INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Interactive Air Pollution Mapping Tool for Experts

Duarte Vaz Correia Vital

Mestrado em Engenharia Informática

Orientador:
Doutor Pedro Figueiredo Santana, Professor Auxiliar, ISCTE-IUL

Coorientador:
Doutor Pedro Mariano, IST-ID

Dezembro, 2020

Acknowledgements

While it is solely my name that is defined as the author of this dissertation, to some extent, this was a collaborative effort. For this reason I would like to acknowledge everyone who had a hand in the making of this project.

I would like to acknowledge Pedro Santana and Pedro Mariano, whom I am deeply grateful for helping to make this project a reality; for always pointing me in the right direction; and for continuously pushing me to do better.

I would like to recognise how crucial the help of Prof. Marta Almeida from CTN-IST, along with all anonymous participants, was during the testing phase. For that, thank you.

I would like to thank my girlfriend, Catarina, for the unwavering support and availability and also for managing to keep me sane in the most stressful moments.

And last, but certainly not least, I would like to thank my family for the unconditional love and support. Particularly, my parents, for being excellent role models and inspiring me to be as hard-working as them.

This work has been funded by FCT - Foundation for Science and Technology, I.P., within the framework of the project ExpoLIS (LISBOA-01-0145-FEDER-032088).

Abstract

Air pollution, especially in city areas, has been a big concern for decades now. Real-time air quality monitoring stations have become the standard in measuring air pollution for their accuracy and reliability. As a result, extensive pollution maps are nowadays created mainly using information from these stations. Two types of pollution mapping solutions are most prominent: maps that display sparse monitoring stations' locations and respective gathered time varying air pollution data; and estimated dense pollution heatmaps resulting from a combination of air quality sensor data and additional data, such as meteorological and traffic information. In alternative, this dissertation proposes the use of expert knowledge as a complementary means for generating air quality maps. The goal is to allow experts to express their knowledge about how air pollution is emitted and diffused as a function of the presence of key topological elements, such as buildings and roads. To this end, a tool was developed and validated with a set of 30 participants. The obtained results confirm the tool's usability and highlight key future research directions to bring the proposed concept closer to a fully functional solution.

Keywords: Air Quality Monitoring, Human-Machine Interface, Pollution Representation, Expert Knowledge, Interface Usability.

Resumo

Há décadas que a poluição do ar, especialmente em zona citadinas, tem sido uma grande preocupação. Estações de monitorização da qualidade do ar em tempo real tornaram-se o padrão para obter medições de poluição no ar devido à sua precisão e confiabilidade. Como resultado, atualmente, mapas de poluição extensivos são criados com o uso de informação recolhida destas estações. Dois tipos de soluções para mapeamento de poluição são proeminentes: mapas que exibem localizações de estações de monitorização e respetivos dados de poluição do ar; e mapas de calor de poluição estimados a partir de uma combinação de dados recolhidos de sensores de qualidade do ar e dados adicionais, como informação meteorológica e de tráfego. Em alternativa, esta dissertação propõe o uso de conhecimento de peritos como uma forma complementar de geração de mapas de qualidade do ar. O objetivo é permitir a partilha de conhecimento de peritos acerca de como a poluição do ar é emitida e difundida em função da presença de elementos topológicos chave, como edifícios e estradas. Para este fim, uma ferramenta foi desenvolvida e testada com um conjunto de 30 participantes. Os resultados obtidos confirmam a usabilidade da ferramenta e realçam futuras direções de investigação para aproximar o conceito proposto de uma solução completa e funcional.

Palavras-chave: Monitorização da Qualidade do Ar, Interface Pessoa-Máquina, Representação de Poluição, Conhecimento de Peritos, Usabilidade de Interface

-

Contents

List of Figures

List of Tables

Glossary

ADS – Abstraction-Decomposition Space

AH – Abstraction Hierarchy

API – Application Programming Interface

AQI – Air Quality Index

BAQI – BreezoMeter's Global Air Quality Index

CO – Carbon Monoxide

CSS – Cascading Style Sheets

CSV – Comma Separated Value

CTA – Control Task Analysis

CWA – Cognitive Work Analysis

EPA – Environmental Protection Agency

HTML – Hypertext Markup Language

IPS – Information-Processing Activities

LCS – Low Cost Sensor

NO_2 – Nitrogen Dioxide

NAQI – National Air Quality Index

NPM – Node Package Manager

O_3 – Ozone

OSM – OpenStreetMap

PM_1 – Atmospheric Particulate Matter inferior to 1 micrometre

PM_{10} – Atmospheric Particulate Matter inferior to 10 micrometres

$PM_{2.5}$ – Atmospheric Particulate Matter inferior to 2.5 micrometres

RGB – Red Green Blue

SO_2 – Sulphur Dioxide

SUS – System Usability Scale

VOC – Volatile Organic Compounds

WAQI – Word Air Quality Index

WCA – Worker Competencies Analysis

WDA – Work Domain Analysis

Chapter 1

Introduction

1.1 Motivation and Problem Definition

Currently, various tools/applications are able to create and continuously update pollution maps. Some of these tools present general pollution values, while others even show individual pollutants values. Either way, this information is collected directly from air quality monitoring stations and then, in most cases, converted to Air Quality Index (AQI) (Agency & Division, 2014). However, many countries have their own AQI and some companies are known to use their own index as well. This poses a problem, namely, lack of data standardization, which means that comparisons between field tests and different studies can prove to be difficult and may lead to misleading conclusions.

A second problem the aforementioned applications face, and arguably the most important one, is the lack of available information. WAQI (waqi.info, n.d.) presents a map containing information of over 10,000 monitoring stations across the world, which, nevertheless, is insufficient to cover every block of typical large cities This relatively low number derives from the fact that only monitors that report on fine particulate matter ($PM_{2.5}$) are accepted. When it comes to air quality monitors, QualAr (QualAr, n.d.) reports information exclusively from Portugal, but a quick look at their map shows a significant increase in the number of installed monitors, since $PM_{2.5}$ sensors are not a requirement. This being said, the aforementioned problem still stands since areas of numerous squared kilometres are currently unmonitored in Lisbon alone (see Figure 1.1). Consequently, air quality assessment in these areas become but an estimate, if available at all.

Air quality monitors are the main source of information in this subject thanks to their accuracy and reliability, qualities that cannot be matched by humans. However, by enabling environmental experts to express their knowledge on the subject by associating entities in a map and expected air pollution emission and diffusion, the tool developed in this dissertation intends to provide a new source of information that could be a complement to existing air quality monitoring networks and an alternative to them in unmonitored areas.

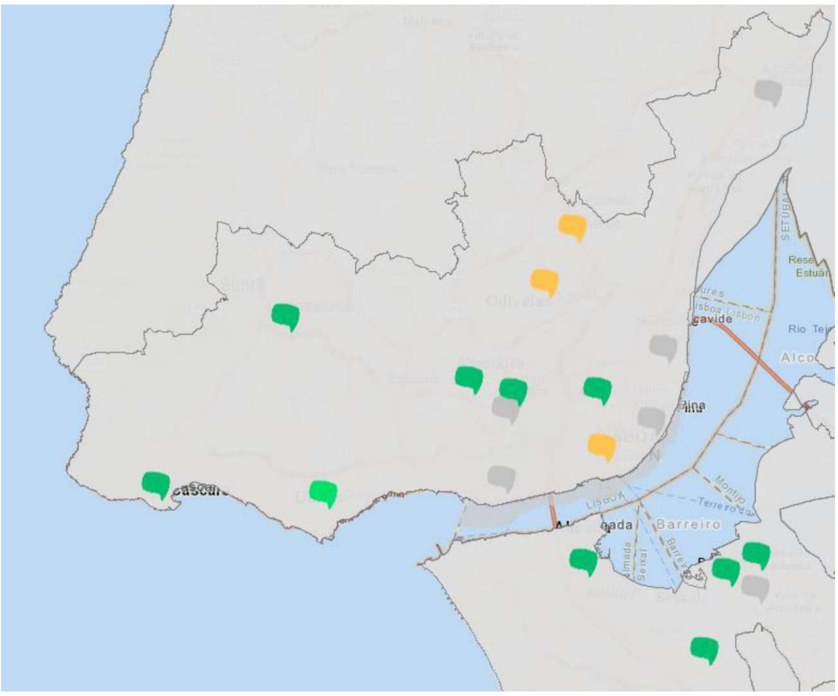

Figure 1.1: Monitored Lisbon according to QualAr (QualAr, n.d.). A proprietary index with its own colours is used. Light green represents the best scenario, while colour approaching red (like orange) represents higher pollution values. Some monitors are greyed out, because at the given time not enough data was gathered in order to do calculations.

1.2 Goals and Research Questions

Given the problem defined previously, the main goal of this study is to develop a tool that enables environmental experts to express their knowledge on air pollution through an intuitive interface.

To achieve this goal, three questions must be answered:

1. How can environmental experts express their knowledge in a way that this knowledge can be used for the generation of city-wise pollution maps?
3. How accurate the expressed expert knowledge is on predicting air quality, when compared to *in-situ* sensors?
4. How should expert knowledge be visually represented through a graphical user interface?

1.3 Approach

In order to get an intuitive and efficient design, literature survey should be made regarding other interfaces of the same nature. As such, searching for map-based applications began, to get a general idea of what the interface should look like. The focus of this survey went beyond user interface though, it was also important to determine which were the most used sources and APIs to render and manipulate maps, particularly open-source ones.

By surveying state-of-the-art applications, an API was chosen, and consequently programming languages, multiple mock-up designs were drawn, and some hypothetical use case scenarios were informally tested to evaluate every components' purpose and placement. After establishing the groundwork, a desktop application was developed to render an interactable open-source map and display a simple interface that allows the selection and pollution associating individual map entities.

At this point the application was working, but its effectiveness was yet to be tested. As such, a series of usability tests were conducted by giving participants a set of hypothetical scenarios to be completed in the app, as well as a questionnaire to further test the interface's usability and the introduced approach's viability

1.4 Document Structure

In this current chapter, the developed project in view of this dissertation was introduced, which will be further detailed throughout this document, presented the research questions to explore and the approach taken in this work. The remaining of this document is structured as follows:

- In Chapter 2 relevant work to this project is discussed. Starting with the chosen air quality index, followed by detailed descriptions of existing pollution maps and, finally, currently available and chosen map rendering tools.
- Chapter 3 is where thorough descriptions of the application's interface and functionalities can be found.
- In Chapter 4 the development process is described, from preliminary work, like requirements gathering and design planning, to individual functionality implementation.

- In Chapter 5 the testing process is detailed, and respective results presented and analysed.
- Chapter 6 contained all concluding thoughts and potential future work based on this dissertation.

Chapter 2

Related Work

In the developed application within the scope of this thesis, users are allowed to associate pollution values, based on the Air Quality Index (AQI), to individual, or groups, of entities in a map (e.g., buildings, roads). Accordingly, this chapter will start with a brief explanation of what AQI exactly is, how it is calculated and some alternative indexes, as all these points will be relevant when discussing existing pollution mapping solutions afterwards.

2.1 Air Quality Index

Air pollution measurements are commonly used to obtain the level of severity of pollution in a given area. The United States (US) version of the Air Quality Index (AQI) (Agency & Division, 2014), developed by the Environmental Protection Agency (EPA), is amongst the most widely used. It converts pollutant concentrations into a color-coded scale, where higher values indicate increased health risk (Karuppasamy et al., 2020), as depicted in Table 2.1.

Data used to calculate this AQI is often collected via monitoring stations that return values for different pollutants. Typical pollutants being ground-level ozone (O3), coarse particulate matter (PM_{10}), fine particulate matter ($PM_{2.5}$), carbon monoxide (CO), sulphur dioxide (SO2), and nitrogen dioxide (NO2).
To obtain the US's version of AQI, based on the listed pollutants, the following formula must be applied on each of them, the highest value being reported as the final AQI:

$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}} (C - C_{low}) + I_{low},$$

Equation 2.1: Formula used to obtain an AQI value

where:

I stands for (Air Quality) index,

C stands for pollutant concentration,

C_{low} stands for concentration breakpoint that is $\leq C$,

C_{high} stands for concentration breakpoint that is $\geq C$,

I_{low} stands for index breakpoint corresponding to C_{low} ,

I_{high} stands for index breakpoint corresponding to C_{high}.

Table 2.1: Table representing every level of the AQI, each with a meaning, range of values and colour.

Health concern level	Good	Moderate	Unhealthy for sensitive groups	Unhealthy	Very unhealthy	Hazardous
Numerical value	0-50	51-100	101-150	151-200	201-300	301-500
Colour						

A problematic issue is that not all countries use this index. For instance, India uses its own National Air Quality Index (NAQI) (Kumar & Goyal, 2011; Selvam et al., 2020), which returns more optimistic results than US's AQI. NAQI applies the same formula to monitored pollutants, but in this case, the lowest result is reported as the AQI for the monitored location, therefore better air quality is returned. There are other differences between indexes. In particular, not all of them report on the same pollutants. While the US AQI reports on the six aforementioned pollutants, Denmark's' index only reports on NO2, CO, and O3 (Friedman, n.d.). This lack of standardization may difficult comparisons between indexes of different countries.

Since many do not approve the simplistic approach regarding how AQI is calculated, many studies have been conducted to create alternatives (Sowlat, Gharibi, Yunesian, Tayefeh Mahmoudi, & Lotfi, 2011; Stieb et al., 2008). However, to our knowledge, most of the current pollution mapping solutions use AQI to present results and some use their own proprietary indexes (e.g. QualAr). This being said, alternative air quality indexes will not be addressed here, unless they prove relevant in upcoming solutions.

2.2 Pollution Maps

2.2.1 World Air Quality Index

The World Air Quality Index (WAQI) project is a non-profit organization that collects data from more than 30,000 monitoring stations in 2000 major cities. These data contain information on individual pollutants concentrations and, in some cases weather forecasts. However, not all gathered data is published. Out of all known stations, only about 12,000 are published. The reason behind this decision is consistency. To keep a consistent set a of results throughout every station, only the ones that are able to obtain particulate matter ($PM_{2.5}/PM_{10}$) readings are published.

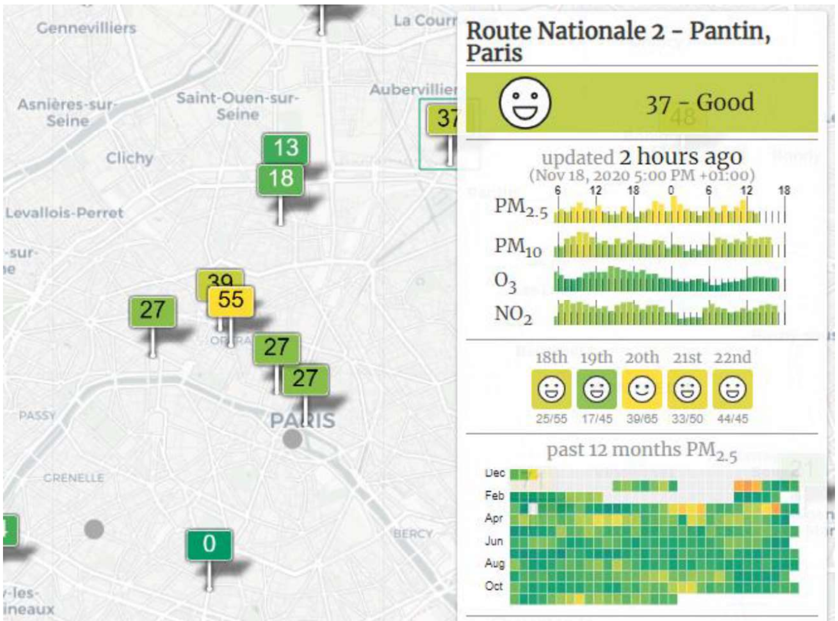

Figure 2.1: Map located at Paris, France, displaying all published monitors. One of these monitors is selected, hence the window information at the right. Map provided by WAQI.

WAQI presents a world map containing every accepted monitoring station accompanied by an AQI value. If one clicks on a station all of its information is displayed followed by when it was last updated. The sensors inside the accepted monitoring stations are very precise at their exact location, but as demonstrated in Figure 2.1, there may considerably wide spaces in between stations, making it difficult, if not impossible, to get very accurate and reliable measurements where no station is available (Helen, n.d.).

2.2.2 BreezoMeter

BreezoMeter (Breezometer, n.d.) is currently, arguably and to our knowledge the most complete pollution mapping application. It collects data from monitoring stations and adds layers of information from satellites, weather sources, traffic reports and even active fire reports, therefore resulting in over 1.8 Terabytes of validated data per hour.

Air quality is dynamic and, in a world where people are always moving around, monitoring stations alone become hardly useful because of its stationary nature. BreezoMeter aimed to combat this issue. By processing all collected data using proprietary algorithms, machine learning techniques, big data analytics, and air pollution dispersion modelling, they claim to be able to provide block level reliable air quality estimates. Regarding accuracy, they also claim that 90% of estimates are correct, because of their usage of multiple layers of (reliable) information and a continuous cross-validation via the Leave One Out Method (Friedman, 2018). The result is a world heatmap representing each location's air quality that, as claimed, goes all the way to block level. By clicking a region, a box is displayed showing every monitored pollutants' concentration, BAQI (BreezoMeter's Global Air Quality Index) and the local AQI, health tips and a forecast for the next five hours.

However, black-box machine learning models should not be entirely trusted since mistakes are always a possible reality. Rudin (Rudin, 2019) exemplifies this by mentioning an incident during the 2018 California wildfires, where Google, using BreezoMeter, reported air quality in this region as "good" when, in reality, it was dangerously bad. Regardless, to our knowledge, this was a one-time incident and should not be considered standard behaviour.

It is worth noting that BreezoMeter also has its own proprietary index, BAQI, whose colour coding is demonstrated in Figure 2.1. However, they also include each country's local AQI, if it exists, when checking a location.

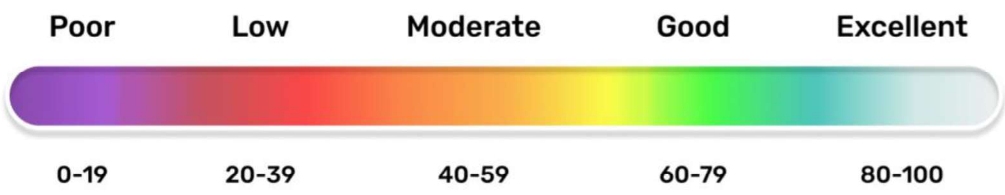

Figure 2.2: BreezoMeter's BAQI colour code.

Figure 2.3: BreezoMeter's heatmap in some scenarios. In a) Lisbon is shown, where more dense city areas produce more pollution. b) displays a really good pollution map from London, although, seemingly, a bit incomplete in some areas. c) is located at Miami, US, and its heatmap is somewhat different. This leads to the suspicion that, due to limited monitoring (as shown in waqi.info) a lot more estimations were made to produce the heatmap, leading to less specific heat points. As for d), in Havana, Cuba, no heatmap is available, despite available satellite imagery. This leads us to conclude that monitoring stations are a depended upon to create the heatmap.

Regardless of all layered information used by BreezoMeter, monitoring stations are still very much relied upon. Without these stations, an air quality estimate can hardly be calculated, even if every other layer contains information. For example, BreezoMeter does not contain information on Havana, Cuba, because there are no known air quality stations, even though there is satellite imagery of this city, as well as traffic data and weather forecasts (see Figure 2.2). Note that the first two types of data are provided by Mapbox, which is the same map tile provider used in the tool developed in this dissertation. This leads to the main issue with current solutions in pollution mapping, the dependence on installed air quality monitors. Due to this limitation, whole cities and event countries are without any kind of air quality monitors. The tool introduced in this project aims at introducing a new source of information to generate dense pollution estimates, potentially reducing dependence of monitoring stations.

2.2.3 PlumeLabs

PlumeLabs (PlumeLabs, n.d.-b) is a France based company specialized in providing air quality information to the public. They have three main relevant focuses to this work: world pollution heatmap, street-by-street map built by using low-cost air quality sensors, and published research.

