Cartography M.Sc.

Master thesis

Creation of an automated workflow for the production of a printed map

Enock Seth Nyamador

Technical University of Munich

Technische Universität Wien
Vienna University of Technology

Technische Universität Dresden

University of Twente. ITC

2022

TECHNISCHE UNIVERSITÄT DRESDEN

Creation of an automated workflow for the production of a printed map

submitted for the academic degree of Master of Science (M.Sc.)
conducted at the Department of Geosciences, Institute for Cartography,
Technical University of Dresden.

Author:	Enock Seth, Nyamador
Study course:	Cartography M.Sc.
Supervisor:	Prof. Dipl.-Phys. Dr.-Ing. habil. Dirk Burghardt (TUD)
	Mathias Gröbe M.Sc. (TUD)
Reviewer:	Olesia Ignateva (TUW)

Chair of the Thesis
Assessment Board: Prof. Dipl.-Phys. Dr.-Ing. habil. Dirk Burghardt

Date of submission: 10.10.2022

Statement of Authorship

Herewith I declare that I am the sole author of the submitted Master's thesis entitled:

"Creation of an automated workflow for the production of a printed map"

I have fully referenced the ideas and work of others, whether published or unpublished. Literal or analogous citations are clearly marked as such.

Dresden, 10.10.2022 Enock Seth Nyamador

“Sharing knowledge is the most fundamental act of friendship. Because it is a way you can give something without loosing something.”
– Richard Stallman

Acknowledgement

First, I would to thank my supervisors, Prof. Dr.-Ing. Habil. Dirk Burghardt and Mathias Gröbe M.Sc. for their support, motivation, time and continuous feedback on this research is very much appreciate.

Secondly, I wish to thank the reviewer, Olesia Igneteva for the useful feedback and remarks through the research process of this master thesis.

Thirdly, I wish to thank Juliane Cron for many things. I wish to thank my friends, family and loved ones who contributed to this journey in several ways that words can not describe.

Finally, I wish to thank everyone who contributes to OpenStreetMap, uphold its values and not for quantity. Thanks to projects such MapOSMatic, OpenTopoMap, and openmap.lt where inspirations were also drawn from. I am also grateful to Free/Libre and Open Source Software advocates, developers, contributors and users, let the FOSS be with you.

Typesetting of this document was done with LaTeX and some vector graphics were created with Excalidraw or modified with Inkscape.

Abstract

Cartography is well known as the art, science and technology of making maps. Map production is an evolving and involving investment. Choosing the right tools is crucial to the achievement of goals. Selecting the most appropriate tool and software stack for the map production workflow is a Multi-Criteria Decision-making process. This thesis focuses mostly on the technology side of map production by exploring and selecting free and open source software and tools for map production using open data from OpenStreetMap in Georgia and Ghana as a proof of concept. This research investigated, tested and proposed a combination of tools that can be chained to achieve a workflow for print map production from the collection, processing, transformation, and visualisation of a final map. In order to keep derived changes and modifications, an adjustable data model was implemented to handle updates using feature history and one measure of similarity to handle both semantic labels and geometric changes.

Keywords: printed maps, OpenStreetMap, map production, free and open source software, cartography

Contents

Acronyms

AHP Analytic Hierarchy Process

DCM Digital Cartographic Model

DLM Digital Landscape Model

FOSS4G Free and Open Source Software for Geospatial

GDAL Geospatial Data Abstraction Library

GIS Geographic Information Systems

GPS Global Positioning System

GUI Graphical User Interface

MCDA Multiple-Criteria Decision Analysis

OSM OpenStreetMap

SDI Spatial Data Infrastructure

VGI Volunteered Geographic Information

WPM Weighted Product Model

WSM Weighted Sum Model

List of Figures

List of Tables

1 Introduction

1.1 Motivation and Problem Statement

Map production has long been a core practice of cartography (Virrantaus et al., 2009), as cartography itself is the art, science and technology of map making. The advent of technology and the web have resulted in novel ways of making and using maps. Notwithstanding, print maps continue to play an essential role in urban planning, hiking, tourism, etc., as there will always be demand for paper maps (Virrantaus et al., 2009). Print map production results from a series of geo-processing steps involving several spatial layers and tools. Traditionally, the process of map making is manual, requiring a considerate amount of time and efforts which is costly (Çobankaya & UluğtekiN, 2013). Automation of the process and workflow involved in print map production is key to reducing both time and minimising cost.

Automated map production workflow is a machine-driven process which results in the completion of tasks that relate to the compilation, construction, or output of a map product (Buckley & Watkins, 2007). They represent the physical world and communicate information and knowledge of an area of interest at a specific time.

National, Regional, and District Mapping Agencies and companies are often involved in print map production. A vital part of the map making process is spatial data. The responsible institution must maintain up-to-date geodatabases used in map production to ensure that the printed maps are up-to-date. In addition, an important aspect is that data is not always readily available for all places worldwide. In this regard, other data sources could be used as the primary source or as a supplement for map production.

Volunteered Geographic Information (VGI) systems have provided alternatives to the traditional sources of data through citizen sensing (Goodchild, 2007). OpenStreetMap (OSM), which Steve Coast started in 2004 is a global project and community creating Wikipedia-like geodatabase available to the public under Open Data Commons Open Database License (Ramm et al., 2011). The OSM data is created by volunteers and institutions around the world to map the world from scratch without copyright. OSM cartography comprises 20 zoom levels at different scales ranging from 1:1 000 to 1:500 000 000 (Davis & Kent, 2022). Contributions are made with applications referred to as editors. These editors allow data modification with the help of local knowledge, satellite imagery, GPS traces, and other sources as references. Open Data released by National Mapping Agencies (NMAs) have found their way into OSM, allowing updates to be made by volunteers such as in the case of Canada (Bégin, 2012) and some European countries (Olteanu-Raimond et al., 2017). OSM has been used as primary data where datasets do not exist for quick visualisation. It has also been used as secondary data or integrated to complement existing primary datasets provided by NMAs (Kunz & Bobrich, 2019; Sarretta & Minghini, 2021).

In contrast with Spatial Data Infrastructure (SDI) systems, which are used to manage and store data, OSM data lacks standard conformity, provenance information, quality standards, and the capability of complex feature modelling (Müller et al., 2012). Notwithstanding, OSM has been successfully transformed and used for map production at both national (Kunz & Bobrich, 2019) and regional levels (Olteanu-Raimond et al., 2017). Furthermore, diverse evaluations performed by Haklay and Weber (2008) and Zielstra and Zipf (2010) have proven that, especially in urban regions, OSM can compete or even surpass data offered by commercial data providers or governmental authorities (Goetz et al., 2012).

In Cartography, a well-structured data model is crucial for both generalisation and visualisation. Generally, when OSM data is to be used for map production, it is uncertain where to commence. This is due to the availability of different ways in which the data can be acquired. Even though several tools and software exist for handling OSM data, the mapmaker is not always aware. Especially how these tools and software better relate to their needs such as data preparation, transformation, visualisation, and integration. Manual processing is time-consuming, especially when it becomes repetitive and datasets become larger leading to overburdening of Graphical User Interface (GUI) tools involved and the mapmaker.

This thesis aims to provide an overview of Free and Open Source for Geospatial and solutions that can be used to create an automated workflow for the preparation of OpenStreetMap and other data sources leading to the production of a printed map. It will therefore contribute to the field of cartography by providing an overview and evaluation of free and open source tools for preparation, transformation, visualisation, and integration of OSM data for map production.

1.2 Research Objectives

Free and open source software and open data for many represents accessibility to otherwise inaccessible geospatial workflows in terms of cost and availability. Commercial data used in Geographic Information Systems (GIS) is available through a relatively small number of merchants or vendors which produce highly accurate, precise, and detailed information. This is produced, however, at a cost that many small and large businesses, private consultants and startups cannot afford (Markieta, 2012); the same applies to off-the-shelf proprietary software.

This thesis focuses on exploring and selecting suitable combinations amongst available free software and tools that can be used for creating a workflow for producing printed maps using open data from OpenStreetMap whilst retaining derived data and changes.

In map production, it is usually necessary to keep adjustments such as generalisation and specific manual changes but retaining these changes during updates are not always the case. This new approach to the problem of handling changes as a result of applying manual changes to OpenStreetMap data or combine other datasets with OSM in a geodatabase for continuous map production.

Generally, this study seeks to achieve the following two main objectives:

1. Create a general workflow for print map production using Open Data and using Open Source tools.
2. Design and implement a method for handling geometry and attribute changes from OSM into a derived geodatabase.

1.3 Research Questions

In order to achieve the objectives of this study, the following main and sub-questions will be answered:

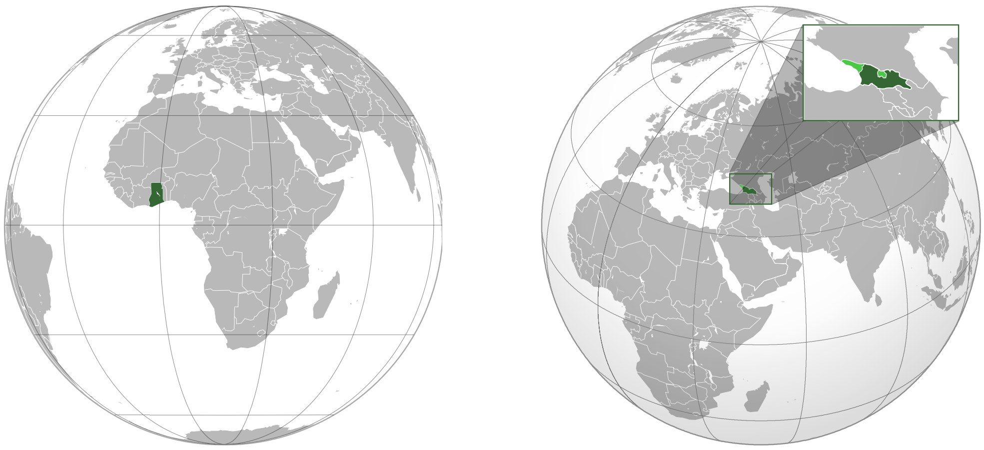
1. RQ1 – What is required to create a general workflow for print map production using Open Data and Open Source?
   a) How can a general workflow for print map production be achieved using OpenStreetMap as the main data source?
   b) Which categories of tools are suitable for the specific tasks in the workflow?
   c) How can various tools for the specific steps identified in RQ1.2 be compared and evaluated according to functions, performance, flexibility, and platform support?
2. RQ2 – How can updates with data changes and adjustments be integrated into existing map data?
   a) How can a data model be designed for handling changes?
   b) How can a suitable data model be implemented to handle updates using selected tools from RQ1?
   c) Which conflicts can result from updates and manual edits, and how can they be resolved?

1.4 Study Area

The initial study area for this thesis is the Mt. Ushba region located in the Svaneti Region of the Republic of Georgia as part of the Mt. Ushba Mapping Campaign at the TU Dresden. Due to data processing constraints and non-familiarity with the region, the workflow is built around Ghana and later applied to Georgia. Ghana is a sub-sahara African country with a growing population of 30 million people according to National census 2021 (Ghana Statisical Service, 2022).

As a country with several economic and political challenges and a low GDP $77.59 billion (The World Bank, 2022), Ghana has sixteen administrative regions which are further subdivided into districts, municipalities, and metropolitan areas on the economic level and constituencies at the political level.

Ghana lacks a well-structured national addressing system and accessible spatial data. In the absence a functional spatial data infrastructure, data from OpenStreetMap, and commercial services such as Google Maps are used by business, national, and local government for providing own services (Dumedah et al., 2022).



(a) Location of Ghana in Africa (Wikimedia Commons, 2011) **(b)** Location of Georgia in Asia (Wikimedia Commons, 2011)

Figure 1.1: Location of Ghana and Georgia

The Survey Department of the Lands Commission is the legally mandated agency for creating up-to-date maps of Ghana. According to the Lands Commission, the last time Ghana was comprehensively mapped in 1974 (GhanaWeb, 2020).

The National Framework for Geospatial Information Management (NAFGIM), which was started in 2000, was Ghana's close attempt to establish a functional SDI. It is highlighted by Masser (2005) as one of the early SDIs in Africa. Unfortunately, NAFGIM has ceased to exist; less or no information seems to be available today (Owusu-Banahene et al., 2013).

Ghana is also among the early pioneers of open data portals on the African continent. This saw the launching of Ghana Open Data Portal[1] in November 2012, thanks to the initial support from the World Wide Web Foundation and the participation of other stakeholders.

[1] https://data.gov.gh

Notwithstanding this project, has also stalked at a point and has so far seen slow growth since its inception (Verhulst & Young, 2017).

1.5 Outline of the Thesis

The **Introdution** provides an overview of the research, highlights the problem statement, and states the motivation. It also outlines the research objectives and research questions. The **Background** introduces data sources, the theoretical background and types of open data, including national/public open data and Volunteered Geographic Information (VGI). It also presents the historical and theoretical background of map production to current workflows and software. The **Methods** introduces software selection and methods for handling updates. Additionally, it presents requirements for the workflow implementation and implementation of an adjustable data model for handling updates. The **Implementation** is the step-by-step description of the implementation of the workflow for map production and implementation of an adjustable data model for handling updates. The **Discussions** elaborates on the outcome of software selection, the workflow, and the data model implemented. Finally, in the **Conclusion**, the research questions formulated for this thesis were answered. Limitations for this work were also outlined to contribute to future research.

2 Background

Map production has a long history from the medieval times involving manual modes to modern day cartography making use of technology and software. This chapter introduces data sources, the theoretical background and types of open data, including national/public open data and Volunteered Geographic Information (VGI). It also presents the historical and theoretical background of map production to current workflows and software.

2.1 Data Sources

An important element of a map production process is data. Data can either be spatial or non-spatial. National Mapping Agencies and private entities are major data collectors and providers. Access to the data is based on a couple of factors, most of which are based on licensing.

2.1.1 Licences

Data availability and use are dependent on the License of the data, copyright and license agreements (Vondrakova & Vozenilek, 2016). When data is produced, it is under some sort of copyright and ownership unless otherwise explicitly stated. National Mapping Agencies collect most data in their jurisdiction since they are mandated by laws to do so. The data source of every cartographic product is of great importance and is considered to be trusted or authoritative when produced by responsible data providers such as NMAs.

2.1.2 Open Data

Open data is data that is openly accessible, exploitable, editable, and shared by anyone for any purpose, even commercially. Open data is licensed under an open license. The issue of open data closely relates to copyright issues, because copyright law relates to a broad range of databases and data files. Knowing copyright issues, licensing and related rights are essential to the right “data opening". The copyright and open data issues are extremely topical subjects of current research and have practical use in Cartography and GI Science (Vondrakova & Vozenilek, 2016).

Generally, open data is the idea that some data should be freely available to everyone as they wish without restrictions (Vondrakova & Vozenilek, 2016). There are many initiatives for opening data at regional, national, and transnational levels. It is the current trend in data policy, which can have a fundamental impact on the development of geospatial applications and cartographic visualization (Vondrakova & Vozenilek, 2016).

Many countries in the developing world such as Ghana lack the availability of a reliable and accessible Spatial Data Infrastructure and spatial data in general (Dumedah et al., 2022). The absence of an SDI and national framework for collecting, storing, and publishing which makes it difficult and frustrating when data needs arise. It is in this regard that Volunteered Geographic Information offers alternatives and use-cases for national map productions.

The abbreviation FAIR is sometimes used to refer to data that meets the principles of Findability, Accessibility, Interoperability, and Reusability (Wilkinson et al., 2016). Findability refers to the ease of finding through metadata. Once data is found it should be accessible either via an open, free or universal protocol. Interoperability allows for the data to be integrated into other datasets and also applications used. Finally, reusable through well-documented metadata that can be replicated.

2.1.3 Open Data Licences

According to Alamoudi et al. (2020) Open Data Licences can help in regulating the legal conditions about how to use, distribute, and modify data. National and Regional Open Data are initiatives at the national or regional levels such as states or districts providing data to the public under one of the open data licences (Alamoudi et al., 2020). The Global Open Data Index lists national portals as compiled by Open Knowledge Foundation (2016). Generally, Open Data Licences can be divided into three main categories as shown in Figure 2.1, depending on the restrictions that they put on the users (Alamoudi et al., 2020).

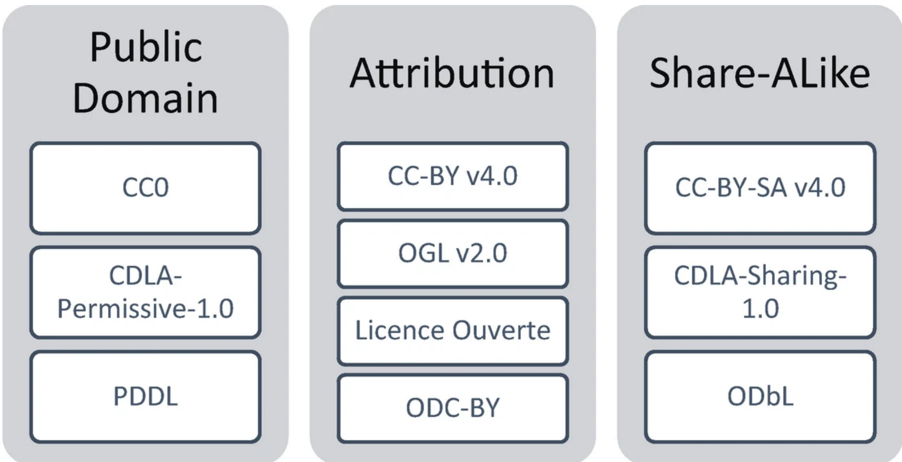


Figure 2.1: Open data licenses divided into different categories according to the restrictions that they place on the user (Alamoudi et al., 2020)

Examples of Open Data licenses

Notwithstanding the three categories indicated in Figure 2.1, data licenses can also be categorised according to their source, i.e. type of issuer (Alamoudi et al., 2020). Some examples of issuers of Open Data Licenses include Creative Commons Licences, Governmental Open Data Licences, and Open Data Commons Licences (Alamoudi et al., 2020; European Commission, 2022).

