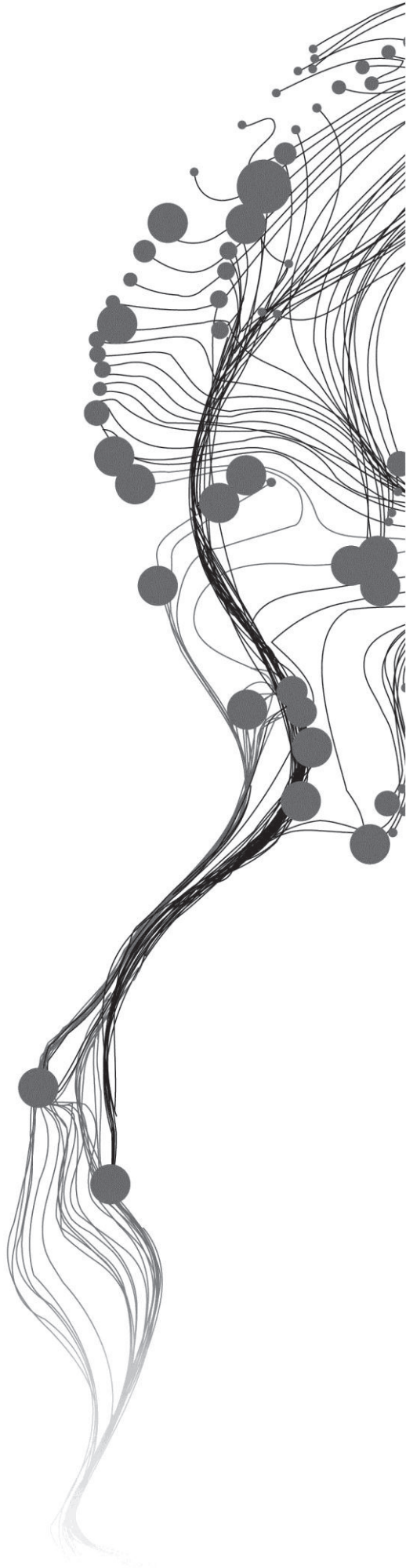


# **A PROTOTYPE FOR THE EXPLORATION OF NEOGEOGRAPHIC POINT DATA WITH CARTOGRAPHIC VISUALIZATION TOOLS**

HOANG ANH DUC  
February, 2012

SUPERVISORS:  
Dr. C.P.J.M. van Elzakker  
Prof.Dr. M.J. Kraak  
Mr. Tanmoy Das (Advisor)



# **A PROTOTYPE FOR THE EXPLORATION OF NEOGEOGRAPHIC POINT DATA WITH CARTOGRAPHIC VISUALIZATION TOOLS**

HOANG ANH DUC

Enschede, The Netherlands, February, 2012

Thesis submitted to the Faculty of Geo-Information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.

Specialization: Geoinformatics

## **SUPERVISORS:**

Dr. C.P.J.M. van Elzakker

Prof.Dr. M.J. Kraak

Mr. Tanmoy Das (Advisor)

## **THESIS ASSESSMENT BOARD:**

Prof.Dr. M.J. Kraak (Chair)

Dr.I.Delikostidis (External Examiner, University of Munster)

#### DISCLAIMER

This document describes work undertaken as part of a programme of study at the Faculty of Geo-Information Science and Earth Observation of the University of Twente. All views and opinions expressed therein remain the sole responsibility of the author, and do not necessarily represent those of the Faculty.

# ABSTRACT

As a result of highly developed information technology, especially after the advent of Web 2.0, people are engaged in contributing any kind of contents (text, photo, video etc.) to the web. We refer to these contents as user-generated content (UGC). UGC applied to location information can be termed user-generated geo-content (UGGC). This phenomenon of creating and sometimes visualizing UGGC as layers on top of base layers from, for example, Google Maps, OpenStreetMap, GeoCommons Maps and Yahoo Maps is known as neogeography and the visualization products can be called neogeography maps. However, neogeography maps have some problems in visualization such as cluttering and using incorrect colours, shapes and sizes of symbols. In this research, web visualization libraries are used together with cartographic design rules to solve those problems in order to make neogeography maps more easily to interpret and to get some meaningful information from the exploration of UGGC.

The thesis starts with the terminologies that surround the concepts of neogeography and cartographic visualization, followed by an overview of uses and users of neogeography maps. Next, the requirements for the exploration of neogeographic point data with the help of maps are discussed. Then, this research gives a description of the design and implementation of a prototype in which web visualization libraries and cartographic design rules are applied. Finally, the evaluation of the prototype is described.

The results show that by applying cartographic design rules, the problems mentioned above may be alleviated. The results also show that with the help of web visualization libraries and the application of cartographic design rules, the prototype in my thesis may have a positive influence on the exploration of neogeographic data. The prototype provides a set of maps with some functions that can help users to answer different geographic questions for various purposes.

**KEYWORDS:** Neogeography, Cartographic Design, Web Visualization Libraries, User-Generated Geo-Content (UGGC), Volunteered Geographic Information, Cartography.



## ACKNOWLEDGEMENTS

First of all, I am grateful to the Vietnamese government, the leaders of Hanoi University of Mining and Geology (HUMG), the dean of Faculty of Information Technology in HUMG who gave me the chance to study at ITC and encourage me to study abroad in the Netherlands.

Secondly, I would like to express my sincere gratitude and admiration to my supervisors, Asst Prof Dr. Corné van Elzaker (First Supervisor), Prof. Dr. Menno-Jan Kraak (Second Supervisor), and my advisor PhD student Tanmoy Das for their invaluable advices, encouragements, guidance and supports during my research. Their advices and constructive criticism lead me on the right track.

Thirdly, I would like to thank Dr. Ulanbek Turdukulov who supervised me during the time of modules 14 and 15, and gave me the first idea of using timemap, timeline in my MSc thesis.

Next, I want to thank Mr Nick Rabinowitz who has developed the Timemap API library for his supports and sharing the code.

I want to thank Ms. Mosa Timeletso Moseme for sharing her works during the thesis with me.

I am thankful to Mr. Nguyen Hoang Long for giving me many advices in programming with Timemap API.

I want to thanks Mr. Willy Kock for the assistance he provided in the usability lab.

I would like to thank Ms. Irma Kveladze, Mr. Ahmed Ibrahim, Mr. Gaurav Singh, Mr. Rehmat Ullah, Ms. Clarisse Kagoyire and Ms. Xiaojing Wu for participating the usability test of my research.

I want to say thanks to all of my Vietnamese friends in Enschede as well as my ITC friends for all the moments we shared together.

Lastly, I want to give my best gratitude to my family and my close friends, especially Ms. Huynh Ngoc Trinh, Ms. Tran Hai Phuong, Ms. Dang Anh Nguyet, Mr. Tran Truong Giang, Mr. Tikaram Baral, Mr. Deepak Raj Awasthi, Ms. Ugyen Eden, Mr. Pema Wangda and Mr. Arun Rai who support, encourage me and give me confidence in difficult moments.

# TABLE OF CONTENTS

---

1.	Introduction.....	1
1.1.	Motivation and problem statement.....	1
1.2.	Research identification .....	3
1.3.	Project set-up .....	4
1.4.	Thesis structure.....	5
2.	Cartographic visualization of neogeographic data.....	7
2.1.	Introduction .....	7
2.2.	Volunteered geographic information .....	7
2.3.	Neogeography .....	8
2.4.	Neogeographic data .....	9
2.5.	Cartographic visualization.....	11
2.6.	Neogeography maps .....	12
2.7.	Uses and users of neogeography maps.....	14
2.8.	Summary .....	15
3.	Requirement analysis for the exploration of neogeographic point data with the help of maps .....	16
3.1.	Introduction .....	16
3.2.	Case study data.....	16
3.3.	Tasks analysis and translation into views and functions .....	21
3.4.	Client side and database server support .....	25
3.5.	Summary .....	26
4.	Basic concepts for the design and implementation of the prototype.....	27
4.1.	Introduction .....	27
4.2.	Cartographic design rules .....	27
4.3.	Web visualization libraries .....	30
4.4.	Summary .....	36
5.	Design and implementation of the prototype.....	37
5.1.	Introduction .....	37
5.2.	Conceptual model of the prototype.....	37
5.3.	Set up of the database .....	38
5.4.	PHP server script and the PostgreSQL database server.....	39
5.5.	Implementation of the web visualization libraries.....	42
5.6.	Summary .....	52
6.	Prototype evaluation .....	53
6.1.	Introduction .....	53
6.2.	Evaluation.....	53
6.3.	Results .....	56
6.4.	Summary about the usability of the prototype .....	59
6.5.	Limitations of the test.....	60
6.6.	Limitations of the prototype .....	60
6.7.	Recommendation for the improvement of the prototype .....	60
6.8.	Summary .....	61
7.	Conclusion and recommendations.....	62
7.1.	Conclusion .....	62
7.2.	Recommendations for further research.....	63
	APPENDICES .....	69
	APENDDDIX A: Scripts and codes .....	69

APPENDIX B: Tables in the research .....	95
APPENDIX C: Figures in the research .....	96
APPENDIX D: Usability test plan, tasks, questions and results.....	101

## LIST OF FIGURES

---

Figure1.1: Examples of web-map with generalization problems [URL 1, URL 2] .....	2
Figure1.2: Example of map with symbol size problem [URL 10] .....	2
Figure2.1: Crisis map of Haiti; Source [URL 15].....	8
Figure 2.2: Examples of Wikimapia [URL 16] (left), Flickr map [URL 17] (center) and Google MyMaps [URL 18] (right) .....	8
Figure 2.3: Examples of neogeographic data's dimensional properties, upper left: point (source: [URL 20] , right: line (source: [URL 29]) and lower left: polygon (source: [URL 30]).....	10
Figure 2.4: Example of 3D visualization of a riding bike's track in Google Earth; Source of .GPX file that used to store the track's data: [URL 32].....	11
Figure 2.5: Neogeography maps - Google Maps [URL 2] with geotagged photos (left) and Haiti Earthquake reports map [URL 15]with the numbers of reports of different areas are shown in the circles(right) .....	12
Figure 2.6 The neogeography maps with 4 topics (hotel, restaurant, forest and school) – Original data (left) and the results of using Circle Tool to solve cluttering problem (middle and right); (MLAY, 2010). 13	
Figure 2.7: Map of hospitals in Pakistan; Source: (DAS & KRAAK, 2011a).....	13
Figure 2.8: The London Profiler interface with vector data (left) and with KML file data (right); Source: (GIBIN et al., 2008) .....	14
Figure 2.9: San Diego Wildfires 2007 interactive fire map; Source [URL 38] .....	15
Figure 2.10: San Diego Wildfires 2007 interactive fire map, shows the structures burned in the region with zip code 92025; Source: [URL 38] .....	15
Figure 2.11: Relationship between types of map use and neogeography map; Source: (DAS & KRAAK, 2011a) .....	15
Figure 3.1: Haiti map with earthquake epicenter and major cities affected; Source: [URL 31] .....	16
Figure 3.2: Crisis Map of Haiti; The numbers in the red circles show the total number of reports submitted in the particular administrative areas where the red circles are centers; Source [URL 15] .....	17
Figure 3.3: Original data of Haiti earthquake reports in CSV format.....	18
Figure 3.4: Haiti boundary map, 10 departments (left) and 41 provinces (right); shown on Arc Map 10.3.21	
Figure 3.5: Questions while exploring spatio-temporal data (PEUQUET, 1994) .....	21
Figure 3.6: Examples of map view (left – source [URL 2]), temporal view (middle – source [URL 36]), and attribute view (right – source [URL 2]) .....	22
Figure 3.7: Map of food newspapers in United State at Google Map zoom level 4; Source [URL 37] .....	22
Figure 4.1: Bertin's six basic graphic variables; Source: (KRAAK & ORMELING, 2009) .....	28
Figure 4.2: Example of Google Maps API and mapmarker.js library; Source: [URL 46].....	31
Figure 4.3: Example of using animation in OpenLayers [URL 12] .....	32
Figure 4.4: Flags of the World map using SIMILE and JSON data type; Source: [URL 23].....	32
Figure 4.5: Department of Peacekeeping Operations (DPKO) missions map using Timemap API; Source: (DUC & MOSEME, 2011) .....	33
Figure 4.6: Map of Shopko Hometown, Walmart, and Target locations, using GeoCommons API; Source: [URL 15].....	34
Figure4.7: Infant mortality rate (0-1 year), statistics from UNdata, year 2005, using Thematic Mapping API; Source [URL 25].....	35
Figure 4.8: Example of using Google Chart API to create a pie chart; Source [URL 40] .....	35
Figure 5.1: Conceptual model of visualizing neogeographic data set of Haiti earthquake using web visualization libraries .....	37

Figure 5.2: Ray casting algorithm's demonstration .....	40
Figure 5.3: Demonstration of algorithm applied for grids .....	41
Figure 5.4: Examples of symbols; 1: point symbols on mapview, 2: point symbols on timeline, 3: proportional point symbols, 4: area symbols .....	42
Figure 5.5: The timeline's interface .....	45
Figure 5.6: Original pop-up window .....	46
Figure 5.7: Improved pop-up window .....	46
Figure 5.8: Infowindow with pie chart .....	47
Figure 5.9: Registration pages .....	47
Figure 5.10: Log-in action results .....	48
Figure 5.11: Interface of the prototype's home page, with nominal point symbols map .....	50
Figure 5.12: Interface of prototype showing choropleth map with options showpast=1 and showgrids=0 .....	50
Figure 5.13: Interface of prototypeshowing choropleth map with grid network .....	51
Figure 5.14: Interface of prototype showing proportional symbols map with grid network .....	51
Figure 5.15: Interface of prototype showing proportional symbols map on departments .....	52
Figure 6.1: Screen shot of users' list page .....	57
Figure 6.2: Screen shot of one participant who gave wrong answer in question 2 of task 2 .....	58

## LIST OF TABLES

---

Table 3.1: Summary of tasks (based on the Visual Information Seeking Mantra spread by SHNEIDERMAN (1996)), functions and views .....	23
Table 3.2: Map use tasks with relation to the geographic questions in the case of Haiti earthquake case study data. ....	24
Table 3.3: List of tasks, required views and functions that will be created in this research .....	25
Table 4.1: Thematic map types and the basic geographic questions to which they provide an answer .....	30
Table 4.2: Criteria to choose the web visualization libraries.....	30



# 1. INTRODUCTION

## 1.1. Motivation and problem statement

A map is a very important and useful tool in our life. KRAAK & BROWN (2001) defined a map as a graphic representation of our environment. Maps are used to visualize geospatial data, which refer to the location and the attributes of objects or phenomena on the Earth (KRAAK & ORMELING, 2009). They have been used for thousands of years and have proved to be very necessary to human life. It can be said that people cannot live comfortably without maps nowadays.

