database was designed after the Loss Database for Disaster Risk Management proposed in latest EU publications (http://dx.doi.org/10.2760/647488). The result was at the same time an abstraction and an extension of that model.

Changes introduced during the development of Risk Data Hub are also reported in "Update of DRMKC Loss Database for disaster risk management" publication.

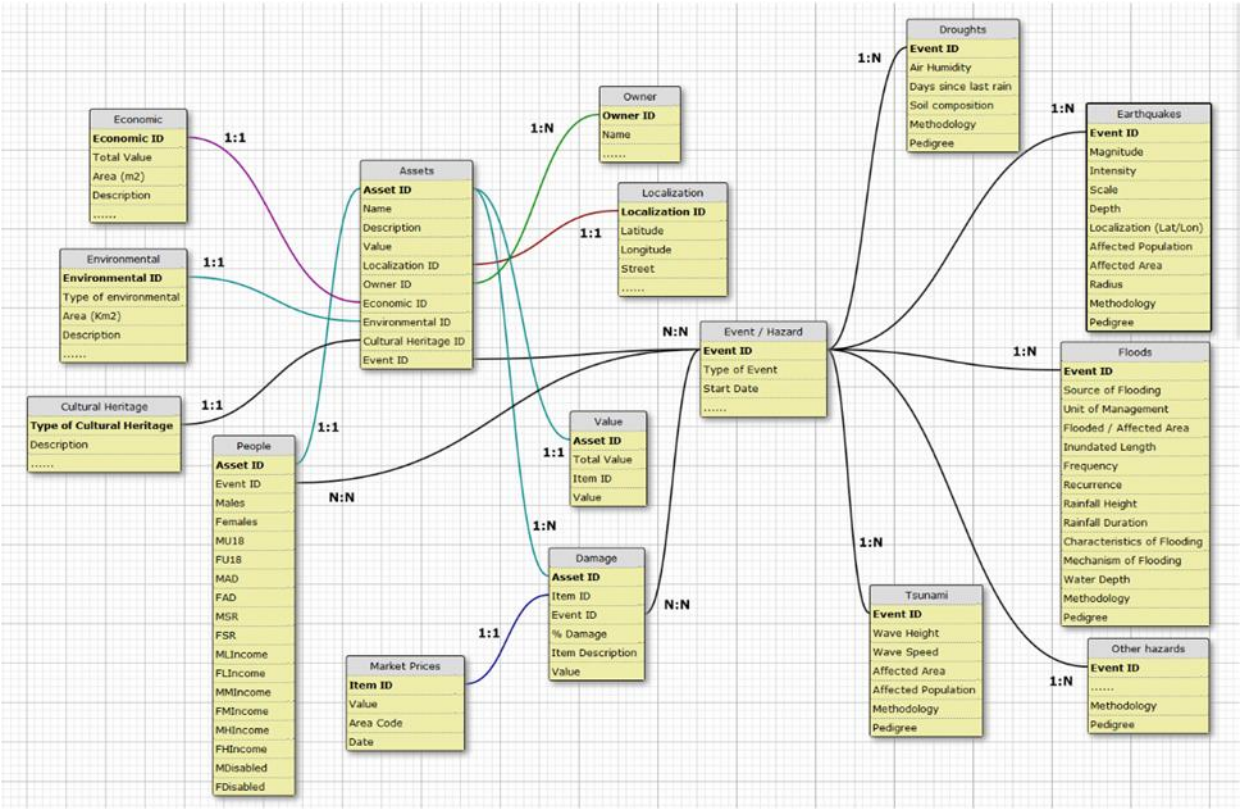

Figure 4. Loss Database diagram as of EUR 29063 EN publication

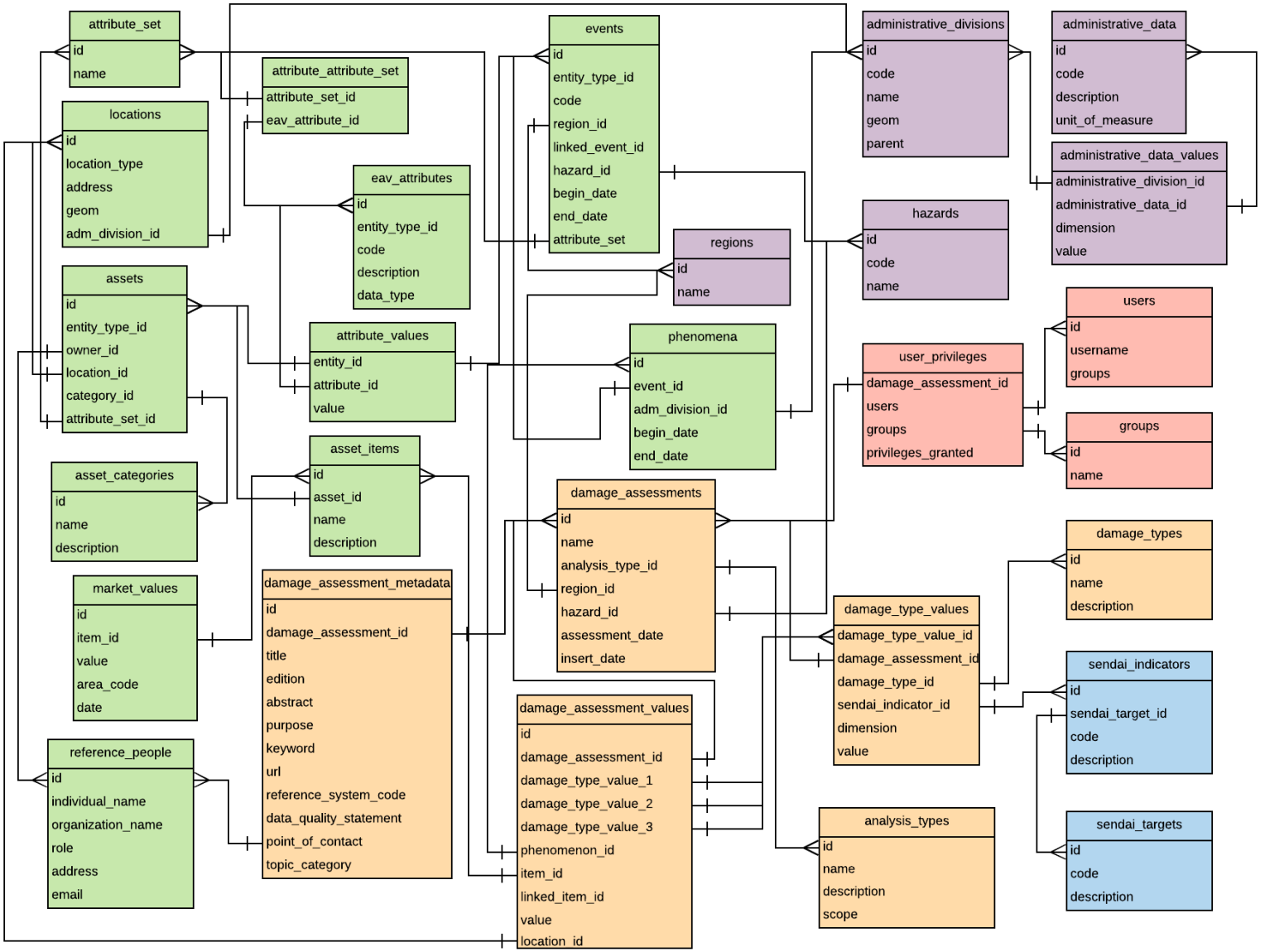

Figure 5. RDH Database diagram

Please, note that in the image above, tables are highlighted in different colours, corresponding to specific functionalities within the application.

What has changed?