Creative Commons Licences known as CC licences are a group of licences used for open content. It gives rights to use, distribute, and adapt existing works. The most recent version 4.0 explicitly considers licensing data. CC licenses are further divided into public domain, attribution, and share alike. **Governmental Open Date Licences** are used mostly by governmental agencies for distributing data such as tax, energy, and others. Governments usually publish their data by using Open Government License (OGL), which is an example of a governmental license that respects open data terms. OGL is compatible with licenses such as Creative Commons Attribution licences. **Open Data Commons License** provides legal solutions for data. It was a project started in 2018 by the Open Knowledge Foundation. Open Data Commons Open Database License (ODbL) license is one example in this category. ODbL is used by the OpenStreetMap project (Wood, 2022). ODbL allows the user to use, share, and distribute the data, requires attribution to release all derived works under the same license

Shuttle Radar Topography Mission (SRTM)

The Shuttle Radar Topography Mission (SRTM) is an example of international/national open data which is available under public domain license. The SRTM collected elevation data over 80% of the earth's land area during an eleven day Space Shuttle mission. With a horizontal resolution of 3 arc sec, SRTM represents the best quality, freely available digital elevation models (DEMs) worldwide. Since the SRTM elevation data are unedited, they contain occasional voids, or gaps, where the terrain lay in the radar beam's shadow or in areas of extremely low radar backscatter, such as sea, dams, lakes, and virtually any water-covered surface (Nikolakopoulos et al., 2006).

Due to the occasional voids there exist void-filled SRTM data. The CGIAR-CSI version 4 provides the best global coverage using interpolation. It provides SRTM 90m Digital Elevation Data for the entire world (CGIAR-CSI, 2018). Viewfinder Panoramas by Jonathan de Ferranti is another void-filled SRTM data. It provides high quality at full SRTM resolution. The data is filled using local survey maps and photographs. The OpenTopoMap website which is based on OpenStreetMap uses Viewfinder Panoramas void-filled data (OpenTopoMap, 2022).

2.1.4 Volunteered Geographic Information

The term Volunteered Geographic was originally coined by Goodchild (2007) as a more general term for user-generated content. VGI has exploded as a result of GPS becoming available for civil use and Web 2.0's era of social media networks which brought about a large amount of user-generated content on the World Wide Web. Other factors include geotags, geo-referencing, GPS, graphics, and broadband communication (Goodchild, 2007). It can also be referred to as the wikification of GIS, which is driven primarily by the massive and voluntary collaboration among both amateurs and experts using Web 2.0 technology (Sui, 2008).

The concept of VGI is mostly attributed to mapping features using OpenStreetMap or the entire OpenStreetMap database as a whole, but there are many other VGI sources available (See et al., 2017). VGI has two components, that is volunteer and spatial information (See et al., 2017). Most researches focus on the spatial information part, but the volunteers are at the heart of VGI projects. Fritz et al. (2017) consider motivational factors for VGI as a critical part of the participation planning phase in any VGI system.

2.1.5 OpenStreetMap

The OpenStreetMap project was started in the year 2004, by Steve Coast in the United Kingdom. It is due to the high cost of accessing spatial data from the Ordnance Survey. OpenStreetMap has become a global community of local, regional, and national communities made of volunteers, and organisations both public and private. The primary goal of OpenStreetMap is to create a free and accessible map of the world through collaboration. OpenStreetMap data is licensed under the OpenData Commons License (Ramm et al., 2011); which allows the creation, modification, and adaptation of OSM data. It is required to provide reference in the form ©*OpenStreetMap Contributors* and on digital maps hyperlinked to *https://openstreetmap.org/copyright* (OpenStreetMap, 2022).

Data Representation

In order to meaningfully use OpenStreetMap data, a fundamental requirement is understanding its data model. This is important because any stakeholder could interact with OSM data in its raw form and manipulate it into formats that are more useful for intended purposes (Bennett, 2010). OSM data is organised using a simple conceptual data model which is a combination of geometric component and semantic component (Ramm et al., 2011).

Nodes

Nodes are the basic geometric representation of OSM data. They are defined by coordinates and they are points in space, which can be identified by an ID and tags. Nodes form the basis of other elements. They are used to represent point features such as benches, traffic signals, peaks, etc (OpenStreetMap Wiki, 2022b).

Ways

Ways are made up of two or more nodes. They can either be open, closed or called an area. Examples of open ways are polylines which do not start and end at the same point, they are used to represent roads, power lines, etc. As the name suggests, closed ways have a common starting and ending point forming a loop-like connection. Roundabouts are typical representations of closed ways (OpenStreetMap Wiki, 2022b).

Areas are filled in closed ways, mostly used to represent an enclosed feature such as buildings, parks, forests, landuse, and others. Specifying a closed-way tag on a closed way implies it is an area. Sometimes a dedicated tag *area=yes* might be required (OpenStreetMap Wiki, 2022b).

Relations

Relations is a group of an ordered list of both, nodes and ways, as well as other relations. The primary goal of relations is to describe either logical or geographical relations between the objects. Members of a relation can have an assigned role which describes the meaning of that element in the relation. Some examples include stops in a public transport route (relation) specifying start and end, stop positions, etc. Another example is a representation of administrative boundaries (OpenStreetMap Wiki, 2022b).

Tags

Tags are `key-value` pairs of strings added to OSM elements i.e. nodes, ways, and relations. Any element can have zero or more tags describing it. Each tag can be any pair of strings up to a maximum of 255 characters, with the only restriction that keys should be unique inside one element (OpenStreetMap Wiki, 2022b).

OpenStreetMap tagging system is a free-tagging system which is flexible. The OSM community have approved a set of tags that are in use. New tags can be created whenever the need arises and have gone through community voting. The OpenStreetMap Wiki is the website used for all documentation and information relating to the OSM project. Free-tagging in OSM has both its merits and drawbacks, but the merits outweigh the negatives.

2.1.6 Quality Assurance

The quality of OpenStreetMap data has been evaluated by Haklay and Weber (2008) and Zielstra and Zipf (2010). These researches have proven, especially in urban regions, that OSM can compete or even surpass data offered by commercial data providers or governmental authorities (Goetz et al., 2012). OpenStreetMap is mostly compared to authoritative data, but other studies have also compared OSM with itself based on intrinsic factors such as history (Madubedube et al., 2021). Authoritative data is data that has some trust or is generated by an institution mandated by law such as the National Mapping Agency of a country. But in the situation where there is no such reference data it is fairly not easy to tell.

In the OpenStreetMap community, there exist tools for quality assurance and data validation. Notwithstanding, the users can always commit their changes to the database by ignoring these issues. These tools create warnings based on validation rules which guide users to correct or update issues (OpenStreetMap Wiki, 2022e). *Osmose*[1] and *OSMCha*[2] are some examples of these tools. Applications for contributing to OpenStreetMap referred to as editors (OpenStreetMap Wiki, 2022a) also implement some level of validation rules and checks when a user tries to commit their changes to the database.

2.2 Map Production

2.2.1 Historical Background of Maps

Humans have long recognised the importance and value of maps in their lives. Indeed, the history of mapping can be traced to more than 5,000 years ago (Intergovernmental Committee on Surveying and Mapping, 2022). Although Anaximander is credited to have created the world's first map in the 6th Century BC, Ptolemy's invention of a method to flatten the world into a 2D representation through the application of geometry and mathematics, laid the foundation for modern map making (Aujac et al., 1987).

Before advances in papermaking and printing, maps were produced through engraving in reverse mostly on copper plates and printed on a hand press. Maps were created manually during the longest period of cartography using hand instruments like brushes and quills (Robinson et al., 1995).

The creation of methods that allowed for the mass manufacture of maps was the first significant step toward automated map production. Mechanical technology accelerated the production

[1]https://osmose.openstreetmap.fr
[2]https://osmcha.org/

of maps, making them more affordable and available to a wider range of customers (Robinson et al., 1995).

2.2.2 Current Workflows

Lithography, or printing from soft stone, largely took the place of engraving in the production of English commercial maps after about 1852. It was a quick, cheap process and had been used to print British army maps during the Peninsular War. Most of the commercial maps of the second half of the 19th century were lithographed and unattractive, though accurate enough (Lynam, 1944).

The advent of GIS has gradually made the map production processes become more partially automated in the digital era. Most map production sub-processes such as data extraction are now almost entirely automated. However, some technical and scientific issues, most notably data generalisation and label placement, remain manual that prevents the production processes from becoming fully automated (Longley et al., 2015).

2.2.3 Map Production Processes

The map production process involves a series of stages from defining the purpose of the map to the final output. Generally, maps are classified into two broad groups: reference maps such as topographic maps that convey general information and are produced by National Mapping Agencies; and thematic maps that convey specific geographic themes, such as population census statistics, soils, climate zones, etc. (Longley et al., 2015).

Since the processes of map production can always vary as described by Virrantaus et al. (2009), map production can also be likened to the factors such as scale and type of medium. According to Longley et al. (2015) map production from GIS can result in two outputs i.e. formal maps and map visualisations. These Formal maps follow established cartographic conventions. They are used as reference or communication products such as topographic maps or geological maps (Longley et al., 2015). Map visualisations are transitory maps or map-like visualisations. Map visualisations which are used to display, analyse, edit, and query geographic information are distinct from formal maps (Longley et al., 2015). An example of map visualisation is a display of database query of school points or routing lines from one location to another. Map visualisations can be interactive on digital media or printed (Longley et al., 2015). Additionally, general maps, thematic maps, and charts are other forms of maps as defined by Robinson et al. (1995).

According to Buttenfield and Hultgren (2005), Hardy et al. (2004), and Longley et al. (2015) modern map production processes are built around a central cartographic database as depicted in Figure 2.2. A Digital Landscape Model (DLM) is a database containing topographic features as captured or compiled (Buttenfield & Hultgren, 2005) or based on a data model that represents scale-free and real-world features (Hardy et al., 2004). The geometric form of the DLM contains explicit or implicit topological information.

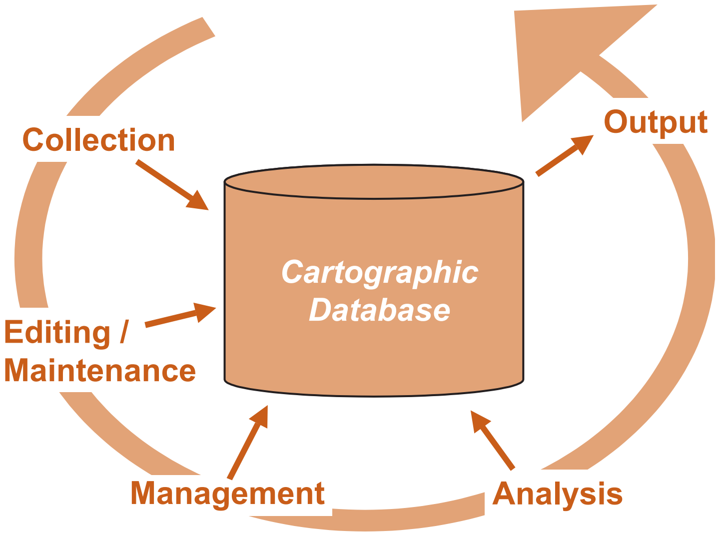


Figure 2.2: GIS processing transformations needed for map production (Longley et al., 2015)

These DLMs are further transformed through abstraction, generalisation, and data modelling to produce Digital Cartographic Models (DCMs) (Buttenfield & Hultgren, 2005; Hardy et al., 2004; Robinson et al., 1995). DLMs differ from DCMs in several ways such as feature representation in addition to being scale dependent (Buttenfield & Hultgren, 2005).

2.2.4 Map Series

Map series are groups of maps that share common elements such as projection, general layout, and symbology (Longley et al., 2015). Map series occur as a result of an area covered by a map needs to be spread across multiple sheets due to the scale of the map. The separate maps sheets, notwithstanding have the required properties to be used independently. Despite the development of novel digital maps and location-based services using mobile and portable devices, paper maps remain widespread use because of their transportability, reliability, ease of use, and the straightforward application of printing technology that they entail. They also convey direct information and messages as they are purposely produced to serve a unique purpose and serve it very well, such examples include city maps and transport maps (Longley et al., 2015). In order to produce multi-scale map series, dedicated DCMs are produced from existing DLMs as described in Section 2.2.3. DCMs can be further transformed into finer DCMs based on scale and geometric requirements (Longley et al., 2015).

2.2.5 Workflow concepts

Workflow refers to the complete or partial digital automation of a process. Workflows can be temporal and logical sequences of functions that are necessary to perform operations on economically relevant objects - with automated transitions, namely processes, whose control logic lies within the control of an information system (Rosemann & Zur Muehlen, 1998). Every workflow is based on a process model that has been enhanced with additional object types and attributes that allow its automation, this can be referred to as the workflow model (Rosemann & Zur Muehlen, 1998). Workflow models are usually described using directed graphs whose knots represent (elementary or composite) functions (Rosemann & Zur Muehlen, 1998). Workflow management systems have an external view of the functions executed and are not concerned with the execution of the internal functions themselves. (Rosemann & Zur Muehlen, 1998).

Workflow is frequently associated with business process re-engineering, which is concerned with evaluating, modelling, defining, and implementing an organization's critical business processes (Hollingsworth & Hampshire, 1995). Workflow management enables the outsourcing of control flow from application systems. The evaluation of a workflow can either be qualitative and quantitative (Rosemann & Zur Muehlen, 1998).

Workflow concepts have is been used within NMAs for the automation of map production such as in the case of Netherlands (Stoter et al., 2014) and the TopoPlus process by the Bundesamt für Kartographie und Geodäsie (BKG) (Kunz & Bobrich, 2019) amongst others.

2.2.6 Software for Map Production

Generally, there are two broad types of software, that is free/libre and open source software and proprietary software. Much like data, the type of software is dependent on the type of license. A software license is a legal instrument that governs how the software is to be used and distributed; software licenses can either be proprietary or open (Löwe et al., 2022).

Like any other original creative work, software is generally copyright protected, unless it is specifically made available in the public domain, where no exclusive intellectual property rights apply. If the software is to be used by a person other than the copyright holder, a license must be granted to the user. Figure 2.3 shows categories of free and non-free software according to the Free Software Foundation.

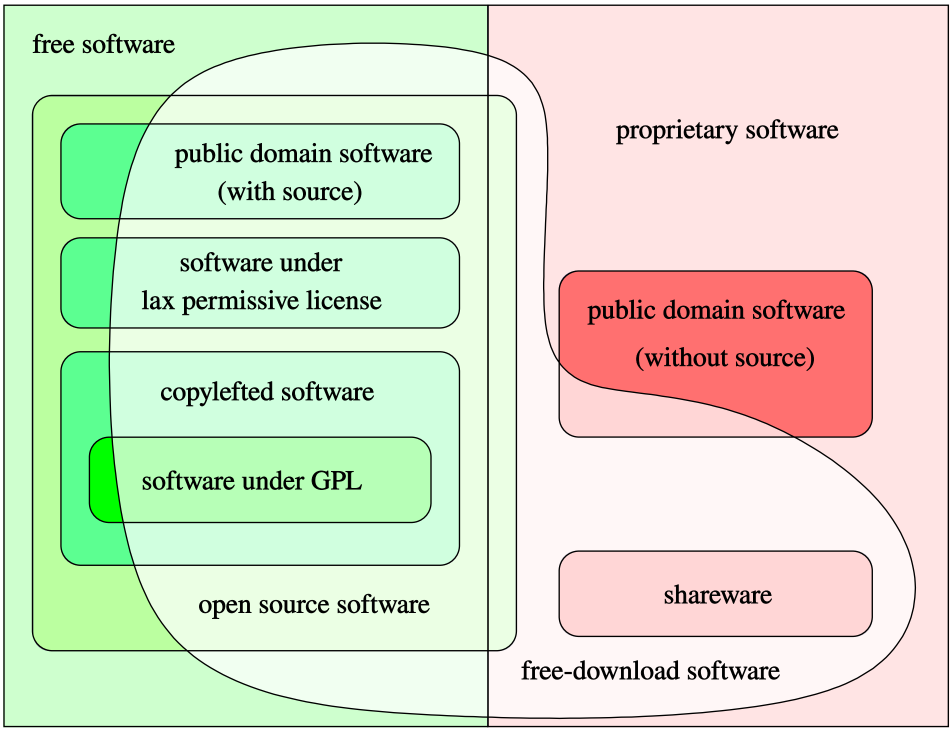


Figure 2.3: Diagram of free and non-free software, as defined by the Free Software Foundation based on Chao-Kuei's Diagram. **Left**: free software, **right**: proprietary software, **encircled**: gratis software (Wikimedia Commons, 2022)

Free Software

According to the GNU project "Free software" means software that respects users' freedom and community. It states that the users have the freedom to run, copy, distribute, study, change, and improve the software (Free Software Foundation, 2022).

Free software is mostly associated with price. However the free is described as a matter of liberty. To understand the concept, the "free" is usually likened "free speech", not as in "free beer". Sometimes it is also referred to as "libre software", inspired by the French or Spanish word for "free" as in freedom, to show the software is gratis. Notwithstanding, free software can be paid for or sold.

The words free software, open source, free/libre and open source software are used interchangeably to refer to one another. The GNU project disagrees the term open source is a loose form of free software which values advantage instead of the levels of freedom (Stallman, 2012). In this thesis, all these words refer to free software and its values. To conclude, the Free Software Foundation defines four kinds of freedom (Free Software Foundation Europe, 2022) that is associated with free software as listed below:

- **Freedom 0**: to run the program without limitations.
- **Freedom 1**: to study the program and its source code; how it works and change it for new purposes.
- **Freedom 2**: to share it with others.
- **Freedom 3**: to distribute copies of modified and improved versions publicly.

Propritary Software

Software is said to be proprietary depending on its license. Usually, proprietary software does not have its source code available which is a pre-requisite for *Freedom 1* of free software; the reason it is also called closed-source software. Proprietary software is always the property of the creator and is only available to their users under certain conditions. To protect the intellectual property invested in developed software is a reason for the refusal to provide the underlying source code. Several commercial off-the-shelf software, mainly produced for sale, are proprietary software. Shareware is an example of proprietary software that can be used free of charge but with limitations such as limited functionalities, and the expiry of trial periods. This explains the fact that software provided free of charge does not mean that it can not be proprietary (Löwe et al., 2022).

2.2.7 The Role of Software in the Map Production Process

Cartography and map production is largely a manual process. Over the years due to the development of technology and the rise in on-demand services and solutions, software has become part of our daily lives.

Map production on one hand requires a lot of processes which computers and software are very good at. It is no doubt that most modern map production workflows are highly dependent on technology, involving novel software and tools that make their work much easier and reproducible such as in the case of NMAs (Kunz & Bobrich, 2019; Käuferle et al., 2015; Stoter et al., 2014).

Software Seletion

The process of selecting software in the map production is a multi-criteria decision. It is important to access available options of software for its intended purpose. Software has importance in many stages from data collection through visualisation and output of a map. The map-maker or the stakeholders involved have to decide if their software needs should be addressed by using a commercial off-the-shelf solutions, free and open-source software or build from scratch. Creating software from scratch is involving and requires both time and financial commitments (Steiniger & Hunter, 2013), even though the benefits of developing from scratch are also enormous allowing full functionality control of the components and needs (Longley et al., 2015).