As the result of highly developed information technology, especially after the advent of Web 2.0, people are engaged in contributing any kind of contents (text, photo, video etc.) to the web. We refer to these contents as user-generated content (UGC). UGC applied to location information can be termed user-generated geo-content (UGGC). This phenomenon of creating and sometimes visualizing UGGC as layers on top of base layers from, for example, Google Maps, OpenStreetMap, GeoCommons Maps and Yahoo Maps is known as neogeography and the visualization products can be called neogeography maps (DAS & KRAAK, 2011b).

TURNER (2006) defines 'neogeography' as a 'new geography' and it consists of a set of techniques and tools that fall outside the realm of traditional GIS (Geographic Information Systems). Neogeography combines the complex techniques of cartography and GIS and places them within the reach of users and developers. Turner (2006) says that "neogeography is about people using and creating their own maps, on their own terms and by combining elements of an existing toolset". There are many types of neogeographic data such as point data, line data, polygon data that are generated by the volunteers. The data will be store on the database server or in files before it is used by the map users to visualize through map displays.

The users of neogeographic map consist of researchers, neogeographers, geographers and the internet map users. Ones, especially who want to build a prototype for the exploration of neogeographic data through cartographic visualization have to deal with some general issues when visualizing neogeographic data on the web. These are, for instances: What are the purposes of neogeographic maps? What do the map users of their product need in a good neogeographic map (for specific purposes)? What cartographic design techniques can be applied in neogeography to improve the current neogeography maps?

Neogeography maps are easy, fast and cheap to create and to be disseminated when compared to conventional maps. Therefore, neogeography can be seen as an alternative or additional source of Geographic Information (maps) for map's users (DAS & KRAAK, 2011b).

However, neogeography maps also have some problems in visualization that need to be solved for the users in order to be able to answer questions like what, when, where and how. Firstly, the neogeography maps have cluttered problem which makes difficulty to interpret or to derive meaningful information from these neogeographic maps. Sometimes, this visualization problem makes the maps unreadable. For example, in Figure1.1 the left map is too cluttered; it is difficult for the user to execute tasks such as to perceive the spatial distribution in this map. In the right map, the designers tried to put as much information as possible into the map and that made the map unreadable as well.



A prototype for the exploration of neogeographic point data with cartographic visualization tools

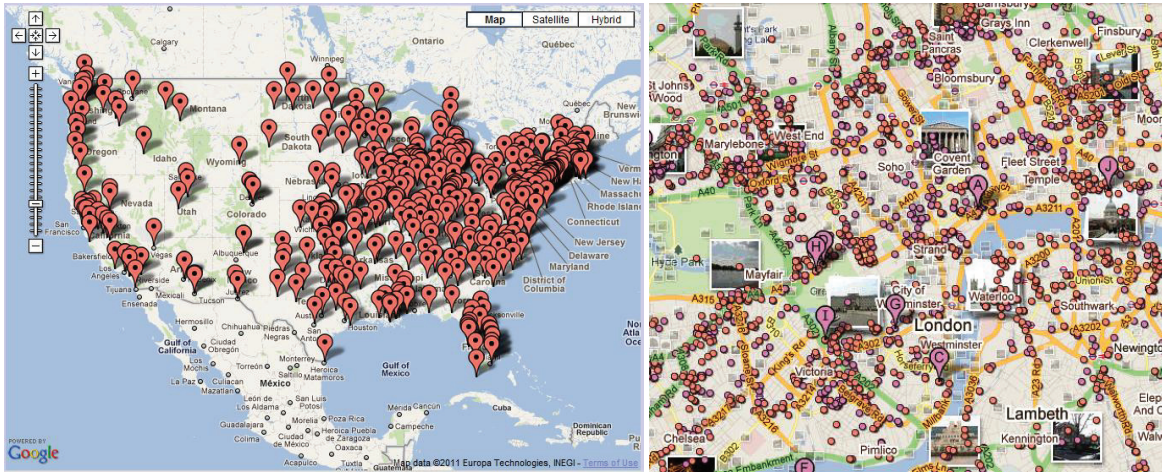


Figure1.1: Examples of web-map with generalization problems [URL 1, URL 2]

Secondly, the colours, shapes and sizes for the symbols in many of current neogeography maps are used incorrectly; therefore, the user may get the wrong information from the symbols. In Figure 1.2, the size of maps' symbols is too big and the sizes remain the same in different zoom-levels.

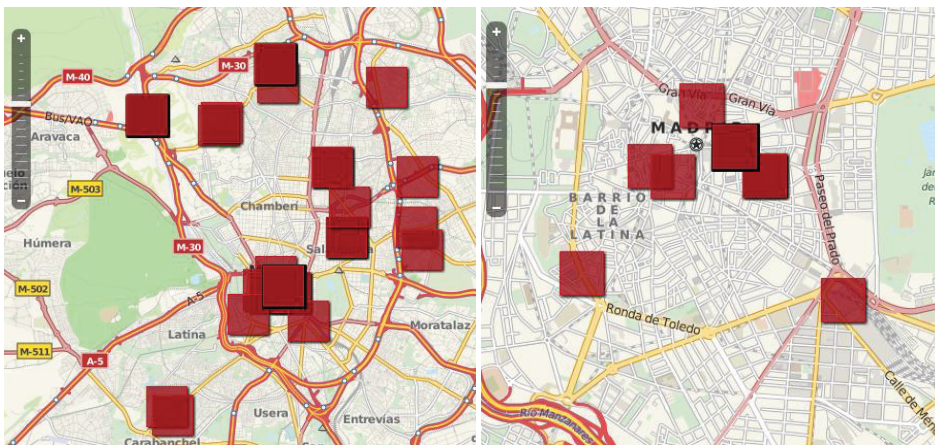


Figure1.2: Example of map with symbol size problem [URL 10]

A solution to the cartographic visualization of neogeographic data may be provided by so-called “interactive visualization libraries on the Web“. Recently, there are many web interactive visualization libraries available on the internet. These include JavaScript libraries such as the Google MAPS API [URL 3], OpenLayers [URL 4], SIMILE [URL 5], Timemap [URL 6], ESRI Javascript API [URL 7] and GeoCommons API [URL 8]. They offer many build-in functions for visualizing and analyzing data on the web page that can be applied as a solution of my thesis. Each of these libraries has its own advantages and disadvantages which need to be figured out to choose the most suitable ones for different geographic questions from the map users.

As mentioned above, there are many types of neogeographic data such as points, lines, polygons; they provide different information for different tasks. Due to the time limit, I decided to concentrate only on neogeographic point data as my research data objective. My research aims at providing a new prototype to visualize neogeographic point data by using one or more web visualization libraries and by applying cartographic design rules to meet the various users' purposes in several ways.

## **1.2. Research identification**

### **1.2.1. Research objectives**

The main objective of this research is to develop a prototype to explore neogeographic point data on top of already existing and freely available base maps, by using web visualization libraries, web programming and applying cartographic design rules to meet various users' purposes.

### **1.2.2. Sub-objectives**

The main objective can be reached by attaining the following sub-objectives:

1. To provide an overview of the different uses of neogeographic point data maps by formulating the geographic questions users may want to have answers to
2. To link these questions to specific cartographic visualization solutions.
3. To gain an overview of relevant existing open source web visualization libraries for visualizing neogeographic data.
4. To design a Graphical User Interface considering the required views and functions for the visualization of neogeographic data.
5. To evaluate the functionalities of the prototype prototype.

### **1.2.3. Research questions**

1. To provide an overview of the different uses of neogeographic point data maps by formulating the geographic questions users may want to have answers to
  - What are the questions users want to have answers to?
  - What are the tasks related to these questions?
2. To link these questions to specific cartographic visualization solutions.
  - What cartographic visualization solutions can be used to answer these questions?
3. To gain an overview of relevant existing open source web visualization libraries for visualizing neogeographic data
  - Which web visualization libraries already exist and what are their characteristics?
  - What are the criteria for choosing a suitable web visualization library for the purpose of this research?
  - Which web visualization library is most suitable for the purpose of this research?
4. To design a Graphical User Interface considering the required views and functions for visualization of neogeographic data
  - Which ways of cartographic visualization are required?
  - Which functionalities are required?
  - How to design and implement a research prototype prototype, taking user tasks and data characteristics into account?
5. To evaluate the prototype
  - How to evaluate the prototype?
  - What are the outcomes of the evaluation of the prototype?

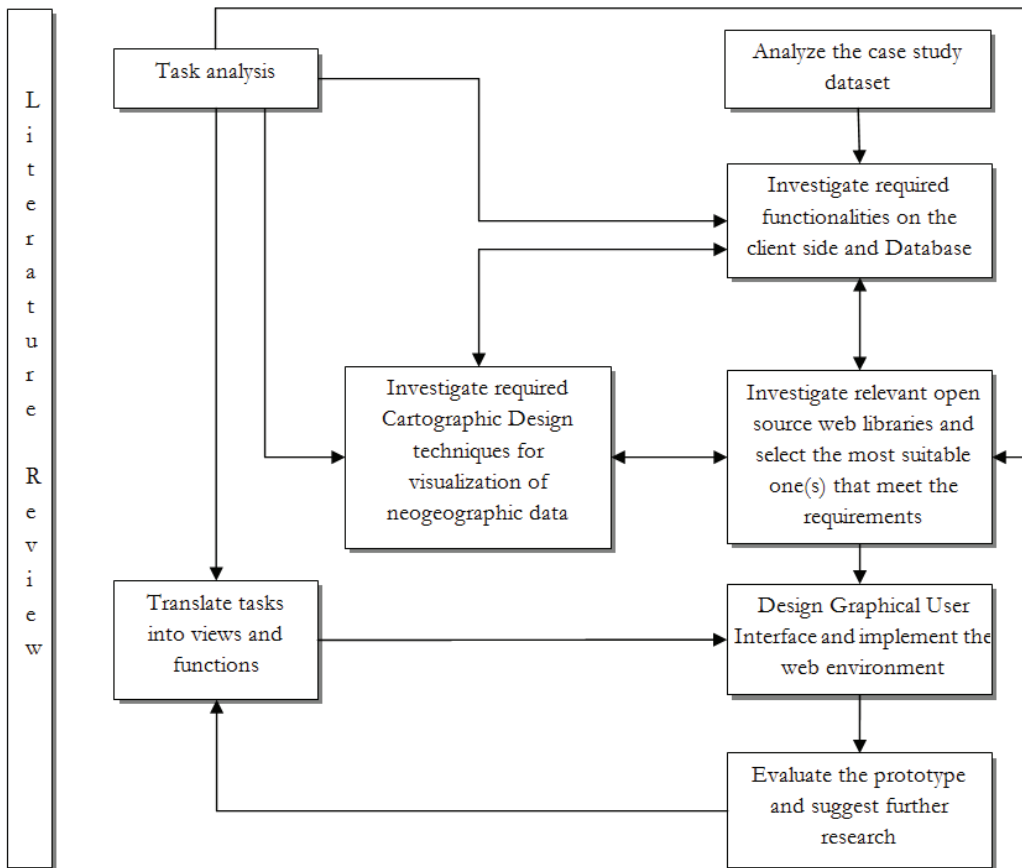
### **1.2.4. Innovation aimed at**

Existing web applications such as GeoCommons Maps, Google Maps, OpenStreetMap and Yahoo Maps provide tools for users to make their own map by their own design from their own data. However, the design is usually used without applying cartographic design rules that lead to the problems which are

mentioned before. The innovation aims at creating a prototype to explore neogeographic point data on top of already existing base map layers by using web visualization libraries, web programming and applying cartographic design rules to meet the various users' purposes in several ways.