A quick look in PlumeLabs' world heatmap (see Figure 2.3) reveals that it covers most of the world, even places where no other solution is able to collect air quality information due to the absence of air quality monitors. To build this heatmap, data is collected from multiple sources. Air quality monitoring stations are still the main and most important source as they are the most accurate, over 10.000 stations are considered. Weather forecasts, anthropogenic emissions datasets, like real-time traffic reports, land classification, population density information and atmospheric models are also used to determine air quality. These latter models are developed by scientific agencies across the world and are able to predict air quality in wide scales like entire countries and even world scale by relying on information regarding: pollutant emitters, for instance, during rush hour in major cities, heavy traffic produces dangerously high levels of CO concentration or, to give another example, a nearby large power plant probably increases SO2 and NO2 concentrations (Popescu & Ionel, 2010); pollutant transport models, developed in order to stay a step ahead of pollutants by predicting how far and in which direction they will travel while considering certain weather conditions, like wind; chemical reaction models, which may be used to predict how certain chemical combinations affect pollutant concentrations.

Unlike BreezoMeter, the creation of the heatmap does not seem to require data from monitoring stations, using this data only when possible to get the most accurate results. It seems this way because no other solution, application or source was able to collect monitored pollution data from, for example, the entire continent of Africa. This leads us to, somewhat, doubt the heatmap's reliability in certain parts of the world, especially since, to our knowledge, no studies were conducted to test this matter.

PlumeLabs also provides another type of map. Some cities are supported in their street-by-street map (see Figure 2.4). This is a fairly ambitious project that is not yet complete but shows a lot of promise. The idea is to create pollution maps at a street level detail by colouring roads by section based on AQI values (see Table 2.1). It uses all of the same sources as their world heatmap to develop pollution models. All these data are then processed by machine learning to determine AQI levels in non-monitored places in between monitoring stations.

Their goal is to create "the Waze of air pollution - a hyper-local, community-powered map of the air in your life" (PlumeLabs, n.d.-a), which is intended to be achieved by combining previously mentioned air quality assessments with collected data from mobile low-cost air quality sensors. *Flow* is a personal and wearable air pollution sensor provided by PlumeLabs. Allegedly, the tool is able to measure concentrations of multiple pollutants, namely PM1, $PM_{2.5}$, PM_{10}, NO2 and volatile organic compounds (VOCs) every 60 seconds. The device has

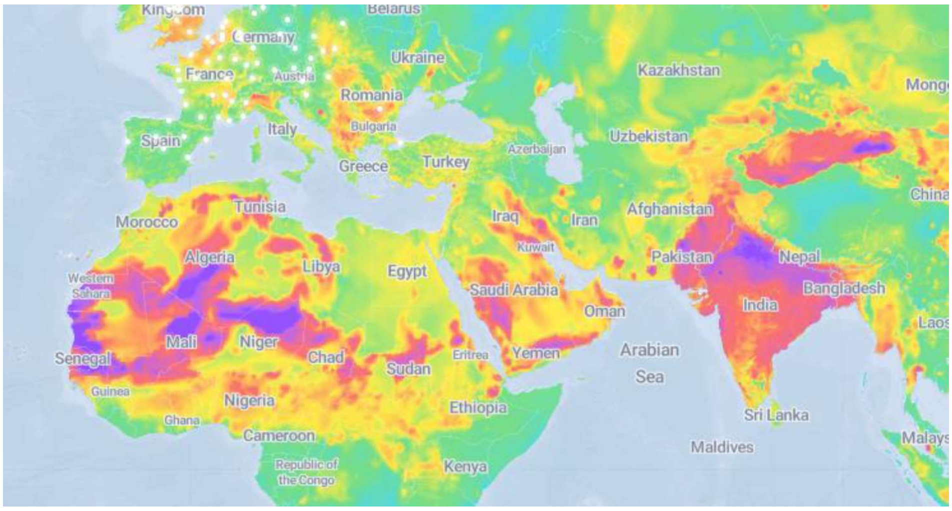

Figure 2.4: PlumeLabs's world-scale pollution heatmap.

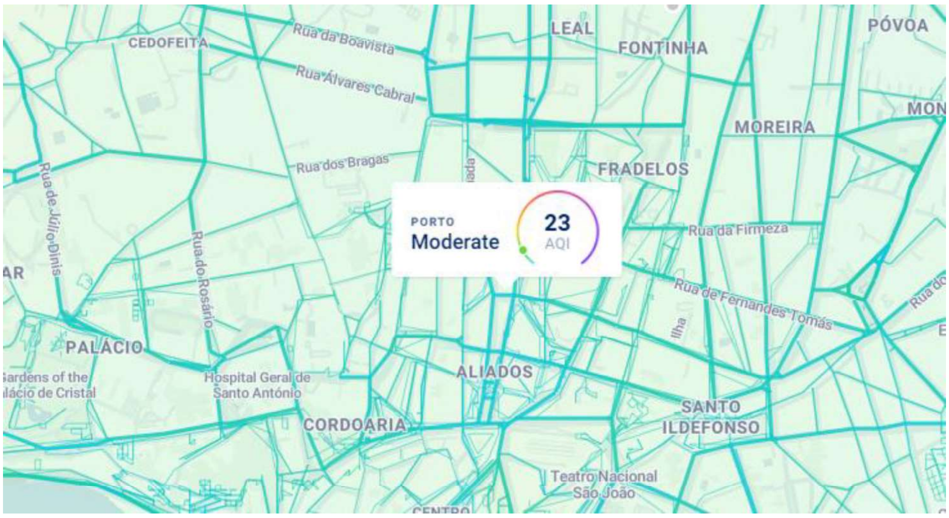

Figure 2.5: PlumeLabs's street-by-street map located in Porto, Portugal.

two main tasks: firstly, update the companion application with new measurements for the user to see; and secondly, to send these data, as well as the devices' location to PlumeLabs' databases and help create crowdsourced street-by-street maps.

By offering an air quality monitor at under 200 euros, the intention is to create a whole community of users providing valuable data to extend the range and accuracy of street maps. Obviously, there are drawbacks to Low Cost Sensors (LCS) when compared to traditional official monitoring stations, usually priced at several tens of thousands of euros, which are mainly related to accuracy and reliability. Recent studies have shown that results are not consistent between pollutants and different meteorological conditions, but acceptable nonetheless showing high correlation when compared to official monitors (Castell et al., 2017; Munir, Mayfield, Coca, Jubb, & Osammor, 2019). Despite some discrepancies, Flow pollutant correlations seem a lot closer to each other, averaging at around 91% (Morawska et al., 2018; PlumeLabs, 2019). Morawska et al. conclude that LCSs are capable of supplementing data to monitoring networks. Furthermore, (Cassard, Jauvion, & Lissmyr, 2020) put this hypothesis to test. In this case study, data from thousands of official monitoring stations and LCSs was collected to build a pollution prediction engine. Results show that, despite being less accurate, LCSs can be used in large scale networks to improve accuracy. Furthermore, they tested two prediction models containing only official monitors stations and LCSs, respectively, and results showed that the latter model was the better performer, leading to the conclusion that the denser a LCS network is, the better results it produces.

Currently, street maps cover hundreds of big cities, which is still very incomplete considering how many are left, as well as smaller ones. However, it does show promise, assuming more and more people acquire and use Flow. As for accuracy, further third-party studies should be conducted in order to test it.

2.2.4 AirVisual and AirVisual Earth

IQAir (IQAir, n.d.) provides pollution information via two pollution maps, while also providing multiple air quality related physical products, although only one of these is relevant to this work.

IQAir Earth is IQAir's 3D rendered pollution map of the planet, which displays detailed representations of wind and values regarding wind direction and force at individual selected points (see Figure 2.5). It also displays $PM_{2.5}$, PM_{10} or wind readings separately heat-mapped with AQI colours. The latter tends to show stronger colours in places where wind force is higher. The PM_{10} heatmap appears to be severely incomplete as the pollutant is barely represented in several places, like the United States or the entire continent of Europe, which is further demonstrated by not even being mentioned in the article introducing IQAir Earth in their website and confirmed in (Parkin, n.d.). $PM_{2.5}$ readings seem to be the map's focus as it is a lot more complete, but not completely though.

IQAir Map is the name of their 2D map. This map displays an AQI heatmap, location of used official monitoring stations, active fires and wind data. Besides the wind animations and a few

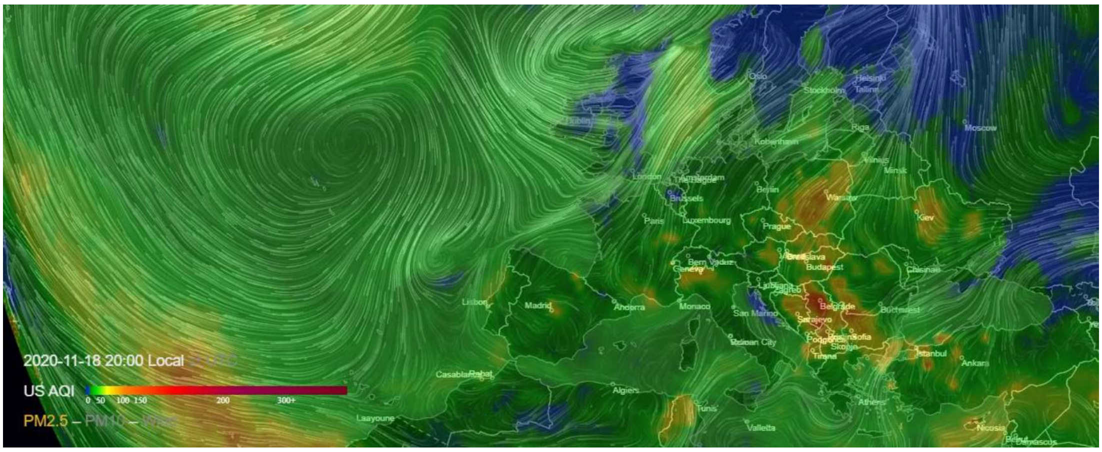

Figure 2.6: IQAir Earth map positioned over Europe. In this image, the $PM_{2.5}$ heatmap is enabled, however PM_{10} or wind respective heatmaps can be toggled at the bottom right.

a)

b)

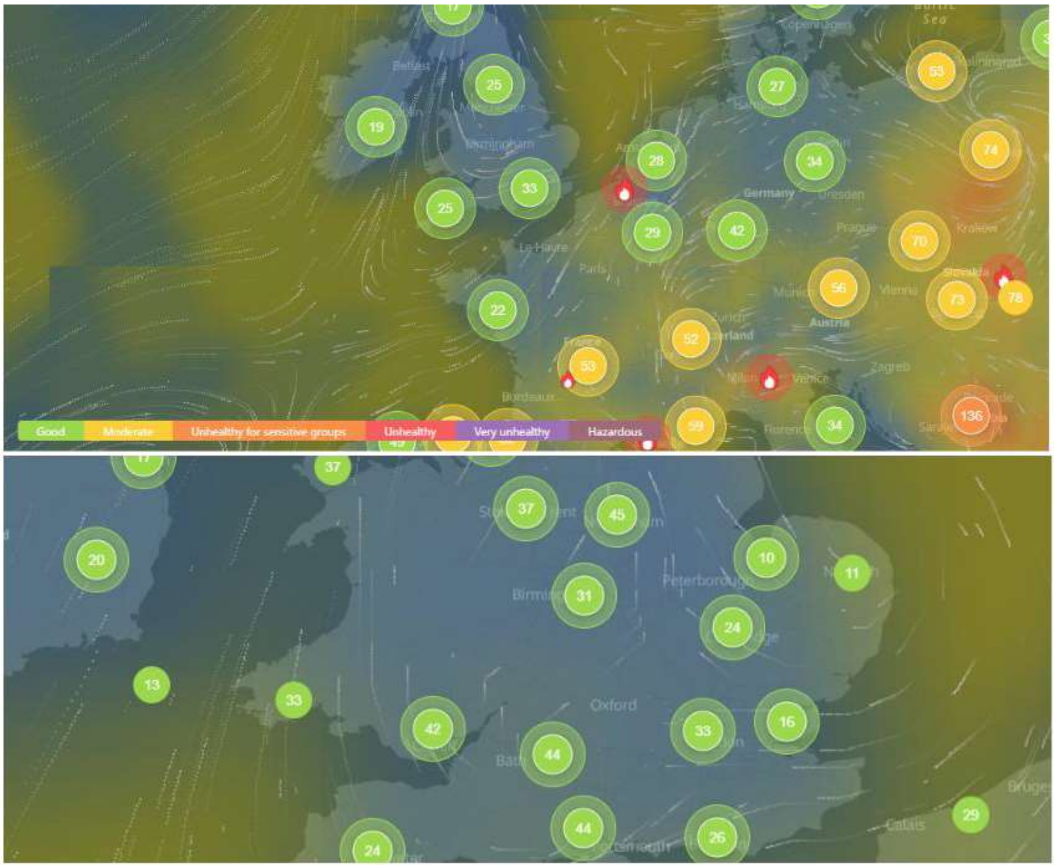

Figure 2.7: IQAir Map. In a) most of Europe is visible, along with monitor data and active fires. As we zoom in, b), more monitors become visible. Also, it is clearly visible in both cases that big land areas are not overlaid by a heatmap.

styling choices, not many differences were found in relation to previously mentioned maps. However, it is worth mentioning that the current state of the displayed heatmap and data sources, seem to be very incomplete.

Case in point, entire countries that are monitored, even according to the standard map, do not display pollution heat. Let us take England, for example depicted in Figure 2.6. Multiple monitors can be seen on the map with AQI values associated, as well as active fires and wind data. However, the heatmap barely reaches the land. As for the reason for this, it is unclear. IQAir's website is incomplete when it comes to information on how their map data is used and heatmaps created.

As for data sources, the air quality data comes from monitoring stations (official and private) and satellite imagery modelling. Details on the latter were not found, however monitoring stations data is somewhat different from other solutions. IQAir provides its own LCS to the public, called the AirVisual Pro and it measures $PM_{2.5}$ and CO2 concentrations, as well as temperature and humidity. Data from these LCS are displayed alongside official monitors in the map, making the users' contributions. Furthermore, for each of monitored area, a list of contributors (official or private) is available.

2.3 Map Rendering APIs

The first step in the development process for this tool was to choose an API to build on. But before that, a few conditions were set for an API to be eligible, so as to facilitate the choice: user interactivity, customizable style, and multiple map providers. This filtered available choices to three popular libraries: Mapbox GL JS, OpenLayers, and Leaflet. All of which use data from OpenStreetMap (OSM), which is a community-driven project that focuses on creating and maintaining open-source geodata (Haklay & Weber, 2008).

The first condition has to do with which of the two most commonly used map rendering technologies should be used, these being raster tiles and vector tiles. While the former downloads tiles at a fixed resolution and styling, the latter downloads the tiles' vector geometries from the server and then styling and processing are done at client-side (Zunino et al., 2020). (Netek, Masopust, Pavlicek, & Pechanec, 2020) conducted a study that thoroughly compared the two with performance in mind. The first point they make is related to interactivity. The raster tiles method is based on customizing and styling at server-side and then downloading everything, therefore interactivity is negatively affected, since each time a change is made to the dataset, either by panning or changing a style parameter, the whole tile generation process must be redone. Contrarily, the vector tiles method does all the rendering and styling client-side, which means that changes can be made to the dataset without having to recall assets. This, combined with reduced server loads, brings to the conclusion that the vector tiles method is the most adequate for this work, since entity selection, dynamic styling, and other interactions are integral parts of the tool.

Table 2.2: Considered criteria when choosing between map rendering methods.

Criteria	Vector	Raster
Dynamic map customization	✔	✖
Reduced server loads	✔	✖
Less hardware intensive	✖	✔

Since vector tiles support is relevant for this work, it was based on this the choice in libraries was made. Leaflet is the less appealing solution for two reasons: it doesn't support vector tiles natively, requiring the external plugin VectorGrid (Leaflet, n.d.); and, even with the plugin installed, it wouldn't be the right choice due to its larger network bandwidth used and overall lower performance. Mapbox GL proved to be the overall best in terms of performance compared to OpenLayers, although by a small margin. Furthermore, Mapbox GL implementation and documentation were found to be to be more intuitive than alternatives.

2.3.1 Mapbox GL JS

Mapbox GL has multiple variants, of which the JavaScript (JS) API (Mapbox GL JS, n.d.-c) was used for being the more complete in terms of documentation, examples and community, as well as to facilitate design and styling with HTML and CSS. Mapbox GL JS is a client-side library that provides tools for building and customizing highly interactive maps. More details about the API will be discussed by parts and addressed one by one.

- Styles. Official information on map style may be confusing as there are multiple ways to implement it, however it essentially is what the base map looks like, before any (more) styling, or adding data sources, or layers (more on these later). Among others, available predefined styles are "streets", "dark", "satellite" and "satellite-streets". One can also design a map style from scratch using Mapbox Studio, however this feature was not used in this dissertation.
- Data sources and layers. Additional visual components can be added to the map, be them points, lines, polygons or many more types. For a layer to work, a data source must be

defined. Mapbox provides numerous different sources, namely "mapbox-streets-v8" (“v8” referring to the latest version) that provides more detailed information on individual entities in a map. Regardless, custom data sources can very well be included as well, enabling third party data to be directly inserted into the map.

- Plugins, which need to be separately installed to be used, are a good way to add some extra functionality and interactivity. The draw plugin (Mapbox GL JS, n.d.-b), for instance, enables polygon and line drawing as well as useful listener functions, for example, when drawing ends or a drew polygon is deleted while the directions plugin, as the name suggests, enables a navigation functionality for several scenarios like driving, cycling or walking.

2.3.2 Consumer Applications

Mapbox became a very powerful and popular tool over the years since its foundation in 2013. Naturally, many big companies' attention was caught, and nowadays multiple consumer applications integrated the Mapbox APIs have been released to the public (Mapbox GL JS, n.d.-a).

Snap Inc, the company behind Snapchat, the social app that lets users share pictures or video clips for 24 hours, developed Snap Map. Created with Mapbox, this application represents user activity with a heatmap in a world map. The more users that show activity in a general location, the hotter is the heatmap. By clicking in a place where some heat exists, a published video-clip published on this location pops up. A couple aspects were found to be interesting for this work: data injection in a layer, as mentioned previously, for instance user activity data being integrated in the heatmap layer; and the map’s query-able aspect, in other words, the fact that a query function over a certain area is executed on click, providing detailed information on all the layers in a location.

AccuWeather is a vastly popular weather information provider. Using Mapbox GL, they now also provide several maps with individual purposes. Their “Radar” map presents four different heatmaps simultaneously (rain, snow, ice, and mix), each with a different set of colours. The “Satellite” map does not have traditional satellite imagery, instead it presents clouds as seen from satellites but coloured to resemble a heatmap, adding stronger colours where more clouds are present. The other (relevant) map is the “Current Conditions” map, where the world is covered by a heatmap created using temperature values. Additionally, individual points are added to the map that show the exact temperature at thousands of locations.

One found peculiarity was that each time a switch between maps was made the whole page was refreshed. Since Mapbox GL support dynamic changes to style and entire layers, it is odd that this functionality is not being used.

2.4 Image Annotation and Segmentation

The basic nature of this tool, and of OSM's edit mode, revolves around image annotation, with the capability of describing objects in an image at a semantic level using labels. Image annotation has many applications, the most popular being building large datasets for later use in supervised machine learning (Weston, Bengio, & Usunier, 2011). Depending on the end goal, labels provide information on fundamental properties of an object. For instance, LabelMe (Russell, Torralba, Murphy, & Freeman, 2005) provides a web-based tool for easy labelling of images as well as a set of 10,000 images, 7,000 of which still to be annotated, and label information should include object classes, shape, locations and other relevant labels. The main functionality of this tool has users drawing a polygon around a targeted object and attaching a label to it. It also allows for a scribble-based selection that consists of drawing rudimentary scribbles on an object, whose boundaries are then estimated by the tool. It is unclear on what model they use for object detection in scribble mode, however the results are very lacking in accuracy. To be fair, this tool was developed in the early 2000's and in the meantime other well implemented solutions were developed (Wu, Zhao, Zhu, Luo, & Tu, 2014; Zemene, E., & Pelillo, 2016).