Software Categorisation

In order to understand what exists, the categorisation of software in the specific domain is vital. It helps both developers and users and stakeholders to know what is required. The OSGeoLive project provides collections of fully-operational versions of popular free geospatial software as self-contained Lubuntu-based bootable Linux distribution for geospatial applications. These collections are also categorised into: desktop GIS, browser facing GIS, web services, data stores, navigation and maps, spatial tools, domain specific GIS, data, geospatial libraries, and geospatial standards (Emde, 2022). The collections are published as bootable ISO images to be used on a USB thumb drive or DVD, and a pre-made virtual machine with additional tools and data to be used in virtual machine applications such as VirtualBox, VMWare, or KVM.

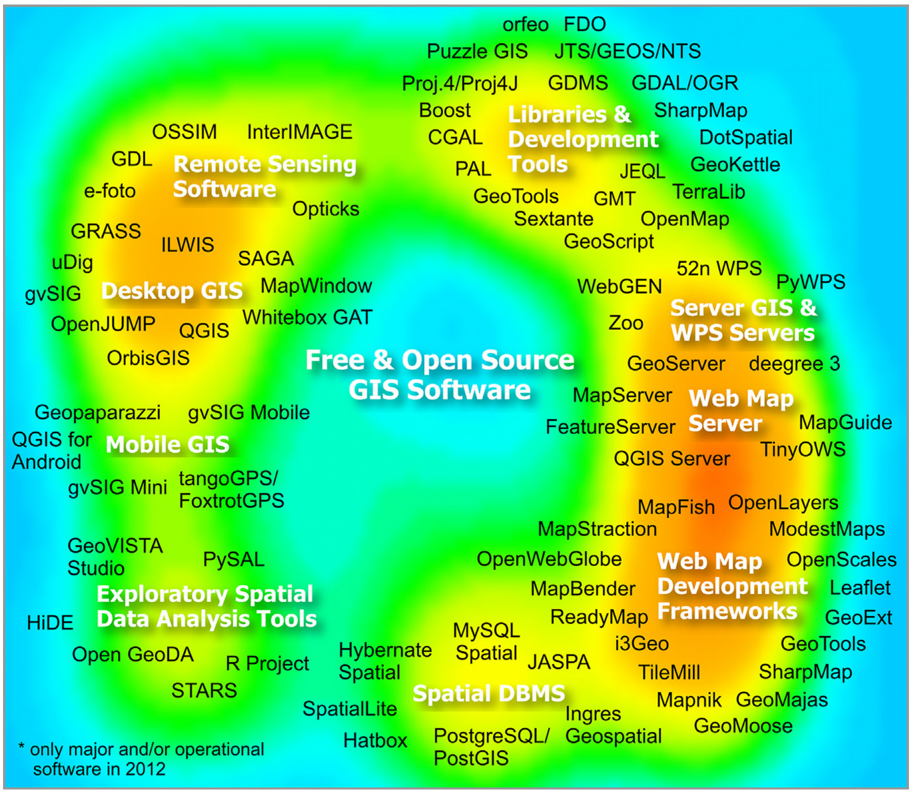


Figure 2.4: The free and open source geographic information software map of 2012 (Steiniger & Hunter, 2013)

Similarly, Steiniger and Hunter (2013) also created a map of available major Free and Open Source Software leading their respective and overlapping categories they fall in. This categorisation resulted in grouping FOSS into nine groups: Remote sensing software, Desktop GIS, Mobile GIS, Exploratory spatial data analysis tools, Libraries and development tools, Server GIS and WPS servers, Web map servers, Web map development frameworks, and Spatial DBMS. It should also be noted that categorisation does not always lead to software falling within one group as one software in most cases will perform multiple functions and hence fall within multiple groups as well. Figure 2.4 shows the result of categorisation and grouping by Steiniger and Hunter (2013).

2.2.8 Free and Open Source Software for Geospatial

FOSS4G is used synonymously to refer to both annual gatherings and regional conferences of the Open Source Geospatial community. The power of Open Source projects is the community, the Open Source Geospatial Foundation. Using Free and Open Source Software has become more accessible to geospatial workflows that would otherwise be inaccessible in terms of cost and availability (Markieta, 2012).

Open Source Geospatial Foundation (OSGeo)

The OSGeo is a not-for-profit organisation devoted to an open philosophy and participatory community-driven development to provide financial, organizational, and legal support to the broader Free and Open-Source Software geospatial community (Tzotzos, 2021). OSGeo projects offer freely available tools and technologies under an open source license. The OSGeo web portal promotes the work of teams and organizations worldwide, while volunteers are organized in over 20 local national or regional chapters and initiatives, reaching out to the GIS industry, education, and academia (Löwe et al., 2022). Some popular OSGeo projects are GRASS GIS, QGIS, PostGIS, GeoServer, MapServer GDAL/OGR amongst others.

2.2.9 Software Libraries

Software libraries are tools which form atomic but very vital and functional components of other software systems. They can either be used as a standalone application or in conjunction with others. Some well-known software libraries include the Geospatial Data Abstraction Library (GDAL/OGR), Proj4 and others. The GDAL/OGR library offers extensive capabilities for data exchange reading and writing to several vector and raster data formats. GDAL uses the Proj library for handling projections.

2.2.10 Desktop GIS

According to Longley et al. (2015), Desktop GIS is mapping software that is installed and run on a single personal computer. They are not accessed or controlled from a different computer through a server. Desktop GIS can be categorised into a wide range of options such simple viewers, desktop mapping and GIS software to high-end applications. Desktop GIS can display, query, modify, edit and analyse geospatial data (Longley et al., 2015). Most desktop GIS are Graphical User Interface (GUI) based applications.

2.2.11 Spatial Data Management Systems

Connolly and Begg (2005) define database management system (DBMS) as a "software system that enables users to define, create, maintain and control access to the database". DBMS can be classified according to the way they store and manipulate data. Three main types of DBMS are available to GIS users today: relational (RDBMS), object (ODBMS), and object-relational (ORDBMS) (Longley et al., 2015). The standard database query language adopted by virtually all mainstream databases is SQL (Structured or Standard Query Language: ISO Standard ISO/IEC 9075) (Longley et al., 2015)

Spatial extensions enables DBMS to support geographical data. These spatial extensions sometimes allow for the execution of spatial functions directly within the database system. PostGIS is an active and well-known spatial extension for the PostgreSQL database management system. PostGIS supports both vector and raster data including functions. Compared to storing data in file systems with single access, DBMS on the other provide multiple access and can store a larger amount of data.

With database triggers the integrity of data is maintained within the database. Database triggers are procedural codes that are automatically executed in response to certain events on particular tables or views in a database. For example, when a new record (representing a new worker) is added to the employee's table, new records should also be created in the tables of the taxes, vacations and salaries. Triggers can also be used to log historical data, for example to keep track of employees' previous salaries.

3 Methods

3.1 Introduction

This chapter describes the software selection process and methods for handling updates. It also presents requirements for the workflow implementation and implementation of an adjustable data model for handling updates.

3.2 Requirements for workflow

In order to achieve a workflow for the production of printed maps some requirements were defined. These requirements are based on the elements of the cartographic process. The cartographic process involves stages from data collection and preparation of an unmapped area to a finished map. It does not only end when a map is produced but also continues with the map users and decisions taken by them. Below are the requirements which defined and adopted for this research:

1. The workflow shall use primarily open data
2. The workflow must use free/libre and open source software
3. Elements of the workflow should be able to work independently and together
4. The workflow should result in the output of a general-purpose map with a scale of 1:25000
5. The workflow should be able to take into account derived changes modifications
6. The workflow should work independently and as whole

3.3 Selecting software for map production

The process of selecting software is challenging. It is a multi-criteria decision-making process that requires a multi-stakeholder engagement and a lot of resources too (Eldrandaly, 2007). In order to select software and tools to be used in this research, existing publications were investigated in addition to the known workflow documentation involving OpenStreetMap map data and Free and Open Source Software. Most of the workflows were targeted at producing tiled web maps such as the map displayed on the main OpenStreetMap website. Considering that there are many options and alternate software, the best-suited option for any required task needs to be carefully chosen. This is a challenging and critical stage as decisions from the selection is used in the final workflow implementation.

3.3.1 Requirements for Software Selection

First of all, before selecting a software for our case, extensive work has been done by studying previous publications on the methods for selecting software in the geospatial domain.

Steiniger and Hunter (2013) proposed a five-step selection process for software, this involves firstly developing software use cases for own context (or "user stories"). Secondly, establishing a set of evaluation criteria based on the use cases. Thirdly, performing the software evaluation with respect to the established criteria. Finally, developing a weighting criteria according to the application context. It should however be noted that weighting is intended to be flexible in allowing for different contexts. Figure 3.2 summarizes the proposed five-steps.

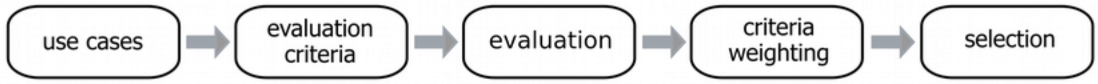


Figure 3.1: Stages for software selection by Steiniger and Hunter (2013)

The above-discussed method is based on a combination of two methods: (a) one used by the Collaborative technological watch group Qualification and Selection of Open Source Software (QSOS) (Team, Qsos core, 2018), which includes the following steps:

1. **definition** of evaluation templates structured as trees of evaluation criteria
2. **evaluation** by assessing competing against criteria
3. **qualification** of evaluation by organising criteria into evaluation axes and defining weights.
4. **selection** of appropriate FOSS by scoring using filtering systems in previous step

and, (b) the four-step process outlined by Sveen (2008), consisting of:

1. define usage scenarios and requirements,
2. gather candidate projects,
3. create evaluation sheet,
4. rank projects and select.

A three-phase software selection process was also proposed by Eldrandaly and Naguib (2013) which consists of justification phase, a screening phase, and an evaluation phase. The justification phase allows justifying the selected GIS software to be used, in the screening suited software are selected based on required and defined capabilities. The evaluation phase leads to ranks according to the order of suitability.

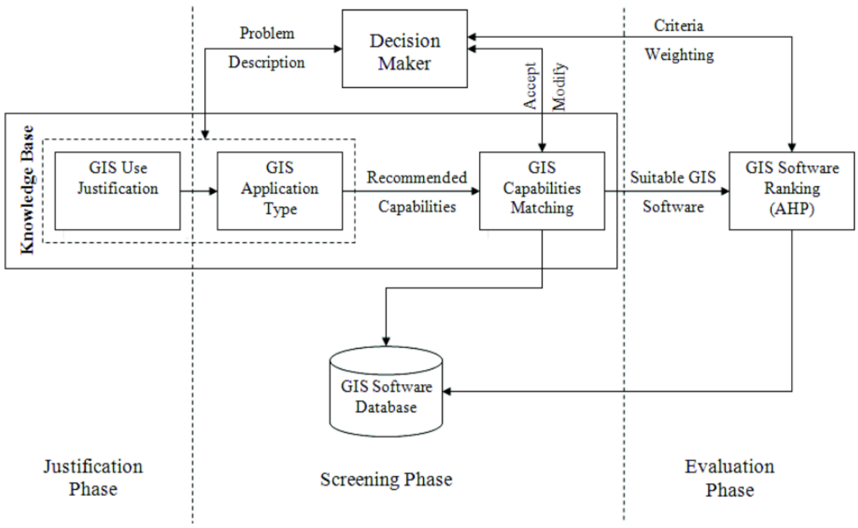


Figure 3.2: Three phase software selection process Eldrandaly and Naguib (2013)

Furthermore, according to Eldrandaly and Naguib (2013), cost, functionality, usability, reliability, and vendor are five important criteria to consider during the evaluation phase of GIS software selection.

This thesis adopts the hybrid five-steps selection process (Section 3.3.1) proposed by Steiniger and Hunter (2013), and shortlisted software and tools for each stage of the map production process. The shortlisting was done by searching existing publications relating to the domain of each stage in the workflow. In addition, other resources such as the OpenStreetMap Wiki, GitHub and GitLab were utilised. A set of criteria were also established based on the needs of each stage in the workflow and how the criteria affect it most. Finally, evaluation is based on ranking scores which is a result of the method of weighing used.

3.3.2 Software overview

The identification of appropriate software alternatives to be used in the various stages composed for the implementation of the map production workflow was based on the cartographic process. Figure 3.3 shows a quick overview of software shortlisted for each stage. For the Data Preparation stage, three alternatives were identified, namely *GDAL/OGR*, *osmium* and *osmconvert*. Table 3.1 shows the shortlisted alternatives and short descriptions.

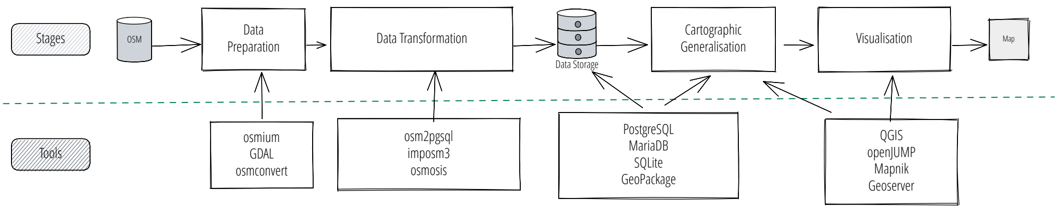


Figure 3.3: Overview of software/tools for stages of the map production process

Table 3.1: Shortlist of data preparation tools

Name	*Description*
GDAL	a translator library for raster and vector geospatial data formats that is released under an MIT style Open Source License by the Open Source Geospatial Foundation (Rouault et al., 2022)
osmium	a fast and flexible C++ toolkit and framework for working with OSM data (OpenStreetMap Wiki, 2022c)
osmconvert	a tool for converting & processing OpenStreetMap files (OpenStreetMap Wiki, 2019)

In the Data Transformation stage, three tools were identified, i.e. *imposm*, *osm2pgsql* and *osmosis*. Table 3.2 shows the identified alternatives and short description for the Data Transformation stage.

Table 3.2: Short list of data transformation tools

Name	*Description*
imposm	an importer for OpenStreetMap data. It reads XML and PBF files and can import the data into PostgreSQL/PostGIS databases (OpenStreetMap Wiki, 2021a)
osm2pgsql	a software to import OpenStreetMap data into a PostgreSQL/PostGIS database (OpenStreetMap Wiki, 2021b)
osmosis	a command line Java application for processing OSM data (OpenStreetMap Wiki, 2022d)

For the Data Storage stage, four alternatives were identified, namely *GeoPackage*, *MariaDB*, *PostgreSQL*, and *SQLite*.

Table 3.3 shows the shortlisted alternatives and short descriptions for Data Storage stage.

Table 3.3: Short list of database management systems

Name	*Description*
GeoPackage	an open, non-proprietary, platform-independent and standards-based data format for geographic information system implemented as a SQLite database container (Wikipedia contributors, 2022b)
MariaDB	a community-developed, commercially supported fork of the MySQL relational database management system (RDBMS), intended to remain free and open-source software under the GNU General Public License (Wikipedia contributors, 2022e)
PostgreSQL	a free and open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance (Wikipedia contributors, 2022f)
SQLite	a database engine written in the C programming language (Wikipedia contributors, 2022g)

For the Visualisation stage, four alternatives were identified, namely *GeoServer*, *Mapnik*, *openJUMP* and *QGIS*. Table 3.4 shows the shortlisted alternatives and short descriptions for visualisation stage.

Table 3.4: Short list of visualisation tools

Name	*Description*
GeoServer	an open-source server written in Java that allows users to share, process and edit geospatial data (Wikipedia contributors, 2022c)
Mapnik	open-source mapping toolkit for desktop and server based map rendering (Wikipedia contributors, 2022d)
openJUMP	a desktop GIS software, written in Java
QGIS	QGIS is a full-featured, user-friendly, free-and-open-source (FOSS) geographical information system (GIS) that runs on Unix platforms, Windows, and MacOS (Contributors, 2022)

3.3.3 Methods for weighting

From the identified requirements in Section 3.3.1 a method needs to be used for weighting. Well known Multiple-Criteria Decision Analysis (MCDA) methods such as Weighted Sum Model and Analytic Hierarchy Process (AHP) were identified to have been used for software selection. Table 3.5 shows a selection of papers and method used.

Table 3.5: MCDA methods for software selection use cases and citations

Method	*Citation*
Weighted sum model	Chen et al. (2010)
Analytic hierarchy process	Eldrandaly (2007), Eldrandaly and Naguib (2013)

3.3.4 Software weighting and selection

The weighted product model (WPM) is one of several MCDA methods for evaluation. Table 3.5 shows MCDA methods used for software selection from existing literature, WPM was not found to have been used. The weighted sum model was the first choice adopted by this thesis but was later changed to WPM due to setbacks in normalising WSM. Considering all these methods have been applied in fields of varying MCDA situations, this thesis instead, decided to use WPM which is not prone to drawbacks in WSM. AHP on the other hand would have been best suited but due to its intensive implementation, fewer alternatives to chose from and time constraints, it was not the best suited.

The weighted product model (or WPM) is very similar to the WSM. The main difference is that instead of addition in the model there is multiplication. Each alternative is compared with the others by multiplying the number of ratios, one for each criterion. Each ratio is raised to the power equivalent to the relative weight of the corresponding criterion.

Assuming that a given MCDA problem is defined on *m* alternatives and *n* decision criteria. All decision criteria *n* are considered benefits, that is, the higher the values are, the better it is. Next suppose that w_j denotes the relative weight of the importance of the criterion C_j and a_{ij} is the performance value of alternative A_i when it is evaluated in terms of criterion C_j. Then, comparing two alternatives A_K and A_L (where $m \geq K, L \geq 1$) then, the following product has to be calculated (Triantaphyllou et al., 1998):

$$P(A_K/A_L) = \prod_{j=1}^{n} (a_{Kj}/a_{Lj})^{w_j}, \text{ for } K, L = 1, 2, 3, \ldots, m. \tag{3.1}$$

Using a simple decision matrix in Table 3.6 which is based on three alternatives represented by A_1, A_2, and A_3 and described in terms of four criteria C_1, C_2, C_3, and C_4 (Triantaphyllou et al., 1998). The relative weight for each criterion denoted is 0.20, 0.15, 0.40 and 0.25 respectively. Values for each alternative per criteria are recorded in each cell. For the score of the first alternative A_1 in terms of the first criteria C_1 is 25. For the WPM the scores of alternatives can be expressed in different units of measure, which is not the case for WSM (Triantaphyllou et al., 1998). It is in this case that WPM is also sometimes called **dimensionless analysis**. The best alternative after WPM is better than or at least equal to all the other alternatives (Triantaphyllou et al., 1998).