### 1.3. Project set-up

#### 1.3.1. Method adopted



- *Literature review:* The first task will be to review literature on cartographic design rules in neogeography and web programming with web visualization libraries. Literature review also will be done in every phase.
- *Task analysis:* By reviewing related literatures about which tasks are needed in exploring neogeographic data or interviewing some representatives of target users such as researchers, cartographers, neogeographers and map users.
- *Translate tasks into views and functions:* in this step, required tasks will be translated into views and functions.
- *Analyze the available data set (neogeographic data set):* data set from the website [haiti.ushahidi.com](http://haiti.ushahidi.com) [URL 15] will be used as my case study. Hence, to get the understanding of the structure of these data is very important in order to use the correct ways of visualization for various data's nature.
- *Investigate relevant open source web visualization libraries and select the most suitable one that meets the requirements:* Each web visualization libraries will be investigated by looking at its examples and functionalities. The most suitable web visualization library (libraries) will be chosen based on the needed views and

functions. However, due to the fact that there are many interactive web visualization libraries and the limited time, not all the web libraries will be investigated.

- *Investigate required functionalities on the client side and database:* investigate required functionalities on the client side and database while considering characteristics of the selected web visualization library (libraries).
- *Investigate required cartographic design techniques for visualization of neogeographic point data:* Investigate required cartographic design techniques from the literatures such as the techniques/map operators in map generalization rules for cartographic symbol design. This is a very important step which decides the output would indeed be better than the current neogeographic maps.
- *Design Graphical User Interface (GUI) and implement the prototype:* A visualization interface prototype will be implemented. Firstly, the data will be prepared by using scripts. After that, the data will be processed by using scripts based on different cartographic techniques, use aggregation operator on the data for example. Finally, the processed data will be put on the map by using web visualization libraries.
- *Evaluate the prototype and suggest for further research:* Finally, the prototype will be tested in order to determine its capabilities and limitations. The user research methods can be used are “think aloud”, “interview”, “video observation”, “screen logging”, and “eye tracking” which can be operated in ITC. After that, with regard to the result of this research, some suggestions are made for further research.

#### 1.4. Thesis structure

This thesis contains seven chapters. Chapter 1 describes the motivation, problem statement, research objectives, research questions and methodology.

In the second chapter, the characteristics of volunteered geographic information, neogeography as well as neogeography data will be examined. Next, some examples of neogeography maps will be observed. Finally, the uses and users of neogeography maps will be discussed.

Chapter 3 will review the requirement for the use of neogeographic point data. Then the general tasks and the tasks that are especially required while exploring neogeographic data will be investigated and translated into views and functionalities. Finally, the issue about functions from the database and the client side will be discussed.

The basic concept for the design and implementation of the prototype will be mentioned in chapter 4. This chapter will describe some cartographic design rules that will be applied into the prototype in design map and investigate some web visualization libraries on their characteristics, advantages as well as disadvantages. Then, based on their characteristics, advantages, disadvantages and requirements which will be mentioned in chapter 3, the most suitable way to implement web visualization libraries into the prototype will be chosen.

Chapter 5 will describe the implementation process of the prototype step by step. Solutions to questions of how to use web open source visualization libraries to visualize neogeographic data taking the cartographic rules into account, as well as how to design a GUI for the prototype will be described.

A short evaluation of functionality of the prototype will be conducted in chapter 6. The evaluation will be based on capabilities and limitations of the prototype from the experience of test participants. From the evaluation, some recommendations for improvement of prototype will be discussed.

Finally, some conclusions and recommendations will be discussed in chapter 7.





## 2. CARTOGRAPHIC VISUALIZATION OF NEOGEOGRAPHIC DATA

### 2.1. Introduction

The technologies are more and more inexpensive, and even simpler to use nowadays than before. Individuals are using those technologies to create their own data, make their own map and/or share their data on the internet. The way of creating, disseminating and visualizing geographic data has changed. This chapter shows the important elements of cartographic visualization of neogeographic data. First, the term “Volunteered geographic information”, a special case of user generated content will be described. After that, the new geography term, namely, neogeography will be mentioned. Next, neogeographic data with its characteristics will be discussed. Finally, neogeography maps will be reviewed with reference to some existing works in relation to the cartographic visualization of neogeographic data.

### 2.2. Volunteered geographic information

GOODCHILD (2007) coined the term “Volunteered geographic information (VGI)”, a special case of user-generated content, to define this remarkable phenomenon: “the widespread engagement of large numbers of citizens, often with little in the way of formal qualifications, in the creation of geographic information, a function that for centuries has been reserved to official agencies. They are largely untrained and their actions are almost voluntary, and the results may or may not be accurate. But collectively, they represent a dramatic innovation that will certainly have profound impacts on geographic information systems and more generally on the discipline of geography and its relationship to the general public”.

There are some examples of this phenomenon which are well-known such as Wikimapia [URL 16], Flickr [URL 17] and Google MyMaps [URL 18]. These websites provide general base map information and allow users to create their own content by marking locations where various events occurred or certain features exist, but are not already shown on the base map. The users of Wikimapia and Google MyMaps can select an area on the Earth’s surface and provide it with a description which may have links to other sources. Everyone can also edit entries and volunteer reviewers monitor the results, check for accuracy and significance. In a similar way, the Flickr site allows its users to upload and locate photographs on the Earth’s surface by latitude and longitude (GOODCHILD, 2007).

A different example of VGI is the “Crisis map of Haiti” [URL 15] which represents the most comprehensive and up-to-date crisis map of Haiti which is available to the humanitarian community. The information here is mapped in near real time and gathered from reports coming from inside Haiti via: SMS, Web, Email, Radio, Phone, Twitter, Facebook, Television, List-serves, Live streams, Situation Reports. Volunteers at Ushahidi’s Situation Rooms at the Fletcher School, in Washington DC, Geneva, London and Portland are mapping the majority of the reports submitted to Ushahidi in near real-time [URL 15]. Then the volunteers identify GPS coordinates for the reports and geo-tag the reports on the Ushahidi crisis map (see Figure 2.1).

A prototype for the exploration of neogeographic point data with cartographic visualization tools

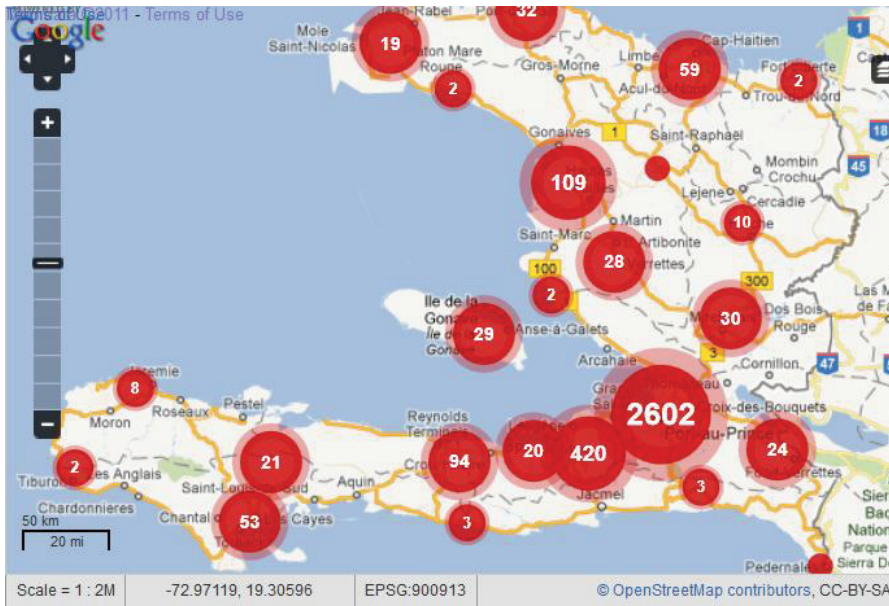


Figure 2.1: Crisis map of Haiti; Source [URL 15]

### 2.3. Neogeography

While geographic information systems (GIS) software are still expensive programs and are mostly used by highly-trained specialists, websites like Wikimapia, Flickr and Google MyMaps brought easy-to-use and free mapping tools to the public. One of the characteristics of neogeography is the term mashup that refers to websites that weave data from different sources into a new integrated single user service (HUDSON-SMITH et al., 2009). For example, the release of the Google Maps API enables users to mix Google base map data with other spatially referenced material such as geo-tagged photos, GPStracks, etc... Figure 2.2 shows examples of Wikimapia, Flickr and Google MyMaps websites which are currently available on the internet.



Figure 2.2: Examples of Wikimapia [URL 16] (left), Flickr map [URL 17] (center) and Google MyMaps [URL 18] (right)

Due to the development of technology, specifically Web 2.0, online mapping using user data has been possible. A non-expert is able to create his/her own maps. The term neogeography was defined the first time by TURNER (2006) as “‘new geography’ consisting of a set of techniques and tools that fall outside the realm of traditional GIS, Geographic Information Systems”. He says “neogeography combines the complex techniques of cartography and GIS and place them within the reach of users and developers”. Turner mentions it is about sharing location information with friends and visitors, helping shape context, conveying understanding through knowledge of place and neogeography is fun.

Nowadays, it is possible for maps to be periodically updated, real time, dynamic and even interactive. The use of satellite imagery and aerial photographs can be considered as the quickest way of obtaining large coverage data for mapping. However, not all information can be sensed or interpreted on satellite images or aerial photographs. For example, in consideration of some of the sudden disasters such as earthquakes, floods, and forest fire, it may happen at the moment when no earth observing satellites pass over the affected area for a number of hours or days. Moreover, images from satellites and aircraft may be obscure by the coverage of clouds and smoke. The condition on the ground may prevent the rapid downloading of digital imagery due to lack of power, internet connection or computer hardware and software (GOODCHILD, 2007). Individuals who are staying or being at these places can report the condition through explaining, voice, mobile phones, pictures, text and audio or video recordings. The availability of this kind of data, also called neogeographic data, which can be used as a replacement of the conventional data mentioned above, may reduce unwanted problems. For example, in emergency events as mentioned above, if the rescue forces have to wait for hours or days, the problems could be extremely serious and unsolvable. However, with the availability of neogeographic data such as messages, pictures, audio or video recordings, the rescue force will be informed immediately after the information is received. Thus, neogeography seems to be a better source of data in this situation.

Neogeography is not limited to a specific technology and is strictly not Web-based. According to D'LOG (2008), the term neogeography has been used since at least 1922. In the early 1950s in the U.S, this term was used in the sociology of production and work. Neogeography was mentioned in the title of a French philosopher François Dagognet's book as "Une Epistémologie de l'Espace concret: Neo-geographie" (An Epistemology of Espace concret: Neogeography) (DAGOGNET, 1977). This word was first used in relation to the study of online communities in the 1990s by Keneth Dowling, the Librarian of the City and County of San Francisco (BROOK et al., 2001). Thus, it cannot be said that neogeography is synonymous with web mapping, although they are commonly conceived as such. My research will only focus on the term of neogeography, which is commonly used in relation to the web 2.0 environment nowadays.

## 2.4. Neogeographic data

Nowadays, the high technology has made the work of collecting geographic data to be very simple and easy for every individual who has GPS enabled devices such as mobile phones, cameras, GPS, navigation systems, etc. These devices offer the individual possibilities for geotagging and, for instance, sharing their photos. TURNER (2006) gave an example of geotagging in adding location information to a document, a photograph, an audio sample, or some other type of data. These data usually consist of latitude and longitude coordinates which are commonly used for photographs, giving geotagged photographs; they can also include altitude, bearing, distance, accuracy data, and place names. However, if a mobile phone does not have a build-in GPS chipset, how can a photograph acquired by this device then be geolocated? In this case, the position of the photograph is obtained based on various other data such as mobile internet, or from mobile networks. Therefore, an individual can use the coordinates from these data without a GPS. These data may then be disseminated to the web where they are called neogeographic data. GPS and cell-phone location systems are not the only way of bringing in the geo-component of neogeographic data. It can also be distributed on the internet by using web 2.0 environments such as blog post, uploading photos, uploading personal tracks, etc by internet users.