In the past two decades, plenty of research was conducted on the topic of semi-automatic and automatic image segmentation (Bhagat & Choudhary, 2018; Cheng, Zhang, Fu, Tu, & Li, 2018; Makadia, Pavlovic, & Kumar, 2008). To our knowledge, no publicly available products exist using the latter type of segmentation, however, semi-automatic approaches seem to be available to some extent, such as in Microsoft's Paint3D or Adobe Photoshop's MagicWand. Using graph-cut optimization technique (Boykov, 2001), GrabCut (Rother, Kolmogorov, & Blake, 2004) popularized semi-automatic image segmentation, outside the realm of drawing minimum cost contours, like Intelligent Scissors (Mortensen & Barrett, 1995). The proposed approach of GrabCut has the user setting a bounding box around the object in question by drawing a rectangle, after which a rough segmentation is automatically initiated to retrieve said object from the scene. The final interactive step is to draw a few scribbles in order to obtain the desired result, as demonstrated in Figure 2.7. Many studies were conducted using GrabCut as a basis, for instance, (Maninis, Caelles, Pont-Tuset, & Van Gool, 2018; Papadopoulos, Uijlings, Keller, & Ferrari, 2017) present similar approaches that would require the annotator to click on a objects' four extreme points, these being the left-most, right-most, top and bottom points, therefore creating a bounding box in which GrabCut is implemented. Results have shown that the extreme clicking method is much quicker than traditional ways, while retaining equal quality.

One of the main functionalities of our tool is to allow users to manually draw polygons to represent entities. However, the drawing process is very simple, only requiring positioning of vertexes to draw a polygon. Here a semi-automatic approach would be useful to accelerate the drawing process. Unfortunately, we were unable to find any open-source tools that allowed for a quicker segmentation of entities. This said, a manual implementation of existing solutions would be required, such as GrabCut, in order to have this functionality. However, open-source satellite imagery tends to have low resolutions which would cause the tool to produce inaccurate results. Perhaps further efforts should be made in future iterations to research and implement reliable semi-automatic approaches for this tool.

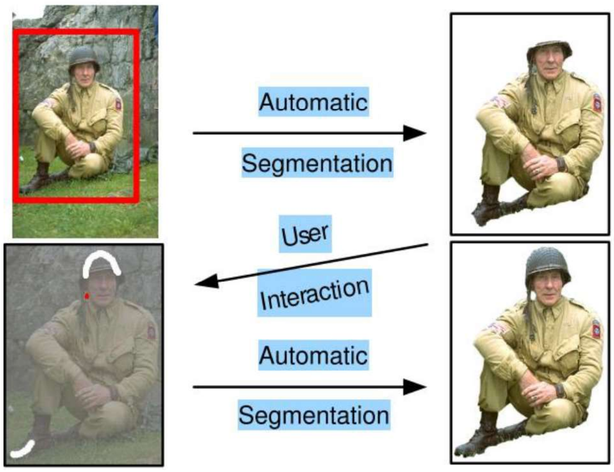

Figure 2.8: Example of GrabCut's object segmentation process. Image taken from (Rother et al., 2004).

Chapter 3

Application Description

Simplicity was a key aspect when it came to the design of the interface. The user is supposed to be able to easily take advantage of every available functionality without much thought. This proved to be a difficult task, as some of the activities that are introduced in this application are, to the best of our knowledge, novel in the industry. Many solutions exist that provide detailed information on air pollution, be it through monitoring stations alone or by mixing data from these with algorithmic estimates. Although these solutions offer some level of interactivity when it comes to choosing which type of information to be displayed or basic map interactions, user collaboration is non-existent. This being said, in this dissertation, a tool is proposed, aimed at environmental experts, that allows them to express and share their knowledge on air quality and pollution sources. The goal is to set an interface design standard for potential future endeavours following this work. In this chapter, the tool interface and functionalities will be explored, which are explained in more detail in Chapter 4.

3.1 Functionalities

When the application is executed, a map is displayed and most main functionalities are disabled, the exceptions being basic map interactions and history entry selections. By zooming in this map to a certain level, topological segmentation based on OSM data begins, overlaying each entity with coloured polygons, depending on topology (building, land use or road), as shown in Figure 3.1. Among the disabled functionalities are entity selection and pollution adding, which result in a multi-coloured heatmap. More details will be revealed further in the current section. Another way to start working in a chosen region is by loading a previously saved use case scenario in the "History" tab.

Figure 3.1: Map after land classification. Blue polygons represent buildings, green represent land use and pink lines represent roads.

A list of the most relevant functionalities implemented in this application includes: selection making; defining pollution sources; adjusting custom variations; profile managing; polygon drawing; defining extra pollution sources; saving and loading; basic map interactions (pan, zoom, rotate); tutorial; and CSV generation.

In the next paragraphs, the most relevant of these functionalities will be detailed.

3.1.1 Selection Making

Once land classification is done, entity selection becomes enabled. There are a few ways for selecting entities, most simple being just clicking on a singular entity's polygon. Other ways include multi-selection using *ctrl-click*, drawing a selection rectangle with *shift-drag*, filtering and selection via the entity table. As shown in Figure 3.2, after a selection, the selected entity's polygon becomes highlighted and a properties panel is displayed at the right side, containing multiple fields (ten if road is selected, or eight otherwise) describing the selected entity, where four are editable/interactable and the rest serve only as information. The latter may include the following fields:

- Source - Describes the category in which the entity resides (building, land use or road).
- Tag - Describes the type of entity (Ex.: apartments).
- Area/Length - Represents the selected entities' dimensions. "Area" if a building or land use is selected and "Length" if road.

- Id - unique number given to each entity for distinction.
- Name - If a road is selected, this field shows its name.
- One Way - Boolean value that defines if road is one way, or not.

As for editable or interactable property fields, these include: Pollution Magnitude; Pollution Range; Pollution Profiles; Extra Pollution Sources. All of which, will detailed further in this section.

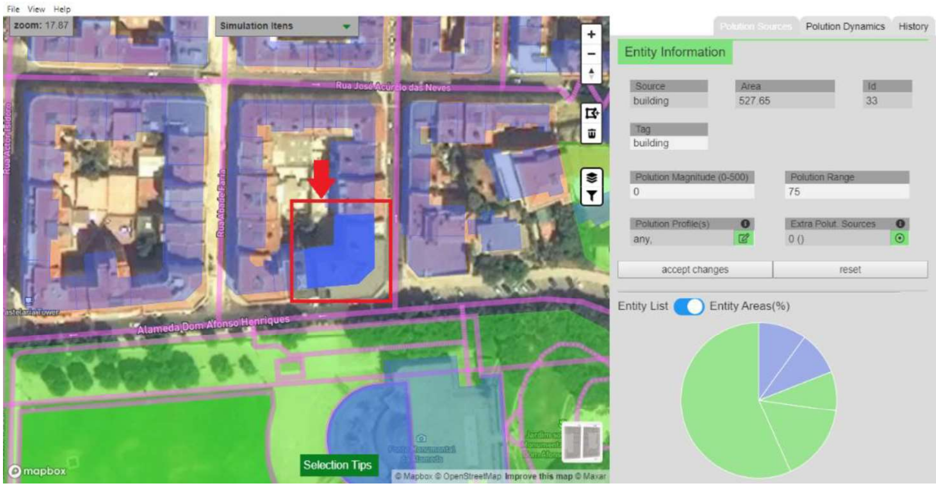

Figure 3.2: After selecting a building, its polygon becomes highlighted and its properties are displayed at the Entity information panel.

3.1.2 Associating Pollution Sources to Map Entities

As hinted previously, the user is able to specify whether an entity emits pollution and how much. To do this, after selecting an entity, the Pollution Magnitude field must be filled by a value ranging from 0 to 500, in the AQI index, the latter being the maximum amount of pollution, and then clicking on "accept changes". After this, the pollution is represented by a heatmap overlaying the selected entity, whose colours depend on the value introduced. Figure 3.3 illustrates what the heatmap looks like after associating a pollution magnitude of 250 and a pollution range of 300 to the map entity (a building) that has been selected in Figure 3.2.

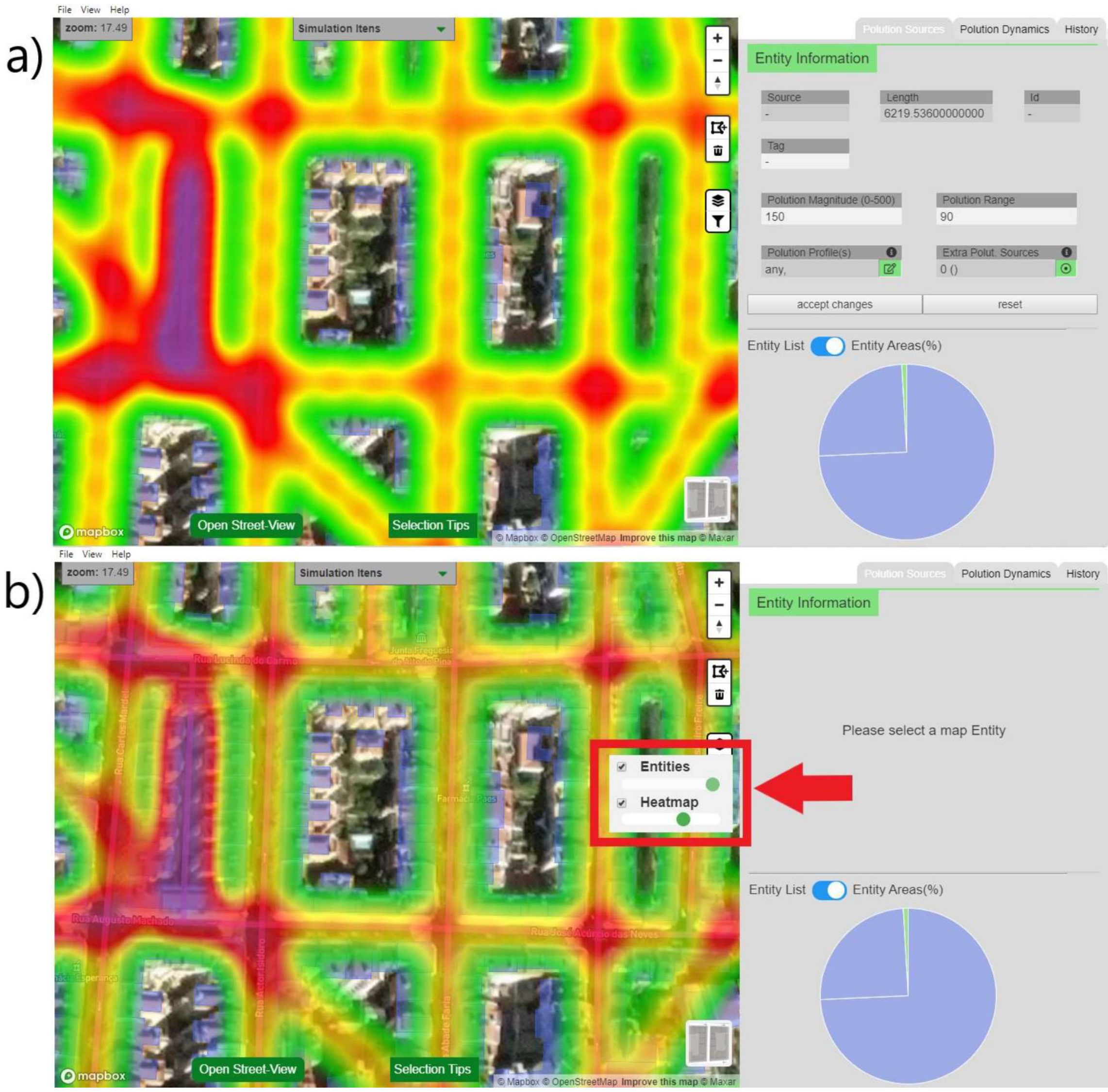

Figure 3.3: Pollution heatmap after adding 150 pollution magnitude and 90 pollution range to all visible roads. Both heatmap and polygons layer's opacity can be altered via a slider. In a) the heatmap layer opacity is at its default value and in b) opacity is set to max.

3.1.3 Adjusting Pollution Variations

Pollution emissions are variably dependent on various factors. For instance, it would not make much sense for a freeway to be associated to a static pollution value, as this varies depending on the time of day and/or amount of traffic. The "Pollution Dynamics" tab is where the user can represent these variations by interacting with existing charts as well as adjust the weight of influence that each chart as on pollution, through the Relevance slider. The weekday graph, for example, represents a weeklong variation, meaning that, according to the user, a particular building could produce more pollution in the weekends than the rest of the week. Figure 3.4

shows what the chart looks like. In case a chart representing a particular variation does not exist, a new chart can be created and customized with a name, profile, relevance meter and some defined bottom-axis items. The graph creation window can be seen in Figure 3.5.

By default, the first item in each chart is selected to affect pollution map data. In other words, considering the weekday chart, only changes made to the "Monday" item dynamically alter map data, represented by the heatmap. To view the simulated effects of Saturday, for example, the respective item must be selected in the "Simulation Items" dropdown, positioned at the top of the map. Here, numerous combinations between charts can be made and dynamically visualized in map (see Figure 3.4).

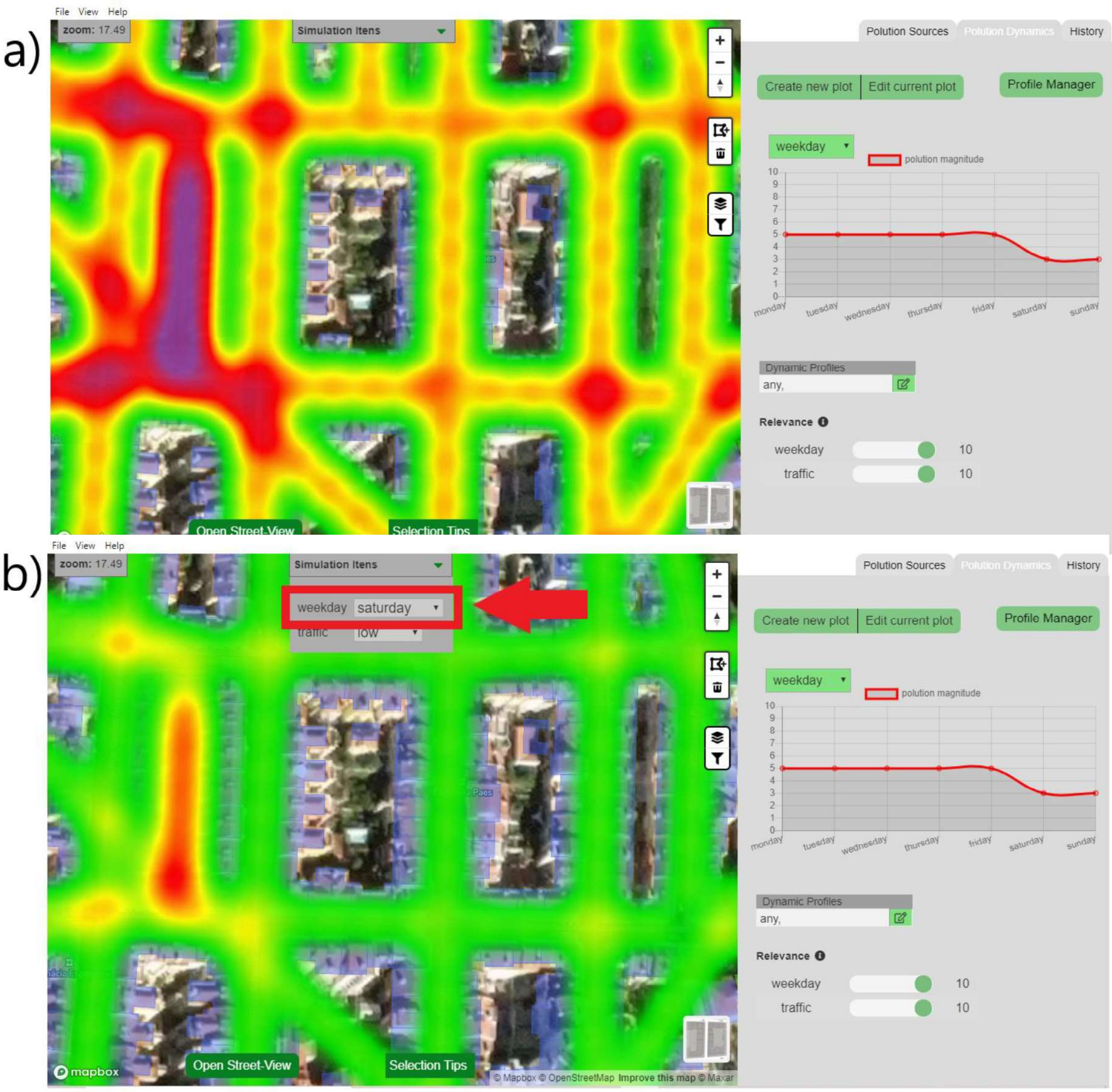

Figure 3.4: Weekday chart altered to indicate that the pollution emitted by any associated entity is lower on weekends than on the rest of the week. In a) there is a heatmap representing pollution on a Monday and in b) Saturday is selected in the "Simulation Items" dropdown, as

a result map pollution decreased. Note that red arrows and rectangles do not belong to the interface and serve only as indicators in this figure.

3.1.4 Pollution Profile Managing

As previously mentioned, for a chart to affect an entity's pollution magnitude, both need to be attached to a mutual profile, as illustrated in the diagram in Figure 3.6. This can be done in the Profile Manager window, which is accessed by clicking on the button next to the "Pollution Profiles" field or the "Profile Manager" button in the charts tab. This new window contains all profiles, each containing two fields, corresponding with entities and charts, respectively. To manage entity associations to a given profile, one must click on the entity field, after which a new window appears containing a list of all existing entities, and then choose which entities are to be associated via checkboxes. All of this process can be seen in Figure 3.7. By default, all entities and all charts are associated with profile "any".

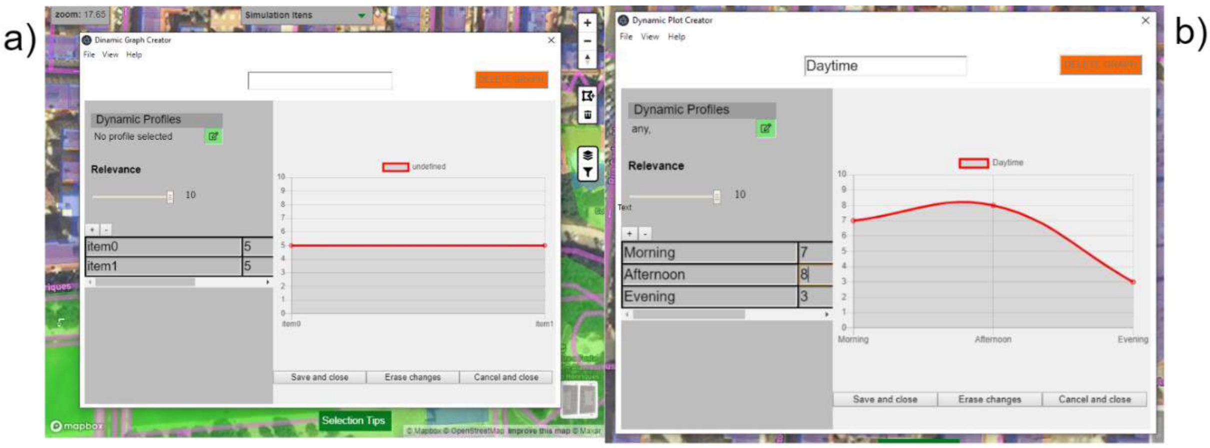

Figure 3.5: Chart creation window. Where a chart can be created with custom properties, such as: name; profile; relevance; number of items; and respective names and values. In a) all fields are yet to be filled. Finally, b) shows a defined daytime chart

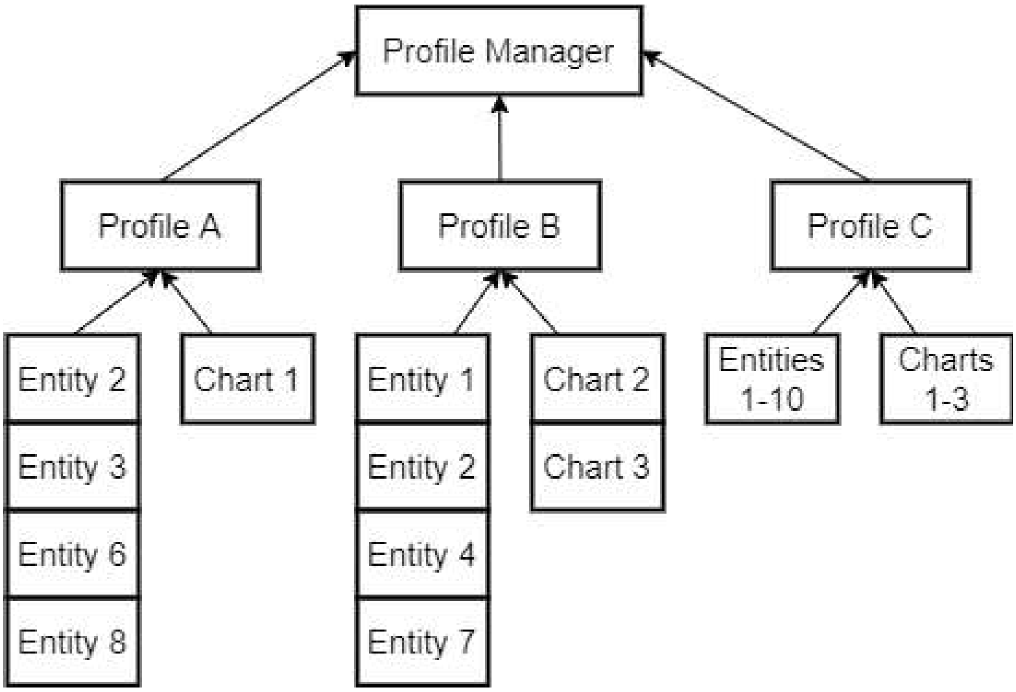

Figure 3.6: Diagram representing how entities and charts are associated through profiles.