4.1.1 Introduction of EAV data model

Entity–attribute–value model (EAV) is a data model to encode, in a space-efficient manner, entities where the number of attributes (properties, parameters) that can be used to describe them is potentially vast, but the number that will actually apply to a given entity is relatively modest. Such entities correspond to the mathematical notion of a sparse matrix. This particular model is well suited for events and assets, as the single entities have many different characterizations, depending on their type.

4.1.2 Events

The Event entity has been split into a "macro event" and a phenomenon, as explained in chapter 1.3. An Event table linked to a number of external tables (Hazards) no longer exists: all event attributes are stored in a centralized table, implementing the EAV (Entity Attribute Value) data model. Since attributes may differ from hazard to hazard, each Event instance is bound to a specific Attribute Set that ideally equals a hazard.

4.1.3 Assets

Similarly to events, also the Asset has been split into a "macro asset" and an item: each asset may contain one or multiple assets (e.g. a house containing pieces of furniture). Damages are linked to items, not to macro assets. Asset attributes are not described by an additional table for every type, but they use the EAV data model and they are also divided in categories. Fox maximum abstraction, People are considered as a specific asset category.

4.1.4 Locations

A location entity still exists, but defines also a type (e.g. fixed location, non-fixed location, people) and it's linked to damages as well; this way every single damage may have a specific location, as damage location may differ from asset location. Damage location could be a point, or a polygon that defines an extent.

As mentioned before, the Risk Data Hub database implements an abstraction of the Loss Database, which can be identified by the Inventory Section (green) of the schema. There are further sections that allow all the functionalities to exist.

Below follows a description of entities implemented, ordered by section (according to the colours).

4.2 Inventory section

This section entities that allow to store all inventory data needed, about both assets and events. The number of columns for assets and events is limited, because all possible descriptive fields are managed via the EAV (Entity Attribute Value) data model, which

allows to define new attributes at any time, without the need of changing the database structure.

Figure 6. Inventory section of RDH database

locations

Description: this entity is useful to store location of any type of asset (fixed, non_fixed, people), or the extent of a single damage

Fields:

- Id (int): unique identifier

- Location_type (enum): eg. Fixed asset
- Address (varchar):
- Geom (binary): geometry (could be Point or Polygon)
- Administrative_division_id (int): reference to administrative_divisions

assets

Description: generic entity affected by event (includes also People)

Fields:

- Id (int): unique identifier
- Entity_type (enum): defines entity type for mapping fitting attributes
- Owner_id (int): reference to reference_people
- Asset_location_id (int): reference to locations
- Asset_category_id (int): reference to categories
- Attribute_set_id: reference to attribute_set

asset_items

Description: single item included in the asset (equals to asset in the simplest case)

Fields:

- Id (int): unique identifier
- Asset_id (int): reference to assets
- Name (varchar):

asset_categories

Description: categories for assets; e.g. Buildings, Infrastructures or People

Fields:

- Id (int): unique identifier
- Name (varchar):
- Description (varchar):

market_values

Description: market value of items

Fields:

- Id (int): unique identifier
- Item_id (int): reference to assets
- Value (decimal):
- Area_code (varchar):
- Date (datetime): start validity date

reference_people

Description: could be the owner of an asset, author of publications, etc.

Fields:

- Id (int): unique identifier
- Individual_name (varchar):
- Organization_name (varchar):
- Role (varchar):
- Address (varchar):
- City (varchar):
- Zipcode (varchar):
- Country (varchar):
- Email (varchar):

eav_attributes

Description: attributes relevant to events and assets (and more) are defined in a single place. This feature allows to define new attributes at any time, without the need to change the structure of database.