All of these data acquired voluntarily by users have different characteristics:

- Measurement levels and scales: neogeographic data can both be qualitative and quantitative. For example, neogeographic data has been shown in different categories of hotel, restaurant, forest and school (qualitative data) in (MLAY, 2010) or shown the number of beds in Pakistan's hospitals (quantitative data) in (DAS & KRAAK, 2011a). According to KRAAK & ORMELING (2009), the measurement scales are



traditionally divided into nominal, ordinal, interval and ratio scales. Nominal scale is used to measure the differences in data which are only of a qualitative nature. For instance, differences in gender, language, religion, land use, etc is measured by nominal scale. To differentiate the data of quantitative nature, ordinal, interval and ratio scales are used for the measurement. Examples for ordinal scale are ‘small – medium – large’ or ‘cool – tepid – hot’; for interval scale, the examples are the measurement of temperature data (0°C, 10°C, 100°C, etc.) or the measurement of height (0m, 4m, 8m, etc.). According to KRAAK & ORMELING (2009), “when ratios can be expressed, one refers to data that can be measured on a ratio measurement scale”. For example, in 2010 the purchase power parity of the Netherlands was 2.5 times more than that of Vietnam, so these are ratio data. Ratio data are absolute or relative ratio data. Absolute ratio data are the result of direct measurement or additions of units. Examples for absolute data are one’s income in money (a specific number of currencies) or the number of total population of a country. Relative ratio data are absolute ratio data related to other data sets, and relative ratio data often tell us more than absolute ratio data, because they have been put into context. For instance, the income of 15000 EUR per year is a high salary for an IT manager in Vietnam in 2010; but a similar income might not be enough for the minimum salary of a worker in the Netherlands.

- Neogeographic data’s dimensional properties can be found as point, line and polygon (see Figure 2.3). Neogeographic point data is the most common form of neogeographic data which are used in top layers of neogeography maps. Neogeographic line data is used on many maps such as the tracks of an outdoor activity in the “Map my tracks” website [URL 21]. Neogeographic polygon or area data are usually used as the base layers of maps; Google Maps [URL 2] and OpenStreetMap [URL 19] are famous examples of neogeographic data in the form of polygon data. Because the main objective of my research is to develop a prototype to explore neogeographic point data on top of already existing and freely available base maps; in my research, I will focus on neogeographic point data to show as thematic maps on top layer of the map and use base map from sources like Google maps, Google Earth, OpenStreetMap etc...

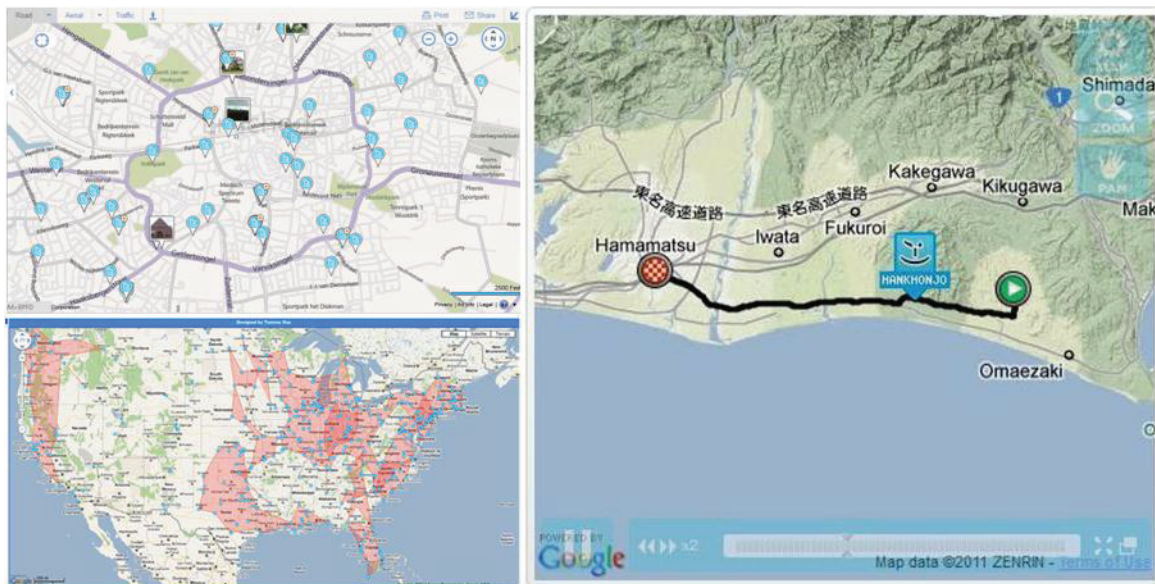


Figure 2.3: Examples of neogeographic data’s dimensional properties, upper left: point (source: [URL 20] , right: line (source: [URL 29]) and lower left: polygon (source: [URL 30])

- Generally, neogeographic datasets contain temporal information. For example, the data characteristic that the objects were captured or created on a particular date/time can be used as temporal data. That means, we that we can use a 4-Dimensional space to show neogeographic data (3-Dimensional space for spatial data (see Figure 2.4) and one additional dimension for temporal data).



Figure 2.4: Example of 3D visualization of a riding bike's track in Google Earth; Source of .GPX file that used to store the track's data: [URL 32]

- One of the important characteristics of neogeographic data is accuracy. COOTE & RACKHAM (2008) mention positional accuracy and temporal accuracy of neogeographic data:

- + Positional accuracy: the accuracy of the position of features or geographic objects whether in two or three dimensions. Positional accuracy is important, for example in a point of interest. If the positional accuracy is low, for example, if the coordinates of a restaurant are considered as the coordinates of the centre of a large postcode area it is very difficult to find exactly where the restaurant is located.
- + Temporal accuracy: the accuracy of the temporal attributes (dates and times) and temporal relationships (for instance, later or earlier than) of features. This is also a very important aspect of neogeographic data. For instance, the data will be invalid if the creation date of an event is generated after its dissolution date. In another example, if the time of time measurement is incorrect, it might happen that a report is recorded in the wrong time according to the error of the recording device that will cause the wrong information to be shown to the users.

## 2.5. Cartographic visualization

There are a number of various definitions and terms used for the concept of cartographic visualization, each with slightly different nuances in meaning, but they generally refer to spatial data analysis. BAILEY & GATRELL (1995) state that “visualization means mapping”. They distinguish visualization from exploratory spatial data analysis by the degree of data manipulation required. Visualization and exploratory spatial data analysis occur on a continuum of varying levels of sophistication, they contend, so the boundary between the two methods becomes blurred. According to MacEachren (1995), visualization works through an iterative process of comparing observations with knowledge, a “seeing that-reasoning why” cycle, unlike the linear nature of language that forces a sequential presentation. In the process of visualization, “there is a continual give-and-take between vision and visual cognition through the intermediary of knowledge schema” (MACEACHREN, 1995). MacEachren created a model that encompasses the traditional map and its use as a tool for communication in the public realm, but it also emphasizes the value of cartographic visualization in truly understanding geographic phenomena. DORLING & FAIRBAIRN (1997) define “scientific visualization” as “the process of learning through the creation and observation of abstract images, providing a method for seeing the unseen”. According to them, the aim of visualization is to go beyond illustration of what is known to the discovery of what is unknown. CRAMPTON (2001) uses the term “geographic visualization” which he defines as “the map’s power to explore, analyze and visualize spatial datasets to understand patterns better”.

Although the terms for cartographic visualization may differ, the definitive goal is the understanding of spatial information and knowledge through interactive visual display. Along with cartographers, geographers in general must be aware of the importance of cartographic visualization to more completely understand spatial data. In my thesis, the term “cartographic visualization”, as defined by MacEachren, will be used.

## 2.6. Neogeography maps

According to DAS & KRAAK (2011a), a neogeography map combines online (base) maps with UGC data. In other words, neogeography maps are often created by using two types of layers, base layers and top layers. The base layer holds the raster base maps containing images, maps or both with a toggling facility. Those base maps are provided by several organisations such as Google Maps [URL 2], OpenStreetMap [URL 19], Microsoft Bing Maps [URL 20] etc. The top layer contains the UGC. Figure 2.5 shows examples of some neogeography maps.

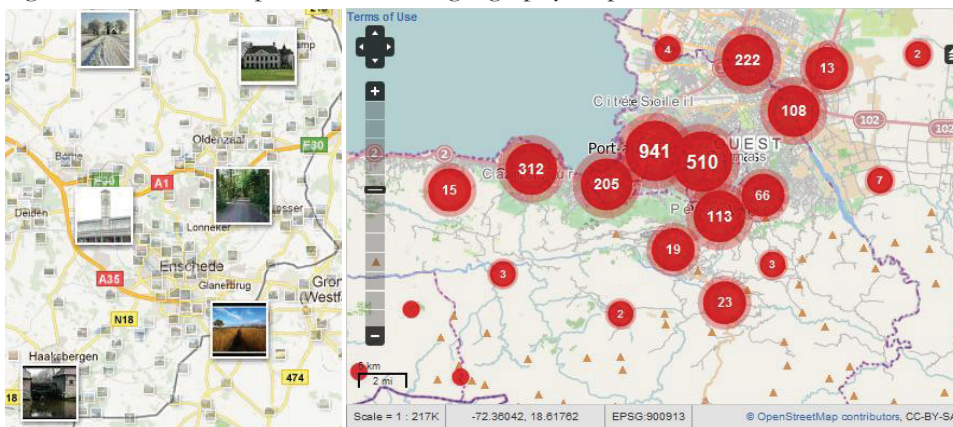


Figure 2.5: Neogeography maps - Google Maps [URL 2] with geotagged photos (left) and Haiti Earthquake reports map [URL 15] with the numbers of reports of different areas shown in the circles (right)

Neogeography maps can be used to present or store the collected data or to explore or consult the available data on the maps. Some users collect data of objects' location and display them on top of the base maps. For example, a user can create a map showing all places of earthquake reports in Haiti [URL 15]. Here the user wants to 'present' the collected data on a map. Other users might use the same map to explore where the reports' locations are to help the people who are in danger there.

There are some problems of neogeography maps that were discussed in the section 1.1 in chapter one, they need to be solved in order for the user to answer the cartographic questions from the map. Previously, there were some researches related to neogeography maps. MLAY (2010) uses two clustering methods, namely, k-means and DBSCAN, to solve cluttered problems of neogeography maps. Some bounding tools such as extent, rectangle, circle and convex hull were also used in summarizing. In Figure 2.6, the left map is the neogeography map with cluttered problem; Mlay uses the Circle tool to change the original point representation into polygon representation to solve the cluttering problem of neogeographic data. The middle map is the result with the original points data, and the right map is the result without these points. However, the solutions in this study need to be improved to make the result maps more informative for the users such as researchers or investigators who need to do research or find out about the data such as to solve the question “how many schools are located in a certain area?”. The other solutions could be giving the total number of points in each circle or the density of points in different areas. Moreover, the shapes of areas should be changed in order to make the cluttered problem really be solved. For example, by using Circle tool, the result is still overlapped and hard to get information from it. If we use alpha shapes as a replacement of circle shapes, the polygons created will not be overlapped



A prototype for the exploration of neogeographic point data with cartographic visualization tools

because each area of the result will be separated by its own boundary. This solution will make the result is more informative than that of Mlay's solution.

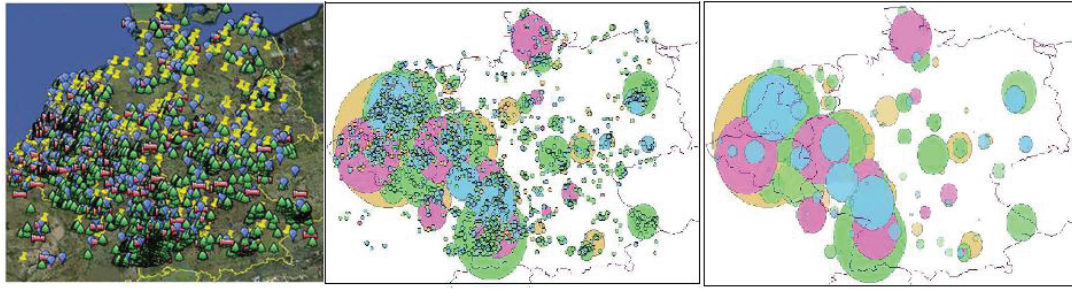


Figure 2.6 The neogeography maps with 4 topics (hotel, restaurant, forest and school) – Original data (left) and the results of using Circle Tool to solve cluttering problem (middle and right); (MLAY, 2010)

In other research, DAS & KRAAK (2011a) applied cartographic design rules using GeoCommons software on neogeographic data for exploratory tasks that were not possible to do on the source neogeography map. In the Figure 2.7, the left map is the original maps of hospitals in Pakistan with the data of capacity of hospital in Pakistan in 1136 records. The dataset of original neogeography map was converted to be used in GeoCommons. The maps generated can show the data in many ways such as the available beds are shown with visual variable value (see upper right map in Figure 2.7). The authors then analysed the data to make some tasks to be available, for example, they subtracted the number of available beds from that of total beds which produced bed occupancy of different hospitals in Pakistan (see lower right map in Figure 2.7). The work of showing bed occupancy cannot be done by using source neogeography data. This research was an attempt to prove that the application of cartographic design rules can improve neogeography maps in order to make them more informative and to be able to do some analysis tasks.

