Open Science Platform for Disaster Risk Reduction

Joost van Ulden, William Chow, Drew Rotheram-Clarke, Damon Ulmi, Anthony Fok

Table of Contents

Executive Summary

This is the abstract. s

1. Introduction

A brief introduction to Pathways and the role of the platform in delivering the science outputs. Emphasis on evloving nature of the platform.

2. Platform Requirements

To support the Pathways activities and deliverables a platform with the following features and functions were desired:

- Support open and collaborative development of science outputs
- Centralize access to the science outputs
- Provide tools and applications to engage the public and support decision making

The generation of science outputs is becoming increasingly reliant on software code to automate processing, quality control, and publishing. The scale and scope of the data involved in the project necessitated an approach that not only streamlined the production of science-based outputs, but also provided for a high degree of collaboration across many disciplines (e.g., policy, technology, and science) and stakeholders (e.g., provincial, municipal).

Decision support requirements were well understood at the outset. A comprehensive set of requirements for a multitude of stakeholders was developed [link to Minerva doc]. The diversity of the stakeholders and their specific needs necessitated a multi-channel approach since a single application was deemed to be insufficient to serve all use cases effectively.

With development being led out of the Government of Canada imposed several requirements for publishing science outputs including standardized metadata, open data, support for both official languages, accessibility, and compliance with scientific integrity and publication policies. The scientific integrity and publishing requirements were particularly problematic as they traditionally pushed the development of science behind closed doors, as such a novel and balanced approach would need to be taken to support the requirements of Pathways project.

3. Overview

GitHub was chosen as the platform to support the development of the science outputs and related software, documentation, and tools. While well known in the software development community it is lesser known in the science community. However, the core concepts behind GitHub (e.g.: versioning, repositories, etc.) were more or less understood by key contributers.

Where possible, runnable code is provided to ensure transparency in the science. For example, an interested party could make a copy of a repository and replicate a particular output (e.g.: dataset) or even the entirety of the OpenDRR infrastucture. The platform makes heavy use of containerization and infrastructure as code technologies for rapid deployment on personal computing devices or on the cloud.

Built in features of GitHub such as continuous integration and deployment, community building, websites, and secure workspaces were seen as ways toward achieving an open and collaborative approach to science, one that seeks to build concensus and drive engagement throughout it's lifecycle.

[OpenDRR GitHub] | *img/opendrr-GitHub-en.png*

Figure 1. OpenDRR GitHub

Due to the diversity of use cases and user profiles for the information products (e.g.: maps, visualizations) it was clear that a single solution would not be sufficient. It was determined that a purpose built web application in addition to a customizable dashboard environment would likely meet the needs of all users.

The dashboard environment is provided by Kibana which connects to an Elasticsearch document store. This environment is highly customizable and allows individuals and organizations to create public or private spaces where they can query and vizualize all available data.

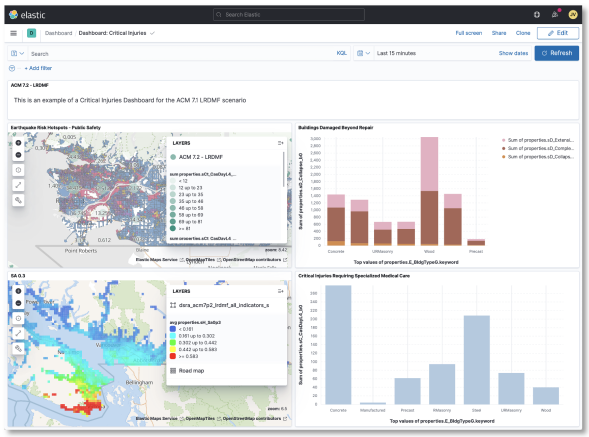

Figure 2. OpenDRR Kibana

The purpose built application, called RiskProfiler, is a custom web application that connects to a variety of services including the Elasticsearch document store. Built to be highly scalable and user friendly, it seeks to communicate the key messages relating to natural hazard (i.e.: earthquake) risk. It provides for some basic filtering and visualization in an area of interest, such as a community or region.