Table 3.6: Sample decision matrix (Triantaphyllou et al., 1998)

	C1	C2	C3	C4
Alts./W	0.20	0.15	0.40	0.25
A1	25	20	15	30
A2	10	30	20	30
A3	30	10	30	10

Applying WPM to Table 3.6, the following values are derived:

$$P(A_1/A_2) = (25/10)^{0.20} \times (20/30)^{0.15} \times (15/20)^{0.40} \times (30/30)^{0.25} = 1.007 > 1. \quad (3.2)$$

Similarly, we also get:

$$\begin{aligned} P(A_1/A_3) &= 1.067 > 1, \text{ and} \\ P(A_2/A_3) &= 1.059 > 1. \end{aligned} \quad (3.3)$$

The interpretation is that the best alternative is A_1, since it is superior to all the other alternatives. In addition, the following ranking of all three alternatives is as follows: A_1 > A_2 > A_3 (the symbol ">" stands for "better than").

An alternative approach to the WPM method is for the decision maker to use only products without the previous ratios (Bridgman, 1922; Miller et al., 1963). That is, to use the following variant of the main formula given in Equation 3.1:

$$P(A_K) = \prod_{j=1}^{n} (a_{Kj})^{w_j}, \text{ for } K = 1, 2, 3, \ldots, m. \quad (3.4)$$

In Equation 3.4 $P(A_K)$ represents the total performance value (i.e., not a relative one) of alternative A_K when all the criteria are considered simultaneously under the WPM model. This will yield the same ranking as using Equation 3.2.

Criteria and Weight Definition

Weights were assigned to criteria defined based on their importance and contribution to each of the stages of the workflow. A performance scale of 0 to 1 is used for pair-wise comparison and assigning scores to alternatives under each criterion. The lowest value of 0 represents *low*, mid value 0.5 representing *mid* and maximum value of 1 presenting *high*. Values between the low–mid–high scale such as 0.1, 0.6, and others can also be assigned. Criterion with similar weight values are of the same importance.

For the Data Preparation stage, six criteria were used for weighing as shown in Table 3.7. The most important criterion here is Customization options (C2). Cross-platform (C1) and Supported data formats (C4) had the same importance. The remaining criteria were also considered to be of similar importance.

Table 3.7: Criterion and weights for data preparation stage

Selection Criteria	*Weight*	*Remarks*
Cross-platform (C1)	0.2	if the software can be used on major operating systems
Customization options (C2)	0.3	if there are further options for customising
Scalability (C3)	0.1	if the software can handle a growing amount of work by adding resources to the system
Supported data formats (C4)	0.2	types of data formats supported (read/write)
Actively supported (C5)	0.1	if the software is under active development
Documentation (C6)	0.1	if there exists up-to-date guides and documentations

For the Data Transformation stage, seven criteria were used for weighing as shown in Table 3.8. The most important criteria were Projections support (C4) and Customization options (C5).

Table 3.8: Criterion and weights for data transformation stage

Selection Criteria	*Weight*	*Remarks*
Cross-platform (C1)	0.15	if the software can be used on major operating systems
Scalability (C2)	0.1	if the software can handle a growing amount of work by adding resources to the system
Processing time (C3)	0.1	the amount of time taken to process the same amount of data
Projections support (C4)	0.2	amount of projections supported for data
Customization options (C5)	0.2	if there are further options for customising
Usability (C6)	0.1	ease of use
Generalisation options (C7)	0.15	if some data generalisation options are immediately supported

For the Data Storage stage, eight criteria were used for weighing as shown in Table 3.9. The most important criteria were Scalability (C2), Community (C3), and Projections support (C4).

Table 3.9: Criterion and weights for data transformation stage

Selection Criteria	*Weight*	*Remarks*
Cross-platform (C1)	0.1	if the software can be used on major operating systems
Scalability (C2)	0.2	able to handle a growing amount of work by adding resources to the system
Community (C3)	0.2	time taken to complete processing for the same area
Adoption (C4)	0.025	how widely it is used; NMAs, etc
Extensibility (C5)	0.1	provision for future growth
Usability (C6)	0.1	ease of use
Useful Functions (C7)	0.075	both spatial functions and other functions
Spatial data support (C8)	0.2	support for spatial data

For the Visualisation stage, nine criteria were used for weighing as shown in Table 3.10. The most important criterion is Layout and design (C8). All other criteria were considered to be of the same importance resulting in the same weight value of 0.1.

Table 3.10: Criterion and weights for visualisation stage

Selection Criteria	*Weight*	*Remarks*
Cross-platform (C1)	0.1	if the software can be used on major operating systems
Scalability (C2)	0.1	able to handle a growing amount of work by adding resources to the system
Community (C3)	0.1	time taken to complete processing for the same area
Adoption (C4)	0.1	support for projections aside Web Mercator
Extensibility (C5)	0.1	provision for future growth
Usability (C6)	0.1	ease of use
Useful Functions (C7)	0.1	other useful functions for geo-processing
Layout and design (C8)	0.2	layout and map design functions
Supported formats (C9)	0.1	supported file/data formats

3.4 Designing an updatable and adjustable data model

At the centre of any GIS creation is a data model, which defines the scope and capabilities of its management and operation (Longley et al., 2015). Since features in the real world are always changing data about them which is stored in the databases also need to be updated or modified according to the needs of the user. The term for Matching and conflation of geographic datasets have a long history in the field of geographic information science.

3.4.1 Data Model

Data Model in the context of Geographic Information Systems is a mathematical and structure for representing phenomena over the Earth. Data models generally represent phenomena about geographic data which may include location information, attributes, change over time, and identity. Vector data model can be represented as points, lines, and polygons whilst raster data represent geographic data as cell matrices that store numeric values (Wikipedia contributors, 2022a).

3.4.2 Data Conflation

Conflation has been researched and defined by several authors with most definitions agreeing to the traditional definition of *data fusion* in computer science and remote sensing fields (Ruiz et al., 2011). Conflation is defined as the process that attempts to replace two or more versions of the same information with a single object that is more accurate with eliminated redundancies and reconciled conflicts, often with an error (Longley et al., 2015; Adams et al., 2010). According to McKenzie et al. (2013), two broad areas of research related to this subject have emerged. One focuses on the geometric or geographic properties of the data and the other is related to the descriptive attributes. In order to update a reference road map by using the OSM data, the two datasets must be first approximately matched. The most popular method is map conflation technology, which consists of geometric conflation, topological conflation, and semantic conflation (Ruiz et al., 2011).

Several approaches have also been developed for data conflation such as progressive buffering for updating official reference road using OpenStreetMap data (Liu et al., 2015). Scheffler et al. (2012) developed an algorithm for matching Points of Interest (POIs) from Qype and Facebook to OpenStreetMap data through an algorithm that uses different similarity measures taking the geographic distance of POIs into account as well as the string similarity of selected metadata fields.

3.4.3 Concept of Similarity

In statistics and related fields, a similarity measure or similarity function or similarity metric is a real-valued function that quantifies the similarity between two objects. The concept of measure of similarity is not only useful in pattern recognition, but also in other fields such as psychology, artificial intelligence, and information-retrieval systems amongst others (Samal et al., 2004). Some known measures of similarity from the literature are: Tversky measure, String Similarity, and Shape Similarity.

Tversky measure is based on a set of theoretical approaches in which similarity between two objects is measured by a function of three arguments: (a) attributes that are common to two objects, (b) attributes that belong to the first but not the second object, and (c) the attributes that belong to the second object but not to the first (Samal et al., 2004). **String Similarity** also known as string metric is a metric that measures the distance ("inverse similarity") between two text strings for approximate string matching or comparison and in fuzzy string searching (Wikipedia contributors, 2022h). **Shape Similarity** deals with comparison of transformed canonical shapes (Samal et al., 2004). An example of shape similarity is used by Goodchild and Hunter (1997) for simple linear (coastlines) feature conflation.

3.4.4 Requirements for Data model

After needs assessment for the creation of a data model, the following requirements for were defined:

1. The data model should be adjustable; an adjustable data model can be changed anytime and is flexible enough to accommodate future changes in original data.
2. The data model should be able to handle point, line, and polygon features; the basic types used to represent features must be supported in the data model.
3. The data model should account for local modified data during imports; when features are added, modified or deleted, they must be accounted for the next time fresh data is imported.
4. It should adopt a measure of similarity for conflation; a measure of similarity is a constraint for identifying similar features.
5. The data model should also handle semantic data; semantic data such as features attributes should also be considered in the data model.

3.4.5 Approaches

Several approaches exist for developing a data model and handling updates. In other words, data conflation and data fusion can be done through several approaches. Some known approaches include:

- **History** using versioning of objects added, changed or deleted.
- **Isolation** allows saving own modifications independent of working layers. An example is a path for labelling large objects such as rivers and national parks which are less likely to change.
- **On-Demand** is an agile way where a dataset is created and features are updated on-demand when necessary.

4 Implementation

This chapter is a step-by-step description of the implementation of the workflow for map production and implementation of an adjustable data model for handling updates.

4.1 Workflow for Map Production

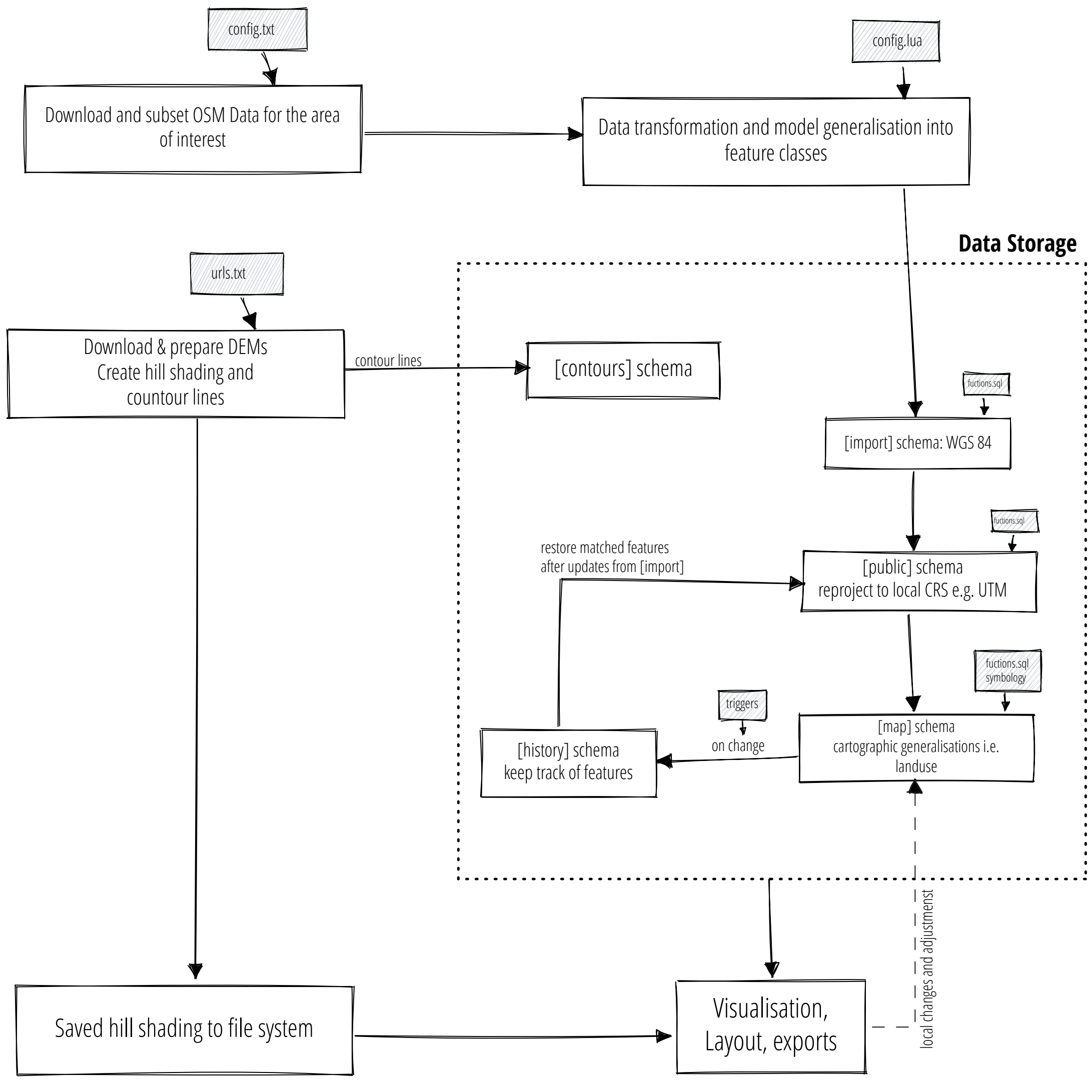


Figure 4.1: The implemented workflow process modes

The workflow for map production developed in thesis builds on the cartographic process and its stages identified in Section 2.2.3. Figure 4.1 illustrates the entire flow lines of the implemented workflow.

4.2 Stages of Map Production Process

At the beginning of the development of the map production workflow, a generalised sketch Figure 4.2, which is based on elements of the cartographic process, geographic data processing, and representation workflows, was used as guide to implementing the workflow. The process illustrates how data from one or several sources are combined, processed and transformed into a finished output, in the case of this research a printed map.

The Cartographic Process entails various steps for transforming an unmapped area into a map. According to the publication by the Open Geospatial consortium, there is no one way for defining the process of map production and it is dependent on the needs for the type map to be produced.

Based on the the stages of Figure 4.2, the development of a map production process was created.

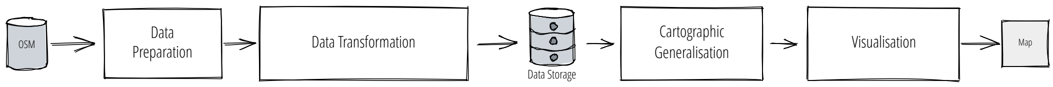


Figure 4.2: Stages of the Cartographic process.

4.2.1 Data Collection

Data from OpenStreetMap is used as the primary for this thesis. OpenStreetMap is both a community and database which anyone can freely contribute to and use, and it is continuously updated and monitored by an online community of contributors Haklay and Weber (2008). OSM data is changing every second, thanks to millions of contributors around.

Getting OpenStreetMap Data

There are many options for getting OpenStreetMap data especially for one-time use purposes. For example, in the QGIS software, QuickOSM, and OSM Downloader are the most popular plugins for directly downloading OpenStreetMap data for a specific area of interest. These plugins however can only retrieve specified data for smaller areas. Table 4.1 below lists popular services for getting OpenStreetMap data, formats, update frequency, and coverage.

Table 4.1: Some OpenStreetMap data services

Service	*Formats*	*Frequency*	*Coverage*
BBBike	Several notable formats	Weekly	Worldwide
Geofabrik	SHPs, PBFs	Daily	Worldwide
OpenStreetMap France	PBFs	Daily	Worldwide
Overpass Turbo	GeoJSON, PBFs, KML, GPX	Regular	Small areas

Notwithstanding, the datasets from most download servers excluding Overpass Turbo which tends to include all available OSM tags is not always modelled according to the needs of the data user. As a result, an extensive preprocessing and creation of a data model tailored to a user's specific need is required. This thesis uses *.pbf* files retrieved from Geofabrik download server on 2 July 2022 but continues to be updated in further preprocessing stages of the workflow implementation.

4.2.2 Automation of Tasks

To ensure the reproducibility and automation of the entire workflow and tasks within it, instructions were scripted using Bash. Bash, also known as **B**ourne **A**gain **Sh**ell is a type of shell interpreter and command language for Unix-like Operating systems. A shell interpreter takes commands in plain text format and calls Operating System services to perform a function. The shell scripts and development of the entire workflow process was carried out on a Lenovo Legion personal computer with *AMD Ryzen 7 4800H (16) @ 2.900GHz* CPU, *64GB* of RAM, and running *Arch Linux* Operating System.

4.2.3 Data Preparation

Data for map production and other use cases is not always readily available in the schema or form to be used. The purpose of data preparation is the foundation for having a working schema for data visualisation.

After the evaluation of alternatives in the Data Preparation stage **GDAL/OGR** was the most suited tool according to defined criteria in Table 3.7. The performance score and ranking at this stage using the Weighted Product Model (WPM) is shown in Table 4.2.

Table 4.2: Evaluation based on weighting and ranking for data preparation stage alternatives

Alts. / Criteria	*C1*	*C2*	*C3*	*C4*	*C5*	*C6*	*WPM*	*Rank*
GDAL/OGR	1.0	1.0	1.0	1.0	1.0	1.0	1.000000	1
osmium	1.0	1.0	0.8	0.5	0.7	1.0	0.821510	2
osmconvert	1.0	0.7	0.5	0.5	0.5	1.0	0.680953	3

Selection and Sub-setting

At the earliest stage of data preparation the needed OSM data is selected and filtered before further processing. This allows for faster processing as we only have to handle what we need, reducing processing time and other resources.

The initial study area for this research was Georgia, unfortunately, there are no regional extracts for Georgia on the Geofabrik download server. It took more time for downloading and importing Asia. Sub-setting of data into smaller units is therefore important and less expensive. Table 4.3 shows processing time for different regions.

Table 4.3: Regions and processing time

Region	*Processing Times*
Africa	38 minutes
Asia	1 hour 2 minutes
Georgia	26 minutes
Ghana	19 seconds

Below in Table 4.4 is a comparison for using *.osm.pbf* file for Ghana without filtering and with filtering, meaning only features that are needed for the initial import process were filtered from the original *.osm.pbf*.

Table 4.4: Storage capacities for each region per feature classes

Features/Region	*Africa*	*Asia*	*Georgia*	*Ghana*
Buildings	20GB	29GB	296MB	256MB
Landuse	1.8GB	2.4GB	37MB	8.8MB
Places	80MB	218MB	1.4MB	1.2MB
Traffic	7.2GB	16GB	152MB	121MB
Peaks	8.4MB	22MB	1.4MB	1.2MB

As part of this thesis, filtering of data did not make much difference as it just created a subset of data which is also selected during Data Transformation (Section 4.2.3). Filtering is much more useful when a selection or data is needed for a specific area.

Hill shading and Contour Lines

Using SRTM data from CGIAR[1], a list of the Uniform Reform Resource Locators (URL) of files within the area of interest were added in a text file with each line representing one file, this allowed for automation of downloads. The downloaded files were then merged and reprojected into a local projection. The contour lines were then imported into the *contours* schema of the database. All processes were carried out using the *GDAL/OGR* library.