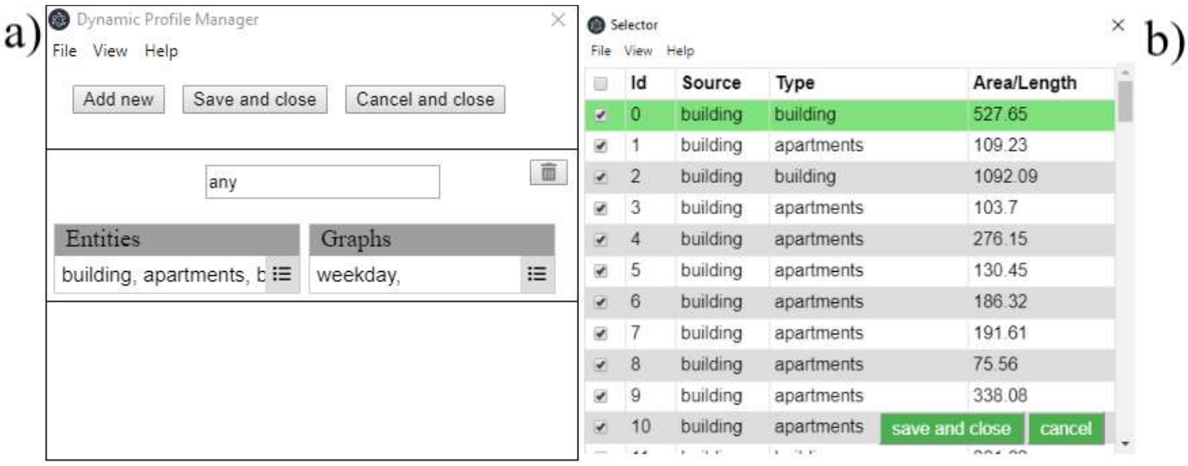

Figure 3.7: Example of managing an entity's profiles. In a) the profile manager is displayed, where all existing profiles are presented, as well as each's list of associated entities and charts. These lists can be altered in the Selector window, which is displayed after clicking on a button with a list icon, as shown in b). To give an example, let's take the selected building with an id of 0. After opening the entity list of profile "any", this entity shows a check mark next to it, which means that it is associated to "any".

3.1.5 User-Specified Map Entity Segmentation

Being so community driven, OSM, which is where Mapbox gets its map data from, is not entirely complete. Meaning it does not contain information on all entities present in the mapped area. This can be visible in the application when map classification is done and a building in the satellite image does not have a polygon assigned to it and, therefore, is not selectable. To counter this issue, the draw plugin from Mapbox (Mapbox GL JS, n.d.-b) was used. With the click of an interface button, draw mode is toggled and a custom polygon can be applied to the non-segmented building. Following this, the Source field becomes editable and, along with the Tag field, must be filled. Having done this, a new entity has been created. This process is shown in Figure 3.8. However, the actual drawing part may not be very visible because of the colour scheme of the tool and the satellite imagery in the background. For a better display of this tool, used in another context, the example in Mapbox's website is the best option.

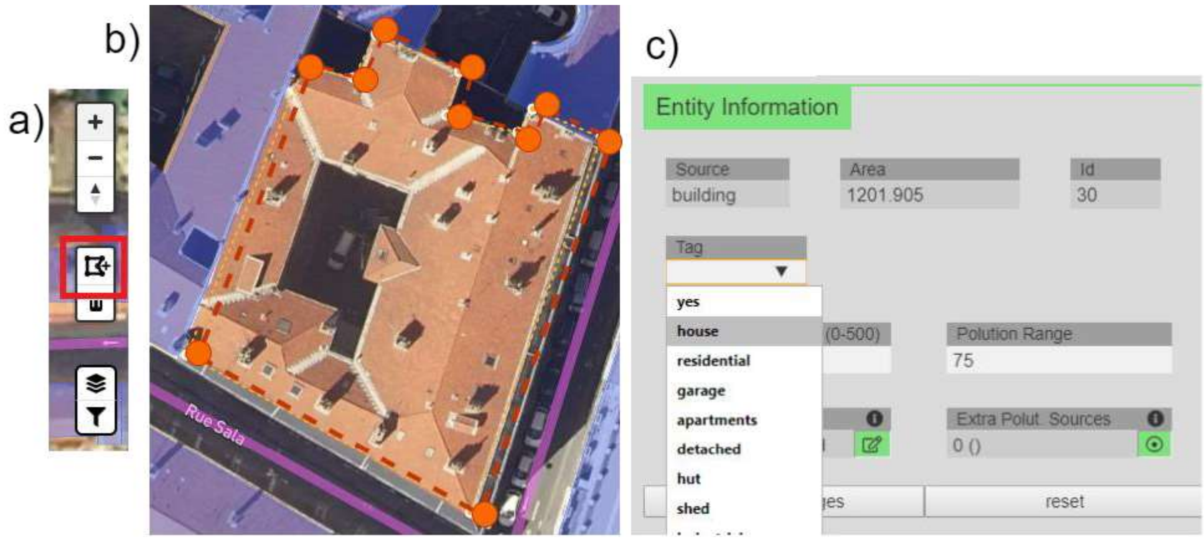

Figure 3.8: Process of creating a new entity using the draw tool. Firstly, the draw button should be clicked, a), then the polygon itself is drawn by clicking on numerous points that act as its vertexes, c). For the polygon to be accepted as an entity, two fields must be filled, starting with the Source field and following by Tag (c).

3.1.6 Defining Extra Pollution Sources

Considering a hypothetical case where a single polygon covers an entire large factory, it might not be very accurate to define a single pollution magnitude value to the building as a whole. Instead, the user may want to add an extra pollution source located, for example, on a chimney that is most likely biggest emitter. To do this, the button next to "Extra Pollution Source" must first be pressed, after which, draw mode is activated, allowing the user to draw a single point where the user clicks inside the original entity's polygon's boundaries. A couple fields are then displayed in the properties panel that must be filled and saved, namely "Pollution Magnitude" and "Pollution Range". An exemplary demonstration can be seen in Figure 3.9.

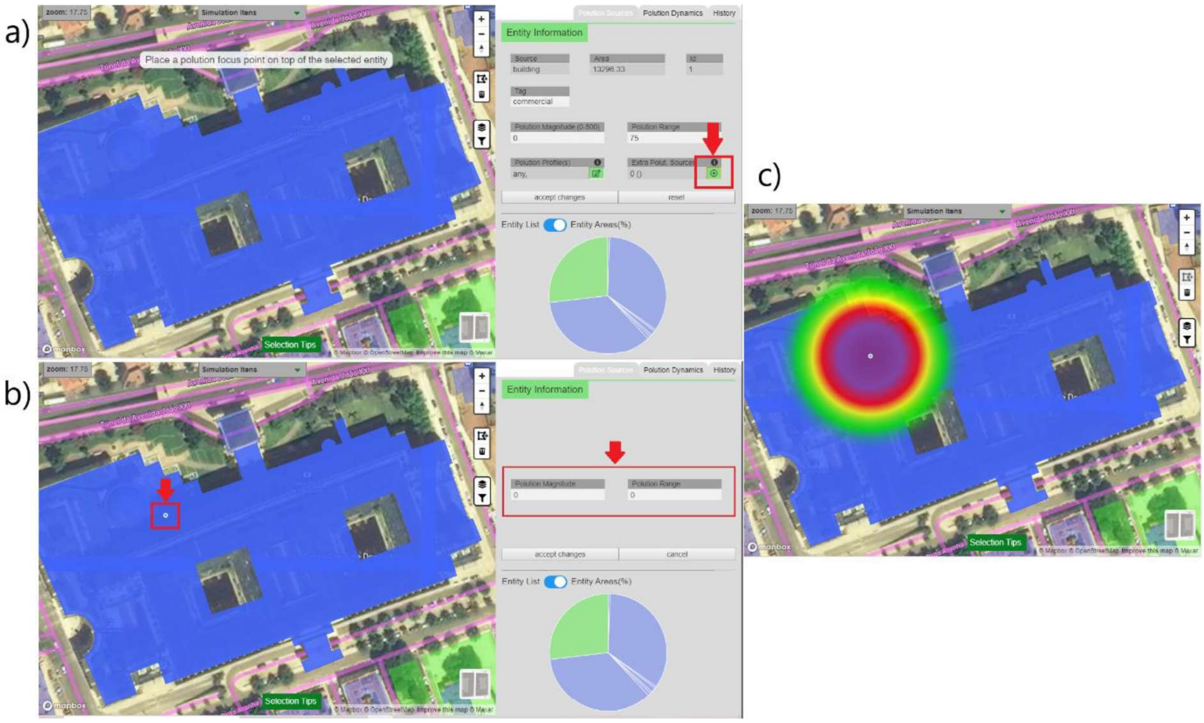

Figure 3.9: Process of adding an extra pollution source to an entity. Assuming that a large building is selected, and the user finds that more pollution is emitted from a particular point of that building. By clicking on the button in a), draw mode is activated, but now it draws a single point. In b) we can see this point and a new properties panel, where only the pollution magnitude and pollution range field are displayed. To attach this pollution focus to the original entity, both fields must be filled and saved. In c) the result is shown. Note that red arrows and rectangles do not belong to the interface and serve only as indicators in this figure.

3.1.7 Knowledge Export Functionality

The knowledge export functionality allows users to generate a Comma Separated Value (CSV) file containing relevant samplings of the introduced entity-pollution associations. This functionality is not aimed to be directly invoked by the user, instead, it is supposed to be automatically executed along with the saving functionality. However, in its current form, the user needs to execute it by clicking on "Generate Points to CSV", under File in the menu bar. What it does is randomly choose a defined number of points in the map and, for each, calculate the pollution magnitude value by interpolation using the detected colour Red-Green-Blue (RGB) value, more on this implementation in Chapter 4. After all points are evaluated and collected, they are stored in a CSV file, which is intended for later use. For instance, the generated file could be used in future work for training a machine learning algorithm.

Chapter 4

Application Development

Software development is composed of numerous steps that go beyond writing code. Before starting implementations, in order to establish clear goals from the start, requirements that the application had to meet were listed. Afterwards, the popular framework Cognitive Work Analysis (CWA) was used to plan the interface's structure beforehand by focusing on a work domain analysis. This method has seen success and usage in many projects (Jones et al., 2006; Lee & See, 2004) thanks to its constraint-based approach.

This chapter will start by presenting the brief list of requirements that the application was supposed to meet, following by the implementation of CWA and, finally, more details of some of the previously mentioned functionalities' implementations will be discussed.

4.1 Requirements Gathering

A set of goals was established from the start for this dissertation and, for these to be met, the tool had to meet a set of requirements of its own. Three main requirements were established to build an application that would allow users, particularly environmental experts, to express and share their knowledge on air pollution, given a categorized map: firstly, and mostly importantly, the user must be able to interact with the map and to associate a pollution value to individual map entities; secondly, since pollution emission and diffusion is not static, a way of representing variations is imperative; thirdly, and this is less geared towards users, the ability to export results from a use case scenario was also required.

Initially, the intention was to allow users to express their knowledge in the form of rules. This would allow the user to write simple conditions that would affect pollution in the current viewport. Conditions such as "`if day==Monday then road.pollution *= 2`", for example, would be written in a simple in-app text editor in its own tab and then stored. This functionality would serve two goals: to allow flexible customization of the pollution map being

created and to train the classifier being developed in parallel to this dissertation, in view of a different project. As more thought was put into this idea the less feasible it became. Creating a new programming language, even if an elementary one, would not only be a difficult task, but also very time consuming, regardless, this was not the main reason why this functionality's implementation was not pursued. To have a programming-based system would significantly increase the learning curve of the application, especially considering that the target audience, environmental experts, is not experienced in computer programming. This would result in longer testing times and, presumably, successfulness would significantly decrease.
As an alternative, a graphical chart-based variations approach was chosen, as it is vastly more intuitive for the user and easier to implement. Even though custom variations can be created, representing a vast number of possible scenarios, a lot of flexibility was lost from the coding approach. For example, a condition that automatically defines pollution values for buildings nearby a coal factory is not possible anymore. Perhaps later studies are able to find an intuitive and straightforward way of implementing this, or similar, functionality.

4.2 Cognitive Work Analysis (CWA)

Traditional, and most common, task analysis methodologies usually focus on defining the best way for a user to conduct a task (Naikar, Moylan, & Pearce, 2006), these being known as normative approaches. While this is a very straightforward approach, some limitations are apparent. For complex systems, where multiple ways to perform tasks are available, the user might run into unforeseen situations that are not predicted by a normative approached to work analysis. Neither the users nor developers are able to predict every possible event in a complex system. Instead of defining the best way for users to perform certain tasks, Rasmussen and, later, Vincente (Rasmussen, J., Pejtersen, A.M., & Goodstein, 1994; Vicente, 1999) took a formative approach with CWA. This constraint-based methodology, instead, focuses on defining constraints under whichM the user must act in order to finish its tasks. It was used in this project to analyse the work environment (interface) in which the user would be going to be part of. By deploying this methodology, a constrain-based analysis of the interface was conducted in order to contain possible user interactions inside set boundaries. Using CWA's multiple phases facilitated early layout planning on what activities can be conducted within the domain, how these can be achieved and what competencies are needed to achieve them. The advantage here is that the finalized application is able to accommodate multiple ways of achieving goals and dealing with unforeseen interactions, which is exactly what the tool's interface was developed to do. The CWA phases are as follows: Work Domain Analysis, Control Task Analysis, Strategies Analysis, Social Organization and Cooperation, and Worker Competence Analysis.

What makes CWA so popular in various different research topics, from medical research to military projects and even autonomous vehicles interfaces, is its flexibility. Meaning that not all phases necessarily need to be realized and no specific order needs to be respected. This methodology can be customized and still producing satisfying results. Like (Debernard, Chauvin, Pokam, & Langlois, 2016), only the first two phases were used to outline this application.

In CWA related studies, the targeted audience for the developed systems are usually referred to as "workers". However, they will continue to be referred to as "users" to maintain coherence with the rest of the thesis.

4.2.1 Work Domain Analysis

The first phase of CWA describes the structure of the system independently of any possible event or user interaction. While possible executable tasks are put aside for now, and analysed in the next phase, it defines the constraints imposed on users when executing them by laying out the available resources and functions. Two tools are usually used to represent a Work Domain Analysis (WDA): The Abstraction-Decomposition Space (ADS) and the Abstraction Hierarchy (AH). Although the latter is mostly a trimmed down version of the former, an AH was created first, to serve as a basis for the ADS.

The AH consists of five levels of abstraction connected to each other by a structural means-end framework: functional purposes, abstract functions, purpose-related functions, entity-related processes and physical entities. For some perspective, the higher (first) level in this hierarchy represents the work domain itself and the lowest (fifth) level corresponds to the physical entities, which would be, for example, a measuring sensor in a system that collects data from air quality monitors. Since no physical components are exist in the developed system, in the present implementation, the last level replaced for "Input Entities", which represent specific inputs, fields, buttons and other components. An applied AH of this project is shown in Figure 4.1.

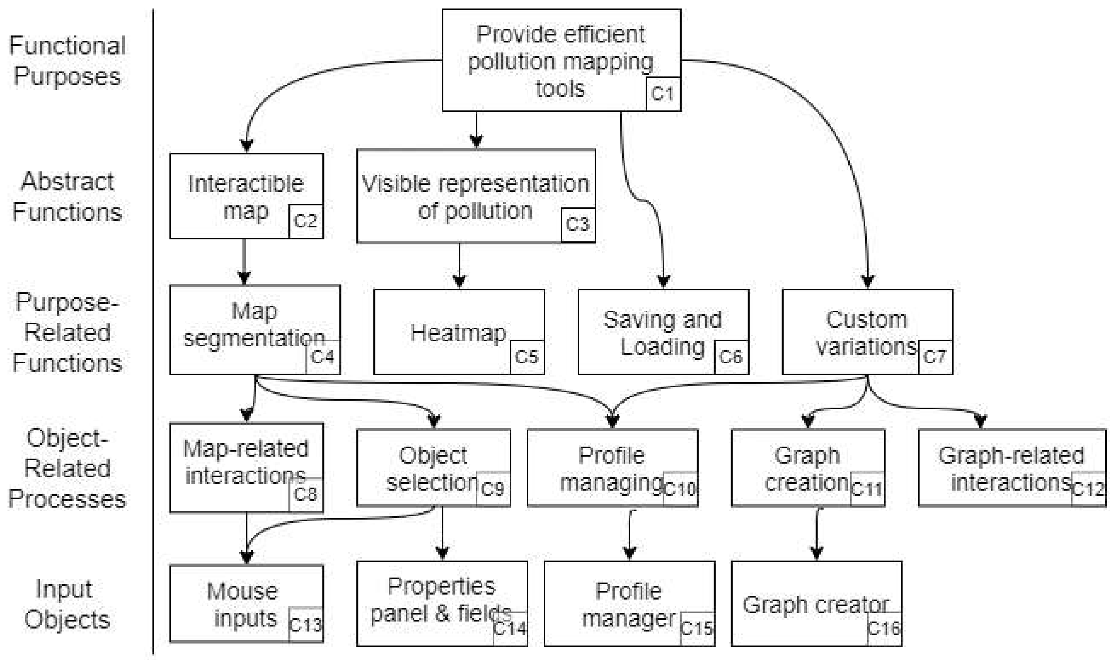

Figure 4.1: Abstraction Hierarchy tool applied to our application. Each node has an associated id to facilitate a subsequent analysis. As the name suggests, this tool represents various levels of abstraction, starting with the overall purpose of the application, to criteria that must be met for this purpose to be realizes, followed by main functionalities. The last two levels are more component-based, starting with tasks directly done by users and ending with the components with which the user interacts.