Fields:

- Id (int): unique identifier
- Entity_type_id (int): defines entity type for mapping fitting attributes
- Data_type (varchar): defines data type (varchar, text, integer, decimal, datetime)
- Name (varchar):
- Description (varchar):

attribute_values

Description: Attributes values are stored in dedicated tables for each type of data (varchar, text, integer, decimal, datetime)

Fields:

- Entity_id (int): identifier of entity (event or asset)
- Attribute_id (int): identifier of eav_attribute
- Value: (see note below)

* The database diagram provided with this document includes a simplified view of the implemented EAV (entity, attribute, value) data model. Actually, a table for each data_type / entity_type exists in the database, e.g. event_attribute_values_varchar, event_attribute_values_text, and so on.

attribute_set

Description: attribute sets are used to link attributes to specific instances of an entity

Fields:

- Id (int): unique identifier

- Name (varchar):

attribute_attribute_set

Description: this is a relation between attribute_set and eav_attribute, so it's basically the content of an attribute set

Fields:

- Attribute_set_id: reference to attribute_set
- Eav_attribute_id: reference to eav_attribute

events

Description: an event is a generic entity which may be the cause of a damage.

Fields:

- Id (int): unique identifier of the event
- Entity_type_id (int): defines entity type for mapping fitting attributes
- Region_id (int): could be Europe, or any country corner
- Linked_event_id (int): optional link to an event identified as cause of the current one (chained events)
- Hazard_id (int): identifier of the hazard (eg. Flood)
- Begin_date (datetime): starting date of recognized event
- End_date (datetime): starting date of recognized event
- Attribute_set_id: reference to attribute_set

phenomena

Description: a phenomenon is part of a major event and has specific location and related assessed damage.

Fields:

- Id (int): unique identifier
- Event_id (int): related event
- Administrative_division (int): maps location of phenomenon
- Begin_date (datetime): starting date of recognized event
- End_date (datetime): starting date of recognized event

4.3 Administrative data section

This section gathers entities used for basic characterization of data stored for the Damage Assessments

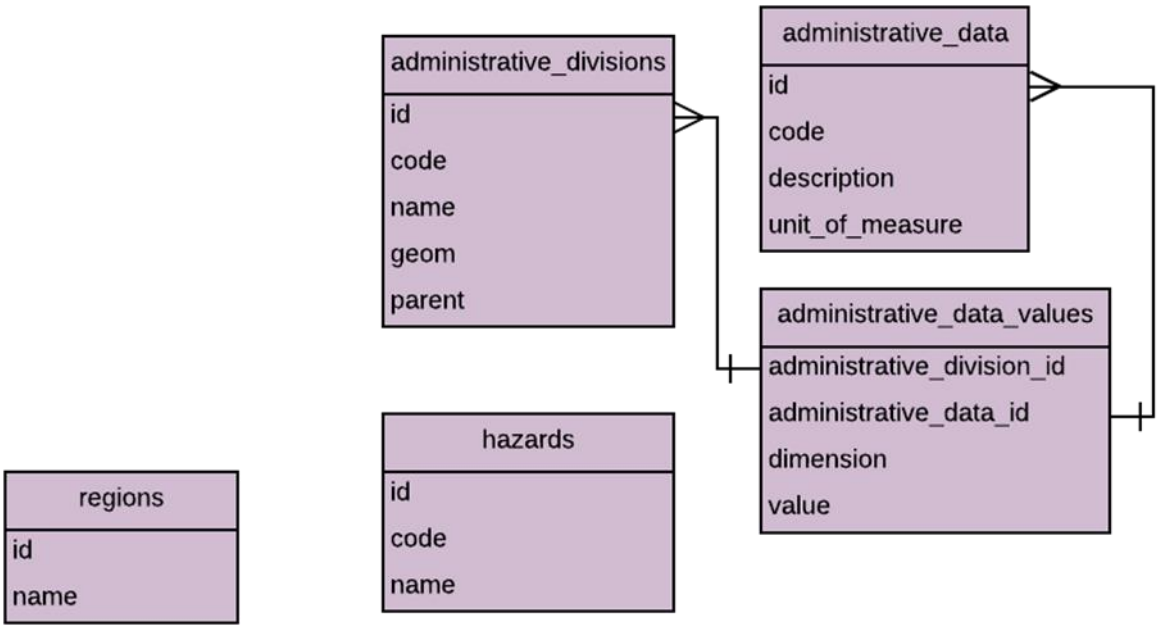

Figure 7. Administrative data section of RDH database

hazards

Description: definition of Hazard (e.g. River Flood)

Fields:

- Id (int): unique identifier
- Code (varchar): e.g. FL for Flood
- Description (varchar):

administrative_divisions

Description: This entity stores basic data of administrative divisions.