Data Transformation

At the data transformation stage of the workflow, model generalisation was carried out. Model generalisation is performed to create feature classes to be used according to the defined data transformation model (Appendix A). Based on the criteria in Table 3.7 and performance scale of 0 to 1.0 as described in Section 3.3.4 scores were assigned to alternatives per criteria.

[1]https://srtm.csi.cgiar.org/

Table 4.5: Evaluation based on weighting and ranking for data transformation stage alternatives

Alts. / Criteria	*C1*	*C2*	*C3*	*C4*	*C5*	*C6*	*C7*	*WPM*	*Rank*
imposm	1.0	1.0	1.0	0.2	0.8	1.0	0.7	0.657035	2
osm2pgsql	1.0	1.0	1.0	1.0	1.0	0.6	0.5	0.856368	1
osmosis	1.0	1.0	0.7	0.1	0.5	0.6	0.1	0.356549	3

Table 4.7 shows the scores and ranking of data transformation stage alternatives. *osm2pgsql* was the most suitable for the data transformation and import. All data is imported into PostgreSQL database schema named *import* as described in Table 4.8. Table 4.6 shows list of features classes used in this thesis.

Table 4.6: Feature classes and descriptions

Name	*Tag(s)*	*Remarks*
buildings	`building=*`	All closed ways and all relations of type multi-polygon with *building* key
traffic	`highway=*`	All ways with *highway* key
places	`place=*`	All nodes with *place* key
peaks	`natural=peak OR volcano`	All nodes with *natural* key and values peaks or volcano
residential	`landuse=residential`	All closed ways and all relations of type multi-polygon with *landuse* key and values residential
landuse	`landuse=*`, except *residential*	All other closed ways and all relations of type multipolygon with *landuse* key
natural	`natural=*`, except *peaks*	All other closed ways and all relations of type multipolygon with *natural* key

Data in the *import* schema is in the WGS 84 projection. The geometries and attributes are the same as their original form as stored in the OpenStreetMap database at the time of data download. *osm2pgsql* works by taking required parameters such as database connection details and data transformation model.

Keeping imported data up-to-date

There are several ways to keep local OpenStreetMap data up-to-date. Generally, it comes down to two broad approaches:

1. A complete re-import of regional or sub-set extract
2. Application of regularly update files (minutely/hourly/daily) to an existing database using tools such as *osmosis osm2pgsql*, *imposm*

As part of this thesis the local OpenStreetMap data in *import* is kept up-to-date with what is in OpenStreetMap using *regional*[2] scripts. *regional* uses *osm2pgsql* in addition to two other tools *osmupdate* and *osmconvert* to update data *import* anytime with focus on using minimal system resources.

4.2.4 Data Storage

At the heart of modern map production workflow is a database management system. PostgreSQL was the main database management system used. PostgreSQL has support for geographic data and functions through the PostGIS extension.

Based on the criteria in Table 3.9 and performance scale of 0 to 1.0 as described in Section 3.3.4, performance scores were assigned to alternatives per criteria. Table 4.7 shows the scores and ranking at the data transformation stage alternatives in which **PostgreSQL**.

Table 4.7: Criterion and weights for data transformation stage

Alts. / Criteria	*C1*	*C2*	*C3*	*C4*	*C5*	*C6*	*C7*	*C8*	*WPM*	*Rank*
GeoPackage	1.0	0.5	0.7	0.8	0.5	1.0	0.7	0.8	0.700307	3
MariaDB	1.0	1.0	1.0	1.0	1.0	0.6	0.5	0.5	0.785293	2
PostgreSQL	1.0	1.0	1.0	1.0	1.0	0.6	1.0	1.0	0.950200	1
SQLite	1.0	0.5	0.7	0.6	0.3	0.6	0.7	0.8	0.627762	4

Defining Database Schemas

In order to separate layers into unique groups, five schemas were created and used in this research. Table 4.8 shows all schemas that were used in this thesis.

[2]https://github.com/Zverik/regional

Table 4.8: Database schemas adopted

Schema name	*Purpose*
import	all imported data
public	reprojected data from *import* schema
map	generalised and ready data used for map design
history	keeps track of changing data in *map* schema
countours	stores contour lines

Schemas represent named collections of database objects such as tables, views, and indexes. Schemas can also contain functions and operators. Different schemas can have the same object names without conflict. Schemas are analogous to folders on a computer system, with the difference that schemas cannot contain other schemas.

The *import* is all data from model generalisation at Data Transformation Stage. In the *public* schema all data from the *import* is reprojected in a local projections, i.e. Leigon/Ghana Metre Grid (EPSG:25000)[3] for Ghana and WGS 84 / UTM zone 38N (EPSG:32638)[4] for Georgia. In the *map* schema, layers are readied for map design and layout in the Visualisation stage. Some pre-processing such as generalisation of residential areas as described in Section 4.2.4 takes place between *public* and *map* schema transition. The *contours* schema as the name implies only stores contour lines. Hill shading as a raster was better accessed from the file system rather than within a database. Finally, the *history* schema which is further described in Section 4.3 is used to store Spatio-temporal modifications of data in the *map* schema which is later used for handling updates.

Generalisation of Landuses

Knowing that all data in OpenStreetMap is created by both experts and amateurs, contributions to OSM vary based on interests and objectives. Residential and wood/forest areas in OSM are tagged with *landuse=residential* and *landuse=woods* respectively. Processing of residential and wood/forest areas was a challenge. This is due to how they were mapped. They were mostly random with less conformity, leaving gaps and overlaps between neighbouring areas and roads as shown in Figure 4.3 for residential areas; each polygon represents a *landuse=residential*. This representation is unsuitable for map design and visualisation purposes.

[3]https://epsg.io/25000
[4]https://epsg.io/32638

Figure 4.3: Residential landuse as imported from OSM

In order to process and handle these landuse issues for visualisation, they were first buffered and clustered within a distance that was tested from several iterations and visualisations. The clustered polygons were simplified to smoothen the edges and holes within these polygons; below a 300 squared meters area value threshold were also removed. The removal of holes was achieved with a group of functions developed by Mapbox[5] named *Sieve*. These functions which work in PostgreSQL rely on the spatial extension PostGIS. After processing the resulting residential landuses looks better for visualisation and map design purposes. The Figures 4.4 and 4.5 show before and after results for residential and wood/forest areas respectively.

[5]https://github.com/mapbox/postgis-vt-util/blob/master/src/Sieve.sql

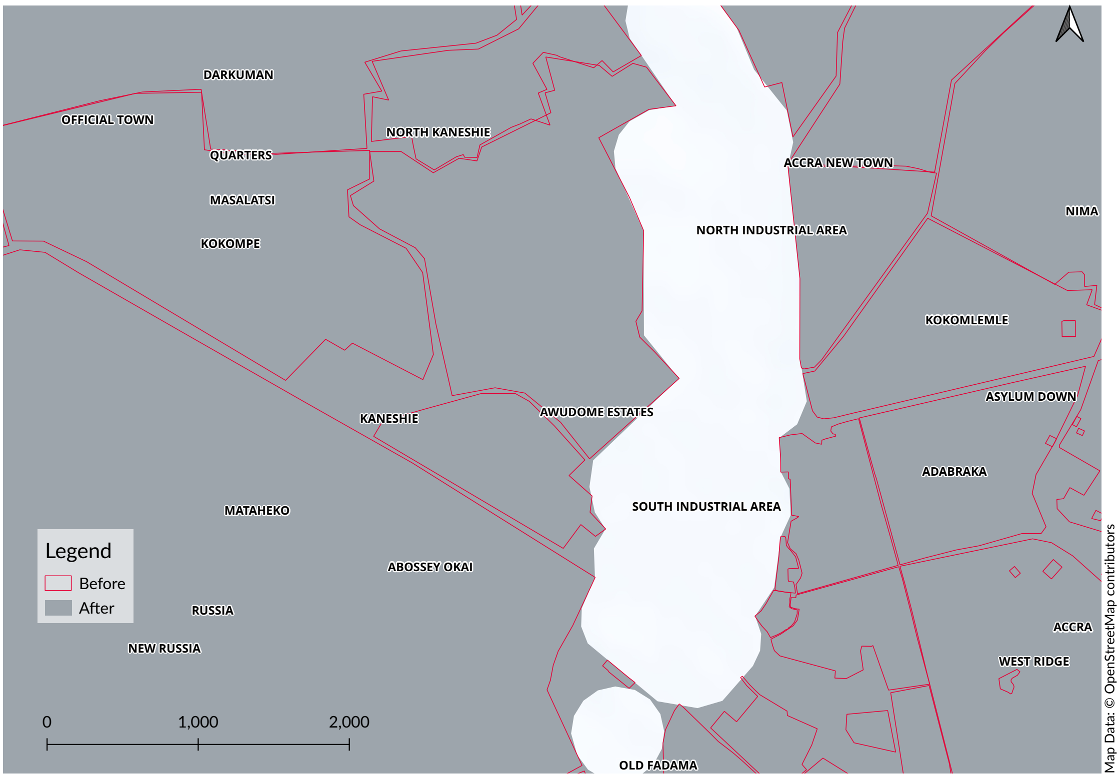


Figure 4.4: Residential landuse before and after processing

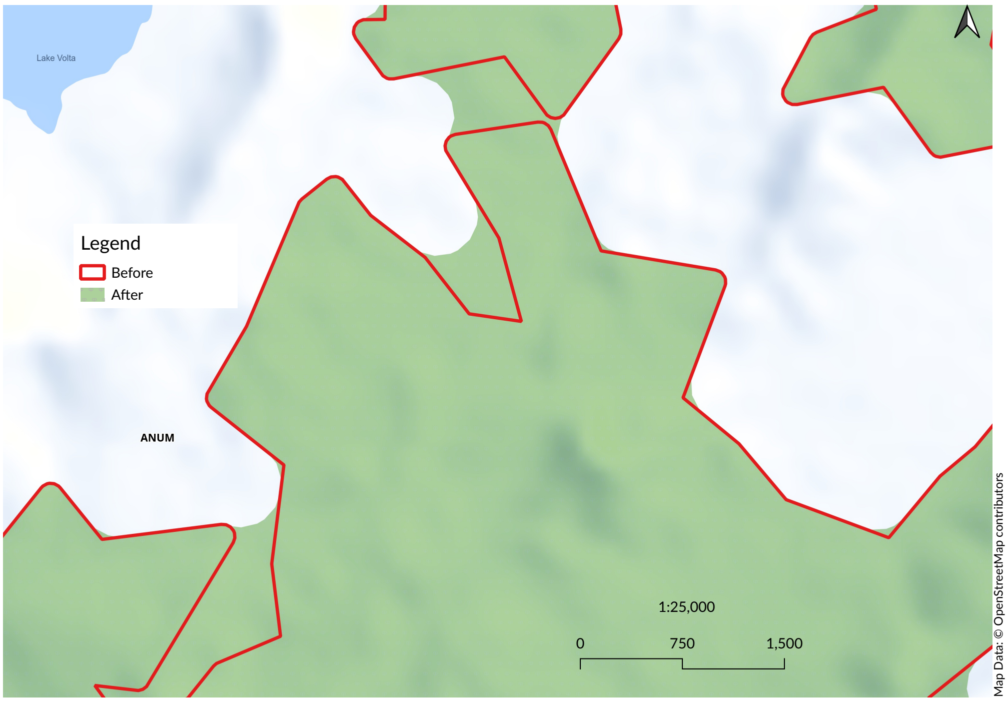


Figure 4.5: Wood/forest landuse before and after processing

4.2.5 Data Visualisation

The data visualisation stage of map production is where some cartographic visualisation approaches were taken into consideration. Based on the criterion in Table 3.10 and performance scale of 0 to 1.0 (as described in Section 3.3.4) scores were assigned to alternatives per criteria. QGIS, which was used for visualisation and layout, was the most suited alternative from the ranking as shown in Table 4.9.

Table 4.9: Evaluation based on weighting and ranking for visualisation stage alternatives

Alts. / Criteria	*C1*	*C2*	*C3*	*C4*	*C5*	*C6*	*C7*	*C8*	*C9*	*WPM*	*Rank*
GeoServer	1.0	1.0	0.7	0.8	1.0	0.5	1.0	0.6	1.0	0.794961	3
Mapnik	1.0	1.0	0.3	1.0	0.9	0.5	1.0	0.9	1.0	0.801460	2
QGIS	1.0	1.0	1.0	1.0	1.0	0.8	1.0	1.0	1.0	0.977933	1
openJUMP	1.0	0.8	0.5	0.6	0.7	0.6	1.0	0.7	1.0	0.740228	4

Firstly, establishing a connection from QGIS with valid credentials to the database is necessary. The layers are then visualised by dragging and dropping them from any of the used schemas. At this stage, the `map` schema is of the most importance. This is because the layers here are ready for map design compared to the other schemas which are for pre and post-processing. QGIS was also used for the designing of map layouts for exporting to final maps as shown in Figure 4.6. The symbology of the layers was based on publicly available map styles from Boundless Legacy (planetfederal, 2022). In order to automate basic and default styling for reuse and ensure uniform symbology, the symbology was saved to the PostGIS database. This enables multiple QGIS users to have the same styling and access to it. During the execution of the workflow styles configuration is taken care of automatically for the layers in the *map* schema. This is done through a dedicated table named *layer_styles* with the public schema of the database which QGIS can recognise.

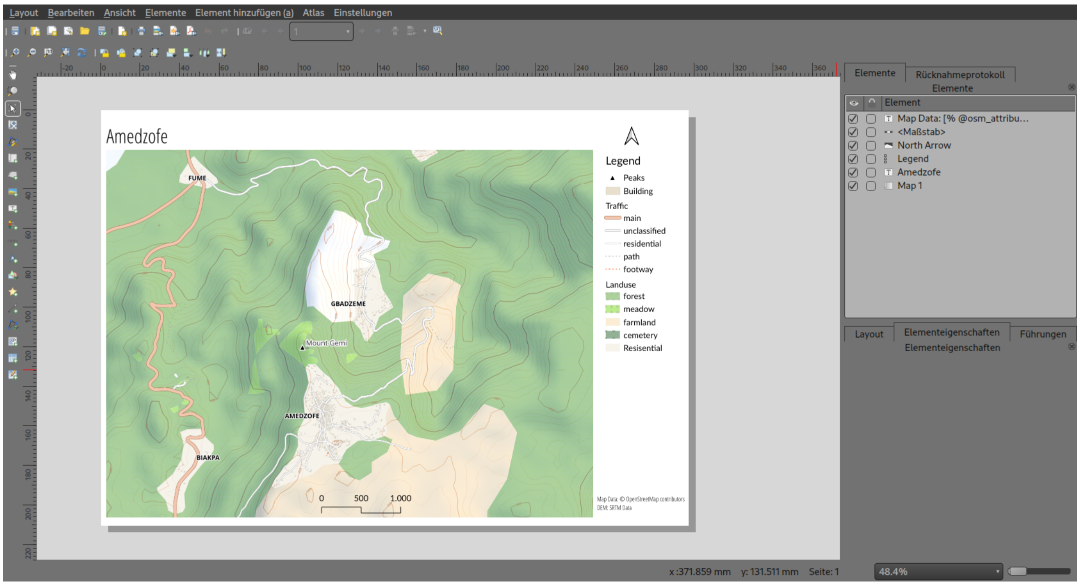


Figure 4.6: Setting out map for export

4.3 Handling Updates

The first step to implementing the handling of updates are the development of a data model. In this case, a data model was developed to handle both semantics and geometric constraints.

It should be noted that the available tools for OpenStreetMap data processing either can keep a local database up-to-date with changes from OpenStreetMap on their own or in combination with an additional tool. It is however uncertain if they can handle or keep changes that have been made locally whilst incorporating upstream changes from OpenStreetMap itself.

In order to handle updates a data model using Spatio-temporal properties of features coupled with a measure of similarity which is a unique ID. First, the process of keeping track of changes was catered for in a dedicated database schema called *history*. This schema stores versions of new, modified, and deleted features from the *map* schema. A feature can be modified several times but the most relevant change is based on its spatiotemporal properties. This is inferred from the most recent timestamp. Figure 4.7 illustrates the connection between the *history* and other schemas implemented in the data storage.

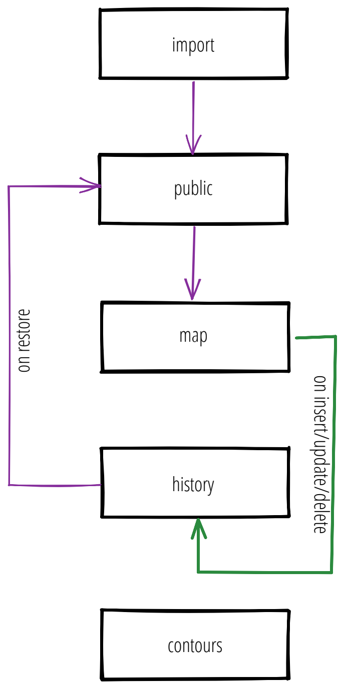


Figure 4.7: Database schemas used

Recording Changes

The history of changes is recorded through versioning. Versioning is made possible and implemented with database triggers. Database triggers, as explained in Section 2.2.11, are functions that ensure transactional integrity in a database during changes. Three types of features change tracking were implemented i.e. (a) insert/add, (b) update/modify, and (c) delete/remove. The power of a database-backed approach to history tracking is that it is software-independent. This is because no matter which tool is used for edits, whether from the SQL command line, a web-based tool or a desktop tool like QGIS, the history is consistently tracked.

4.3.1 Handling of geometric and semantic changes

According to Adams et al. (2010) every geospatial feature on Earth can be represented by a 4-tuple «*G, F, A, T*» where: *G* is Geometry, *F* is Feature type, *A* is a Set of attributes, and *T* is a Set of topological relationships to other features. *F, A*, and *T* could be considered under semantic labels (and even *G*) but Adams et al. (2010) found it useful to make a separation because feature types are salient in the conceptualisation of geospatial features and topology is unique from other attributes. Based on this the data model implemented handles label semantics and and geometry. Listing 4.1 shows an example SQL data model for tracking places.