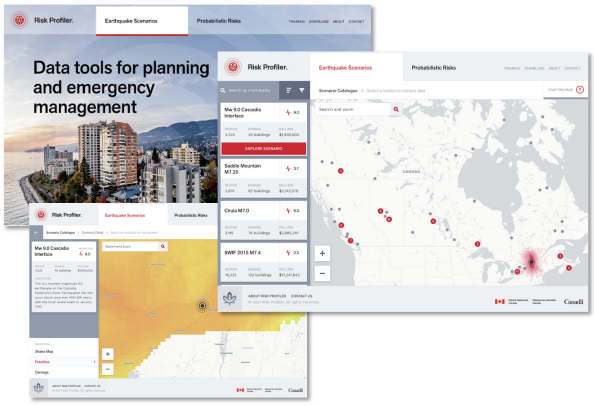

Figure 3. OpenDRR RiskProfiler

4. Architecture

4.1. Data Processing Pipeline

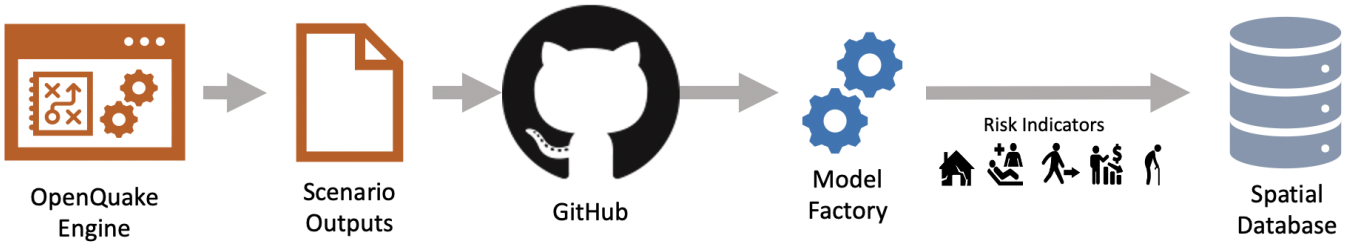

Figure 4. OpenDRR Data Processing Pipeline

The OpenDRR platform data processing pipeline consists of several open source technologies, namely PostgreSQL with PostGIS extension, Python, ElasticSearch/Kibana and PyGeoAPI. Each application is containerized using separate Docker containers which are orchestrated using Docker-Compose.

Source data for this project includes the National Human Settlement Layers (physical exposure and social vulnerability), the National Seismic Risk model for Canada (CanadaSRM2, probabilistic), Canada's National Earthquake Scenario Catalogue (deterministic), and boundary geometries are stored on the Open Disaster Risk Reduction Platform (OpenDRR) GitHub site using a large file storage service called Git LFS. The stack of technologies can be configured to run on a local machine with PostgreSQL and ElasticSearch environments running locally or can be configured to communicate with cloud deployments of these services. For example, the stack can be configured to communicate simultaneously with a PostgreSQL database on Amazon Web Services (AWS) Relational Database Service (RDS) and a Containerized ElasticSearch environment on an AWS Elastic Container Service (ECS). A common use case is to run the ETL process on a local desktop or AWS Elastic Compute Cloud (EC2) with a local containerized PostgreSQL environment and a cloud deployed ElasticSearch environment which can support a public or private application programming interface (API).

The ETL process currently relies on a main shell script titled 'add_data.sh' which downloads the required source data files including comma separated value (csv), and GeoPackages (gpkg) into the

local filesystem and performs necessary transformations to load the data into a PostgreSQL database. The script is written to be flexible enough to load any number of conformant deterministic earthquake scenarios and national seismic risk model data? and allows some flexibility in the contents of the input data as long as a minimal number of required fields are present. Once the required source data has been loaded into the database, a number of Structured Query Language (SQL) scripts help transform the data into a framework of meaningful earthquake risk indicators which in turn gets pushed to ES/Kibana ??? .

Discuss the postgresql database? Briefly? Schemas? The PostgreSQL database is created and has PostGIS extension enabled to allow for spatial data queries. The source datas are loaded into their respected schema names and the results are generated into their respected schemas with results_ prefix.????

Results are calculated and/or aggregated at different spatial scales dependent on the data. For National Human Settlement Layers the results are in settled areas. The probabilistic and deterministic data are calculated at the building level and aggregated up to the settled area and census subdivision levels.

These results are aggregated at several different spatial scales, building, sauid, and csd level. The building level is the finest spatial resolution and aggregates all buildings of simmilar construction type for a given neighborhood. The Sauid level of aggregation groups all building types across the whole neighbourhood polygon. The Census Sub-divison (CSD) aggregation is a coarse aggregtion of all assets within a given CSD as defined by Statistics Canada.