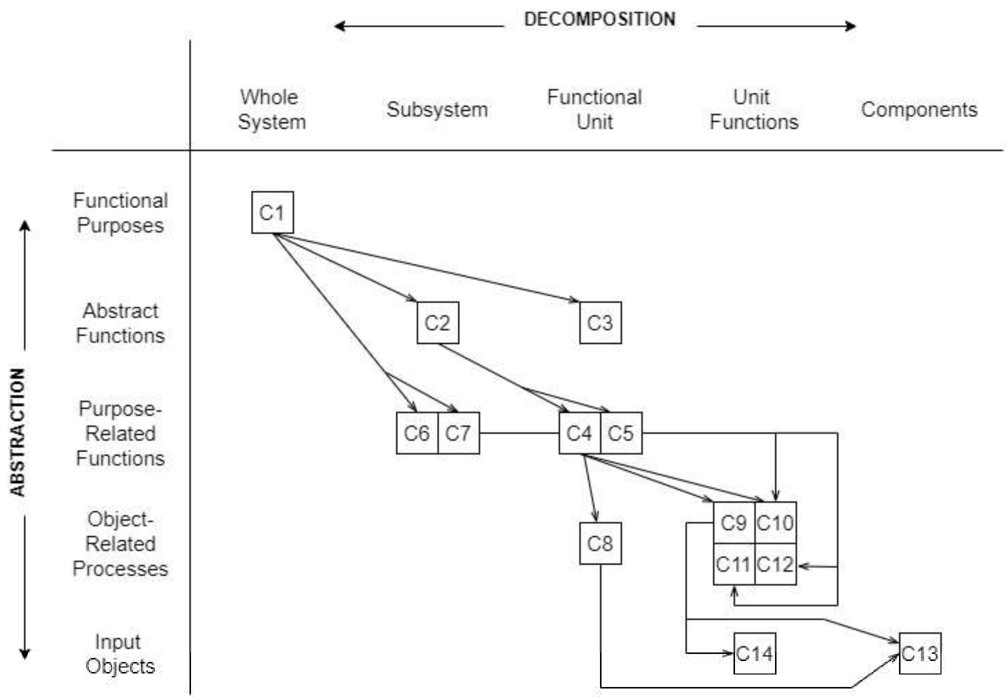

Figure 4.2: The Abstraction Decomposition Space tool applied to our application. The left side represents the abstraction hierarchy, as demonstrated in Figure 4.1. From this figure, the same nodes are used via their ids. Along the top side, is the Decomposition Hierarchy, starting from the system seen as a whole and at each level more specific systems become the focus.

The ADS, in its original form (some "alterations" have been made when used in other studies), can be viewed as a two-dimensional table describing the system trajectory throughout a given task. The left side corresponds to the AH discussed before. Along the top side of the table is the decomposition hierarchy goes from the coarsest level of a system to the most specific. Vincente (1999) explains it better by saying "Moving from left to right is equivalent to zooming in because each successive level provides a more detailed representation of the same work domain". Figure 4.2 presents the ADS framework applied to our application.

4.2.2 Control Task Analysis

This is the second phase of the CWA and it defines the tasks (control tasks) that must be fulfilled in order to reach the previously defined goals, independently of how or by whom.
The tool used to model this analysis is the Decision Ladder, which decomposes activity into a set of actions and results. As mentioned before, the tools used in the various phases of this methodology are not static, instead they intended to be customized depending on the work. As such, a few changes were made to the original Decision Ladder, such as not using some of the nodes. Figure 4.3 shows our implementation of a Decision Ladder, taking into account the task of adding a pollution to a road.

One of the most noticeable features that can be observed in Figure 4.3 are the two node types. Both serve different purposes, for instance, the oval shapes represent "states of knowledge", which are the results (outputs) of the rectangular activities. The rectangles are "Information-Processing Activities" (IPA), which can be seen as the user's train of thought, that is, they represent the cognitive process the user uses to move from one state of knowledge to another. This said, an explanation of each state of knowledge in our diagram follows:

- Goal - Starting with the top-most node, this is not really a part of the main trajectory of the task being evaluated, instead it comes in as an influence when this trajectory reaches the Evaluation IPA. It represents the overall goal of a user when using the application, which is to create a pollution map.
- Alert - This being the first state of the whole task process, it represents the moment that the user knows what to do, which is to associate a pollution level to a map entity (e.g., a road).

- Set of Observations – This is the state that represents the knowledge received after collecting information about the map entity in question (e.g., which pollution is typically present in such an entity).

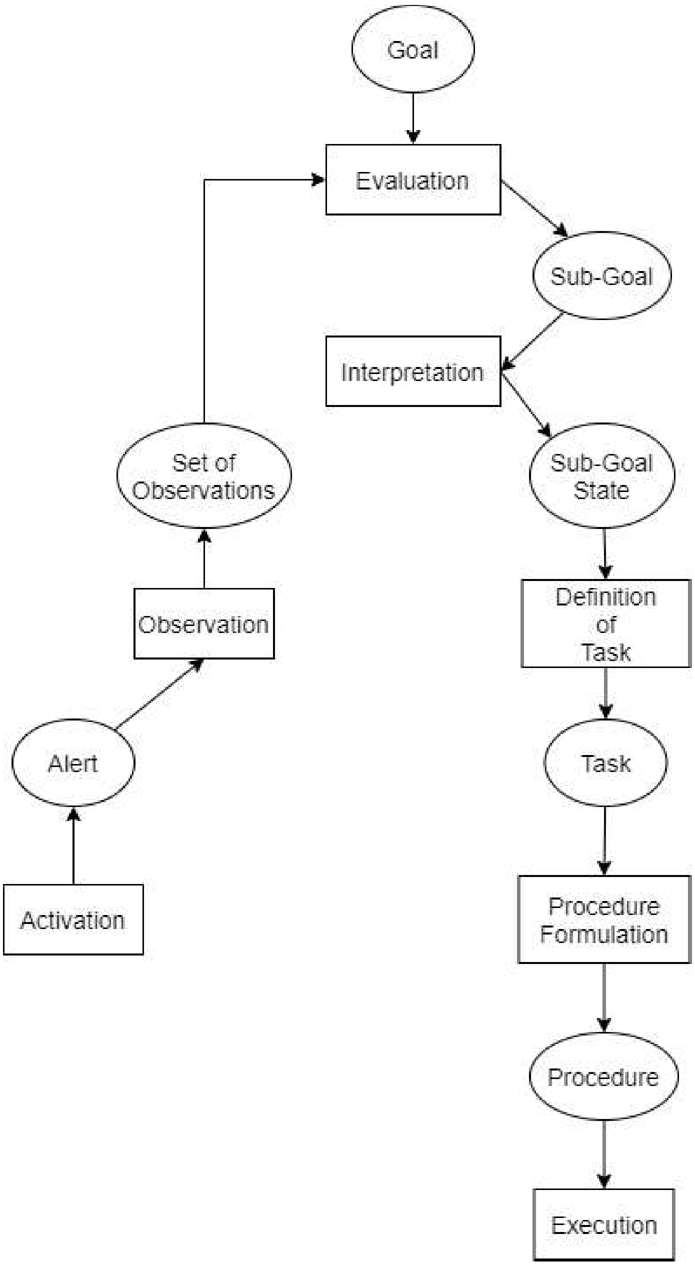

Figure 4.3: Decision Ladder applied to our application.

- Sub-goal - Result of the evaluation process that leads to concrete pollution related values to insert in the road's property fields.
- Sub-goal State - Result of the verification of the sub-goal's current state, that is, for example, if it is not already done.

- Task - Result of the user's decision about the task, or set of tasks, that need to be fulfilled to reach the sub-goal.
- Procedure - Defined strategy on how to complete the task.

4.2.3 Worker Competencies Analysis

The last thing which needed to be determined was the type of user competences needed to properly use our application and deliver somewhat accurate results. To our knowledge, no studies have been conducted in order to evaluate non-expert knowledge related to air pollution evaluation. This being said, another solution was needed to answer the question, so, to this end, some principles of the Worker Competencies Analysis (WCA) were considered. When explaining WCA, Vicente states that Control Task Analysis (CTA) tasks should be considered when defining user requirements to fulfil them. For instance, considering the same example introduced in Figure 4.3, a couple requirements must be met by the user: in the Activation activity, the requirement would be to be able to detect that the road should have some pollution just by looking at the map, although this is relatively rudimentary since the average person assumes that roads, especially main ones, produce some kind of pollution; the other requirement is more expert-focused and it is present in the Sub-Goal activity, as it is considered that expert knowledge is necessary to determine a relatively accurate AQI value to attach to the road.
While non-expert testing was useful to draw conclusions related to the design and intuitiveness of the interface, experts could lead us to conclude whether the main idea of this application is viable or not. Testing and results can be found in Chapter 5.

4.3 Functionalities Implementation

The design of the developed application was extensively discussed in Chapter 3. In this section, the actual implementation of functionalities will be explained.
It is important to note that all external APIs and libraries are open-source. Although Mapbox GL does have paid plans, these are mostly intended for enterprise level usage and in this project their API was used with no additional costs.

4.3.1 Mapbox GL JS

Mapbox GL JS, which will be referred by us as just "mapbox", is a JavaScript library that provides mapping capabilities, an extensive list of useful functions and some plugins that provide extra functionalities. Getting started with mapbox is fairly simple. After installing the API through Node Package Manager (NPM) (npm, n.d.), as is done with any third-party API in Node.js, one must simply import it through the *require* method and create a new map entity with few parameters, like initial zoom level, coordinates, style (various options are provided), and HTML canvas container that is meant to contain the map. Worth noting that, for any developer to use this API, a mapbox account must be created through their website, where a unique token is linked to it. This token must then be inserted, in code, to the mapbox entity for authentication reasons.

Sources/Layers

At this point, a working interactable map was available, with no extra features added to it. The first implementation goal, defined in the planning stage of the project, was to have a topology-based map segmentation. Fortunately, mapbox provides a layer system that overlays various types of shapes on the map. Multiple layers were used for different purposes, such as inserting polygons, a heatmap and representing selections. By adding a layer to the map containing data about entities vertexes coordinates, polygons were automatically added to the map, overlaying the respective entities. Now, in this use case, while data was originally provided by OSM, mapbox's layer system fetches it by accepting a URL as a source parameter pointing to the data location. However, custom layers are also a possibility by inserting custom data into a layer's source, as long as it respects the geoJSON format (Butler et al., 2016). There are other types of sources that do not require this format, like image type sources, although these were not required for any data representation in map, as such, were not used.

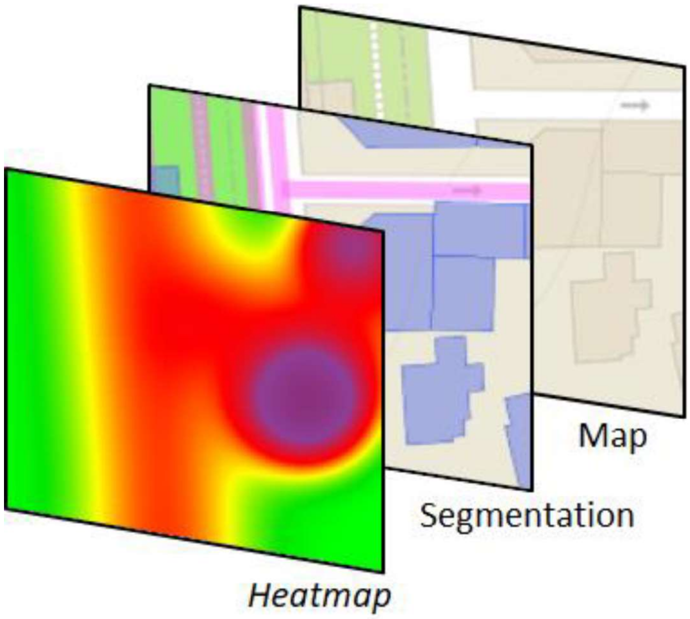

Figure 4.4: Illustration of layer implementation.

Initialization

Once all layers are loaded, a query is executed to save every entity and relevant information in lists and variables for later use, which includes source (e.g., building, landuse or road), tag, id, name (for roads) and area/length. The latter is calculated using the third-party library, Turf.js (turf) (TurfJS, n.d.), which was used multiple times for other functionalities. The information gathering is done by using mapbox's *queryRenderedFeatures* function, which returns useful information in a desired location, like the entire viewport. This information includes the detected entities' original layer and properties parameters, as well as geometry coordinates and more. Furthermore, any layer will only actually be triggered when the zoom level hits a defined threshold. This value is defined by the developer to some extent, but the layers themselves have minimum zoom limits, preventing, for example, a large-scale segmentation. In this implementation these limits were used.

As mentioned before, most functionalities are disabled on start-up, being only enabled when the aforementioned zoom threshold is reached. It was done like this because all initially disabled functionalities depend on the information gathered with the map query function, which does not work if the layers to be queried are not loaded.

Map Entities Selections

In the current state of the tool, when the user selects a map entity (e.g., a building) the respective polygon becomes highlighted, or to be more specific, its colour's opacity is increased. Mapbox does not support single entity colour changing, instead only full layer customization is possible. To get around this issue, a custom layer was used to represent selected polygons. Firstly, the selected entity's coordinates had to be collected by querying the clicked point on the map using the *queryRenderedFeatures* function. The respective coordinates are then inserted into the custom selection layer and a polygon appears overlaying the selected entity. The information gathered from the query function is used for other purposes as well, for instance, the polygon's colour is defined based on the entity's original layer (e.g., if the entity is a building, the respective polygon is blue, for example) and the non-editable fields in the properties panel are filled with the parameters in the query results.

Multi-selection is also a functionality and it can be achieved in multiple ways, as described in Chapter 3. Essentially whenever a new entity is selected, its properties are added to the "selected entities" list. The shift-drag approach draws a bounding box around the desired area, after which, the system queries everything inside it and automatically selects the detected entities. The filter approach simply iterates through all initially detected entities in the viewport and selects the ones matching the user's choice.

User-Specified Map Entity Segmentation

Although the tool relies on mapbox's segmentation functionality, the latter does have some flaws. Since OSM data is community-driven, it also is, consequently, incomplete, which means that properties data are not available for all entities in the world. This leads to the lack of polygons representing several entities. While this is mostly the case in rural areas, even in big cities the layers do not display polygons on some entities. In order to circumvent this issue, mapbox's own *mapbox-gl-draw* plugin was used, which allows for vertex positioning polygon drawing. The draw functionality can be used for two purposes, which were discussed earlier (see Chapter 3): entity creation and extra pollution source definition. Using the plugin's integrated events, it is possible to detect when the drawing process starts, is ongoing or finished and lead the user through it accordingly. If the user is, at any point, in drawing mode, then entity selection is disabled, therefore adding a constraint that is only lifted if the process is finished or the "trash" button is pressed. When adding a pollution focus, in particular, a few more constraints are applied. Using the update event, triggered whenever a point is clicked in drawing mode, it does not allow for a focus to be positioned outside the original entity's polygon, displaying a warning message if this happens. Also, if a focus point is inserted, the displayed pollution magnitude field is required to be filled, otherwise, when trying to save it, the process is interrupted and a message box displays, requesting the user to define a pollution magnitude value.

Pollution Heatmap Generation and Visualisation

A heatmap consists of a cluster of points, the closer these points are to each other more intense heat they produce. Generally and in our case, each cluster of points has a circular form with a central point. Point density is its strongest near the cluster centre, dissipating the further they are from it. For displaying the pollution heatmap, a mapbox layer is used while custom data is inserted. This data consists of coordinates where heat central points (representing the central emission point) are to be positioned. Around this central point, a circular cluster of heat points is formed (assuming an isotropic emission pattern), dissipating the further they are to the centre, illustrating pollution dissipation the further it is from the emitter. Figure 4.4 shows an example of a heatmap with only one central point. To implement this functionality, before inserting any data, firstly, the layer had to be initialized, by filling some global parameters, such as:

- *heatmap-color*. Defines the RGB colours to be displayed as representative of each heat density value (ranging from 0 to 1). By connecting each colour to be displayed, six according to AQI, with a density value, a linearly interpolated set of colours is displayed around the centre point.
- *heatmap-weight*. Defines the aforementioned density of points around the centre. By default, this parameter is set to 1. For example, considering a heatmap containing 100 heat points, by default, each would have an equal weight of 1. Increasing this parameter to 5, would be equivalent to multiplying each point by the same value, meaning that the cluster would contain 500 points. This would result in much stronger colours. Now, this parameter was connected to the selected entity's pollution magnitude. Consequently, a pollution magnitude value of 500 corresponds to the default value of 1 and a pollution value of 250 corresponds to half, meaning that it would result in half the number of points. Therefore, smaller values produce lighter colours, towards green, for example.
- *heatmap-intensity*. Essentially, it can be used as a weight multiplier to every point. While this may seem similar to the previous parameter, the difference is that *heatmap-weight* is data-driven, meaning that it was possible to connect it to inserted pollution magnitude values and produce different heat points accordingly. This parameter affects the whole layer as one, which is to say that one cluster of heat points cannot have its own unique

heatmap-intensity value. Since heat points are summable, when the map is zoomed out, they tend to merge. For example, while not affecting user defined values, numerous lighter heat points (green) at one zoom level may merge into much stronger heat (purple) once zoomed out, looking like a lot more pollution was defined than it actually was. In order to control this phenomenon, this parameter was used alongside the map's zoom level, reducing the intensity with zoom out operations and *vice-versa*.

- *heatmap-radius*. For the same reason explained in the previous parameter, this parameter also used zoom level as an argument. Heatmap points are attached to the map. However, since their size (radius) is fixed, zooming in or out does not affect it in any way, resulting in unwanted heat point addition if multiple clusters exist. To avoid this, radius is controlled as a linear function of the zoom level, meaning that the former is decreased if the latter decreases, and *vice-versa*.
- *heatmap-opacity*. Serves to adjust the overall opacity of this layer. It can be manipulated by the user via a slider in the interface.

Figure 4.5: Pollution heatmap after adding 250 pollution magnitude and 200 pollution range to a single entity (building). The heat-point cluster dissipation is demonstrated by the transition between the colour red to green.

Attaching a single cluster of heat points to an entity with a large polygon is not enough to represent its pollution appropriately. Unfortunately, to our knowledge, there is no way of creating polygon-wide heat using mapbox's API, instead only points can be inserted. When it comes to a smaller polygon, turf's *centerOfMass* function was used that calculates its centroid and then place a single cluster of heat points on the respective coordinates. Representing pollution for larger polygons, however, requires a few extra steps. Using turf's *along* function, multiple coordinates alongside the polygon's outline were collected, after which, heat cluster points are created in each of these positions. The space between points is calculated depending on the respective entity's pollution range value, which affects *heatmap-radius*, in a way that clusters do not end up merging with each other. Figure 4.4 illustrates this implementation. While this implementation is satisfactory, a few drawbacks are evident. For instance, when adding pollution to an entity, heat is only displayed on the polygon's outline ignoring its interior. To simulate pollution in the polygon's interior, using another layer just for this purpose, a new polygon, whose colour depends on the inserted pollution magnitude value, is overlaid directly on top of the original. Another drawback can be found on corners. The current implementation only accounts for distance alongside a polygon's outline trajectory and not physical map distance, causing heat points to be summed together when they are not supposed to, displaying stronger colours, despite not affecting actual pollution magnitude values. Given the defined requirements for this tool, the current heatmap implementation is adequate. However, there is room for improvement. A polygon-wise heatmap solution would fix both conveyed issues. Perhaps some research in this area should be conducted.

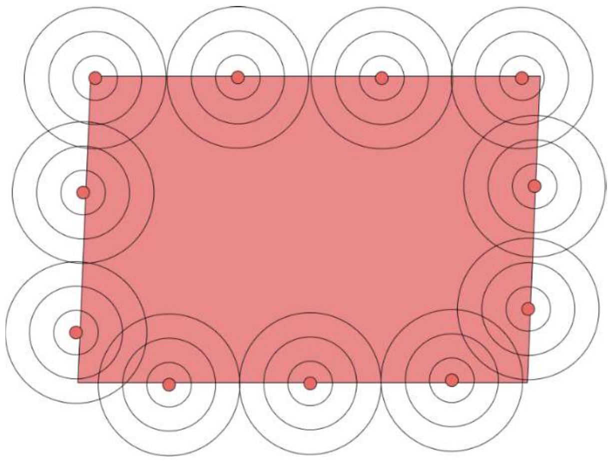

Figure 4.6: Illustration of the point cluster distribution along a polygon's outline. Note that the polygon itself is painted with the same colour as the central point of each cluster to simulate interior pollution emission.

4.3.2 Pollution Profiles and Variation Charts

Pollution Profiles

It was mentioned before that pollution profiles act as a bridge connecting map entities to pollution variation charts, where one can affect the other if both are on the same profile. Profiles are defined in code by entity variables containing two lists, one for map entities and another for charts, all of them being stored in a separate list. This list can be seen represented in the Profile Manager window, which can be accessed in multiple ways described previously, where each profile can be edited, deleted, and new ones created. To share information between windows, the *localStorage* method was used, which allow us to store variables locally so they can be accessed elsewhere in the project. Using an event listener, as soon as the window is closed, any alterations are retrieved from local storage and the main variables are updated, such as the profile list.