Listing 4.1: SQL Data model for tracking places

```
        CREATE TABLE IF NOT EXISTS history.places_history (
        hid SERIAL PRIMARY KEY, -- History ID
        node_id BIGINT, -- Node ID from OpenStreetMap
        fid INTEGER, -- Local ID created during import
        name VARCHAR(200), -- Name of the object
        name_en VARCHAR(200), -- English name of the object
        type VARCHAR(50), -- Type of object i.e village, city, town, etc
        population VARCHAR(50), -- Population
        capital VARCHAR(50), -- If the object is a capital of administrative unit
        ts TIMESTAMP DEFAULT now(), -- Timestamp of change
        geom GEOMETRY(Point,25000), -- Geometry of object i.e. Point
        created_by VARCHAR(32), -- User who modified or created an object
        deleted_by VARCHAR(32) -- User who deleted object
);
```

In this implementation both semantic and geometric changes are handled for all feature types i.e. points, lines, and polygons. A unique ID is used as the measure of similarity for data conflation during future updates. The process works by creating a history of changes either semantic label or geometric ordered by timestamps. The changes that are taken into account are additions, updates, and deletions.

During restoration, modified or new objects get added to replace existing ones if the constraint of unique ID matches. The same applies to deleted features which are instead deleted during conflation. Figure 4.8 illustrates the processing of handling updates implemented.

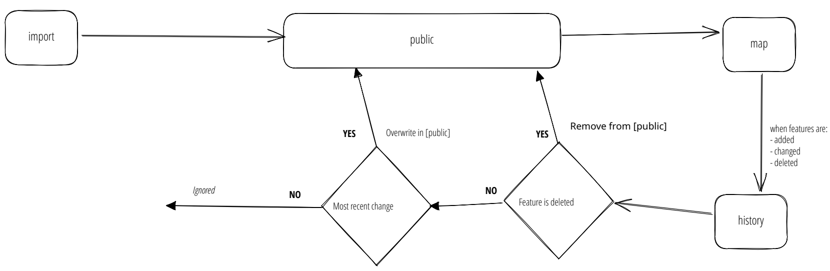


Figure 4.8: Data Conflation flow

Figure 4.9 shows an example point feature (peaks) history and restored version using the most recent changes.

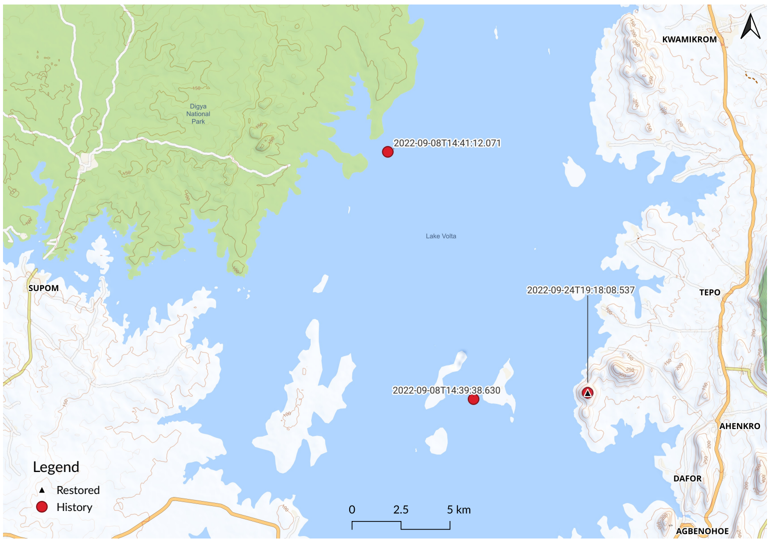


Figure 4.9: History of Peaks restored (Points)

Figure 4.10 shows an example line feature (roads) history and restored version using the most recent changes.

Figure 4.10: History of roads restored (Lines)

Figure 4.11 shows an example polygon feature (water) history and restored version using the most recent changes.

Figure 4.11: History of water restored (Polygons)

5 Discussions

This chapter discusses the outcomes of this research and its implementation, successes, failures, and approaches by responding to the research objectives and research questions.

5.1 Discussion of Software Selection and Workflow

RO1: Creating a general workflow for print map production using OpenStreetMap and Free Software.

RQ1.1: How can a general workflow for print map production be achieved using OpenStreetMap as the main data source?

In order to create a general workflow for print map production using OpenStreetMap and Free Software, a list of needs were first identified resulting in the development of requirements as indicated Section 3.2. These requirements guided the process modes of the workflow to create stages for specific tasks. These stages enabled the execution of individual stages of the workflow without executing the entire process flow. This allowed for quick and easy troubleshooting and maintenance. This also saved time and the cost involved in replacing specific parts of the entire workflow with no or minimal disruption in existing and working components. It is observed that filtering and sub-setting data at the earliest stage of the workflow improved import time and further processing.

One shortcoming is a good understanding of OpenStreetMap data model and it is crucial to create a workflow with the data. With enough understanding, one can be able to implement a comprehensive model transformation from OpenStreetMap to a geodatabase.

RQ1.2: Which categories of tools are suitable for the specific tasks in the workflow?

The categorisation of software in a specific domain is vital to helping both developers, users and stakeholders know what is needed. Categorisation of various tools is not always straightforward as one tool can perform several functions that overlap stages and categories. In the end categorisation of available tools based on the needs of the cartographic process was used to develop a four-stage map production process i.e. Data Preparation, Data Transformation, Data Storage, and Visualisation. Based on these four stages the identified tools were classified.

RQ1.3: How can various tools for the specific steps identified in RQ1.2 be compared and evaluated according to functions, performance, flexibility, and platform support?

Considering that software selection is a multi-criteria decision problem requiring a multi-stakeholder and multiple criteria, direct comparison of software functions was tedious, very exhaustive and less decisive. This research, therefore made use of Multi-Criteria Decision Analysis to identify, evaluate, and select tools.

MCDA is non-linear, complex, objective, and iterative. A more general criterion relating to how useful it was to a stage in the workflow was created. Weights were assigned to the criterion based on their importance. Performance scores were assigned to each alternative based on the pair-wise comparison. Using the Weighted Product Model (WPM) the software alternatives were evaluated and ranked for the best alternative. The ranking process was tested with Weighted Sum Model (WSM) and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) methods which resulted in a similar ranking.

At the Data Preparation stage, **GDAL/OGR** was the best suited noting that it had the highest performance score under all criteria as shown in Figure 5.1. *GDAL/OGR* is well suited for data preparation with support for many data formats, good documentation, active development, and cross-platform support. It is also a library that is used in several applications both free software and proprietary. The presence of *GDAL/OGR* in other software might already make it available to the user without knowing. *GDAL/OGR* was also used for preparing DEMs, hill shading, and generating contour lines. Notwithstanding, both *GDAL/OGR* and the second-best alternative *osmium*, which specifically targets OSM data processing was used in this thesis since OSM data formats.

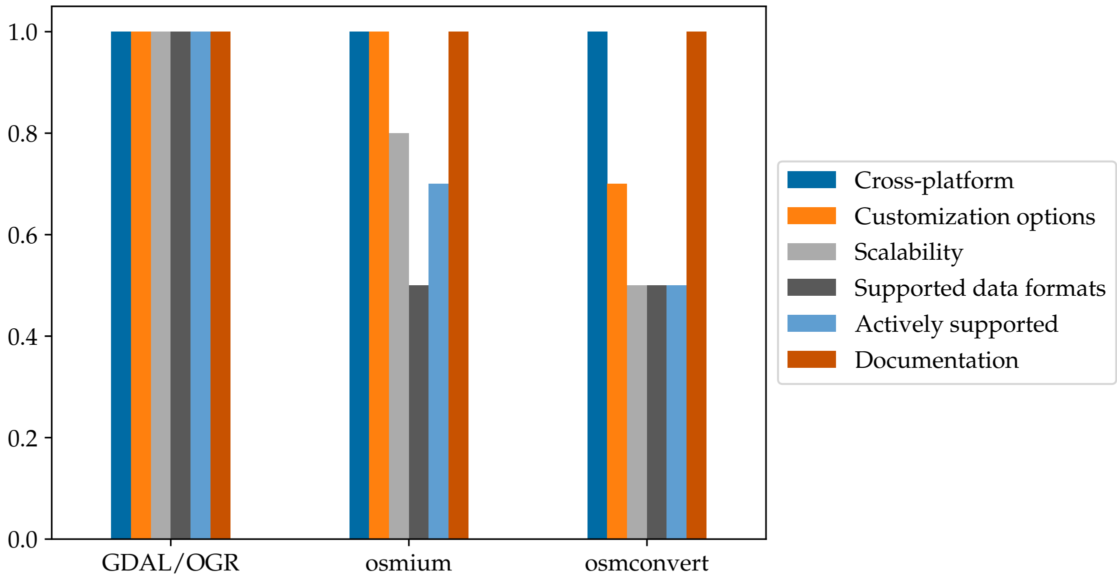


Figure 5.1: Data Preparation alternatives, performance scores, and criteria.

At the Data Transformation stage even though *imposm* has the highest performance for Usability (due to how easy it is to configure data transformation file using YAML; which is a human-readable markup language), it only supports two projections (i.e. EPSG:4325 and EPSG:3857). *osm2pgsql* on the other hand supports as many projections as GDAL/OGR supports because it is compiled with the *Proj* library. *osm2pgsql* also had a higher score for customisation options since its configurations are done via the Lua programming language allowing for extensibility and own functions.

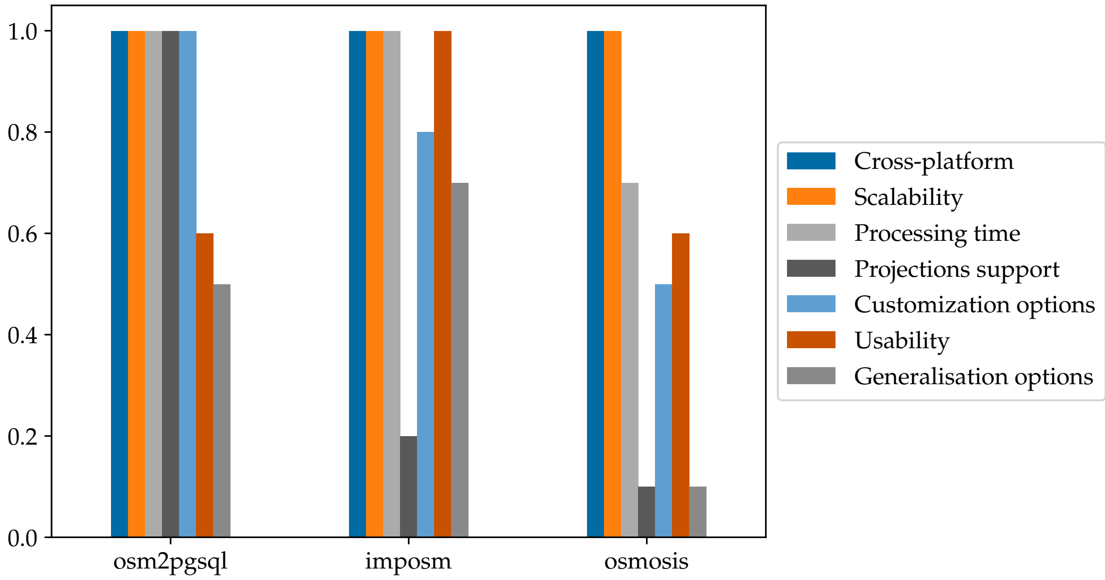


Figure 5.2: Data Transformation alternatives, performance scores, and criteria.

At the Data Storage stage as presented in Figure 5.3, all alternatives appear to have some support for spatial data. However, SQLite with its SpatiaLite extension, PostgreSQL with PostGIS extension, and GeoPackage had high performance scores. PostGIS for PostgreSQL however had more spatial functions than other alternatives. Here the term Community as a living ecosystem that enables the software to thrive via the organisation of events and user groups around spatial data storage. All options appear to have associated communities but PostgreSQL with *PostGIS Day* and other Community events increased its score making **PostgreSQL** the best-suited alternative.

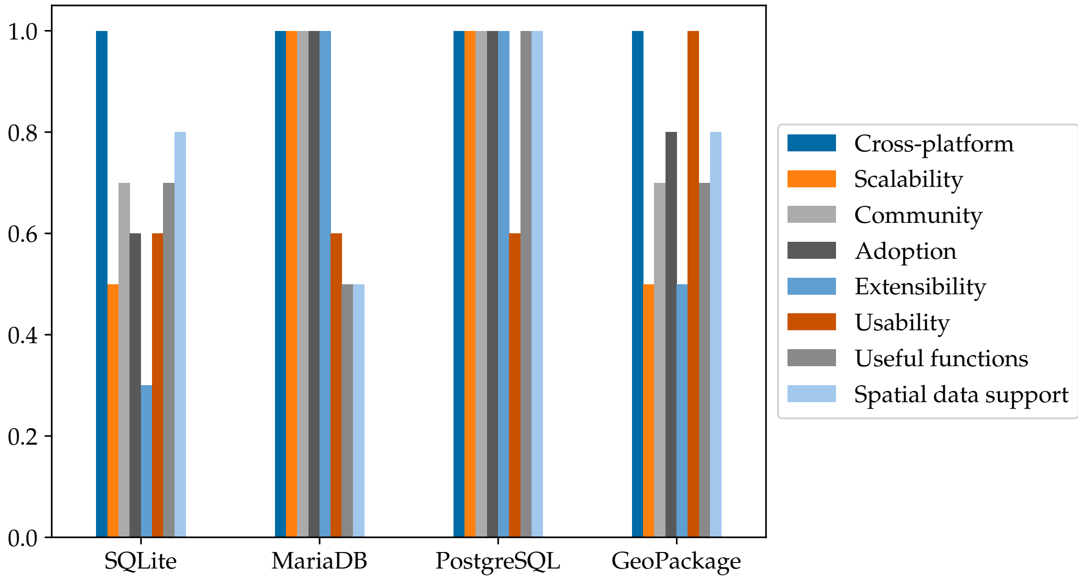


Figure 5.3: Data Storage alternatives, performance scores, and criteria.

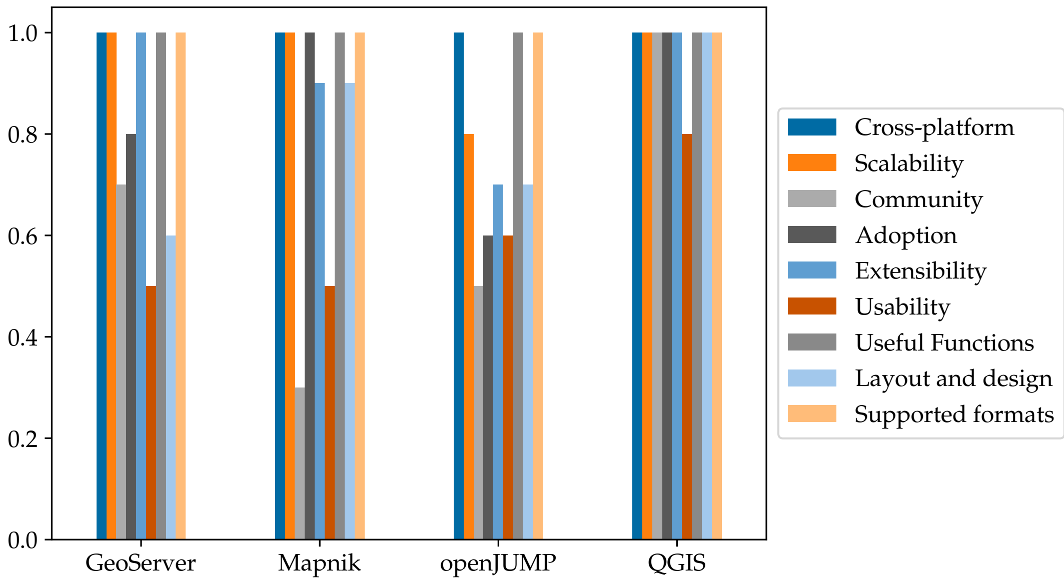


Figure 5.4: Visualisation alternatives, performance scores, and criteria.

At the Visualisation stage, as indicated in Figure 5.4, **QGIS** was the most suited software considering the high-performance scores in most criteria. However, it recorded less usability which might be the case for users using the software for the first time. *Mapnik* and *Geoserver* which ranked second and third respectively had the lowest performance score as they are not primarily made for print production and requires a high level of expertise compared to Desktop GIS as in the case of QGIS and *OpenJUMP*.

5.2 Discussion of Data Model and Handling of Updates

RQ2 – How can updates with data changes and adjustments be integrated into existing map data?

RQ2.1: How can a data model be designed for handling changes?

A Data Model was implemented using SQL Triggers. The power and usefulness of SQL Triggers for tracking changes and restoring modified features is that they are software independent, meaning that so far as the GIS software or tool can connect to the database with the appropriate rights for reading and modification any changes will be taken into account. The data model implemented in this research can handle both geometric and semantic labels i.e. attribute changes for points, lines, and polygons per feature. This is handled on creation, update and deletion. It, however, adopted only one measure of similarity, which is based on a unique ID for data conflation. Geometric changes handling is strongly tied to the measure of similarity

RQ2.2: Which conflicts can result from updates and manual edits, and how can they be resolved?

Conflicts that could arise are both geometric and semantic label related. The conflicts arise on point, line and polygon features.

Geometries of point features were not a problem as their conflation was much more straightforward using the measure of similarity adopted; an example is shown in Figure 4.9, 4.10, and 4.11 representing point, line, and polygon respectively. For all feature types, semantic label conflicts were resolved by overwriting existing data with derived data from history *schema* when features are matched. This is done by using their unique ids (Figure 4.10 shows IDs of objects and timestamps) and most recent changes ordered by timestamp from the history schema.

RQ2.3: How can a suitable data model be implemented to handle updates using selected tools from RQ2.1?

In order to handle updates and keep own changes and using an adjustable data model was implemented using PostgreSQL and PostGIS. The data model makes use of database triggers which ensure history tracking on creation, updating, and deletion. Using SQL functions made the data model software/tool independent. This is because any tool that access and modifies data will be accounted for.

6 Conclusion

In this thesis, a workflow for general and basic map production was conceptualised and implemented. The workflow is broken down into stages. These stages also represent the categorisation of tools and software employed in the workflow. In order to understand which free and open source software exists for the map production process, a shortlist of identified tools were created for each stage.

To select the best-suited tool from the shortlisted application to complete the building of flow lines for the workflow, the method of software selection method proposed by Steiniger and Hunter (2013) was implemented.