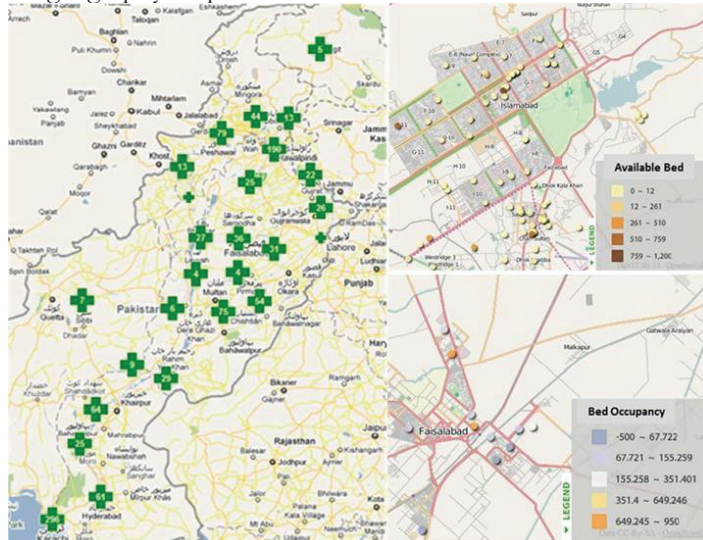


Figure 2.7: Map of hospitals in Pakistan; Source: (DAS & KRAAK, 2011a)

In the research of GIBIN et al. (2008), the authors described the ways in which the GMap Creator tool developed under the ESRC National Centre for e Social Science (NCeSS) programme enables users to 'mashup' thematic choropleth maps using the Google API. The London Profiler website [URL 28] made with this tool visualizes geographic information as a series of choropleth maps on top of Google Maps rather than as simple points (push pins). The purpose of the website is to engage with decision and policy-makers from a variety of audiences and encourage them to make more informed choices based on publicly available spatial information. By overlaying these data onto Google Maps data, this enables contextual information to be taken into account when making decisions (see the left map in Figure 2.8).

Google Maps API enables vector shapes to be overlaid on their map data; however this is limited to small datasets which are not suitable for extensive geographical areas. To solve this problem, the vector data can

be transformed into an image format similar to the Google background map, thereby enabling this information to be integrated seamlessly with Google Maps information. The London Profiler website provides a feature which enables users to incorporate their own data into London Profiler; that feature is the ability to load publicly available files in Google Earth standard (known as Keyhole Markup Language - KML) onto the map. For example, using KML feeds from Nestoria, a vertical search engine for finding UK property [URL 33], property information can be added to the London Profiler website, thus enabling contextual information to be considered when searching for a property.

GIBIN et al. (2008) used GMap Creator software on the London Profiler website's data to create a KML file which consists of data of houses for sale in London. After that, the houses for sale from the KML file can be displayed onto the map (the right map in Figure 2.8 shows the data which was provided to Nestoria by the houses' owners about their houses' price). This research is an example of using KML to show neogeographic data on a web map by using Google API. On the negative side, the map uses static ranges of colours and classification, and, therefore, the user cannot explore the information in more detail. Furthermore, the application is inherently cartographical and void of any analytical capacity.

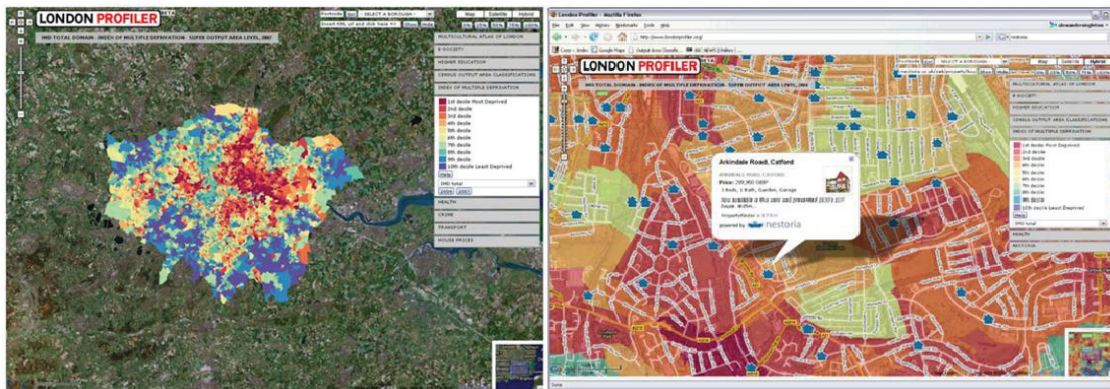


Figure 2.8: The London Profiler interface with vector data (left) and with KML file data (right); Source: (GIBIN et al., 2008)

In the three researches given above, different purposes were satisfied with the help of maps. MLAY (2010) solved cluttered problem by using bounding tools. DAS & KRAAK (2011a) used GeoCommons software to generate maps that were used for exploratory tasks which were not possible to do on the source map. GIBIN et al. (2008)' research is an example of using KML to show neogeographic data on a web map. In my research, with the objective of developing a prototype for the exploratory of neogeographic point data, I will use web visualization libraries, web programming and applying cartographic design rules to explore the neogeographic point data with the help of maps.

## 2.7. Uses and users of neogeography maps

DAS & KRAAK (2011a) mentioned two kinds of maps use, they are presentation and exploration. Some users collect data of objects' location to display them on top of base maps. An example is San Diego Wildfires 2007 interactive fire map [URL 38] which shows map of fire in San Diego, the eighth largest city in the United States in the year of 2007 (see Figure 2.9). The positional data of fire occurred in San Diego were collected and displayed on top of Google Map. Here, the map is used to present the collected data. This map might be used by the other users who want to explore the detail of fire in the specific place by using filtering or zooming to that specific region. Figure 2.10 is the example of showing all structures burned in the region of zip code 92025. According to DAS & KRAAK (2011a), both kinds of maps use (present and explore) might require different kinds of designs. Figure 2.11 shows the relationship between the two kinds of maps use and neogeography map. In my research, I will focus on the use of exploration

A prototype for the exploration of neogeographic point data with cartographic visualization tools

on neography maps which help the users to explore neogeographic point data on top of already existing and freely available base maps to meet various users' purposes.



Figure 2.9: San Diego Wildfires 2007 interactive fire map; Source [URL 38]



Figure 2.10: San Diego Wildfires 2007 interactive fire map, shows the structures burned in the region with zip code 92025; Source: [URL 38]

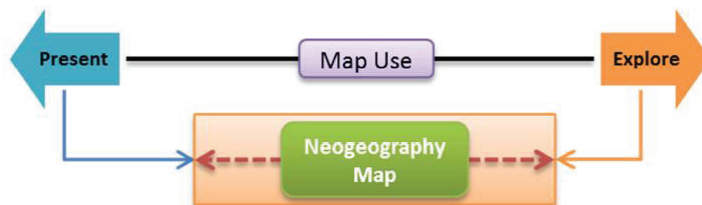


Figure 2.11: Relationship between types of map use and neogeography map; Source: (DAS & KRAAK, 2011a)

## 2.8. Summary

In this chapter, the terms VGI, neogeography, neogeography data, cartographic visualization, neogeography maps were reviewed from existing literatures. VGI and neogeography were considered as source of neogeographic data. Later, neogeographic data and its characteristics were mentioned. Next, some descriptions and examples of neogeography maps were reviewed from existing literatures to show what has been done in neogeography mapping for different purposes which is necessary as starting-point of my research. Finally, the uses and users of neogeography maps were mentioned. The next chapter will present the requirements analysis for the exploration of neogeographic point data with the help of maps and the overview of the case study data which will be used in my research.



### 3. REQUIREMENT ANALYSIS FOR THE EXPLORATION OF NEOGEOGRAPHIC POINT DATA WITH THE HELP OF MAPS

#### 3.1. Introduction

In the previous chapter, we have presented a general view of neogeographic data and on how people use digital devices to generate data and later share it on the internet. After that we have reviewed some characteristics of neogeographic data and some existing examples of neogeography maps. In this chapter, we will present a case study with neogeographic data. Next, the general tasks which need to be implemented in the prototype will be reviewed. Later, the tasks in visual exploration of neogeographic data and case study data will be analyzed. Then, some functions of the database server, server and client will be discussed. Finally, those tasks will be translated into required views and functions.

#### 3.2. Case study data

On 12<sup>th</sup> January 2010, an earthquake struck Haiti at 04:53 PM local time. The earthquake epicenter was 18.457° N, 72.533° W, 25 km WSW of Port-au-Prince on or near the Enriquillo fault (see Figure 3.1) (EERI, 2010a). According to EERI (2010b), the effects of the earthquake were felt over a wide area, including the provinces of Ouest, Sud-Est, and Nippes. The metropolitan Port-au-Prince region includes the cities of Carrefour, Petionville, Delmas, Tabarre, Cite Soleil, and Kenscoff was hit extremely hard. In the city of Léogâne, located on the epicenter, 80% of the buildings collapsed or were critically damaged. Over 1.5 million people (approximately 15% of the national population) have been directly affected by the earthquake. It is estimated that over 105,000 homes were completely destroyed and more than 208,000 damaged. Approximately 1,300 educational institutions and over 50 medical centers and hospitals collapsed or were damaged; 13 out of 15 key government buildings were severely damaged. The Haitian government estimates that the damage caused by the earthquake totals approximately \$7.8 billion, which is more than 120% of Haiti's 2009 gross domestic product. According to EERI (2010b), by 24<sup>th</sup> January 2010, at least 52 aftershocks measuring 4.5 or greater had been recorded. An estimated three million people were affected by the earthquake; the Haitian government reported that an estimated 316,000 people had died, 300,000 had been injured and 1,000,000 made homeless. International agencies, including the United States Agency for International Development, have suggested that the death toll is much lower at somewhere between 46,000 and 92,000 and 220,000, with around 1.5 million to 1.8 million homeless. The government of Haiti also estimated that 250,000 residences and 30,000 commercial buildings had collapsed or were severely damaged.



Figure 3.1: Haiti map with earthquake epicenter and major cities affected; Source: [URL 31]

Recently, there has been growing importance of using Geographic Information(GI) to disseminate information for emergency or early warning purposes. The GI from traditional sources like satellite images or aerial photography is not free, they might have restrictions of use and it is time-consuming to have a newest update of the emergency situation like the earthquake situation in Haiti earthquake. There is a need to use the other source of GI in order to make a near real time map that is needed for the emergency situation of Haiti. In this circumstance, neogeographic data is a suitable choice. The case study data are obtained from the Crisis Map of Haiti [URL 15] which represents the most comprehensive and up-to-date crisis map available of the Haiti earthquake in January 2010 to the humanitarian community (see Figure3.2). The data are mapped in near real time and gathered from reports coming from inside Haiti via:SMS, Web, Email, Radio, Phone, Twitter, Facebook, Television, List-serves, Live streams, Situation Reports. According to [URL 15], volunteers at the Ushahidi's Situation Rooms at the Fletcher School in Washington DC, Geneva, London and Portland were mapping the majority of the reports submitted to Ushahidi website [URL15] in near real-time. The volunteers identified GPS coordinates for the reports and geo-tagged the reports on the Ushahidi map. Each report was first read at least once by the Situation Rooms before being shown on the map (see Figure3.2). The map shows the total number of reports submitted inside the red circles whose centers are the centers of administrative areas. In different zoom levels, the administrative areas are as big as the whole Haiti, much smaller as one of 141 different communes or a specific place on a small road of Haiti at maximum zooming level. On the website [URL 15], we can also select different categories of reports based on their attributes and see the details of each report by looking for it in a new window.

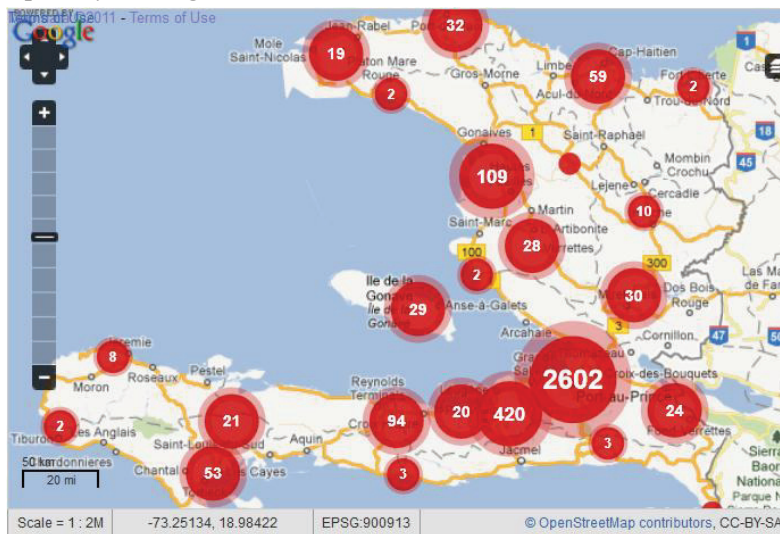


Figure3.2: Crisis Map of Haiti; The numbers in the red circles show the total number of reports submitted in the particular administrative areas where the red circles are centers; Source [URL 15]

The data of Haiti earthquake reports can be downloaded from the Ushahidi website [URL 15] in Comma-Separated Value (CSV) file format (see Figure3.3).