Pollution Variation Charts

Pollution variation charts can be found under the “Pollution Dynamics” tab, along with other functionalities. As far as we know, JavaScript does not offer chart functions natively, so the open-source alternative Chart.js (Chart.js, n.d.) was used for its intuitiveness and aesthetically pleasing and customizable charts. Displaying a chart is fairly simple. Firstly, an HTML canvas is to be the container and so must be placed where the chart is supposed to be. Secondly, the chart itself must be initialized, by filling some essential parameters, such as its name, the dataset (values), and item names. What makes this tool so powerful, however, is its customization level available, from the overall colour scheme to more specific options, like padding on the sides, and, more importantly, event listeners. Now, the goal here was for chart changes to be immediately reflected in the map's pollution to allow the user to be more in control of everything. To this end, the chart just had to be interactable in some way. In order to add this missing feature, a third-party plugin for Chart.js (Christoph Pahmeyer, n.d.) was used. It enables dragging individual points in the chart. It also provides some extra event listeners, the most notable one listening to active dragging and values changing.

With all this, the only step left is to connect the chart to the map's entities and heatmap layer. Using the aforementioned event listeners, all entities with a common profile are updated when the chart is altered. Furthermore, the heatmap layer is automatically updated as well, therefore displaying heat points changing in real-time.

Charts values range between 0 and 10. By default, incrementing an item's value by 1 means that associated entities' pollution magnitude is incremented by 50. However, this can be changed. If the user wishes to increase or decrease the chart's weight of influence through the Relevance slider, positioned at the bottom of the tab panel. By default, this is set to 10, meaning pollution incrementations of 50 points. If a value of 5 is defined, these incrementations drop to 25. This process is demonstrated in Figure 4.5, where different Relevance produce different results.

Creating or editing a chart is done in a separate window, where, as detailed previously, the user is able to customize the chart's name, relevance, profile, and data (items and values). Similarly to Profile Manager, to save the new/edited chart, *localStorage* is used to store new variables and an event listener is triggered when the window closes to update main variables, like the charts list.

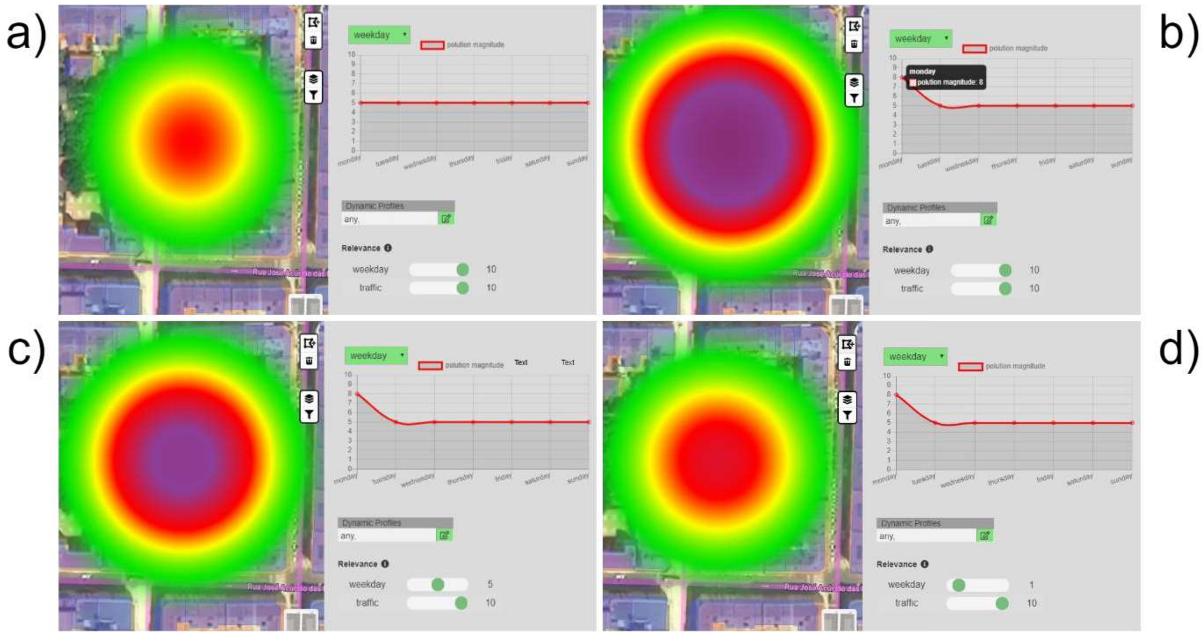

Figure 4.7: Demonstration of chart and Relevance functionality. In this example, an initial pollution magnitude value of 250 was added to a map entity, as we can see in a). In b), this entity's pollution is manipulated using the weekday chart by incrementing 3 points. Note that the "Monday" item is selected in the map and Relevance is set to 10. By decreasing Relevance to 5 and repeating the process, c) is the result. In d), the process is repeated again with Relevance at 1.

4.3.3 JSON Extraction and CSV Generation

JSON Extraction

The saving/loading system was implemented with end users in mind. Being able to save the current document in any work-oriented application is essential and, since creating a pollution map may be take some time, it is important to be able to return to previous work without having to redo everything. By pressing in "Save current document" under "File" in the menu bar, a JSON file containing all information about the current state of the application is created using the *JSON.stringify* method, this includes map coordinates, zoom level, profiles, charts and entities (including drawn). A timestamp is also included, as well as a unique id for the file.

Loading, on the other hand, can be done in the History tab by selecting the wanted entry representing a work scenario. Each entry displays information, such as a timestamp, the average pollution magnitude across all entities and how many altered entities, charts and profiles exist. When an entry is selected, *JSON.parse* is used to extract all information from the file into a variable. Afterwards, *flyTo* function is applied to position the map's viewport at the saved location and all relevant variables are defined by the information gathered.

CSV Generation

This functionality is not aimed at end-users. It is, however, intended to be automatically executed when alongside the saving function and used externally, in putative future work, to train a machine learning algorithm, the goal being automatic map entity classification in terms of air pollution emission, given its topological context. Its implementation consists on picking numerous random points in the viewport and following the remaining steps for each point:

- Query the selected point as to obtain the entity's properties, if one was detected;
- Obtain the point's RGB values (corresponding to the AQI and represented by the heatmap) using mapbox's *readPixels* function;
- Considering the AQI's range of colours, used in the heatmap layer, the two closest ones to the point's RGB are obtained. By defining all colours' RGB as three-dimensional coordinate vectors, we start by assuming a line between each two AQI colours, after which, for each line, the Equation 4.1 is used, where *t* represents the gradient respective to the original selected point. This process is illustrated in Figure 4.6. This variable will be useful later on in the process.
- In this step, the distance between the original point and *t*'s coordinates is calculated. The result is used to verify the closest AQI colours.
- Now that the closest AQI colours have been obtained, their pollution values and the *t* gradient, the final step is to interpolate the two extreme pollution values with *t* as weight using Equation 4.2. The result is the pollution value which better represents the selected point's colour.

$$t = \frac{(P_1 - P_0) \cdot (P_2 - P_1)}{|P_2 - P_1|^2}$$

Equation 4.1: Considering a point P_0 and a line between points P_1 and P_2, t is the closest line point to P_0.

$$result = Pol_1 + \left(t \times (Pol_2 - Pol_1)\right)$$

Equation 4.2: Considering that Pol_1 and Pol_2 represent the pollution values corresponding to the two closest AQI colours, this equation returns an interpolated pollution of this interval with t as weight.

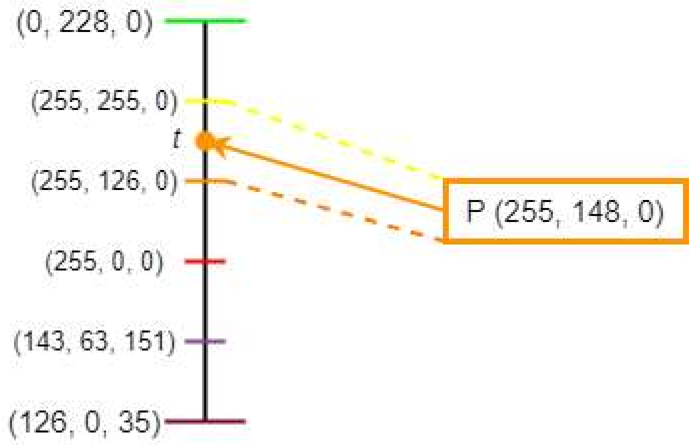

Figure 4.8: Illustration of finding the gradient (t) between two colours corresponding to a respective colour.

One issue that arose was that the satellite imagery in the background was affecting the results, since colours belonging to buildings, roads and other objects obviously aren't aligned with AQI's. To counter this, a plain green background (representing lack of pollution) was displayed overlaying this imagery. This way, if a random point hits this green surface a pollution value of zero is returned.

4.4 Interface Design

Despite being experts, end-users are not expected to be computer-savvy. Therefore, usability and simplicity are imperative for the tool's success. To build such an interface it is imperative that it follows Nielsen's heuristics, published in revised form in 1994 (Nielsen, 1994). These are widely used rules of thumb meant to find potential usability issues and, consequently, build efficient interface designs. This set of heuristics are composed by 10 in total, just as follows:

1. Visibility of system status. In order to keep the user informed of any relevant change to the system, feedback should be returned as some activities are executed. In this case, for example, visual feedback is displayed after selecting an entity, by highlighting the respective polygon, or when a chart is interacted with, the "Simulation Items" dropdown flashes green.
2. Match between system and the real world. Users usually approach a new application with a defined mindset based in their real-world experience. This was taken into account when naming or describing interface components and choosing button icons. The former task can be seen when specifying map entities, which are referred to as "entities" in app, while for the latter task font-awesome's (FontAwesome, n.d.) icons for being simple and familiar.
3. User control and freedom. When using an application, the user is bound to make mistakes, especially in initial interactions. Emergency exits, should be readily available whenever this happens, for instance, when drawing mode is enabled there are two ways of disabling it, either through the "esc" key or by clicking the "trash" button. Another example should be the "cancel and close" button in Profile Manager and Chart Creator. The ability to revert changes should also be available and, while it was implemented in Chart Creator as the result of clicking "Erase Changes", a full undo functionality should have been implemented for the whole application. This limitation is to be circumvented in tool's future iterations may address this.
4. Consistency and standards. As mentioned in the second heuristic, the user's pre-defined mindset may affect its interactions with the application's interface. This led us to choose commonly known words and expressions to describes components. Another important aspect of this heuristic is consistency, in other words, avoiding using different expressions to describe the same functions.
5. Error prevention. As the name suggests, this heuristic consists on preventing users from making errors. For instance, when adding an extra pollution source to an entity, the system prevents positioning the new pollution point outside the entity's polygon. Another example can be seen when trying to save an entity with no associated profiles, where an alert message is displayed warning that no charts will affect the entity in question's pollution.
6. Recognition rather than recall. A good interface should not require constant information remembering from one area of the application to another, instead tips or visual cues should be used to help minimize the user's memory load. When opening the entity selector, from

Profile Manager, currently selected entities are highlighted in the table, so the user does not have to remember them.

7. Flexibility and efficiency of use. While much of the application was built to accommodate initial users with simple and intuitive design, advanced users might expect something more, for instance, they might want to customize their experience or accelerate the whole process. Several functionalities were built in with these users in mind, like changing layers opacity or changing the map's style (satellite/street). Multiple ways of making selections were discussed, and, while multi-selection may be seen as a more advanced task despite it being taught by the tutorial, another, more hidden, way is available. In the bottom part of the Pollution Sources tab, by toggling from "Entity Areas" to "Entity List", a table containing all entities is displayed, from where it is possible to single-select and multi-select entities basing on their parameters.
8. Aesthetic and minimalist design. It is important that the user understands all aspects of the interface, although extensive or to many descriptions can deteriorate the overall experience by cluttering the interface. An aesthetically pleasing interface should be able to limit the amount of information available while still be descriptive enough to be intuitive. Some information in the application is hidden, while still easily accessible, such is the case with tips, which are only displayed when the mouse hovers over the respective containers, and some non-prioritized functions, like entity filtering or layer opacity sliders. Icons, when well designed and well placed, are also a good way of hiding/replacing descriptions, for example, entity filtering is enabled when hovering a button containing a filter icon.
9. Help users recognize, diagnose and recover from errors. As much as user error is avoided, sometimes these are inevitable. Accordingly, it is important that the system is capable of dealing with user mistakes by displaying helpful and descriptive error messages. The aforementioned warning triggered when saving an entity with no profiles is a good example. Also, for example, when trying to create a new chart and not all fields are completely filled, a message pops up describing the issue.
10. Help and documentation. In a best use case scenario, the interface should be sufficiently intuitive to be correctly used without much help, either by recognizable icons and brief descriptions or comprehensive error messages. However, sometimes, or to some users, some more information may be useful. The first time the application is executed, a tutorial window consisting of seven slides, each related to a task with a description and a video of

it being completed, is shown. This automatic characteristic can be turned off by the user, although, it is easily accessible via the menu bar, under "Help".

Chapter 5

Testing and Discussing

In this chapter, the testing process and collected results will be presented and discussed. The testing process was divided in two parts. Firstly, informal formative tests were conducted to identify application errors and bugs, as well as evident interface design issues. This problem discovery strategy was paramount to inform the several revisions throughout the tool's development. Subsequent formal summative tests were aimed at: evaluating the interface's efficiency and usability; validating the viability of the solution presented in this dissertation; identifying additional application errors; collecting feedback and improvement suggestions.

5.1 Testing Methodology

Starting with informal tests. Each test began with participants just exploring the interface and becoming familiar with it. After exploring, a set of three exercises were presented, given a certain map location to work in: generate a heatmap by associating pollution magnitude values to map entities; associate edited entities to profile "any"; and represent weekly variations using a chart.

Moving on to formal testing. Each test session started with a brief description of the project and testing process itself. It was mentioned that the application's goal is to allow users to express their knowledge regarding the correlation between environments' topologies and air pollution emission and diffusion, independently of concrete geographical locations. In an effort to avoid any kind of pressuring and getting the most out of each test, it was also mentioned that the application was being evaluated, not the participant, there are no wrong answers and every interaction is crucial. Finally, participants were asked to think out loud, so it was evident which interface aspect was being focused on, and which train of though was following when facing any potential challenge.

After this introduction, a tutorial consisting of seven slides was shown to the participant. Each slide describes a functionality of the application by displaying a video of its use, along with a short textual description. It was also informed that this tutorial could be reviewed at any point

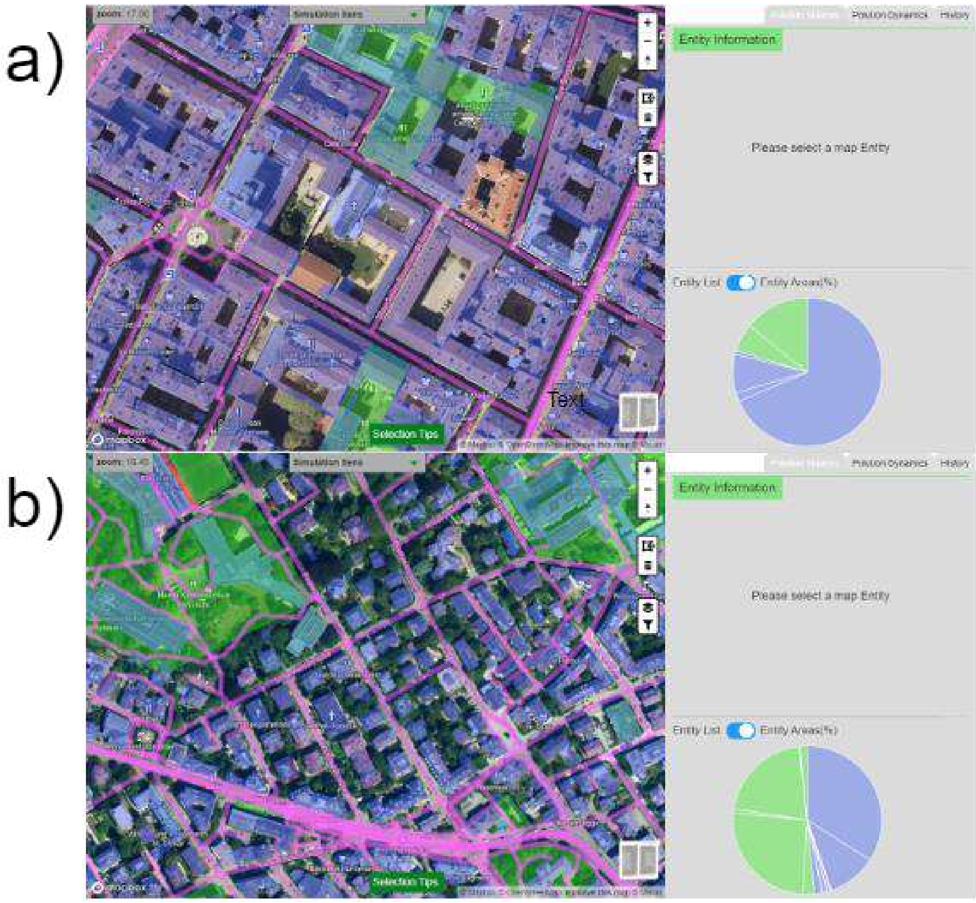

Figure 5.1: Chosen locations for testing. Location in a) was used in the first phase (Lyon, France). Location in b) was used in the second phase (Zurich, Switzerland).

in the application. This option follows from the fact that the tool is to be used by experts in a professional context, rather than by casual users in cold-start scenarios. Therefore, the time spent on following a tutorial should not be an issue, that is, it is not expected to trigger early tool's time-to-leave.

After the tutorial, the actual testing session begins. Three different phases compose this stage, the first two being interaction -oriented, whereas the third is opinion-oriented. Firstly, a set of six tasks with specific goals and related to main functionalities (see Table 5.1), to be achieved using the interface, was presented to the participant. Secondly, a separate task requiring interacting with the application in order to produce a pollution map from scratch in a new location was presented. Finally, participants were asked to fill the widely used user interface evaluation questionnaire, System Usability Scale (SUS) (Diot, Zarka, & Lemarié, 2002), consisting of ten items, to assess the application's usability, as follows:

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

Additionally, the following three items were also included to the questionnaire, to specifically evaluate the last task:

1. In the last task, the tool allowed me to create a pollution map, given a zone and its' properties.
2. In the last task, the tool allowed me to express my knowledge on pollution emission, given a zone and its' properties.
3. The pollution map I created given a zone, and its' properties, is representative of what I expect to be correct.

To each of these questions, five possible answers could be given: strongly disagree; disagree; neutral; agree; strongly agree. If the participant did not know what answer to choose, "neutral" should be selected. During debriefing, participants were then asked for any extra feedback regarding improvement suggestions.

Table 5.1: List of tasks presented to participants in the first testing phase, along with functionalities tested in each one.

Tasks	**Related Functionalities**
Add Associate a pollution magnitude value of 100 to a specified road.	Single selection; Adding Associating pollution to map entity

Select and a pollution magnitude value of 150 to every building alongside a specified road.	Multi-selection; Adding Associating pollution to map entities
Represent weeklong variation in the viewport considering that at the weekend pollution is reduced to half.	Pollution variation
Select and add associate a pollution magnitude value of 100 to all roads currently in the viewport.	Entity filtering; Adding Associating pollution to map entities
Add Associate a pollution magnitude of 500 to a specified non-segmented building.	Polygon drawing and map entity creation
Represent daylong variation of the previous task's building, considering that this variation has half the impact of the weeklong variation.	Pollution variation chart creation; Pollution profile management

5.1.1 Informal Formative Tests

The set of informal formative tests were conducted in order to identify rudimentary issues and bugs. A total of six tests were carried out. However, to avoid every participant running into the same issues, these were divided into two phases of three tests each. Following each phase, necessary changes to the application's interface were made. Examples of changes applied after the first phase include: some map buttons were repositioned; opacity in the zoom and "Simulation Items" boxes was increased so as to improve visibility; the autocomplete functionality on the "Tag" field was redone, now showing the entire list of possible tags on click, instead of just showing results after a character was inserted, which improved tag association; and a game-breaking bug was discovered and fixed, where entities would not load if the map was dragged, causing selections to not work as well.