Fields:

- Id (int): unique identifier
- Code (varchar): ISO2 for countries, or relevant NUTS code according to Eurostat
- Name (varchar): name of administrative division
- Geom (binary): spatial data
- Parent_id (int): parent adm division

regions

Description: this is crucial for ownership management of data and visibility. Each user in the system belongs to a specific Region and so are the data owned by that user.

Fields:

- Id (int): unique identifier

- Name (varchar): name of Region (e.g. Europe, or country corner, like Austria)

administrative_data

Description: definition of data related to Administrative Divisions, like GDP, Population, Area, and so on.

Fields:

- Id (int): unique identifier
- Code (varchar): e.g. GDP
- Description (varchar): description of data
- Unit_of_measure (varchar): e.g. Mln EUR

administrative_data_value

Description: relation between Administrative Data and Administrative Divisions.

Fields:

- Administrative_division_id (int):
- Administrative_data_id (int):
- Dimension (varchar): e.g. Year 2018 of GDP
- Value (decimal):

4.4 Damage assessment section

This section represents the core of RDH, as defines the Damage Assessments and how the datasets are organized. The Analysis_type entity basically defines a dataset in terms of data analyzed (Buildings, People) and of scope (Risk Analysis or Historical Events). The Damage_type defines the dimensions used to measure data within the Assessment (e.g. Climate Change scenarios, return periods of events).

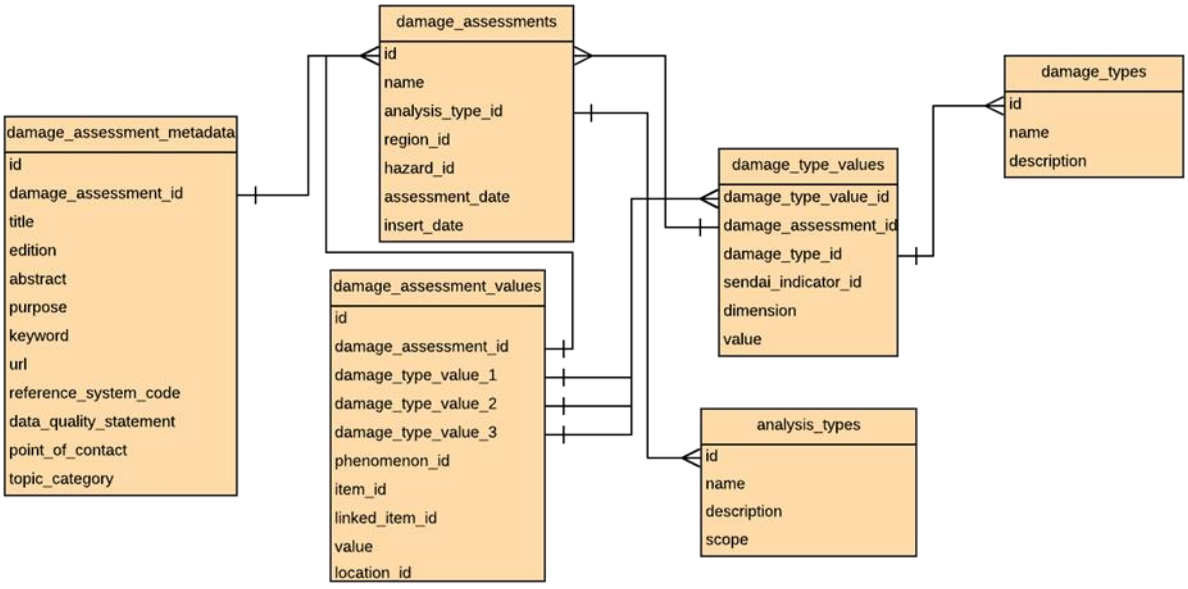

Figure 8. Damage Assessment section of RDH database

analysis_types

Description: defines the type of data analysed (e.g. Polulation, Buildings, Econonic values)