	A	B	C	D	E	F	G	H	I	J
1										
2	#	INCIDENT	INCIDENT DATE	LOCATION	DESCRIPTION	CATEGORY	LATITUDE	LONGITUDE	APPROVED	VERIFIED
3	4073	missing pe	9/7/2011 1:51	haiti	We are yet to find Mr Raynold,	6. Natural Hazards,	18.546674	-72.33398	NO	NO
4	4070	Premios Y	5/10/2011 5:51	Madrid	muchos premios ir's cree	8f. Other,	40.332706	-3.766433	NO	NO
5	4067	Passage d	3/23/2011 14:45	Madagascar	Depuis la matinÃ©e du lundi	6. Natural Hazards, 6a. F	-18.76695	46.869107	NO	NO
6	4065	The truth	1/25/2011 10:55	Bonga	God forgive me for	1. Urgences   Emergency	18.442778	-72.35083	NO	NO
7	4062	Another C	12/13/2010 9:45	Port-au-Pi	http://www.youtube.com/wa	8. Autre   Other, 8f. Oth	18.539167	-72.335	NO	NO
8	4061	Domestic	12/10/2010 13:47	Samoa	Bashing	4c. Group violence,	-13.75903	-172.1046	NO	NO
9	4059	cholera di	10/28/2010 13:55	Port-au-Pi	Cholera disease starts to	3. Public Health, 3a. Infe	18.546674	-72.34222	NO	NO
10	4057	good	9/22/2010 8:23	Bac d'Aq	good	3. Public Health,	19.233333	-72.63333	NO	NO
11	4056	Job	8/20/2010 3:10	Port-au-Pi	I am golam suckline . I am bang	7. Secours   Services Av	18.539167	-72.335	NO	NO
12	4054	Pharmacy	7/27/2010 20:20	lamare	Pharmacy at Lammare Street ir	7. Secours   Services Av	49.295769	-0.892294	NO	NO
13	4052	* URGENT	7/5/2010 17:26	Jacmel, Ha	Birthing Clinic in Jacmel #Haiti	1. Urgences   Emergency	18.233333	-72.53333	YES	NO
14	4051	Food-Aid	6/28/2010 23:06	fondwa	Please help food-aid.org deliv	1. Urgences   Emergency	50.226029	5.729886	NO	NO
15	4050	how haiti	6/24/2010 16:21	centrie	i feel so bad for you i know i ar	2. Urgences logistiques	22.278381	114.17429	NO	NO
16	4049	Lost persc	6/20/2010 21:59	Genoca	We are family members of	1. Urgences   Emergency	44.407062	8.933989	NO	NO
17	4042	Citi Soleil	5/18/2010 16:26	Citi Soleil	We are working with Haitian	1. Urgences   Emergency	18.571084	-72.33467	YES	NO
18	4041	Radio Con	4/26/2010 13:14	Radio Con	i'm Louinel from Sarthe. I'd to	5e. Communication line	18.593707	-72.31008	YES	NO
19	4040	Contamin	4/26/2010 14:19	Marc near	How do we treat water in	4. Menaces   Security Th	18.4828	-73.6388	YES	NO
20	4039	Violence d	4/26/2010 14:27	unable to	Goodnight at (arcahaie bas	4. Menaces   Security Th	18.415	-73.195	YES	NO

Figure3.3: Original data of Haiti earthquake reports in CSV format.

Figure3.3 shows that the collected data of Haiti earthquake reports' attributes include:

- **#:** ID of the reports
- **INCIDENT TITLE:** The title of incident for reports. Every time when the victims or volunteers submit a report, they need to provide a title of incident to make an overview of what happened at their places. The titles are freely named by the victims or volunteers; they might be very short as "SOS!!!" or very long which consists of victims' name, address and their request.
- **INCIDENT DATE:** The date and time when the report was sent
- **LOCATION:** Name of the place where the report was sent (named by the volunteers). This information is not exactly provided because many victims or volunteers are not from Haiti. They do not know exactly where they were at that time, thus the location might be as big as a city which is very hard to locate where they are. In this case, the coordinates of messages need to be used as a replacement of positional information which can give more accurate information about the messages' location.
- **DESCRIPTION:** Description about the incident, the victims or volunteers created this content to describe the real time situation when they sent the report. Based on these descriptions, Ushahidi's Situation Rooms verified and classified the reports into different categories that listed below.
- **CATEGORY:** The officers in Ushahidi's Situation Room categorized the reports into eight categories which will be used for the users to see the data in different aspects. The categorization is based on the contents of the attribute DESCRIPTION. There are some difficulties in categorizing the reports. Firstly, the reports are written in different languages of different victims or volunteers from different countries. Therefore, the officers need to translate the reports' descriptions before categorizing. Secondly, many reports' descriptions are too short for the officers to know which categories they are in. In this case, if the information are not enough to categorize, other information from incident's title need to be used before the report have to be put into an unclear category namely "Other". The reports are categorized into eight categories, they are:

**1. Emergency:** includes incidents that relate to emergency situations such as highly vulnerable, medical emergency, people trapped, fire emergency, etc. In this category, there are four small categories in relation to emergency, they are:

- 1a. Highly vulnerable
- 1b. Medical Emergency
- 1c. People trapped
- 1d. Fire

**2. Vital lines:** includes incidents that relate to vital lines such as food shortage, water shortage, contaminated water, fuel shortage, power outage, shelter needed, etc... There are seven small categories in this category, they are:

- 2a. Food Shortage
- 2b. Water shortage
- 2c. Contaminated water
- 2d. Shelter needed
- 2e. Fuel shortage
- 2f. Power Outage
- 2g. Security Concern

**3. Public health:** consists of the incidents related to public health problems like infectious human diseases, chronic care needs, psychiatric needs, animal illness/death, etc... There are six small categories in relation to public health, they are:

- 3a. Infectious human disease
- 3b. Chronic care needs
- 3c. Medical equipment and supply needs
- 3d. OBGYN/Women's Health
- 3e. Psychiatric need
- 3f. Animal illness/death

**4. Security threats:** comprises the incidents that relate to security problems such as looting, theft of aid, group violence, riot, etc... In this category, there are five small categories, they are:

- 4a. Pillage | Looting
- 4b. Theft of aid
- 4c. Group violence
- 4d. Riot
- 4e. Water sanitation and hygiene promotion

**5. Infrastructure damage:** includes the incidents relevant to infrastructure damage, such as collapsed structure, unstable structure, road blocked, compromised bridge, communication lines down, etc... In this category, there are five small categories in relation to infrastructure damage, they are:

- 5a. Collapsed structure
- 5b. Unstable Structure
- 5c. Road blocked
- 5d. Compromised bridge
- 5e. Communication lines down

**6. Natural hazards:** consists of incidents with problems in relation to natural hazards like earthquake and aftershocks, floods, landslides, etc... There are six small categories relate to natural hazards, they are:

- 6a. Floods
- 6b. Landslides
- 6c. Earthquake and aftershocks
- 6d. Deaths
- 6e. Missing Persons
- 6f. Asking to forward a message

**7. Services available:** includes incidents that have information related to services such as hospital/clinics operating, food distribution point, non-food aid distribution point, rubble removal, feeding centers available, internet access, etc... Ten small categories below are related to the main category:

- 7a. Food distribution point
- 7b. Water distribution point
- 7c. Non-food aid distribution point
- 7d. Hospital/Clinics Operating
- 7e. Feeding centers available
- 7f. Shelter offered
- 7g. Human remains management
- 7h. Rubble removal
- 7i. Financial services available
- 7j. Internet access
- 7k. Port open

**8. Other:** consists of incidents that do not relate to any of the seven categories above.

Six small categories below are in the group of other incidents:

- 8a. IDP concentration
- 8b. Aid manipulation
- 8c. Price gouging
- 8d. Search and Rescue
- 8e. Persons News
- 8f. Other

- **LATITUDE, LONGITUDE:** Coordinates of the places where the reports were sent. Because the reports are submitted from different devices such as mobile devices (SMS), GPS devices, computers (by using internet connection to submit via email, blog post, etc), radio, etc. Because each device has different measurement accuracy in location, each report that was submitted by these devices will have different accuracies of coordinates. For example, if the report was submitted by using a GPS device the maximum accuracy of few decimetres can be achieved (can be converted to decimal degrees as a few micro-degrees). However, if another report was submitted by using a device that uses cell ID (mobile network station) to locate it, the accuracy would be very low (hundreds of meters, which can be converted to decimal degree as a few milli-degrees).

- **APPROVED:** The status of the reports in which YES means the reports were read and ready to be visualized on the map, NO means the reports were not read and cannot be visualized until they are read at least once.

- **VERIFIED:** The status of the reports in which YES means the reports' information is correct, NO means the reports' information is incorrect or has not been verified yet.

To make further analysis from case study data and to be able to create the different types of maps that need the boundary data such as proportional map, choropleth map, etc., I use two shape-files that store boundary data of ten different departments (see left map in Figure 3.4) and 41 provinces (see right map in Figure 3.4) in Haiti, downloaded from [URL 27].

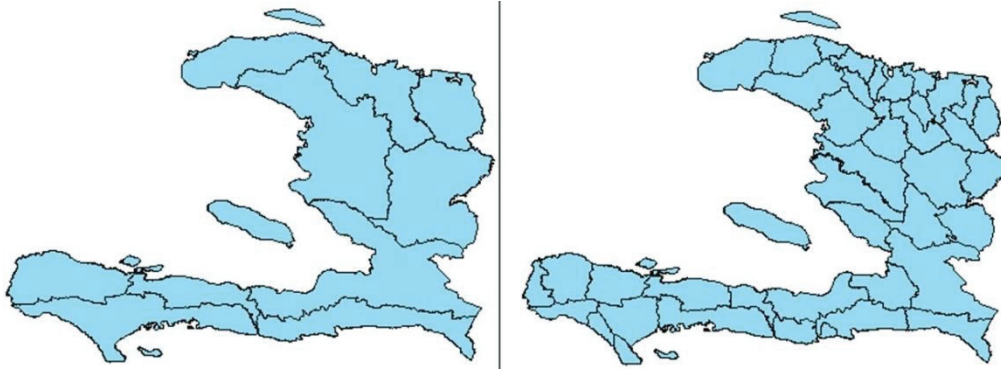


Figure 3.4: Haiti boundary map, 10 departments (left) and 41 provinces (right); shown on Arc Map 10.3.

### 3.3. Tasks analysis and translation into views and functions

#### 3.3.1. General taskswith neogeographic data

PEUQUET (1994) distinguishes three components of spatio-temporal data: space (where), time (when) and objects (what) (see Figure 3.5). Due to that, there are three possible basic kinds of questions when working with geospatial data:

When + where → what: Users want to know the objects or group of objects at the specific location or set of locations at a given time.

When + what → where: Users want to know the location or set of locations occupied by a specific object or group of objects at a given time.

Where + what → when: Users want to know the times that a specific object or group of objects occupied a specific location or set of locations.

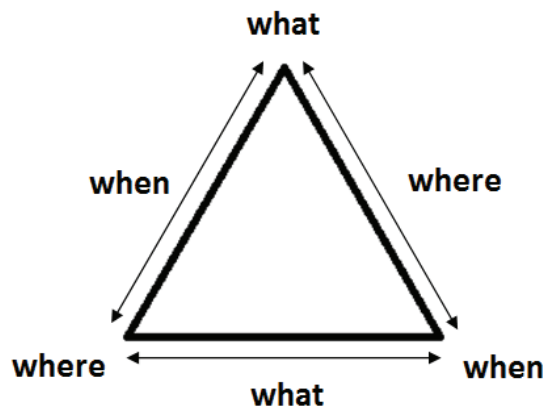


Figure 3.5: Questions while exploring spatio-temporal data (PEUQUET, 1994)

To answer the kinds of questions above, users need something like a view where users can find information to answer the questions. In this case, a map view may help users to answer the “where” question. A temporal view may help users to answer the “when” question. An attribute view may help users to answer the “what” question. As shown in the Figure 3.5 the questions what, where and when are related to each other. Hence, the three views (see Figure 3.6) should be also linked together to give answers to the questions.