The second testing phase showed smoother interactions already, thanks to previously made changes. However, some users still felt that some basic functionalities were not yet fully obvious, such as which steps should be taken to associate pollution levels to map entities. To circumvent this difficulty, the already mentioned slideshow-style tutorial was added to the application so that it could be displayed to the participant whenever the application started. This tutorial describes, via a video and some brief text, in each slide, how some of the most important functionalities work. Also, the colour scheme of the map buttons was changed from green/black to white/black, as it was observed to improve visibility.

5.1.2 Formal Summative Tests

To assess the usability of the application's final version, a set of 30 summative tests were carried out. Although, in an ideal scenario, every participant involved in this process would be an expert, such was not the case. A total of 7 environmental experts were available to take part in these tests, the other 23 users being non-experts. Although all tests were used and considered when making conclusions about the usability aspect of the interface, only expert tests were taken into account when validating the viability in the approach introduced in this dissertation, which consists of using expert knowledge as a new source of information to create pollution maps. To clarify, the expert category was considered to include researchers in environmental sciences, as well as masters' degree students in the same field, the latter being surrogates of established researchers.

On average, the summative tests lasted approximately 40 minutes, varying from 30 minutes up to an hour. Out of the experts, 2 were men and 5 women. As for non-experts, these consisted of 16 men and 7 women. All non-experts were students, 13 of which currently in their masters' degree in computing area, while the rest of them in the bachelors' degree, 7 of which were in the computing area, and other 3 were in economics and management.

5.2 Results

In this section results from the previously described interface assessment process will be presented and discussed, starting with the interface design effectiveness, following by the viability of the developed tool.

5.2.1 Interface's Usability Evaluation

Constructive criticism and useful feedback were received throughout testing and will presented in this section. Overall, data collected from testing related to the interface design's usability was considered satisfactory. Results from all 30 tests have shown that every task, except the sixth (chart creation and profile managing), was carried out with a success rate higher than 70% (see Table 5.1). Generally, each task has a single goal, which needs to be achieved for it to be considered a success. However, a "middling success" status was given to tasks which were taking too long (presumably, each task should take no more than five minutes to be concluded) and/or required some kind of hint (by the researcher conducting the test) to be concluded. A task was considered a failure if the participant is not able to conclude it, even after receiving a hint.

The first task, in which participants were required to select a road and associate a given pollution level to it by filling the "Pollution Magnitude" field, exhibited a success rate of 70%; however, the execution of this task highlighted an issue: the roads' selectable lines are too thin, requiring the participant to be very precise when clicking on it. Although, eventually, most participants managed to select the road on their own, some required a hint to do it, therefore achieving a middling success. Failure scenarios saw participants trying alternative methods, like associating pollution levels to nearby buildings or even trying to select the road using the draw tool.

In the second task, focused on multi-selection, participants were expected to select multiple buildings alongside a specified road. This task could be solved with some flexibility, as numerous ways to multi-select are available. The easiest way of completing it was to *cntrl-click* the buildings and associate the given pollution value; this method was used in 90% of all successful cases. This task showed a success rate of 73%. However, it is plausible that this percentage would dramatically increase with subsequent uses (resulting from learning effects). Among middling success cases, participants were using the shift-drag multi-selection, while also selecting the road (which should not be included). In these cases, a hint was needed to correct their approach. Some of the 13% of unsuccessful cases were already showing usage of multi-selection capabilities, such as the selection rectangle with *shift-drag*, but were unable to complete the task, even with a hint. This leads to the conclusion that more attempts would result in more proper uses of the selection rectangle, or even lead to multi-selection with *cntrl-click*.

The third task aimed at testing pollution variation chart interactivity and required participants to interact with the weekday chart in a way that values corresponding to items “Saturday” and “Sunday” were decreased to half of the other days’ values. This was the most successful task, with a success rate of 90%. However, in retrospective, this task should have included required interactions with the “Simulation Items” dropdown, since most users did not realize that chart variation could be dynamically visualized through the map’s heatmap. Even though the tutorial contains a panel dedicated to charts, explaining this, perhaps a more comprehensive explanation should be attempted.

The fourth task is relatively simple given the small number of required interactions. It required the participant to select all roads by clicking on the corresponding option in the "Filter" dropdown and to associate a specified pollution value to them. A success rate of 70% was observed in this task. About half of the unsuccessful task executions were due to the participants opting for manually selecting all roads, which is too time consuming and error prone. All of the middling success cases (17%) were due to testers taking too long at finding the "Filter" dropdown and, eventually, requiring a hint when giving up and trying a different method. This interface component is located at the right side of the map along with other buttons and, while its position was not out of the ordinary, the problem resided in the fact that the "Layers" dropdown overlaid it when triggered. This happened often, since participants tended to explore the right-side buttons starting from the top, when the mouse touched "Layer" button, the respective dropdown was displayed, therefore hiding the filter functionality. A quick rearranging of the buttons would resolve this issue, which will be addressed in a future version of the tool.

The fifth task, which showed a success rate of 87%, evaluated the draw functionality's usability in adding a polygon to a building which OSM does not contain information about. Essentially, the task involved creating a new map entity by drawing a polygon over a non-segmented building, identifying it with a tag, and associating a given pollution level to it. Usually, after a couple of tries at selecting the building, participants realized that drawing was the solution. It is worth noting that some confusion was evident when testers had to choose a tag, as it proved difficult for them to determine which tag was the most adequate just by looking at the satellite image of the building. Middling success cases saw participants trying to draw by via click-drag motions or getting stuck at choosing a tag. Perhaps a default tag should be automatically added to a drawn object, such as "drawn", allowing users to change it even they happen to know which type of building it is.

The sixth task, although presented as one, involves three implicit sub-tasks: create a new pollution variation chart, adjust its relevance to half, and associate it to the same profile as the map entity created in the previous task. Some of these sub-tasks required themselves multiple steps to be completed, which prove difficult to a new user. A few unique criteria were used to analyse this task's results. Respectively, the task was considered to be a success if all three sub-tasks were realized, a middling success if only two, and a failure if one or none. The success rate of this task was 20%, middling success was 37%, and 43% of users failed to complete it (see Table 5.2). One immediate conclusion to be drawn from these results would be that chart creation was a success, as the majority of users managed to do it. However, adjusting the "Relevance" parameter showed more mixed results. In the tutorial panel regarding charts, this parameter was not mentioned, leaving users with a small hint box next the respective slider component. Perhaps a more comprehensive explanation would have decreased this issue's occurrence. Profile management proved to be the most misunderstood functionality, being used properly by only 33% of participants. User feedback obtained in debriefing revealed that the tutorial description and the accompanying video could have been more clarifying. Furthermore, once Profile Manager is opened only one profile exists, "any". Perhaps more pre-defined profiles could also have helped in obtaining a better understanding of the functionality.

Table 5.2: List of tasks presented to participants in the first testing phase, along with functionalities tested in each one.

	Task	**Success Rate**	**Middling Success Rate**
1	Associating a pollution magnitude value of 100 to a specified road.	70%	23%
2	Select and associate a pollution magnitude value of 150 to every building alongside a specified road.	73%	14%
3	Represent weeklong variation in the viewport considering that at the weekend pollution is reduced to half.	90%	7%
4	Select and associate a pollution magnitude value of 100 to all roads currently in the viewport.	70%	17%
5	Associate a pollution magnitude of 500 to a specified non-segmented building.	87%	13%
6	Represent daylong variation of the previous task's building, considering that this variation has half the impact of the weeklong variation.	20%	37%

Table 5.3: Success and middling success rates of sixth task's sub-tasks.

Sub-task	**Success Rate**
Create new chart.	76%

Adjusting relevance to half.	43%
Associating the new chart to the same profile as the previous task's entity.	33%

In most tasks, testers frequently recurred to the tutorial. The majority of users took advantage of this help tool when performing the first, second, third, and fifth tasks. This level of attention to the tutorial was unexpected given the results obtained during the formative tests and suggests that tool's usability needs further improvements. Testers feedback showed that some of them found the text and videos in some tutorial slides somewhat confusing, mostly due to small or large amount of text included in them. Perhaps, reformatting the text descriptions in order to render them clearer, as well as adding some simple animations to the videos that could help the participant identifying expected mouse interactions, would improve the tutorial experience.

5.2.2 Tool Viability

This section mainly presents results from the second phase's task, which asked participants to create a pollution map to the best of their knowledge. Furthermore, any discussed thoughts and conclusions are derived from expert feedback, as it is more likely to be reliable information, although some of the feedback was shared by the non-experts as well. These results support the analysis regarding the value of the developed tool as a means for environmental experts to express and use their knowledge for classifying satellite imagery in terms of pollution emission and diffusion. A proportion of 53% of the participants, which tried to create a pollution map in this task, wound up selecting multiple entities, mainly roads, and then associated the same pollution magnitude to all of them, while some, 13%, tried to introduce different values depending on the type and apparent width of the road. Every participant, however, had difficulty in defining a general/aggregate AQI value, as they felt the need to be more specific, pollutant-wise. Therefore, although experts recognised the novelty and value of the tool to their field, which is a strong positive outcome of this dissertation, they displayed some difficulties in using it as a means to express their knowledge in its current form. Fortunately, their rich feedback generated a list of improvements that will be included in future versions of the tool.

The most common complaint, even by some non-experts, was their difficulty in determining the most adequate global/aggregate pollution value (AQI) for the map entity in question, given the limited amount of information provided by the tool. Currently, map information consists of topology segmentation, labels, and entity properties: type (e.g., apartments, residential roads), physical area (or length, in case of roads), road names and whether they are one way or not. However, testers felt that details regarding traffic density would help them. Although Mapbox provides these data, it was not included in the tool due its low update rate for the city of Lisbon. Regardless, in retrospective, this data could have been integrated nonetheless, since it would provide the user with additional context and, ultimately, yield better results. Some more information was also reported by the testers as valuable in helping them expressing their knowledge, such as: sea or river traffic, which affects coastal cities; road width (including number of lanes); buildings' height, useful to identify street canyons; and the existence of certain smaller entities, such as bus stops, trees, and crosswalks. Actually, OSM provides building height data; however, it is very incomplete in most cities, returning a default height of 3 for most buildings. Future endeavours should do more research aimed at finding and including this type of data in the tool.

In its current state, the tool only allows the user to associate a global/aggregate (not pollutant-wise) pollution index to each map entity, in the form of AQI (see Chapter 2). However, some users reported that they felt that in some situations it would be easier for them to express specific pollutant-wise indexes, upon which the global/aggregate AQI would be automatically computed. For instance, it would be more natural for them to state the emission magnitude of nitrogen dioxide (NO2) along a given road than specifying a global AQI for it. In addition, this approach would also be beneficial since it would add a new layer of knowledge extracted from the users.

Despite the goal being to associate pollution to entities, independently of their geolocation, testers reported location to be crucial. However, the argument can be made that geolocation cruciality may decrease by improving on the approach taken with this application using collected feedback and some other ideas, which are discussed in Chapter 6.

5.2.3 Usability Questionnaire

In order to obtain some more statistical information about the application's usability, in the third testing phase, a usability questionnaire consisting of 13 items was presented to the testers. Ten of these items are from SUS, whereas the other 3 were appended so as to handle the specificities of the last task (see Tables 5.2 and 5.3). Results are presented in Tables 5.4, 5.5 and 5.6.

Responses to the first question showed mixed results with non-experts, which was expected since they were not the target audience; conversely, satisfactory results were obtained with experts, 57% of which agreeing that they would frequently use this system. Because the latter percentage represented most experts, while not being definitively high, it leads to the conclusion that, in its current state, the application is, at least, useful and showing promise. By including the feedback received during testing in the tool's forthcoming versions the feedback received during testing, it is expected that more users will agree to the first question.

Results from the other nine usability questions generally demonstrate good or satisfactory results with both participant groups. Some non-expert participants reported that they selected "Agree" in the last usability question because of their inability to define pollution magnitude values, saying that perhaps some previous knowledge was required to do it. The nineth question obtained at least apparently incoherent results, when compared to other questions. Finstad (Finstad, 2006) argues that the use of the word "very" tends to confuse people, as it may conflict with "strongly". This may explain the higher percentage in neutral answers than expected, seeing that selecting "agree" or "disagree" may be seen as equivalent as removing the word "very" from the question and selecting a "strongly" option. The same argument can be made considering the seventh question. Possibly users felt that the system was somewhat cumbersome to use, but not "very". Using this word may have compelled them to choose disagreeing options.

In the study introducing SUS (Diot et al., 2002), a method of scoring a set of questionnaires was also introduced. It is known that each question must be ranked from 1 to 5 based on users' level of agreement. In order to calculate a SUS score, a sum of each question points across all tests must be done, although with a few rules. For each odd numbered question, user response is subtracted by 1. For each even number question, user responses are subtracted from 5. By following these two rules and after summation is done, the result is then multiplied by 2,5. The final score ranges between 0 and 100. After the testing process was finished, a total SUS score of 67,75 was obtained.

By the time SUS was nearly ten years old, a new study (Bangor, Kortum, & Miller, 2008) was conducted where data from 206 studies, consisting of a total of 2,324 surveys, was collected and analysed to evaluate SUS as a usability questionnaire as well as to put into perspective how to determine a SUS score. It was concluded in this study that acceptable scores would be at least a 70, with good results hovering around the high 70s are 80s. Scores just below the acceptable threshold, in the 60s, are considered usable but to be continued improved. More recently, however, Jeff Sauro (Sauro, 2011) analysed data from 500 different studies, totalling at over 5000 surveys. In his conclusion, the acceptable threshold dropped by two points to 68.

Having our score accordingly approximate to this value, leads to the conclusion that, considering usability, the interface developed for this application is satisfactory, although continued improving should be implemented.

The three questions related to the last task showed very different results from a usability standpoint, performing much worse, which is not surprising, considering the success rate of the task. Non-experts seemed to return the best results overall, however, similarly to the last task's analysis (see Section 5.2.2), expert results are the most likely to return reliable results. The first question shows the most positive results, as pollution maps were indeed created. However, most scenarios saw the participant performing multiple selections of various buildings and/or roads and associating the same pollution magnitude value, a behaviour already observed in the usability tasks (when the expected behaviour was to consider available entity information when associating pollution). The second and third question appeal to the participant's knowledge. While results from these questions were (mostly) unsatisfactory, by submitting participants to the tool, considerably valuable constructive criticism and suggestions were collected. Returned feedback suggest that building upon and improving the developed tool by altering existing functionalities (e.g., individual pollutant definition) and adding new ones (e.g., including more data sources, and entity searching by tag) should produce better viability results. Information such as this, will be extremely useful in development of future iterations of the tool introduced here.

Table 5.4: Results of questionnaire conducted on experts.

Items \ Answers	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	0%	14%	29%	57%	0%
2	14%	72%	14%	0%	0%
3	0%	0%	29%	71%	0%
4	0%	71%	29%	0%	0%
5	0%	14%	29%	57%	0%
6	0%	57%	29%	14%	0%
7	0%	0%	14%	29%	57%
8	14%	57%	29%	0%	0%
9	14%	0%	57%	29%	0%
10	14%	86%	0%	0%	0%

specific to the second phase					
11	14%	42%	14%	28%	0%
12	28%	28%	14%	28%	0%
13	20%	0%	60%	20%	0%

Table 5.5: Results of questionnaire conducted on non-experts.

Answers / Items	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	13%	39%	30%	17%	0%
2	48%	35%	13%	4%	0%
3	0%	4%	17%	57%	22%
4	35%	35%	17%	13%	0%
5	0%	0%	17%	83%	0%
6	26%	44%	22%	9%	0%
7	0%	0%	9%	44%	48%
8	17%	49%	17%	17%	0%
9	0%	30%	30%	35%	4%
10	26%	30%	30%	13%	0%
specific to the second phase					
11	4%	13%	13%	61%	9%
12	9%	26%	30%	35%	0%
13	0%	13%	78%	9%	0%

Table 5.6: Results of questionnaire considering both experts and non-experts.

Answers / Items	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	10%	33%	30%	27%	0%
2	40%	44%	13%	3%	0%
3	0%	3%	20%	60%	17%
4	27%	43%	20%	10%	0%
5	0%	3%	20%	77%	0%
6	20%	47%	23%	10%	0%
7	0%	0%	10%	47%	43%
8	17%	50%	20%	13%	0%

9	3%	23%	37%	34%	3%
10	23%	44%	23%	10%	0%
specific to the second phase					
11	7%	20%	13%	53%	7%
12	13%	27%	27%	33%	0%
13	3%	10%	77%	10%	0%

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In view of this dissertation, a tool was developed as a means for environmental experts to share their knowledge using an interactive map.
Current available solutions to pollution mapping are heavily reliant on air quality fixed monitoring stations, which, even in highly monitored cities, leave vast gaps of unmonitored areas. Alternatives have been designed in order to fill these gaps, such as using mobile sensors, or even estimate pollutant quantities in completely unmonitored countries exploiting various sources of complementary information based on machine learning solutions. Nevertheless, there are still several situations in which is not viable to apply these alternative solutions to properly produce dense air pollution distributions estimates. For instance, the coverage of mobile sensors is no unlimited.

This dissertation addressed an alternative and novel way of complementing existing air quality monitoring solutions. Concretely, a tool was developed in order to allow environmental experts to share their knowledge regarding how different city entities (e.g., buildings) affect emission and diffusion of air pollution. The ultimate goal is to use this knowledge to extend the air quality predictions beyond the areas addressed by the expert while introducing their knowledge in the tool. Experts express their knowledge by associating air pollution magnitude, in the form of AQI, and its dynamics, to selected map entities.

The tool's validation was divided into two phases. Firstly, interface design and usability were evaluated. Then, the viability of the tool as a means to allow experts to adequately express their knowledge was assessed. Overall, the obtained results demonstrate good usability levels. Testers generally found the interface to be intuitive and easy to navigate. Functionalities such as entity selection, multi-selection, and polygon drawing were quickly recognizable. In some cases, pollution adding by editing map entity parameters and chart interactions required the tutorial use after which, were easily realized. Some functionalities were not as easily used though. For instance, switching between simulation items, to being able to observe chart variation represented in map, whereas rarely used. Nevertheless, restructuring the third task to accommodate this action could have improved its usage. Pollutions profiles management was the most misunderstood functionality. Users were frequently confused by it and feedback revealed that the tutorial panel related to profile management should have been more clarifying. A participant suggested to remove profiles as the intermediary between entities and charts, directly linking both through an entity parameter instead.

When faced with the idea underlying the tool, experts recognised its value and novelty. When requested to express their knowledge with the tool, a few improvements for forthcoming tool's versions became evident, which will be discussed in the next section. The elements pointed out by the experts as the most valuable to bring the tool to the next level are: to allow the user to associate specific pollutant magnitude values to map entities, rather than a single global/aggregate AQI value; to provide the user with additional topological information, such as road width and buildings height, as these are known to greatly influence air dynamics; and provide the user with traffic information as, again, is key to predict air pollution magnitude.

Concluding, the results indicate that the developed tool is usable and is in the right direction for allowing experts to express their knowledge for expanding air quality predictions beyond sensor data. The novelty of the tool rendered difficult to anticipate all design elements that maximises user experience. Tool's wireframes presented to environmental experts in early design iterations helped us in directing the design but were still incapable of uncovering some of the already discussed issues. These only became evident when the experts were exposed to the functional prototype. We believe that the lessons learned, what works and what needs to be improved, are of value for those who intend to develop air pollution knowledge extraction tools and will surely be useful to improve forthcoming versions of the tool herein presented.

6.2 Future Work

As mentioned, we believe that future iterations of this tool have the potential of drastically improving it in terms of usability and viability. After resolving all usability issues and adding required functionalities, research should be made on additional data sources that provide some of the information mentioned in Chapter 5. Mapbox's own traffic layer, would be an option to consider for its relatively high coverage, despite the update rate not being very frequent in some cities.

As the application reaches higher levels of improvement, new functionalities and ideas occur, to build upon the foundation laid in this project. For instance, having the option to view the map with 3D rendered buildings might help users to better visualize it and identify street canyons, which generally increase pollution. Mapbox provides way to easily implement this functionality, as all it takes is to add the respective 3D rendering layer to the map. It should be noted that, in most cities, the majority of buildings would be represented with the same height, being that height data is very incomplete. However, when it is complete in some cities, such as in Manhattan, New York, it could be very useful. An alternative could be to provide the user with link to Google Street View, based on which the user could estimate, at least qualitatively, the height of the buildings.