Fields:

- Id (int): unique identifier
- Name (varchar):
- Description (varchar):

damage_assessments

Description: definition of data measured

Fields:

- Id (int): unique identifier
- Name (varchar): name given (unique)
- Analysis_type_id (int): reference to analysis type
- Region_id (int): reference to Region, needed for Risk Analysis that do not use events
- Hazard_id (int): reference to Hazard, needed for Risk Analysis that do not use events
- Assessment_date (datetime): date declared for the assessment
- Insert_date (datetime): date of insertion in the database

damage_types

Description: definition of considered scenario. It is useful for complex analysis with predicted values in different declinations of a given scenario (e.g. climate change)

Fields:

- Id (int): unique identifier
- Name (varchar): name given (unique)
- Description (varchar):

damage_type_values

Description: relation between Damage_Assessment and Damage_Type

Fields:

- Id (int): unique identifier
- Damage_assessment_id (int): reference to damage assessment
- Damage_type_id (int): reference to damage type
- Sendai_indicator_id (int): reference to sendai indicator
- Dimension (varchar): e.g. Axis of a chart
- Value (varchar): value of damage type for given assessment and dimension

damage_assessment_value

Description: value assigned to the loss for given phenomenon, damage assessment, damage type and item

Fields:

- Id (int): unique identifier
- Damage_assessment_id (int): reference to damage assessment
- Damage_type_value_1(2,3)_id (int): damage type specific to DA
- Phenomenon_id (int): reference to phenomena
- Item_id (int): reference to asset_items
- Linked_item_id (int): eg. allows to map people into a building
- Value (decimal):
- Location_id (int): reference to locations, to store location (extent) of the single damage

damage_assessment_metadata

Description: complementary description of a damage assessment publication

Fields:

- Id (int): unique identifier
- Damage_assessment_id (int): reference to damage assessment
- Title (varchar):
- Edition (varchar):
- Abstract (varchar):
- Purpose (varchar):
- Keyword (varchar):
- Url (varchar):
- Reference_system_code (varchar):
- Data_quality_statement (text):
- Point_of_contact (int): point of contact for the publication (reference_people)
- Author (int): author of publication (reference_people)
- Topic_category: e.g. Environmental, Structure, etc.

4.5 Authorization section

These entities ensure the datasets are properly managed by their owners which may allow other users to perform operations (view, create, edit or delete).