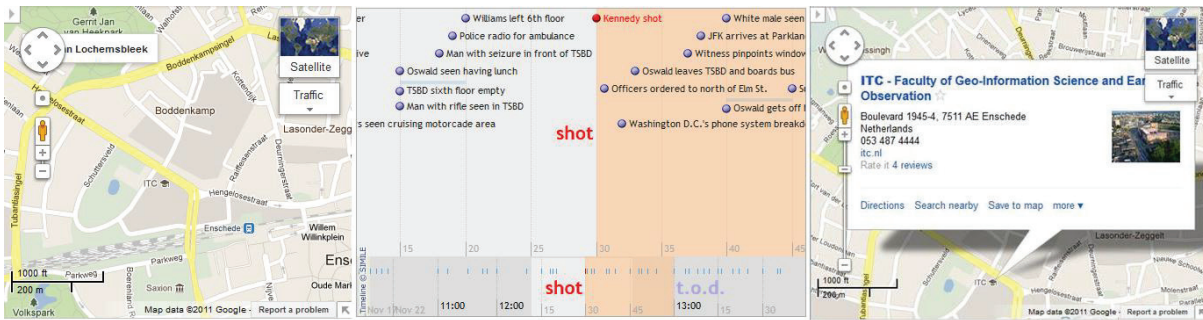


Figure 3.6: Examples of map view (left – source [URL 2]), temporal view (middle – source [URL 36]), and attribute view (right – source [URL 2])

As mentioned above, for visual exploration of neogeographic data some generic tasks will be described. Moreover, tasks will be translated into views and functions. Although there are many visual design guidelines, the Visual Information Seeking Mantra spread by SHNEIDERMAN (1996): “Overview first, zoom and filter, then detail-on-demand” will be considered as a basic guiding principle of visualization in my research:

**Provide overview of the whole data set:** The goal of the overview is to give users a first impression of the data. Because we are dealing with geo-referenced neogeographic data, in order to give users an overview of the whole dataset to get the overall pattern, all the points should be visualized on a map view. However, since the data contain too many points, users may have difficulties to get information when all the points are shown at a small map scale (see example in Figure 3.7). In this situation, depending on the total number of points on the map, we can apply a clustering function, which can be defined as a process to group a set of objects into classes of similar objects. After clustering we can visualize representative points of each cluster instead of visualizing all the area’s data. In some cases, it is required not only to visualize one type of entity, but two or more types of entities (e.g. to visualize reports of natural hazards and also emergency reports to see the relationship between them). To deal with these requirements, a classification function should be applied to classify entities into numbers of classes. Classification can also be applied to classify entities that have different properties (e.g. classify the entities based on their categories).



Figure 3.7: Map of food newspapers in United State at Google Map zoom level 4; Source [URL 37]

**Spatial Zooming:** As users may be interested in some portions of the maps, we need a tool to enable them to focus on a particular region. However, users also want to see the map of a larger area after they finish the work of looking for information in a small area. So, we can provide a zooming mechanism that lets users zoom-in and zoom-out. A useful way to zoom-in is by pointing to the location and double click or using the wheel on the mouse to perform the zooming command. Furthermore, in some cases,



depending on the zoomfactor users may need a panning tool that lets them navigate on the screen by dragging the map.

**Filtering/query:** While the purpose of overview is to give users an idea about the whole dataset, the aim of filtering is to focus on the items of interest from a user's perspective. If there are too many points visualized on a map view, users find it very difficult to get the information that they need. In addition, sometimes users are just interested in some particular objects, or they want to get information about the patterns in a certain period of time. They may also be interested in a group of objects that have some characteristics in common in a specific period. In principle, there are two ways to filter and query data:

- Add requested information to what is already displayed on the user's view to highlight it.
- Eliminate from user's view the points that do not meet the requirements.

**View detail-on-demand:** As mentioned before, neogeographic data do not only contain data about objects like ID, latitude, longitude, but also have their own characteristics that users want to know like location name, description. A simple approach that is normally used in this situation is to click on an item to get a pop-up window with the values of all needed attributes, or to open another web page to get the information from that item. Linking those views together will provide a coherent image of the data (see Table 3.1).

Tasks	Required views	Required functions
Overview	<ul style="list-style-type: none"> <li>- Map view</li> <li>- Temporal view</li> <li>- Attribute view</li> </ul>	<ul style="list-style-type: none"> <li>- Classification</li> <li>- Clustering</li> </ul>
Zoom		<ul style="list-style-type: none"> <li>- Zoom in</li> <li>- Zoom out</li> <li>- Panning</li> </ul>
Filtering / Query		<ul style="list-style-type: none"> <li>- Add items on user's views</li> <li>- Remove items from user's views</li> </ul>
View detail-on-demand		<ul style="list-style-type: none"> <li>- Pop up windows</li> </ul>

Table 3.1: Summary of tasks (based on the Visual Information Seeking Mantras spread by SHNEIDERMAN (1996)), functions and views

### 3.3.2. Specific tasks with case study data

Four main tasks that were discussed in section 3.3.1 are only the basic tasks in analyzing geographic data in general. Like other data, questions about case study data are also related to "what", "where" and "when". VAN ELZAKKER et al. (2004) mention some tasks with relation to the geographic questions. Applying these tasks into the case of Haiti earthquake of 2010 data, Table 3.2 shows tasks and questions that the users may want to have answers to from the prototype.

Geographic questions	Tasks
What is there?	To recognize objects
At a given place, what is there? Which categories/types the report is in?	To identify objects
At a given place, how many reports were submitted? At a specific time period, how many reports were submitted? How many reports are in the category X? What is the density of sent reports in different areas?	To estimate amounts
Where is the particular report?	To locate an object
Where the most/least number of reports is sent?	To quantify spatial anomalies
Have the reports' distribution changed in a period of time?	To establish trends
Are there relationships between types of reports in a particular place?	To discover correlations/conflicts
Which factors cause the reports' distribution?	To structure the geographic information

Table 3.2: Map use tasks with relation to the geographic questions in the case of Haiti earthquake case study data.

Based on the Table 3.2, some additional tasks that focus on the exploration of neogeographic point data are:

**Searching reports:** To recognize and identify reports those are located on the map. This function also can give the users the total numbers of reports in many ways such as how many reports were submitted in location A, how many reports in the category X, etc. Searching reports can be seen as an advanced function to do the filtering/query task which was mentioned in section 3.3.1.

**Moving to an object's place:** The reports can be shown in lists. However, it is difficult to know the location of a report on the map if we can only see the names with some descriptions of the reports or even with their coordinates in the description. One comfortable way to look for the place is moving the map to the location of the report. It can be done by clicking on the report; the center point of the map view will then move to the report's location.

**Emphasize appearance (e.g. the newest appearance of reports):** Information about the newest reports is important because the newest reports show the most emergency incidents in Haiti at that time. The numbers of new reports' appearance affect the number of reports that are visualized on the view. Hence, using dynamic visualization variables to emphasize objects (e.g. blinking), or using visualization variables, such as colour to differentiate the newest reports from the others, can be applied.

**Show the changes of thematic properties:** Sometimes, users want to know the trends, such as trends in total number of reports that appeared after a period of time. This is very important for the emergency force to know the place to distribute the manpower for help. It is the temporal dimension, so it is very difficult to visualize on the map. Hence, providing a time plot view with play, stop and change speed functions which is linked with the map view can be an option.

**Show the different categories of incidents:** The original dataset of case study data includes eight big categories and some more small categories of incident which separate the concentration of data into different aspects (i.e. attribute data). Many incidents were put into more than one category, for example, from the database we have three incidents I1, I2 and I3; I1 is in categories "5a. Collapsed structure" and "1. Emergency", I2 is in categories "1. Emergency", "5a. Collapsed structure" and "4. Security Threats", and I3 is in category "1. Emergency". One of the questions is "In group of three incidents, how many incidents the categories 1, 4 and 5a have?". To solve this question, we need to calculate the total number of reports in different categories. In this case, the total number of reports in categories 1, 4 and 5a are 3, 1 and 2 respectively. Thus, based on the CATEGORY attribute in the data, we can show the reports in different categories. For example, the support force for health should be sent to the places where reports of health problems are located, instead of to the places where reports of infrastructure damage are located. To solve this problem, selection based on existing classification is the only choice. After the selection, we can select different categories of incident to show them on the map; this will help us to discover correlations or conflicts between types of reports in a particular place.

**Show the number of reports in different locations:** To answer the question "Where the most/least number of reports is sent?" in order for distributing the support forces to the required locations, classification by the total number of reports is also necessary. It shows the places with the most serious situations (the place where reports were sent the most in Haiti) or the places with much better situations (the place where reports were sent the least), where only a small support force is needed.

**Show the details of incidents:** The incidents need to be shown with their detail in order to help the users to know about the descriptions of the reports which consist of real-time situations in these places where the reports were sent. This is an important task that has different solutions, such as using pop-up windows

or opening a new window to show the detail. Considering the convenience of using a webpage and improve the speed of processing, using pop-up windows can be applied.

**Change the status of reports:** The reports that were received need to be approved to show on the map and need to be verified that they are telling the truth. There is a need to make a confirmation function for the administrator to change the status of reports to show them on the map or change the status of verified. It could be provided inside the pop-up windows or shown in the list of reports which can only be accessed by the administrators who know that the information of the report is correct or not.

**Submit new reports:** Suppose that the prototype is working during the period of an earthquake. In that situation there may be a need to provide a page for volunteers or victims to submit their own reports. It is also a required function in the case we use this prototype for other events such as for an election in a country, for other natural hazards, etc. A submit form can be applied for this task.

**Change map types:** Last but not least, the prototype needs to be designed based on cartographic design rules that make the maps useful and informative. According to VAN ELZAKKER et al. (2004), for different purposes, we need to use different types of maps such as using choropleth map to demonstrate the density of sent reports in different areas. One of the ways to do this task is to support a list of different purposes/different cartographic questions; users can select the purpose/question which they need to have the answer to and the map view will be changed to different types of maps depending on which purpose/question has been chosen.

Table 3.3 shows the summary of the tasks, required functions and views that will be implemented into the prototype later.

Tasks	Required views	Required functions
Overview	<ul style="list-style-type: none"> <li>- Map view                             <ul style="list-style-type: none"> <li>• Visualize reports</li> <li>• Use visualization variables for events</li> <li>• Use dynamic visualization variables</li> <li>• Use dynamic legends</li> <li>• Show list of users' purposes</li> </ul> </li> <li>- Temporal view/Timeline</li> <li>- Attribute view(s)</li> <li>- Animation</li> </ul>	<ul style="list-style-type: none"> <li>- Search</li> <li>- Zoom-in</li> <li>- Zoom-out</li> <li>- Panning</li> <li>- Auto pan to selected report</li> <li>- Pop up window</li> <li>- Clustering</li> <li>- Selection/filtering</li> <li>- Classification</li> <li>- Add items to user's views</li> <li>- Remove items from user's views</li> <li>- Play</li> <li>- Stop</li> <li>- Change speed</li> <li>- User management</li> <li>- Change status</li> <li>- Submit form</li> <li>- Select map types by purposes</li> </ul>
Zoom		
Filtering / Query		
View detail-on-demand		
Searching reports		
Moving to an object's place		
Emphasize appearance		
Show the changes of thematic properties		
Show the different incidents in categories		
Show the number of reports in different locations		
Show the details of incidents		
Change the status of reports		
Submit new reports		
Change map types		

Table 3.3: List of tasks, required views and functions that will be created in this research

### 3.4. Client side and database server support

In client and server database systems, the database is stored and controlled by the server while users send requests to the server and receive information from it via the client. Some functions can be performed on both the server and the client side. There is a trade-off between executing functions on the database server and on the client side. For example: the classification function can be implemented on both the client side or on the server side. On the one hand, if the dataset is very big and the function runs on client side, the client application has to download all the data from the server before performing the classification function on this data. That will take a lot of resources on the client side. On the other hand, in case of



small datasets, clients can easily download from the server and do the processing on the client side. It can reduce the tasks that have to be performed on the server side. If there are too many clients send requests to the server simultaneously and the capabilities of the server are limited, the server may be down or it cannot work properly.

Neogeographic datasets usually are very big. Hence, we have to consider using which functions for which side of the system. I suggest creating some supported functions from the server side to increase the performance on the clientside such as clustering and classification. According to [URL 22] some main functions of client and server are described:

#### **a. Functions of the Database Server**

The database is stored and controlled by the server; it allows clients to add information into the database and extract information from the database. The main functions of the database server are:

- Managing the database.
- Receiving queries or requests from the client, executing the queries and sending the results to the clients.
- Managing the database recovery, security and backup services.
- Handling database access, users management, and all control functions.

#### **b. Functions of the Client**

Clients use front-end applications to communicate with the server. The main functions of clients are:

- Presenting and managing the Graphical User Interface through front-end applications and some data manipulation functions.
- It sends the queries/requests to the database server.
- It interprets the results received from the server and displays the results to users.
- The client can also process further the queries results received from the server to get the more informative information that users need.

### **3.5. Summary**

In this chapter, we have discussed case study data from the Haiti earthquake with their characteristics and structure. Further, general tasks that are required in the visual exploration of neogeographic data and some specific tasks associated with the case study data have been discussed. Then, all the discussed tasks were translated into views and functions. Finally, some client and database server support functions were reviewed. In the next chapter, we will discuss some cartographic design rules and investigate some web visualization libraries; look at their characteristics, advantages as well as disadvantages. After all, we will choose the most suitable way to implement web visualization libraries into the prototype based on their characteristics and requirements above.

## 4. BASIC CONCEPTS FOR THE DESIGN AND IMPLEMENTATION OF THE PROTOTYPE

### 4.1. Introduction

In the previous chapter, we have discussed the characteristics and structure of the case study data. Later, the general tasks that are required in the visual exploration of neogeographic data and some specific tasks associated with the case study data were discussed. Then, all the tasks were translated into views and functions. This chapter describes some cartographic design rules that will be applied into the prototype in design map and investigates some web visualization libraries on their characteristics, advantages as well as disadvantages. Finally, we will choose the most suitable way to implement web visualization libraries into the prototype based on their characteristics, advantages, disadvantages and requirements mentioned in chapter 3.