The *waqi.info* API (Aqicn.org, n.d.) overlays the map with all accepted air quality monitors and respective results, similarly to Table 2.1. By implementing this API in the application, a new study could be conducted in which expert knowledge was compared with official sensor data. This could answer the question of how relevant direct expert knowledge would be in pollution mapping.

Another way in which usability could be improved would be to implement some gamified aspects. In (Teles, Mariano, & Santana, 2020), embedded tutorials inspired by videogames were implemented. A similar set of tutorials would fit well in the application, perhaps even improving understanding of all functionalities and fixing the aforementioned issues regarding the original tutorial. Furthermore, including game-like challenges to the tool could be a great way to inspire further user exploring of the interface. Finally, providing game-like goals and perhaps even including a point system could involve users into an active learning process to master every functionality related to challenges (Domínguez et al., 2013).

Bibliography

Agency, U. S. E. P., & Division, I. (2014). Air Quality Index (AQI). *Encyclopedia of Quality of Life and Well-Being Research*, (February), 120–120. https://doi.org/10.1007/978-94-007-0753-5_100115

Aqicn.org. (n.d.). API - Air Quality Programmatic APIs. Retrieved from https://aqicn.org/api/pt/

Bangor, A., Kortum, P. T., & Miller, J. T. (2008). An empirical evaluation of the system usability scale. *International Journal of Human-Computer Interaction*, *24*(6), 574–594. https://doi.org/10.1080/10447310802205776

Bhagat, P. K., & Choudhary, P. (2018). Image annotation: Then and now. *Image and Vision Computing*, *80*(October), 1–23. https://doi.org/10.1016/j.imavis.2018.09.017

Boykov, Y. Y. (2001). *Interactive Graph Cuts*. (July), 105–112.

Breezometer. (n.d.). GLOBAL AIR QUALITY MAP. Retrieved from https://breezometer.com/air-quality-map/

Butler, H., Daly, M., Doyle, A., Gillies, S., Hagen, S., & Schaub, T. (2016). The geojson format. *Internet Engineering Task Force (IETF)*.

Cassard, T., Jauvion, G., & Lissmyr, D. (2020). *High-Resolution Air Quality Prediction Using Low-Cost Sensors*. Retrieved from http://arxiv.org/abs/2006.12092

Castell, N., Dauge, F. R., Schneider, P., Vogt, M., Lerner, U., Fishbain, B., … Bartonova, A. (2017). Can commercial low-cost sensor platforms contribute to air quality monitoring and exposure estimates? *Environment International*, *99*, 293–302. https://doi.org/10.1016/j.envint.2016.12.007

Chart.js. (n.d.). Simple yet flexible JavaScript charting for designers & developers. Retrieved from https://www.chartjs.org/

Cheng, Q., Zhang, Q., Fu, P., Tu, C., & Li, S. (2018). A survey and analysis on automatic image annotation. *Pattern Recognition*, *79*(June 2018), 242–259. https://doi.org/10.1016/j.patcog.2018.02.017

Christoph Pahmeyer. (n.d.). chartjs-plugin-dragdata. Retrieved from https://github.com/chrispahm/chartjs-plugin-dragdata

Debernard, S., Chauvin, C., Pokam, R., & Langlois, S. (2016). Designing Human-Machine Interface for Autonomous Vehicles. *IFAC-PapersOnLine*, *49*(19), 609–614. https://doi.org/10.1016/j.ifacol.2016.10.629

Diot, P., Zarka, V., & Lemarié, E. (2002). SUS - A quick and dirty usability scale. *Revue Des Maladies Respiratoires*, *19*(1), 87–89.

Domínguez, A., Saenz-De-Navarrete, J., De-Marcos, L., Fernández-Sanz, L., Pagés, C., & Martínez-Herráiz, J. J. (2013). Gamifying learning experiences: Practical implications and outcomes. *Computers and Education*, *63*, 380–392. https://doi.org/10.1016/j.compedu.2012.12.020

Finstad, K. (2006). The system usability scale and non-native English speakers. *Journal of*

Usability Studies, *1*(4), 185–188.

FontAwesome. (n.d.). Font Awesome. Retrieved from https://fontawesome.com/

Friedman, S. (n.d.). BreezoMeter's Global Air Quality Index: Real-time & Micro-Local (BAQI). Retrieved from https://blog.breezometer.com/breezometers-air-quality-index

Friedman, S. (2018). BreezoMeter's Continuous Accuracy Testing for Reliable Air Quality Data. Retrieved from https://blog.breezometer.com/air-quality-accuracy-testing

Haklay, M., & Weber, P. (2008). OpenStreet map: User-generated street maps. *IEEE Pervasive Computing*, *7*(4), 12–18. https://doi.org/10.1109/MPRV.2008.80

Helen, A. (n.d.). The Ultimate Guide to Understanding Air Quality Data in 2020. Retrieved from https://blog.breezometer.com/ultimate-guide-understanding-air-quality-data

IQAir. (n.d.). IQAir. Retrieved from https://www.iqair.com/

Jones, W., Bruce, H., Bates, M. J., Belkin, N., Bergman, O., & Marshall, C. (2006). Personal information management in the present and future perfect: Reports from a special NSF-sponsored workshop. *Proceedings of the American Society for Information Science and Technology*, *42*(1), n/a-n/a. https://doi.org/10.1002/meet.1450420151

Karuppasamy, M. B., Seshachalam, S., Natesan, U., Ayyamperumal, R., Karuppannan, S., Gopalakrishnan, G., & Nazir, N. (2020). Air pollution improvement and mortality rate during COVID-19 pandemic in India: global intersectional study. *Air Quality, Atmosphere and Health*. https://doi.org/10.1007/s11869-020-00892-w

Kumar, A., & Goyal, P. (2011). Forecasting of daily air quality index in Delhi. *Science of the Total Environment*, *409*(24), 5517–5523. https://doi.org/10.1016/j.scitotenv.2011.08.069

Leaflet. (n.d.). VectorGrid. Retrieved from https://github.com/Leaflet/Leaflet.VectorGrid

Lee, J. D., & See, K. A. (2004). Trust in automation: Designing for appropriate reliance Human Factors (In press). *Human Factors*, *46*(1), 50–80.

Makadia, A., Pavlovic, V., & Kumar, S. (2008). A new baseline for image annotation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *5304 LNCS*(PART 3), 316–329. https://doi.org/10.1007/978-3-540-88690-7-24

Maninis, K. K., Caelles, S., Pont-Tuset, J., & Van Gool, L. (2018). Deep Extreme Cut: From Extreme Points to Object Segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 616–625. https://doi.org/10.1109/CVPR.2018.00071

Mapbox GL JS. (n.d.-a). Consumer Apps (B2C). Retrieved from https://www.mapbox.com/industries/consumer

Mapbox GL JS. (n.d.-b). *mapbox-gl-draw*. Retrieved from https://github.com/mapbox/mapbox-gl-draw

Mapbox GL JS. (n.d.-c). Mapbox GL JS. Retrieved from https://docs.mapbox.com/mapbox-gl-js/api/

Morawska, L., Thai, P. K., Liu, X., Asumadu-Sakyi, A., Ayoko, G., Bartonova, A., … Williams, R. (2018). Applications of low-cost sensing technologies for air quality monitoring and exposure assessment: How far have they gone? *Environment International*, *116*(February), 286–299. https://doi.org/10.1016/j.envint.2018.04.018

Mortensen, E. N., & Barrett, W. A. (1995). Intelligent scissors for image composition. *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, (January 1995), 191–198. https://doi.org/10.1145/218380.218442

Munir, S., Mayfield, M., Coca, D., Jubb, S. A., & Osammor, O. (2019). Analysing the performance of low-cost air quality sensors, their drivers, relative benefits and calibration in cities—a case study in Sheffield. *Environmental Monitoring and Assessment*, *191*(2). https://doi.org/10.1007/s10661-019-7231-8

Naikar, N., Moylan, A., & Pearce, B. (2006). Analysing activity in complex systems with cognitive work analysis: Concepts, guidelines and case study for control task analysis. *Theoretical Issues in Ergonomics Science*, *7*(4), 371–394. https://doi.org/10.1080/14639220500098821

Netek, R., Masopust, J., Pavlicek, F., & Pechanec, V. (2020). Performance testing on vector vs. Raster map tiles - Comparative study on load metrics. *ISPRS International Journal of Geo-Information*, *9*(2). https://doi.org/10.3390/ijgi9020101

Nielsen, J. (1994). Enhancing the explanatory power of usability heuristics. *Conference on Human Factors in Computing Systems - Proceedings*, 152–158. https://doi.org/10.1145/259963.260333

npm. (n.d.). Node Package Managar. Retrieved from https://www.npmjs.com/

Papadopoulos, D. P., Uijlings, J. R. R., Keller, F., & Ferrari, V. (2017). Extreme Clicking for Efficient Object Annotation. *Proceedings of the IEEE International Conference on Computer Vision*, *2017-Octob*, 4940–4949. https://doi.org/10.1109/ICCV.2017.528

Parkin, C. (n.d.). What is AirVisual Earth and how does it work? Retrieved from https://support.iqair.com/en/articles/3029339-what-is-airvisual-earth-and-how-does-it-work

PlumeLabs. (n.d.-a). How we create street by street maps. Retrieved from https://plumelabs.zendesk.com/hc/en-us/articles/360025031993-How-we-create-street-by-street-maps

PlumeLabs. (n.d.-b). PlumeLabs Air Quality Map. Retrieved from https://air.plumelabs.com/air-quality-map

PlumeLabs. (2019). *Evaluation of Flow, a personal air quality sensor*. *2*, 1–9.

Popescu, F., & Ionel, I. (2010). Anthropogenic Air Pollution Sources. *Air Quality*. https://doi.org/10.5772/9751

QualAr. (n.d.). Mapa QualAr. Retrieved from https://qualar.apambiente.pt/indices

Rasmussen, J., Pejtersen, A.M., & Goodstein, L. P. (1994). Cognitive systems engineering. *New York: Wiley*, *16*(6), 1023–1023. https://doi.org/10.3109/15513819609168726

Rother, C., Kolmogorov, V., & Blake, A. (2004). GrabCut - Interactive foreground extraction using iterated graph cuts. *ACM SIGGRAPH 2004 Papers, SIGGRAPH 2004*, 309–314. https://doi.org/10.1145/1186562.1015720

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, *1*(5), 206–215. https://doi.org/10.1038/s42256-019-0048-x

Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2005). LabelMe: a database and web-based tool for image annotation. *MIT Computer Science and Artificial Intelligence Laboratory (CSAIL) Technical Reports*. Retrieved from http://18.7.29.232/bitstream/handle/1721.1/30567/MIT-CSAIL-TR-2005-056.pdf?sequence=2%5Cnhttp://dspace.mit.edu/handle/1721.1/30567

Sauro, J. (2011). *MEASURING USABILITY WITH THE SYSTEM USABILITY SCALE (SUS)*. Retrieved from https://measuringu.com/sus/

Selvam, S., Muthukumar, P., Venkatramanan, S., Roy, P. D., Manikanda Bharath, K., & Jesuraja, K. (2020). SARS-CoV-2 pandemic lockdown: Effects on air quality in the industrialized Gujarat state of India. *Science of the Total Environment*, *737*, 140391. https://doi.org/10.1016/j.scitotenv.2020.140391

Sowlat, M. H., Gharibi, H., Yunesian, M., Tayefeh Mahmoudi, M., & Lotfi, S. (2011). A novel, fuzzy-based air quality index (FAQI) for air quality assessment. *Atmospheric Environment*, *45*(12), 2050–2059. https://doi.org/10.1016/j.atmosenv.2011.01.060

Stieb, D. M., Burnett, R. T., Smith-Doiron, M., Brion, O., Hwashin, H. S., & Economou, V. (2008). A new multipollutant, no-threshold air quality health index based on short-term associations observed in daily time-series analyses. *Journal of the Air and Waste Management Association*, *58*(3), 435–450. https://doi.org/10.3155/1047-3289.58.3.435

Teles, B., Mariano, P., & Santana, P. (2020). Game-like 3d visualisation of air quality data. *Multimodal Technologies and Interaction*, *4*(3), 1–19. https://doi.org/10.3390/mti4030054

TurfJS. (n.d.). Turf.js. Retrieved from https://turfjs.org/

Vicente, K. J. (1999). Cognitive Work Analysis Toward Safe, Productive, and Healthy Computer-Based Work [Book Review]. In *IEEE Transactions on Professional Communication* (Vol. 46). https://doi.org/10.1109/tpc.2002.808348

waqi.info. (n.d.). World’s Air Pollution: Real-time Air Quality Index. Retrieved from https://waqi.info/

Weston, J., Bengio, S., & Usunier, N. (2011). WSABIE: Scaling up to large vocabulary image annotation. *IJCAI International Joint Conference on Artificial Intelligence*, 2764–2770. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-460

Wu, J., Zhao, Y., Zhu, J. Y., Luo, S., & Tu, Z. (2014). MILCut: A sweeping line multiple instance learning paradigm for interactive image segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 256–263. https://doi.org/10.1109/CVPR.2014.40

Zemene, E., & Pelillo, M. (2016). Interactive Image Segmentation Using Constrained Dominant Sets. *European Conference on Computer Vision*, 278–294. https://doi.org/10.1007/978-3-319-46484-8

Zunino, A., Velázquez, G., Celemín, J. P., Mateos, C., Hirsch, M., & Rodriguez, J. M. (2020). Evaluating the Performance of Three Popular Web Mapping Libraries: A Case Study Using Argentina’s Life Quality Index. *ISPRS International Journal of Geo-Information*, *9*(10), 563. https://doi.org/10.3390/ijgi9100563

Appendices

Appendix A

Testing Script

Guião

No âmbito de um projeto de investigação estamos a desenvolver uma ferramenta através da qual se pretende que qualquer pessoa consiga descrever aquela que considera ser a relação entre a topologia do ambiente e as fontes de poluição do ar. Por exemplo, quanto é que a presença de um edifício ou de uma estrada influencia no nível de poluição do ar naquele lugar? Esta informação deve depender apenas da topologia do ambiente e não do local em específico, ou seja, não deve ser importante se essa localização se encontra em Portugal ou na China. Para analisarmos potenciais problemas na nossa ferramenta, estamos a conduzir um conjunto de testes, sendo este um deles. Note que o que está a teste é a nossa ferramenta e não o senhor(a). Não existem respostas certas ou erradas, todas elas contribuem para a melhoria da nossa ferramenta e, portanto, são todas importantes e uma mais valia para nós. Agradecia que fosse pensando em voz alta de forma a que possamos entender a que aspetos da ferramenta está a prestar atenção no seu processo de decisão. Quero que se sinta à vontade para terminar o teste assim que o entender. Desde já gostaríamos de agradecer a sua disponibilidade para nos ajudar nesta tarefa.

Antes de iniciar o teste vou começar por mostrar um pequeno tutorial do funcionamento da ferramenta, ao qual pedia-lhe a maior atenção.
Depois do tutorial irei apresentar uma lista de tarefas a serem realizadas por si utilizando a ferramenta. O objetivo é perceber até que ponto a ferramenta é intuitiva o suficiente na realização de tarefas que consideramos típicas. Para além disso, ao realizar estas tarefas ter oportunidade de explorar a ferramenta e suas funcionalidades.

Pré-freeplay

Agora que já teve oportunidade de explorar a ferramenta, peço-lhe para criar o mapa de poluição da zona que se encontra visível no ecrã. Faça como pretender, levando o tempo que entender, e quando sentir que terminou a tarefa por favor diga.

Pré-questionário

Agradeço-lhe desde já o tempo que tem estado a disponibilizar na avaliação da nossa ferramenta. Para terminar, vou apenas pedir-lhe que preencha um simples questionário composto por 10 itens. As respostas que der a este questionário servirão para avaliarmos a usabilidade da ferramenta de forma a podermos tirar conclusões que nos ajudem a melhorá-la.

Leia atentamente cada afirmação presente no questionário e responda numa escala de 1 a 5 quanto é que concorda com essa afirmação. 1 se discorda totalmente, 2 se concorda parcialmente, 3 se não concorda nem discorda, 4 se concorda parcialmente e 5 se concorda totalmente. Se não sabe como responder a um dado item, coloque por favor o valor 3. Tenha em atenção que algumas afirmações são colocadas pela positiva enquanto que outras pela negativa, pelo que lhe pedia que lesse com calma cada uma das afirmações antes de responder.

Appendix B

First Phase Tasks

Tarefas

1. Sabemos que a estrada residencial (*residential*) Rua *Saint-Fraçois de Sales* emite um nível de poluição de 100 unidades e com largura suficiente para influenciar os edifícios ao lado, mas tal não está representado no mapa. Como faria para demonstrar isto?

2. Imagine que todos os edifícios ao longo da rua *Impasse Catelin* emitem o mesmo nível de poluição de 150 unidades. Como faria para representar isto?

3. Imagine que ao fim-de-semana existe metade da poluição nesta zona em relação ao resto da semana. Como faria para representar isto?

4. Imagine agora que todas as estradas visíveis no mapa emitem o mesmo nível de poluição de 100 unidades.

5. Imagine que o edifício situado na esquina entre a Rua *Sale* e Rua *Boissac* emite um nível de poluição de 500 unidades. Como faria para atribuir poluição a este edifício?

6. Imagine que a poluição deste edifício varia consoante a parte do dia (manhã, tarde e noite), mas que esta variação tem apenas metade do impacto nas entidades associadas do que a variação semanal. Como representaria esta variação?

Appendix C

Usability Questionnaire

System Usability Scale

1. I think that I would like to use this system frequently.
   ☐ Strongly Disagree
   ☐ Disagree
   ☐ Neutral
   ☐ Agree
   ☐ Strongly Agree

2. I found the system unnecessarily complex.
   ☐ Strongly Disagree
   ☐ Disagree
   ☐ Neutral
   ☐ Agree
   ☐ Strongly Agree

3. I thought the system was easy to use.
   ☐ Strongly Disagree
   ☐ Disagree
   ☐ Neutral
   ☐ Agree
   ☐ Strongly Agree

4. I think that I would need the support of a technical person to be able to use this system.
   ☐ Strongly Disagree
   ☐ Disagree
   ☐ Neutral
   ☐ Agree
   ☐ Strongly Agree

5. I found the various functions in this system were well integrated.
   ☐ Strongly Disagree

- ☐ Disagree
- ☐ Neutral
- ☐ Agree
- ☐ Strongly Agree

6. I thought there was too much inconsistency in this system.
   - ☐ Strongly Disagree
   - ☐ Disagree
   - ☐ Neutral
   - ☐ Agree
   - ☐ Strongly Agree

7. I would imagine that most people would learn to use this system very quickly.
   - ☐ Strongly Disagree
   - ☐ Disagree
   - ☐ Neutral
   - ☐ Agree
   - ☐ Strongly Agree

8. I found the system very cumbersome to use.
   - ☐ Strongly Disagree
   - ☐ Disagree
   - ☐ Neutral
   - ☐ Agree
   - ☐ Strongly Agree

9. I felt very confident using the system.
   - ☐ Strongly Disagree
   - ☐ Disagree
   - ☐ Neutral
   - ☐ Agree
   - ☐ Strongly Agree

10. I needed to learn a lot of things before I could get going with this system.
    - ☐ Strongly Disagree
    - ☐ Disagree
    - ☐ Neutral
    - ☐ Agree

☐ Strongly Agree

Tool Specific

1. In the last task, the tool allowed me to create a pollution map given a zone and its' properties.
   ☐ Strongly Disagree
   ☐ Disagree
   ☐ Neutral
   ☐ Agree
   ☐ Strongly Agree

2. In the last task, the tool allowed me to express my knowledge on pollution emission given a zone and its' properties.
   ☐ Strongly Disagree
   ☐ Disagree
   ☐ Neutral
   ☐ Agree
   ☐ Strongly Agree

3. The pollution map I created given a zone, and its' properties, is representative of what I expect is correct.
   ☐ Strongly Disagree
   ☐ Disagree
   ☐ Neutral
   ☐ Agree
   ☐ Strongly Agree