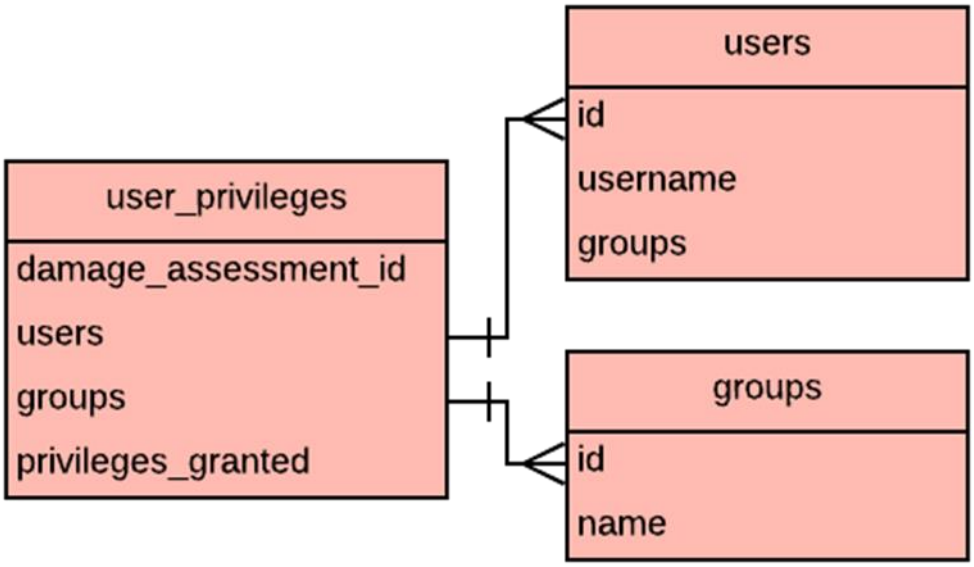

Figure 9. Authorization section of RDH database

users

Description: users registered

Fields:

- Id (int): unique identifier
- Username (varchar):
- Groups (array): list of groups the user belongs to

groups

Description: group of users for permission purposes

Fields:

- Id (int): unique identifier
- Name (varchar): name given (unique)

user_privileges

Description: privileges assigned to group or single user to perform actions against a Damage Assessment (view, create, edit, delete)

Fields:

- Damage_assessment_id (int): reference to damage_assessment
- Users (array): list of users for current entry
- Groups (array): list of groups for current entry
- Privileges_granted (array): list of privileges granted for current entry

4.6 Additional section

This additional section collects entities that are not strictly relevant to the main functionalities of the application. At this time, there are the definitions of Sendai Targets and Indicators. Please, note that these tables are used only to store a mapping between the Assessments performed by RDH and the Sendai Indicators, while outputs useful for Sendai reporting are generated, when data available is consistent, using a logic implemented in the source code of the application.

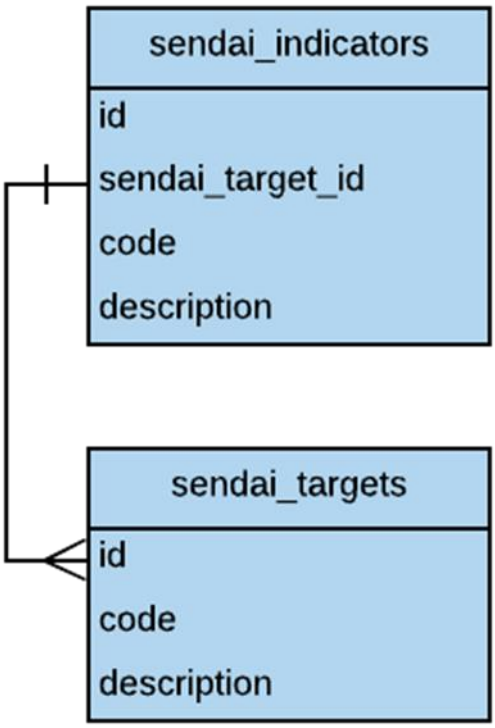

Figure 10. Additional section of RDH database

sendai_targets

Description: Sendai Target as defined by UNISDR specifications

Fields:

- Id (int): unique identifier
- Code (varchar): (unique)
- Description (varchar):

sendai_indicators

Description: Sendai Indicator as defined by UNISDR specifications

Fields:

- Id (int): unique identifier

- Sendai_target_id (int): reference to target
- Code (varchar): (unique)
- Description (varchar):

5 Conclusions

The way of using Risk Data Hub at local level is completely up to the user, who basically has two options: activate its account on the EU hosted platform, or deploy the whole application on a separate infrastructure of its own choice. While the first option is definitely faster to implement and does not include any costs for the user, the second one could be preferable if specific needs or constraints exist, for example restricted access to the internet, managing of extremely large datasets, customization of base models like Hazards or Analysis Types, or even concerns about privacy of some sensible data.

The application is still on a development phase and different collaborations have been established with both scientists and end users from several areas. Future work will then focus on new topics, like technological disasters, critical infrastructures and cultural heritage; this is why the identification of new features to be integrated is expected. The database architecture is by design hopefully flexible enough to handle all the complexity introduced, but as the application continues to grow, we cannot exclude that upgrades will be needed also for this part, which is typically the most critical to change.

