27. January 2018
Matthew Moy de Vitry
Version 0.7-web

water-fountains.org | **work package 2**

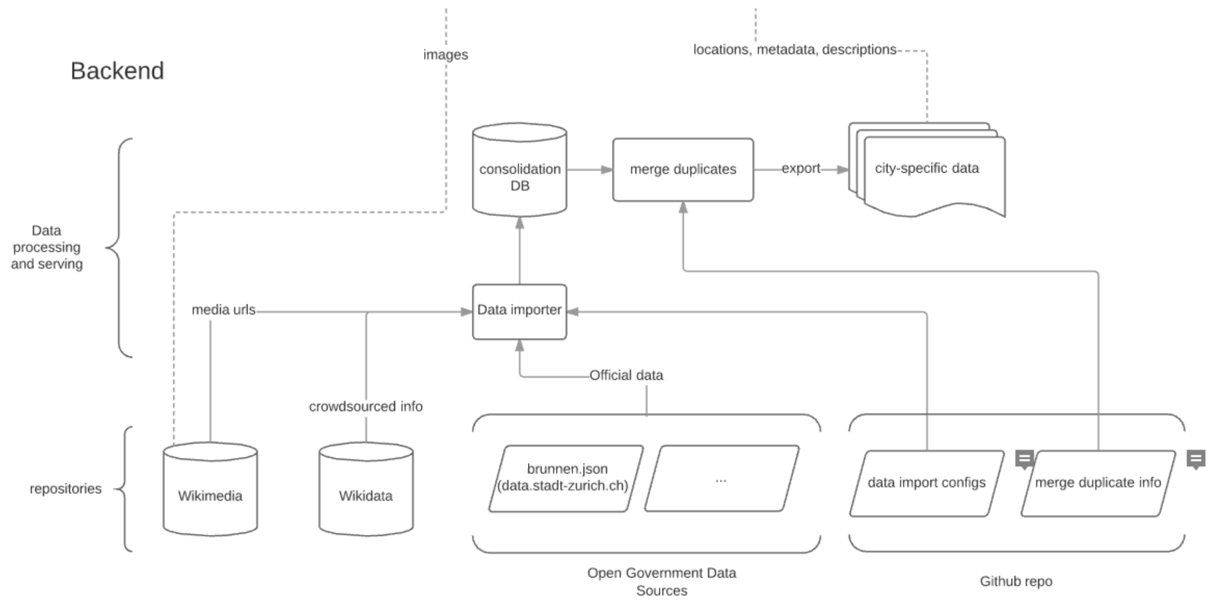
Multi-source data consolidation

FEATURES

- Consolidate data from Open Data Zurich, WikiData and WikiMedia
- Conserve data trace (date of download, and source of data)

DATA PROCESSING CONCEPT

The consolidation of official and unofficial data sources is the most innovative part of the work package. The trick is in merging an arbitrary number of data sources with arbitrary data structures.



https://www.lucidchart.com/invitations/accept/59c350e2-310e-4279-8169-7044839b7307

Data collection

Data is collected from a selection of repositories (see figure below), either on a schedule or triggered by webhooks, if possible. In the case of media files hosted on Wikimedia, only the metadata and a link to the media file is collected.

To increase transparency and protect against data sources going offline, backups of the data will be made on a weekly basis.

Data assimilation

The data collected from the different sources is imported into a data structure (e.g. database table, see "consolidation DB" in figure below) where each row corresponds to information for a single fountain as read from a single data source. It is thereby possible to have multiple rows for a single fountain, and the origin of each information is memorized.

Data importation (see "data importer" in figure below) into the data structure thanks to scripts and configuration files which

1. map property names from the source to the data structure (e.g. "name_fr": "nom")
2. Indicate nodata values (e.g. 'inconnu')
3. Provide missing metadata (e.g. city = 'Geneva' or water_quality='excellent')
4. Set the authority level of the datasource (Zurich OGD is of higher authority than Wikidata), relevant for the merging process
5. Provide information on the estimated accuracy of the fountain coordinates (e.g. +/- 1 m)

Data exporting/merging

The data served to the web app must meet certain quality standards (no duplicates, certain fields required). The data export step polishes the data quality and formats the data as a json for the web app:

1. **Merge duplicates**:

   a. The rows of the data structure are grouped by similarity of location and given name. For the location, a distance threshold can be defined. For the comparison of names, many algorithms are available: Hamming distance, Levenshtein distance, Damerau–Levenshtein distance, Jaro–Winkler distance. A smart combination of the two distances must be designed (e.g. if the name matches perfectly, then the location doesn't matter as much). It would be clever to normalize the geometric distance with the estimated accuracy of the coordinates. Warning: two empty names must have a non-zero distance.

   b. Within each group, rows are squeezed together into a single row. If for a given property multiple proposals (coming from multiple data sources) exist, then the proposal with the highest authority level is conserved.

   c. In the merging process, the origin and language of each element is carried along.

2. **Quality check**: The remaining rows should represent unique fountains. The quality should be guaranteed as follows:

   a. Fountains without coordinates are removed.
   b. Fountains with unknown drinking water quality should be labeled as such
   c. Default values can be set in absence of information. (e.g. year = ‘unknown’)

3. Save as JSON: The resulting dataset is saved as a json to be easily read by the app. One JSON is saved for each city.

SOFTWARE / LANGUAGES

The data will be collected and preprocessed using Python, an efficient scripting language. As the coverage of the app grows and users are empowered to make contributions, a database will need to be set up and an API will need to be developed. We would then consider a NodeJS server with a database like PostGres, MariaDB, or MongoDB for the consolidation database.
For the case that data is migrated to a DB, the license must make sure that the database content remains in public domain and amendments will be public domain as well.

HARDWARE

Hosting

Static data files on an Apache server are sufficient for version 1 of the app. When the app develops so that users can contribute to open datasets, an API and virtual machine will be necessary for managing users and advanced requests.

Data processing

The data preprocessing and will be conducted on a server, which then uploads the generated data files to the hosting server via FTP.

TASKS

1	**Data collection and pre-processing**
1.a	Data collection and backup Open Data Zurich Wikidata Wikimedia
1.b	Group data by similarity Merge sources Quality check Export and upload
1.c	Task scheduling and coordination
2	**Consolidation data structure and**

	backup
3	**Deployment**
3.a	Install requirements on VM, set up automated deployment and run scheduling.
3.c	Infrastructure: Debian VM