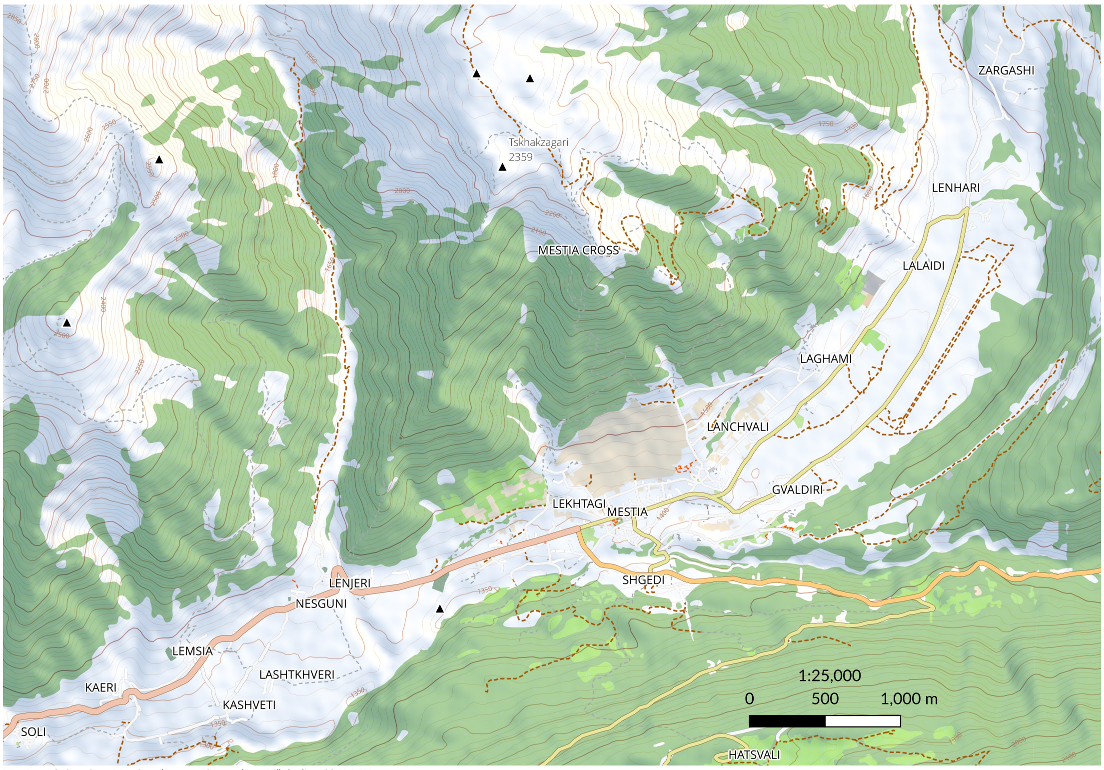
Since the process of selecting the best suited software from a list of alternatives is a multi-criteria problem, the Weighted Product Model (WPM), one of the most popular multi-criteria decision method was implemented. A set of criteria were developed for each stage due to the varying needs of these stages to fulfil with some of them repeating or unique at stage.

A pair-wise comparison based on the scale 0, 0.5, and 1 representing *low*, *mid*, and *high* respectively was used to assign weights to each criterion. The same scale was used to assign performance scores per alternative. All criteria were considered as benefits, meaning the higher the weight the more important they are. The evaluation resulted in using GDAL, osm2pgsl, PostgreSQL (PostGIS), and QGIS for the implementation of the workflow.

With the appropriate data transformation tool such as osm2pgsql the process of retrieval could become much easier.

For **RO2**, an adjustable data model was also developed to handle changes and updates. This is based on using historical changes in the dedicated database schema. The spatiotemporal recentness of features based on timestamps is used for restoring modified features.

Notwithstanding, it has been proven that VGI sources such as OpenStreetMap are potential source of data for cartographic work productions in places where less is known about authoritative data. Since OpenStreetMap data model is worldwide, the implemented workflow was easily adapted to Georgia. The implementation produced the same results as shown in Figure 6.1, with just a few parameters to modify.



Map Data: © OpenStreetMap Contributors. Contour lines, Hill shading: CGIAR DEM

Figure 6.1: Result of applying workflow to Georgia

It is also recommended that the local institutions responsible for data collection and map production take advantage of this leverage where volunteers and large communities are responsible for creating such enormous data.

The processing and transformation of OpenStreetMap data into a ready spatial database is time-consuming, technical, and requires a lot of resources. Notwithstanding, most information about how to process and transform OSM is available on the web and also in previous research and academic publications. Several iterations and technical knowledge are also required in order to implement such a workflow and sometimes use it (Kunz & Bobrich, 2019).

6.1 Limitations and Outlook

A simple method, the Weighted Product Model was used for weighting and ranking of software in this thesis. A more sophisticated method such as Analytic Hierarchy Process (AHP) which have been used by Eldrandaly (2007) and Eldrandaly and Naguib (2013) could be implemented. The identification of alternatives and assignment of performance scores could also be done through local stakeholder engagements.

This thesis only created a workflow for one scale which is 1:25000. In order to create a multi-scale workflow further and several generalisation operations need to be carried out. Map production is still a difficult task and requires multiple processing to arrive at the best results.

Currently, the output of a map is done manually after the workflow is executed. QGIS can be extended with Python programming language to make map exporting partial or completely automated by taking into account dynamic map elements. Alternatives such as Mapnik and GeoServer can also be employed in this regard, they however required much technical knowledge compared to QGIS.

The implemented data model for handling updates used only one measure of similarity and one approach i.e. unique IDs and spatiotemporal records. It can be further improved by using multiple measures of similarity and approaches such as topological constraints and shape similarity.

Bibliography

Adams, B., Li, L., Raubal, M., & Goodchild, M. F. (2010). A general framework for conflation.

Alamoudi, E., Mehmood, R., Aljudaibi, W., Albeshri, A., & Hasan, S. H. (2020). Open source and open data licenses in the smart infrastructure era: Review and license selection frameworks. In R. Mehmood, S. See, I. Katib, & I. Chlamtac (Eds.), *Smart infrastructure and applications: Foundations for smarter cities and societies* (pp. 537–559). Springer International Publishing. https://doi.org/10.1007/978-3-030-13705-2_22

Aujac, G., Harley, J., & Woodward, D. (1987). The foundations of theoretical cartography in archaic and classical greece. *The history of cartography*, *1*, 130–147.

Bégin, D. (2012). Towards integrating VGI and national mapping agency operations-A Canadian case study. *Proceedings of the GIScience Conference, Columbus, OH, USA*, 18–21.

Bennett, J. (2010). *OpenStreetMap: Be your own cartographer*. Packt Publ.

Bridgman, P. W. (1922). *Dimensional analysis*. Yale university press.

Buckley, A., & Watkins, D. (2007). Automated Map Production Workflows. *24th International Cartographic Conference*.

Buttenfield, B. P., & Hultgren, T. (2005). Managing multiple representations of "base carto" features: A data modeling approach. *Proceedings, AUTO-CARTO 2005*.

CGIAR-CSI. (2018). CGIAR-CSI SRTM – SRTM 90m DEM Digital Elevation Database [publisher: CGIAR-CSI]. Retrieved September 26, 2022, from https://srtm.csi.cgiar.org/

Chen, D., Shams, S., Carmona-Moreno, C., & Leone, A. (2010). Assessment of open source GIS software for water resources management in developing countries. *Journal of Hydro-environment Research*, *4*(3), 253–264. https://doi.org/10.1016/j.jher.2010.04.017

Çobankaya, O. N., & UluğtekiN, N. (2013). Multi-scale Representation: Modelling and Updating, 8.

Connolly, T. M., & Begg, C. E. (2005). *Database systems: A practical approach to design, implementation, and management*. Pearson Education.

Contributors, Q. (2022). *Qgis* (Version 3.22.3). Zenodo. https://doi.org/10.5281/zenodo.5869838

Davis, M., & Kent, A. J. (2022). Soviet City Plans and OpenStreetMap: A comparative analysis. *International Journal of Cartography*, 1–14. https://doi.org/10.1080/23729333.2022.2047396

Dumedah, G., Marfo, C. O., & Andam-Akorful, S. A. (2022). An interoperability mobile application for location and travel planning services in ghana: Mytroski. *International Journal of Applied Geospatial Research (IJAGR)*, *13*(1), 1–17.

Eldrandaly, K. (2007). Gis software selection: A multi-criteria decision making approach.

Eldrandaly, K., & Naguib, S. (2013). A knowledge-based system for gis software selection. *Int. Arab J. Inf. Technol.*, *10*(2), 152–159.

Emde, A. (2022). *Osgeolive* (Version 14.0.0) [OSGeoLive is a project of the Open Source Geospatial Foundation (OSGeo)]. Zenodo. https://doi.org/10.5281/zenodo.5884641

European Commission. (2022). Why do we need to license? [[Online; accessed 9. Oct. 2022]]. https://data.europa.eu/elearning/en/module4/#/id/co-01

Free Software Foundation. (2022). What is Free Software? - GNU Project - Free Software Foundation [[Online; accessed 12. Sep. 2022]]. https://www.gnu.org/philosophy/free-sw.html

Free Software Foundation Europe. (2022). Free Software - FSFE [[Online; accessed 12. Sep. 2022]]. https://fsfe.org/freesoftware

Fritz, S., See, L., & Brovelli, M. (2017). Motivating and Sustaining Participation in VGI. In S. Fritz, L. See, G. Foody, P. Mooney, A.-M. Olteanu-Raimond, C. C. Fonte, & V. Antoniou (Eds.), *Mapping and the Citizen Sensor* (pp. 93–118). Ubiquity Press. Retrieved May 30, 2021, from https://www.jstor.org/stable/j.ctv3t5qzc.8

Ghana Statisical Service. (2022). 2021 Population and Housing Census - Preliminary Report [[Online; accessed 21. Sep. 2022]]. https://web.archive.org/web/20220704023146/https://census2021.statsghana.gov.gh/gssmain/fileUpload/reportthemelist/PRINT_COPY_VERSION_FOUR%2022ND_SEPT_AT_8_30AM.pdf

GhanaWeb. (2020). Lands Commission to use auto maps to address land acquisition challenges [[Online; accessed 21. Sep. 2022]]. https://www.ghanaweb.com/GhanaHomePage/NewsArchive/Lands-Commission-to-use-auto-maps-to-address-land-acquisition-challenges-1062403

Goetz, M., Lauer, J., & Auer, M. (2012). An Algorithm Based Methodology for the Creation of a Regularly Updated Global Online Map Derived From Volunteered Geographic Information, 9.

Goodchild, M. F. [Michael F.]. (2007). Citizens as sensors: The world of volunteered geography. *GeoJournal*, *69*(4), 211–221. https://doi.org/10.1007/s10708-007-9111-y

Goodchild, M. F. [Michael F], & Hunter, G. J. (1997). A simple positional accuracy measure for linear features. *International journal of geographical information science*, *11*(3), 299–306.

Haklay, M., & Weber, P. (2008). Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, *7*(4), 12–18.

Hardy, P., Briat, M.-O., Eicher, C., & Kressmann, T. (2004). Database-driven cartography from a digital landscape model, with multiple representations and human overrides. *submitted to ICA Commission on Generalization and Multiple Representation–Research Workshop, Leicester, UK*.

Hollingsworth, D., & Hampshire, U. (1995). Workflow management coalition: The workflow reference model. *Document Number TC00-1003*, *19*(16), 224.

Intergovernmental Committee on Surveying and Mapping. (2022). History of Mapping | Intergovernmental Committee on Surveying and Mapping [[Online; accessed 18. Aug.

2022]]. https://icsm-prod.oxide.co/education/fundamentals-mapping/history-mapping

Käuferle, D., Streit, C., & Forte, O. (2015). New national maps for switzerland. *1st ICA European Symposium on Cartography*, 50.

Kunz, P., & Bobrich, J. (2019). Multiscale cartographic visualization of harmonized datasets. *International Journal of Cartography*, *5*(2-3), 178–194. https://doi.org/10.1080/23729333.2019.1610931

Liu, C., Xiong, L., Hu, X., & Shan, J. (2015). A Progressive Buffering Method for Road Map Update Using OpenStreetMap Data [Number: 3 Publisher: Multidisciplinary Digital Publishing Institute]. *ISPRS International Journal of Geo-Information*, *4*(3), 1246–1264. https://doi.org/10.3390/ijgi4031246

Longley, P. A., Goodchild, M. F., Maguire, D. J., & Rhind, D. W. (2015). *Geographic information science and systems*. John Wiley & Sons.

Löwe, P., Anguix Alfaro, Á., Antonello, A., Baumann, P., Carrera, M., Durante, K., Hugentobler, M., Lime, S., Mitasova, H., Müller, D., Neteler, M., Reed, J., Strobl, C., & Wessel, P. (2022). Open source – gis. In W. Kresse & D. Danko (Eds.), *Springer handbook of geographic information* (pp. 807–843). Springer International Publishing. https://doi.org/10.1007/978-3-030-53125-6_30

Lynam, E. (1944). *British maps and map-makers* (Vol. 73). London: W. Collins.

Madubedube, A., Coetzee, S., & Rautenbach, V. (2021). A contributor-focused intrinsic quality assessment of openstreetmap in mozambique using unsupervised machine learning. *ISPRS International Journal of Geo-Information*, *10*(3), 156.

Markieta, M. (2012). Using openstreetmap data with open-source gis. *Cartographic Perspectives*, (71), 91–104.

Masser, I. (2005). *Gis worlds: Creating spatial data infrastructures* (Vol. 338). Esri Press Redlands.

McKenzie, G., Janowicz, K., & Adams, B. (2013). Weighted multi-attribute matching of user-generated points of interest. *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 440–443. https://doi.org/10.1145/2525314.2525455

Miller, D. W., et al. (1963). Executive decisions and operations research.

Müller, M., Wiemann, S., & Grafe, B. (2012). A framework for building multi-representation layers from OpenStreetMap data, 9.

Nikolakopoulos, K. G., Kamaratakis, E. K., & Chrysoulakis, N. (2006). Srtm vs aster elevation products. comparison for two regions in crete, greece. *International Journal of remote sensing*, *27*(21), 4819–4838.

Olteanu-Raimond, A.-M., Hart, G., Foody, G. M., Touya, G., Kellenberger, T., & Demetriou, D. (2017). The Scale of VGI in Map Production: A Perspective on European National Mapping Agencies. *Transactions in GIS*, *21*(1), 74–90. https://doi.org/10.1111/tgis.12189

Open Knowledge Foundation. (2016). Place overview - Global Open Data Index [[Online; accessed 9. Oct. 2022]]. https://web.archive.org/web/20220901002214/http://index.okfn.org/place

OpenStreetMap. (2022). OpenStreetMap - Copyright and Licence [[Online; accessed 21. Sep. 2022]]. https://www.openstreetmap.org/copyright

OpenStreetMap Wiki. (2019). Osmconvert — OpenStreetMap Wiki [[Online; accessed 7-October-2022]]. https://wiki.openstreetmap.org/w/index.php?title=Osmconvert&oldid=1920339

OpenStreetMap Wiki. (2021a). Imposm — OpenStreetMap Wiki [[Online; accessed 7-October-2022]]. https://wiki.openstreetmap.org/w/index.php?title=Imposm&oldid=2190286

OpenStreetMap Wiki. (2021b). Osm2pgsql — OpenStreetMap Wiki [[Online; accessed 7-October-2022]]. https://wiki.openstreetmap.org/w/index.php?title=Osm2pgsql&oldid=2215447

OpenStreetMap Wiki. (2022a). Editors — OpenStreetMap Wiki [[Online; accessed 21-September-2022]]. https://wiki.openstreetmap.org/w/index.php?title=Editors&oldid=2386082

OpenStreetMap Wiki. (2022b). Elements — OpenStreetMap Wiki [[Online; accessed 21-September-2022]]. https://wiki.openstreetmap.org/w/index.php?title=Elements&oldid=2394875

OpenStreetMap Wiki. (2022c). Osmium — OpenStreetMap Wiki [[Online; accessed 7-October-2022]]. https://wiki.openstreetmap.org/w/index.php?title=Osmium&oldid=2376950

OpenStreetMap Wiki. (2022d). Osmosis — OpenStreetMap Wiki [[Online; accessed 7-October-2022]]. https://wiki.openstreetmap.org/w/index.php?title=Osmosis&oldid=2411898

OpenStreetMap Wiki. (2022e). Quality assurance — openstreetmap wiki [[Online; accessed 9-October-2022]]. https://wiki.openstreetmap.org/w/index.php?title=Quality_assurance&oldid=2401429

OpenTopoMap. (2022). Opentopomap [[Online; accessed 22. Sep. 2022]]. https://opentopomap.org/about

Owusu-Banahene, W., Mensah, F., Coetzee, S., Cooper, A. K., Rautenbach, V., Sinvula, K. M., Nangolo, E., & Hippondoka, M. (2013). A description of spatial data infrastructure stakeholders in ghana using the ica model. GSDI Association.

planetfederal. (2022). osm-styles [[Online; accessed 8. Oct. 2022]]. https://github.com/planetfederal/osm-styles

Ramm, F., Topf, J., & Chilton, S. (2011). *OpenStreetMap: Using and enhancing the free map of the world*. UIT Cambridge Cambridge.

Robinson, A. H., Morrison, J. L., Muehrcke, P. C., Kimerling, A. J., & Guptill, S. C. (1995). *Elements of Cartography* [Google-Books-ID: mUyAAAAAMAAJ]. Wiley.

Rosemann, M., & Zur Muehlen, M. (1998). Evaluation of workflow management systems-a meta model approach. *Australian Journal of Information Systems*, *6*(1).

Rouault, E., Warmerdam, F., Schwehr, K., Kiselev, A., Butler, H., Łoskot, M., Szekeres, T., Tourigny, E., Landa, M., Miara, I., Elliston, B., Kumar, C., Plesea, L., Morissette, D., Jolma, A., & Dawson, N. (2022). *Gdal* (Version v3.5.1). Zenodo. https://doi.org/10.5281/zenodo.6801315

Ruiz, J. J., Ariza, F. J., Urena, M. A., & Blázquez, E. B. (2011). Digital map conflation: A review of the process and a proposal for classification. *International Journal of Geographical Information Science*, *25*(9), 1439–1466.

Samal, A., Seth, S., & Cueto 1, K. (2004). A feature-based approach to conflation of geospatial sources. *International Journal of Geographical Information Science*, *18*(5), 459–489.

Sarretta, A., & Minghini, M. (2021). Towards the integration of authoritative and openstreetmap geospatial datasets in support of the european strategy for data. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLVI-4/W2-2021*, 159–166. https://doi.org/10.5194/isprs-archives-XLVI-4-W2-2021-159-2021

Scheffler, T., Schirru, R., & Lehmann, P. (2012). Matching points of interest from different social networking sites. *Annual Conference on Artificial Intelligence*, 245–248.