References

Loss Database Architecture for Disaster Risk Management

(https://ec.europa.eu/jrc/en/publication/eur-scientific-and-technical-research-reports/loss-database-architecture-disaster-risk-management)

Ríos Díaz, F., Marín Ferrer, Montserrat: 2018 – Update of the DRMKC Loss database architecture for disaster risk management

Antofie, Tiberiu-Eugen, Luoni, S., Faiella, A., Marín Ferrer, Montserrat: Risk Data Hub – web platform to facilitate management of disaster risks

Entity-attribute-value model (EAV) – Wikipedia

(https://en.wikipedia.org/wiki/Entity-attribute-value_model)

Extract Transform Load (ETL) – Wikipedia

(https://en.wikipedia.org/wiki/Extract,_transform,_load)

GeoSAFE (https://geonode.ingc.gov.mz/)

Rasor (http://www.rasor.eu/rasor/)

ThinkHazard! (http://thinkhazard.org/en/)

Afghanistan Disaster Risk (http://disasterrisk.af)

World Bank (https://www.worldbank.org/)

GFDRR (https://www.gfdrr.org/)

CIMA Research Foundation (http://www.cimafoundation.org/)

EFAS (https://www.efas.eu/)

EFFIS (http://effis.jrc.ec.europa.eu/)

EDO (http://edo.jrc.ec.europa.eu/)

GHSL (https://ghsl.jrc.ec.europa.eu/)

Copernicus (https://emergency.copernicus.eu/)

GDACS (http://www.gdacs.org/)

EMM (http://emm.newsbrief.eu/NewsBrief/clusteredition/en/latest.html)

EM-DAT (https://www.emdat.be/)

HANZE (https://data.4tu.nl/repository/uuid:5b75be6a-4dd4-472e-9424-f7ac4f7367f6)

Python programming language (https://www.python.org/)

Django (https://www.djangoproject.com/)

PostgreSQL (https://www.postgresql.org/)

Geonode (http://geonode.org/)

GeoServer (http://geoserver.org/)

OGC services – OpenGeoSpatial (https://www.opengeospatial.org/standards/owc)

React JS (https://reactjs.org/)

Mapstore (https://mapstore.geo-solutions.it/)

Leaflet maps (https://leafletjs.com/)

List of abbreviations and definitions

RDH	Risk Data Hub
DRMKC	Disaster Risk Management Knowledge Centre
EAV	Entity Attribute Value
GFDRR	Global Facility for Disaster Risk Reduction
ETL	Extract Transform Load
EFAS	European Flood Awareness System
EFFIS	European Forest Fire Information System
EDO	European Drought Observatory
GHSL	Global Human Settlement Layer
GDACS	Global Disaster Alerting Coordination System

List of figures

GETTING IN TOUCH WITH THE EU

In person

All over the European Union there are hundreds of Europe Direct information centres. You can find the address of the centre nearest you at: http://europea.eu/contact

On the phone or by email

Europe Direct is a service that answers your questions about the European Union. You can contact this service:

- by freephone: 00 800 6 7 8 9 10 11 (certain operators may charge for these calls),

- at the following standard number: +32 22999696, or

- by electronic mail via: http://europa.eu/contact

FINDING INFORMATION ABOUT THE EU

Online

Information about the European Union in all the official languages of the EU is available on the Europa website at: http://europa.eu

EU publications
You can download or order free and priced EU publications from EU Bookshop at: http://bookshop.europa.eu. Multiple copies of free publications may be obtained by contacting Europe Direct or your local information centre (see http://europa.eu/contact).

EN-N

JRC Mission

As the science and knowledge service of the European Commission, the Joint Research Centre's mission is to support EU policies with independent evidence throughout the whole policy cycle.

EU Science Hub
ec.europa.eu/jrc

@EU_ScienceHub

EU Science Hub - Joint Research Centre

Joint Research Centre

EU Science Hub

doi:xx.xxxx/xxxx

ISBN xxx-xx-xx-xxxxx-x