4.2. Data Dissemination

4.2.1. API Stack

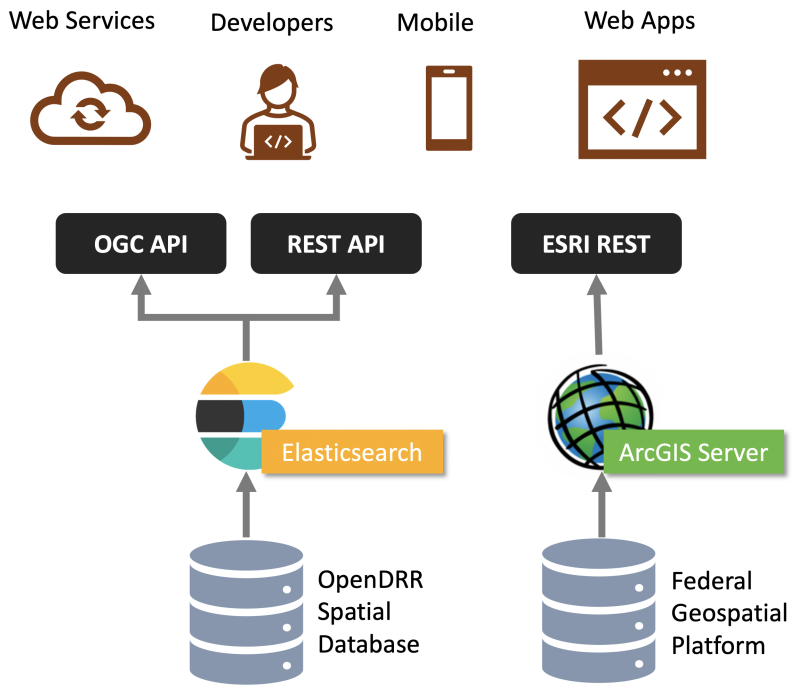

Figure 5. OpenDRR API Stack

4.2.2. Application Stack

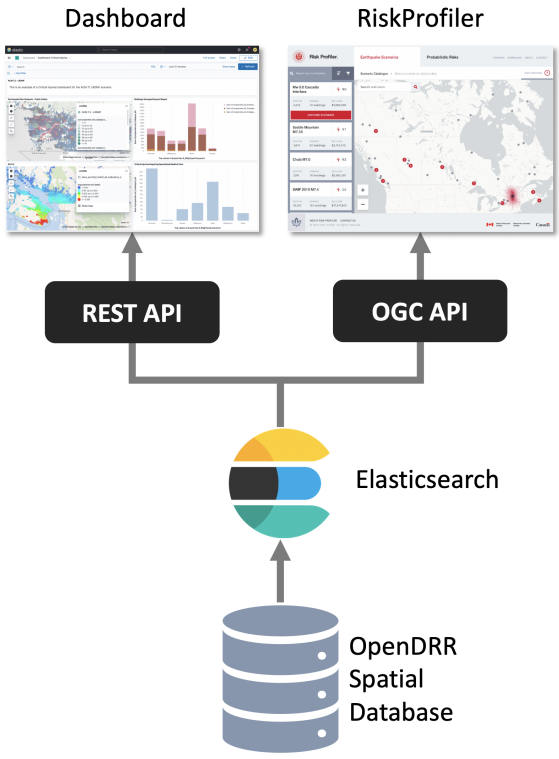

Figure 6. OpenDRR Application Stack

5. Lessons Learned

5.1. Open Source, Open Data, Open Science, Open Process

- (May seem scary at first?) but many benefits
- full openness, transparency, accountability
- GitHub platform offers a lot of features and flexibility that we need
- Industry best practices
- Mutually beneficial relationship with GEM (OpenQuake)
- pygeoapi
- etc.

5.2. Security

Example: Apache Log4j vulnerability

- We got confirmation from ITSEC that our ES endpoint was attacked. Since we were running 7.12.0 the attack failed. Lesson: keep your software up to date. (Credit: Joost,)
  - Mitigate Log4j2 / Log4Shell in Elasticsearch
- (a little bit about) keeping up-to-date vs. compatibility-breaking changes (e.g. Python 3.8 to 3.9); be prepared for extra work hours (manpower?) to resolve compatibility issues during upgrade…

5.3. [optional] The use of Agile principles on GitHub; benefits and challenges

5.4. [optional] opendrr.github.io got falsely flagged as phishing or malicious

- CIRA, Google, VirusTotal; took a while to get off e.g. Microsoft SmartScreen
- Possibly due to the use of non Canada.ca template on a non .ca domain?
- https://github.com/OpenDRR/opendrr/issues/122

6. Future Development

6.1. Risk Profiler (v1)

- Expected to be published before April 2022 (kudos to HabitatSeven)

6.2. Increase in scope: Import and process other datasets besides earthquake

- Flood: https://github.com/NRCan/CanFlood (Open-source flood risk modelling toolbox)
- Wildfire
- Mudslides

6.3. Continued automation, optimization, etc.

- Increased use of GitHub Actions workflows
- Full CI (Continuous Integration, Continuous Delivery)
- Cost-savings (e.g. move off Git LFS wherever possible; cheaper alternatives…)
- More automated testing, to catch errors as early as possible (preferrably at the pull request stage)
- Test-driven development (TDD) / Bahaviour driven development (BDD)?

6.4. Promote the project

- as a model with example that that other groups (countries/governments/companies etc.) can use
- (as open-source project) invite participation from open-source, scientific community, and the general public
- through web site(s), seminars/talks, publications

6.5. More…