See, L., Foody, G., Foody, G., Fritz, S., See, L., Mooney, P., Fritz, S., Olteanu-Raimond, A.-M., Mooney, P., Fonte, C. M. P. d. C., Olteanu-Raimond, A.-M., Antoniou, V., Fonte, C. C., & Antoniou, V. (2017). *Mapping and the Citizen Sensor*. Ubiquity Press. https://doi.org/10.5334/bbf

Stallman, R. (2012). Why open source misses the point of free software–gnu project–free software foundation [[Online; accessed 13. Sep. 2022]]. *GNU Project*. https://www.gnu.org/philosophy/open-source-misses-the-point.en.html

Steiniger, S., & Hunter, A. J. (2013). The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems*, *39*, 136–150. https://doi.org/10.1016/j.compenvurbsys.2012.10.003

Stoter, J., Post, M., van Altena, V., Nijhuis, R., & Bruns, B. (2014). Fully automated generalization of a 1: 50k map from 1: 10k data. *Cartography and Geographic Information Science*, *41*(1), 1–13.

Sui, D. Z. (2008). The wikification of gis and its consequences: Or angelina jolie's new tattoo and the future of gis. *Computers, environment and urban systems*, *1*(32), 1–5.

Sveen, A. F. (2008). Use of free and open source gis in commercial firms. *Masters thesis, Technology and Science, Norwegian University.*

Team, Qsos core. (2018). Method [[Online; accessed 14. Sep. 2022]]. https://www.qsos.org/method

The World Bank. (2022). GDP (current US) $-Ghana|$ Data [[Online; accessed 21. Sep. 2022]]. https://data.worldbank.org/indicator/NY.GDP.MKTP.CD?end=2021&locations=GH&start=2016&view=chart

Triantaphyllou, E., Shu, B., Sánchez, S. N., & Ray, T. G. (1998). Multi-criteria decision making: An operations research approach.

Tzotzos, A. (2021). Open source geospatial foundation (osgeo) [https://doi.org/10.5446/55248 (Last accessed: 01 Oct 2022)]. https://doi.org/10.5446/55248

Verhulst, S. G., & Young, A. (2017). *Open data in developing economies: Toward building an evidence base on what works and how*. African Minds. https://books.google.de/books?id=c1dDDwAAQBAJ

Virrantaus, K., Fairbairn, D., & Kraak, M.-J. (2009). ICA Research Agenda on Cartography and GI Science. *The Cartographic Journal*, *46*(2), 63–75. https://doi.org/10.1179/000870409X459824

Vondrakova, A., & Vozenilek, V. (2016). Licences and Open Data in Cartography. In G. Gartner, M. Jobst, & H. Huang (Eds.), *Progress in Cartography: EuroCarto 2015* (pp. 239–250). Springer International Publishing. https://doi.org/10.1007/978-3-319-19602-2_15

Wikimedia Commons. (2011). File:Ghana (orthographic projection).svg - Wikimedia Commons [[Online; accessed 21. Sep. 2022]]. https://commons.wikimedia.org/wiki/File:Ghana_(orthographic_projection).svg

Wikimedia Commons. (2022). File:categories of free and nonfree software.svg — wikimedia commons, the free media repository [[Online; accessed 13-September-2022]]. https://commons.wikimedia.org/w/index.php?title=File:Categories_of_free_and_nonfree_software.svg&oldid=665746090

Wikipedia contributors. (2022a). Data model (gis) — Wikipedia, the free encyclopedia [[Online; accessed 7-October-2022]]. https://en.wikipedia.org/w/index.php?title=Data_model_(GIS)&oldid=1069899166

Wikipedia contributors. (2022b). Geopackage — Wikipedia, the free encyclopedia [[Online; accessed 7-Oktober-2022]]. https://en.wikipedia.org/w/index.php?title=GeoPackage&oldid=1071852893

Wikipedia contributors. (2022c). Geoserver — Wikipedia, the free encyclopedia [[Online; accessed 7-Oktober-2022]]. https://en.wikipedia.org/w/index.php?title=GeoServer&oldid=1080305974

Wikipedia contributors. (2022d). Mapnik — Wikipedia, the free encyclopedia [[Online; accessed 7-Oktober-2022]]. https://en.wikipedia.org/w/index.php?title=Mapnik&oldid=1077252889

Wikipedia contributors. (2022e). Mariadb — Wikipedia, the free encyclopedia [[Online; accessed 7-Oktober-2022]]. https://en.wikipedia.org/w/index.php?title=MariaDB&oldid=1112541326

Wikipedia contributors. (2022f). Postgresql — Wikipedia, the free encyclopedia [[Online; accessed 7-Oktober-2022]]. https://en.wikipedia.org/w/index.php?title=PostgreSQL&oldid=1114504561

Wikipedia contributors. (2022g). Sqlite — Wikipedia, the free encyclopedia [[Online; accessed 7-Oktober-2022]]. https://en.wikipedia.org/w/index.php?title=SQLite&oldid=1112521160

Wikipedia contributors. (2022h). String metric — Wikipedia, the free encyclopedia [[Online; accessed 9-October-2022]].

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., et al. (2016). The fair guiding principles for scientific data management and stewardship. *Scientific data*, *3*(1), 1–9.

Wood, H. (2022). 10 Years of ODbL | OpenStreetMap Blog [[Online; accessed 9. Oct. 2022]]. https://blog.openstreetmap.org/2022/09/14/10-years-of-odbl

Zielstra, D., & Zipf, A. (2010). A comparative study of proprietary geodata and volunteered geographic information for Germany. *13th AGILE international conference on geographic information science*, *2010*, 1–15.

A Appendix I

Listing A.1: osm2pgsql model generalisation configuration

```
-- Requires osm2pgsql Version >= 1.7.1
print('osm2pgsql␣version:␣' .. osm2pgsql.version)

local tables = {}
local mySchema = 'import'

tables.landuse = osm2pgsql.define_area_table('landuse', {
    {column = 'fid', sql_type = 'serial', create_only = true},
    {column = 'type', type = 'text'}, {column = 'name', type = 'text'},
    {column = 'geom', type = 'multipolygon'}
}, {schema = mySchema})

tables.residential = osm2pgsql.define_area_table('residential', {
    {column = 'fid', sql_type = 'serial', create_only = true},
    {column = 'type', type = 'text'}, {column = 'name', type = 'text'},
    {column = 'geom', type = 'multipolygon'}
}, {schema = mySchema})

tables.natural = osm2pgsql.define_area_table('natural', {
    {column = 'fid', sql_type = 'serial', create_only = true},
    {column = 'type', type = 'text'}, {column = 'name', type = 'text'},
    {column = 'geom', type = 'multipolygon'}
}, {schema = mySchema})

tables.building = osm2pgsql.define_area_table('building', {
    {column = 'fid', sql_type = 'serial', create_only = true},
    {column = 'name', type = 'text'}, {column = 'type', type = 'text'},
    {column = 'geom', type = 'multipolygon'}
}, {schema = mySchema})

tables.traffic = osm2pgsql.define_way_table('traffic', {
    {column = 'fid', sql_type = 'serial', create_only = true},
    {column = 'name', type = 'text'}, {column = 'highway', type = 'text'},
    {column = 'railway', type = 'text'}, {column = 'service', type = 'text'},
    {column = 'usage', type = 'text'}, {column = 'tracktype', type = 'text'},
    {column = 'oneway', type = 'text'}, {column = 'bridge', type = 'text'},
    {column = 'tunnel', type = 'text'}, {column = 'layer', type = 'text'},
    {column = 'ref', type = 'text'}, {column = 'geom', type = 'linestring'}
}, {schema = mySchema})

tables.places = osm2pgsql.define_node_table('places', {
    {column = 'fid', sql_type = 'serial', create_only = true},
    {column = 'name', type = 'text'}, {column = 'name_en', type = 'text'},
    {column = 'type', type = 'text'}, {column = 'population', type = 'text'},
    {column = 'capital', type = 'text'}, {column = 'geom', type = 'point'}
}, {schema = mySchema})

tables.peaks = osm2pgsql.define_node_table('peaks', {
    {column = 'fid', sql_type = 'serial', create_only = true},
    {column = 'name', type = 'text'}, {column = 'name_en', type = 'text'},
```

```
    {column = 'type', type = 'text'}, {column = 'ele', type = 'text'},
    {column = 'geom', type = 'point'}
}, {schema = mySchema})

-- local tag generalisation functions
local function building_type(object)
    local _type = {}
    if object.tags.building then _type = 'yes' end
    return tostring(_type)
end

function osm2pgsql.process_node(object)

    if object.tags.place then
        tables.places:insert({
            name = object.tags.name,
            name_en = object.tags['name:en'],
            type = object.tags.place,
            population = object.tags.population,
            capital = object.tags.capital,
            geom = object:as_point()
        })
    end

    if object.tags.natural == 'peak' or object.tags.natural == 'vulcano' then
        tables.peaks:insert({
            name = object.tags.name,
            name_en = object.tags['name:en'],
            type = object.tags.natural,
            ele = object.tags.ele,
            geom = object:as_point()
        })
    end

end

function osm2pgsql.process_way(object)

    -- A closed way that also has the right tags for an area is a polygon.
    if object.is_closed and object.tags.landuse ~= 'residential' and
        object.tags.landuse ~= 'forest' then
        tables.landuse:insert({
            type = object.tags.landuse,
            name = object.tags.name,
            geom = object:as_multipolygon()
        })
    end

    -- Residential only
    if object.is_closed and object.tags.landuse == 'residential' then
        tables.residential:insert({
            type = object.tags.landuse,
            name = object.tags.name,
            geom = object:as_multipolygon()
        })
    end
```

```
-- Process national parks
if object.is_closed and object.tags.boundary == 'national_park' then
    tables.landuse:insert({
        type = object.tags.boundary,
        name = object.tags.name,
        geom = object:as_multipolygon()
    })
end

if object.is_closed and object.tags.natural ~= 'wood' then
    tables.natural:insert({
        type = object.tags.natural,
        name = object.tags.name,
        geom = object:as_multipolygon()
    })
end

-- forests
if object.is_closed and object.tags.landuse == 'forest' and
    object.tags.natural == 'wood' then
    tables.landuse:insert({
        type = object.tags.landuse,
        name = object.tags.name,
        geom = object:as_multipolygon()
    })
end

-- woods only
if object.is_closed and object.tags.landuse ~= 'forest' and
    object.tags.natural == 'wood' then
    tables.landuse:insert({
        type = object.tags.natural,
        name = object.tags.name,
        geom = object:as_multipolygon()
    })
end

if object.is_closed and object.tags.building then
    tables.building:insert({
        name = object.tags.name,
        type = building_type(object),
        geom = object:as_multipolygon()
    })
end

if object.tags.highway or object.tags.railway then
    tables.traffic:insert({
        name = object.tags.name,
        highway = object.tags.highway,
        railway = object.tags.railway,
        service = object.tags.service,
        usage = object.tags.usage,
        tracktype = object.tags.tracktype,
        oneway = object.tags.oneway,
        bridge = object.tags.bridge,
```

```
            tunnel = object.tags.tunnel,
            layer = object.tags.layer,
            ref = object.tags.ref,
            object:as_multilinestring()
        })
    end

end

function osm2pgsql.process_relation(object)

    local type = object:grab_tag('type')

    -- Store multipolygon relations as polygons
    if type == 'multipolygon' and object.tags.landuse ~= 'residential' and
        object.tags.landuse ~= 'forest' then
        tables.landuse:insert({
            name = object.tags.name,
            type = object.tags.landuse,
            geom = object:as_multipolygon()
        })
    end

    -- Residential only
    if type == 'multipolygon' and object.tags.landuse == 'residential' then
        tables.residential:insert({
            type = object.tags.landuse,
            name = object.tags.name,
            geom = object:as_multipolygon()
        })
    end

    if type == 'multipolygon' and object.tags.boundary == 'national_park' then
        tables.natural:insert({
            name = object.tags.name,
            type = object.tags.boundary,
            geom = object:as_multipolygon()
        })
    end

    if type == 'multipolygon' and object.tags.natural ~= 'wood' then
        tables.natural:insert({
            type = object.tags.natural,
            name = object.tags.name,
            geom = object:as_multipolygon()
        })
    end

    -- forests
    if type == 'multipolygon' and object.tags.landuse == 'forest' and
        object.tags.natural == 'wood' then
        tables.landuse:insert({
            type = object.tags.landuse,
            name = object.tags.name,
            geom = object:as_multipolygon()
        })
```

```
    end

    -- woods only
    if type == 'multipolygon' and object.tags.landuse ~= 'forest' and
        object.tags.natural == 'wood' then
        tables.landuse:insert({
            type = object.tags.natural,
            name = object.tags.name,
            geom = object:as_multipolygon()
        })
    end

    if type == 'multipolygon' and object.tags.building then
        tables.building:insert({
            name = object.tags.name,
            type = building_type(object),
            geom = object:as_multipolygon()
        })
    end

end
```

B Appendix II

Listing B.1: 'Trigger functions for peaks (point) traffic (line) and natural (polygon)

```
-- Traffic
CREATE OR REPLACE FUNCTION map.traffic_history_trigger() RETURNS trigger AS
    $$
        BEGIN
            IF TG_OP = 'INSERT' THEN
                INSERT INTO history.traffic_history (way_id, fid, name, highway,
                    railway, service, usage, tracktype, oneway, bridge, tunnel,
                    layer, ref, ts, geom, created_by)
                    VALUES (
                            NEW.way_id,
                            NEW.fid,
                            NEW.name,
                            NEW.highway,
                            NEW.railway,
                            NEW.service,
                            NEW.usage,
                            NEW.tracktype,
                            NEW.oneway,
                            NEW.bridge,
                            NEW.tunnel,
                            NEW.layer,
                            NEW.ref,
                            current_timestamp,
                            NEW.geom,
                            current_user
                        );
                    RETURN NEW;
            ELSIF TG_OP = 'UPDATE' THEN
                INSERT INTO history.traffic_history (way_id, fid, name, highway,
                    railway, service, usage, tracktype, oneway, bridge, tunnel,
                    layer, ref, ts, geom, created_by)
                    VALUES (
                            NEW.way_id,
                            NEW.fid,
                            NEW.name,
                            NEW.highway,
                            NEW.railway,
                            NEW.service,
                            NEW.usage,
                            NEW.tracktype,
                            NEW.oneway,
                            NEW.bridge,
                            NEW.tunnel,
                            NEW.layer,
                            NEW.ref,
                            current_timestamp,
                            NEW.geom,
                            current_user
                        );
                    RETURN NEW;
```

```
                ELSIF TG_OP = 'DELETE' THEN
                    INSERT INTO history.traffic_history (way_id, fid, name, highway,
                        railway, service, usage, tracktype, oneway, bridge, tunnel,
                        layer, ref, ts, geom, deleted_by)
                        VALUES (
                                OLD.way_id,
                                OLD.fid,
                                OLD.name,
                                OLD.highway,
                                OLD.railway,
                                OLD.service,
                                OLD.usage,
                                OLD.tracktype,
                                OLD.oneway,
                                OLD.bridge,
                                OLD.tunnel,
                                OLD.layer,
                                OLD.ref,
                                current_timestamp,
                                OLD.geom,
                                current_user
                            );
                        RETURN OLD;
                END IF;
            END;
        $$
        LANGUAGE 'plpgsql';

    CREATE OR REPLACE TRIGGER traffic_history_trigger
    AFTER INSERT OR UPDATE OR DELETE ON map.traffic
      FOR EACH ROW EXECUTE PROCEDURE map.traffic_history_trigger();
    --- end traffic


     -- Peaks
     ----------------
    CREATE OR REPLACE FUNCTION map.peaks_history_trigger() RETURNS trigger AS
        $$
            BEGIN
                IF TG_OP = 'INSERT' THEN
                    INSERT INTO history.peaks_history (node_id, fid, name, name_en,
                        ele, ts, geom, created_by)
                        VALUES (
                                NEW.node_id,
                                NEW.fid,
                                NEW.name,
                                NEW.name_en,
                                NEW.ele,
                                current_timestamp,
                                NEW.geom,
                                current_user
                            );
                        RETURN NEW;
                ELSIF TG_OP = 'UPDATE' THEN
                    INSERT INTO history.peaks_history (node_id, fid, name, name_en,
                        ele, ts, geom, created_by)
```

```
                        VALUES (
                                NEW.node_id,
                                NEW.fid,
                                NEW.name,
                                NEW.name_en,
                                NEW.ele,
                                current_timestamp,
                                NEW.geom,
                                current_user
                            );
                        RETURN NEW;
                ELSIF TG_OP = 'DELETE' THEN
                    INSERT INTO history.peaks_history (node_id, fid, name, name_en,
                        ele, ts, geom, deleted_by)
                        VALUES (
                                OLD.node_id,
                                OLD.fid,
                                OLD.name,
                                OLD.name_en,
                                OLD.ele,
                                current_timestamp,
                                OLD.geom,
                                current_user
                            );
                        RETURN OLD;
                END IF;
            END;
        $$
        LANGUAGE 'plpgsql';

    CREATE OR REPLACE TRIGGER peaks_history_trigger
    AFTER INSERT OR UPDATE OR DELETE ON map.peaks
      FOR EACH ROW EXECUTE PROCEDURE map.peaks_history_trigger();
    -- end Peaks

     -- Natural
     ----------------
    CREATE OR REPLACE FUNCTION map.natural_history_trigger() RETURNS trigger AS
        $$
            BEGIN
                IF TG_OP = 'INSERT' THEN
                    INSERT INTO history.natural_history (area_id, fid, name, type, ts
                        , geom, created_by)
                        VALUES (
                                NEW.area_id,
                                NEW.fid,
                                NEW.name,
                                NEW.type,
                                current_timestamp,
                                NEW.geom,
                                current_user
                            );
                        RETURN NEW;
                ELSIF TG_OP = 'UPDATE' THEN
                    INSERT INTO history.natural_history (area_id, fid, name, type, ts
                        , geom, created_by)
```

```
                        VALUES (
                                NEW.area_id,
                                NEW.fid,
                                NEW.name,
                                NEW.type,
                                current_timestamp,
                                NEW.geom,
                                current_user
                            );
                        RETURN NEW;
                ELSIF TG_OP = 'DELETE' THEN
                    INSERT INTO history.natural_history (area_id, fid, name, type, ts
                        , geom, deleted_by)
                        VALUES (
                                OLD.area_id,
                                OLD.fid,
                                OLD.name,
                                OLD.type,
                                current_timestamp,
                                OLD.geom,
                                current_user
                            );
                        RETURN OLD;
                END IF;
            END;
        $$
        LANGUAGE 'plpgsql';

    CREATE OR REPLACE TRIGGER natural_history_trigger
    AFTER INSERT OR UPDATE OR DELETE ON map.natural
      FOR EACH ROW EXECUTE PROCEDURE map.natural_history_trigger();
      -- end natural
```