### 4.2. Cartographic design rules

Maps need to be well designed so that the users can easily answer questions like what, when, where and how. SLOCUM et al. (2009) defined cartographic design as “partly a mental and partly a physical process in which maps are conceived and created”. The design process will be repeated until the final map has been completed because if the users find the map is not useful and informative, it needs to be designed and constructed again.

According to SLOCUM et al. (2009), cartographic design refers to the conceptualization and visualization of the map to be created. It aims to two goals: (1) to create a map that appropriately serves the map user based on the map’s intended use and (2) to create a map that communicates the map’s information in the most efficient manner, simply and clearly. Cartographic design is generally directed by rules, guidelines and conventions, but it is also relatively unstructured.

After cartographers examined the data considering measurement scale (nominal, ordinal, interval and ratio), the graphic options to use will be determined. The graphic options are the graphic variables, scale and resolution, symbolization, and text on the map. They will be described as follows:

#### 4.2.1. Graphic variables

According to KRAAK & ORMELING (2009), graphic variables are used to represent the locations and attributes of features existing on the Earth’s surface on maps by using symbols. These symbols are: point symbols, line symbols and area symbols. Difference in symbol size, grey value or lightness, texture, colour saturation, shape and orientation can be used to represent quantitative and qualitative variation of attribute data. Bertin’s six basic graphic variables shown in Figure 4.1 can be used in understanding the differences. These visual variables, namely size, value, grain, colour, orientation and shape can be applied to point, line and area symbols.

differences in:	symbols		
	point	line	area
size			
value			
grain			
colour			
orientation			
shape			

Figure 4.1: Bertin's six basic graphic variables; Source: (KRAAK & ORMELING, 2009)

The six variables in Figure 4.1 can be used for different application based on scales of measurements of the data. For example, the variable size can be used for ratio data; size, grey value and grain can be used for ordinal and interval data, while colour, orientation and shape can be used for nominal data (KRAAK & ORMELING, 2009). From Figure 4.1, we can give the meaning for each of the visual variables as follows:

- Size refers to the dimensions of the symbols.
- Gray value refers to the gray scale or lightness values ranging from white to black.
- Grain refers to the variation in density of the graphic element.
- Colour refers to colour hue of the symbols.
- Orientation refers to direction in which symbols are placed (or the direction of the elements inside the symbols).
- Shape refers to the shape of the symbol.

According to KRAAK & ORMELING (2009), each of the visual variables has one or more of the following perception properties: differentiation (and association) properties, order properties, distance properties and proportional properties. These are related to the measurement levels of data. These perception properties are explained below:

**Differentiation properties:** A visual variable is differentiating if the symbols to which it has been applied are spontaneously perceived as different.

**Association properties:** A visual variable is associative if the symbols to which it has been applied are spontaneously perceived as of equal importance. No symbol stands out visually above the others.

**Order properties:** A visual variable is ordered if spontaneously all symbols to which it has been applied can be placed in an unambiguous order.

**Distance properties:** A visual variable has distance perception properties if spontaneously all symbols to which it has been applied can be placed in an unambiguous order and an estimate of the distance between the symbols within the range can be made.

**Proportional properties:** A visual variable is proportional if the differences between the symbols to which it has been applied can be expressed in distinct amounts.

There is an important process, namely symbolization that needs to be used to design the symbols in the maps of the prototype by applying graphic variables in the stage of implementation. Symbolization is the process of creating graphic symbols to represent feature attribute values (ROBINSON et al., 1995). Geographic features like points, lines and polygons can be symbolized differently. Both qualitative and quantitative point data can be represented by point symbols. These point symbols can be either purely geometric or pictorial. Symbols representing a quantitative data on the map can give a visual measurement according to its size, importance or number. In contrast, the symbols are represented in equal importance

with qualitative data such as showing different names of areas by labels, showing different categories by using pictorial point symbols which represent categories based on their nature.

#### 4.2.2. Scale and resolution

According to HUISMAN & DE BY (2009), map scale is the ratio of a unit distance on the map to the corresponding distance on the ground. For example, a 1:100,000 scale map means that 1 cm on the map represents 100,000 cms, which is 1 km in the terrain. “Large-scale” means that the ratio is large, so typically it means there is much detail, like in a 1:1,000 map. In contrast, “small-scale” means a small ratio, hence less detail, like in a 1:2,500,000 map. Depending on the scale of the map that is going to be created, data can be collected, conceptualized and treated differently. For example, at a certain scale a building can be displayed as an area object while at much smaller scales it is shown as a point. In this research, for different zoom levels, different map scales are used. Applying this concept to zooming function that mentioned earlier in chapter 3, the data need to be shown depending on different zoom level. For example, in small zoom a level, the data of the whole Haiti is shown, when the zoom level is bigger, the data need to be shown within smaller areas like Departments (in middle zoom levels) and Provinces (in large zoom levels).

According to HUISMAN & DE BY (2009), when applied to spatial data, the term resolution is associatively used with the cell width of the tessellation applied. According to [URL 45], resolution can be defined as the number of pixels that constellated together from an image or a photograph. Digital spatial data is stored in a GIS without scale: “Scale is a ratio notion associated with visual output, like a map or on screen display, not with the data that was used to produce the map” (HUISMAN & DE BY, 2009).

#### 4.2.3. Text on the map

KRAAK & ORMELING (2009) mention the phrase “text on the map” by the meaning of the text within the map’s frame, and not the additional information (title, legend, etc.) in the map’s margin. Text on the map can express qualitative, quantitative, ordered and selective perceptual properties. It has the primary function of providing geospatial address or indicating the nature of objects. Applying text on the map can be a challenge as it involves the choice of elements to be labelled, label design and placement. Although labelling can be automated by the computer, still expertise and human intervention are needed; because text can contain other information such as the text’s direction could be the direction of a river and even follows the river’s bend that is hard to achieve with the current mapping software. Hence, human needs to interfere expertly to the work of labelling.

#### 4.2.4. Thematic map types

According to VAN ELZAKKER et al. (2004), in exploratory cartography, different map types provide answer for different basic geographic questions. VAN ELZAKKER et al. (2004) mentioned some types of map and the basic geographic questions which can be answered by using those map type. Applying those map type into the case of Haiti earthquake of 2010 data, considering the characteristics of the data,

Map types for exploration of neogeographic point data	Basic geographic questions
Nominal point symbol map	Where are these different point features?
Choropleth map	What is the density/intensity of a particular feature?
Proportional point symbol map	Where and how much of a discrete point or area features?

Table 4.1 shows some thematic map types and geographic questions can be answered by those types of map. Those map types will be used as solutions for different purposes of the users in the prototype.

Map types for exploration of neogeographic point data	Basic geographic questions
---	----------------------------

Nominal point symbol map	Where are these different point features?
Choropleth map	What is the density/intensity of a particular feature?
Proportional point symbol map	Where and how much of a discrete point or area features?

Table 4.1: Thematic map types and the basic geographic questions to which they provide an answer

### 4.3. Web visualization libraries

Web visualization libraries are the source code libraries that can be used for graphic applications on the web. There are many web visualization libraries that can be used for cartographic visualization available on the internet. However, due to the time limit, I decided to examine the libraries by looking at some criteria which were mentioned in the requirements of the prototype (see Table 3.3) as described in Table 4.2. Following that, the libraries should allow the users to do some tasks such as map view, attribute view, temporal view, selection/filtering, interaction, and changing visual variables of data such as size, colour of symbols when do zooming, etc... After investigating which criteria could be met by the libraries, I decided to limit further investigation to seven first web visualization libraries in Table 4.2, namely, Google Maps API, OpenLayers, SIMILE, Timemap, GeoCommons API, Thematic maps API and Google Chart API. This section describes the characteristics, advantages and disadvantages of these web visualization libraries. Then, depending on the functions that those libraries offer, and considering the functions and views that meet the requirements of different geographic questions, I will choose the libraries that offer us most of the required functions and views to apply into the implementation of the prototype. Hence, I do not have to build those functions and views that is a time-consuming process.

	Web libraries	Programming language	Map view	Attribute view	Temporal view	Animation	Select /Filter	Interaction
1	Google Maps JavaScript API	JavaScript	Yes	Yes	No	No	Yes	Yes
2	OpenLayers API	JavaScript	Yes	Yes	No	Yes	Yes	Yes
3	SIMILE	JavaScript	Yes	Yes	Yes	No	Yes	Yes
4	Timemap	JavaScript/Java	Yes	Yes	Yes	Yes	Yes	Yes
5	GeoCommons API	JavaScript	Yes	Yes	Yes	Yes	Yes	Yes
6	Thematic maps API	JavaScript	Yes	Yes	Yes	Yes	Yes	Yes
7	Google Chart API	JavaScript	No	Yes	No	Yes	Yes	Yes
8	Processing	JavaScript	Yes	No	Yes	No	No	Yes
9	Pchart	PHP	No	No	No	No	No	No
10	Jfree	Java	No	No	No	No	No	No

Table 4.2: Criteria to choose the web visualization libraries

#### 4.3.1. Google Maps JavaScript API

The Google Maps JavaScript API [URL 3] is a JavaScript web library that allows users to embed Google Maps onto their web pages. This web visualization library is free and available for any website which uses JavaScript. It supports most recent web browsers on most recent operating systems. The Google Maps JavaScript API requires an API key which you can easily and freely get at [URL 3] to embed Google Maps in your web pages.

The Google Maps JavaScript API offers users interaction with some kinds of base layers: map, terrain, satellite and hybrid. It offers the ability to display points, lines and polygons in different sizes, shape and colours on the map view. Users can overlay data from KML (Keyhole Markup Language), KMZ (KML zip file with .KMZ extension) and GeoRSS files on top of the map. The Google Maps API also provides various functions for mapping such as zooming, panning, filtering, pop-up windows, directions etc.



A prototype for the exploration of neogeographic point data with cartographic visualization tools

Furthermore, GoogleMaps API provides `markermanager.js` [URL 11] that allows users to handle a large number of Google's markers which can be used to manage the symbols for point data on the maps. Figure 4.2 gives an example of handling a large number of markers for a weather map using the Google Maps API and `markermanager.js`. However, the Google Maps API has limitations in animation and timeline; it does not support animation and the data cannot be shown in temporal scale. Thus, if the users need to work with temporal data like the case study data, they need to use the other libraries.

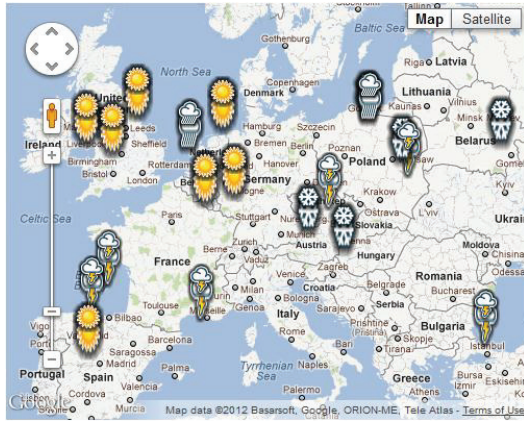


Figure 4.2: Example of Google Maps API and `mapmarker.js` library; Source: [URL 46]

#### 4.3.2. OpenLayers API

According to [URL 4], OpenLayers is a pure open source JavaScript library that allows users to display map data on any web page in web browsers, with no server-side dependencies. It facilitates users to display map tiles and markers from any source. OpenLayers is released under the 2-clause BSD License, which is “a class of extremely simple and very liberal licenses for computer software that was originally developed at the University of California at Berkeley” [URL 47]. Like the Google Maps API, OpenLayers implements a JavaScript API for creating rich web-based geographic applications. It has been developed for and by the Open Source software community.

OpenLayers can load map data from many sources such as Web Mapping Service (WMS), Web Feature Service (WFS), Google Maps, from the files KML, GML, GeoRSS, GeoJSON etc. OpenLayers separates map tools from map data so that users can use all the tools on all the data sources.

By looking at examples on the internet, we can clearly see the capabilities of OpenLayers. OpenLayers provides many capabilities including visualizing user data on a map view, interactive mapping functions (e.g. zooming, panning) with the map, filtering/querying on data source, display the geometry of data (e.g. points, lines and polygons), animation etc. However, it does not provide a timeline to show the temporal space of case study data. OpenLayers presents animations as a sequence of layers or images.

Figure 4.3 shows an example of an animation in OpenLayers. Unfortunately, for each animation step, users have to create a layer class. Moreover, OpenLayers does not provide the temporal brushing for large time-series data which is necessary to show the temporal scale in some of the neogeographic data. Therefore, when the users need to work with temporal data which is as big as the usual neogeographic data, the other libraries need to be investigated.

A prototype for the exploration of neogeographic point data with cartographic visualization tools

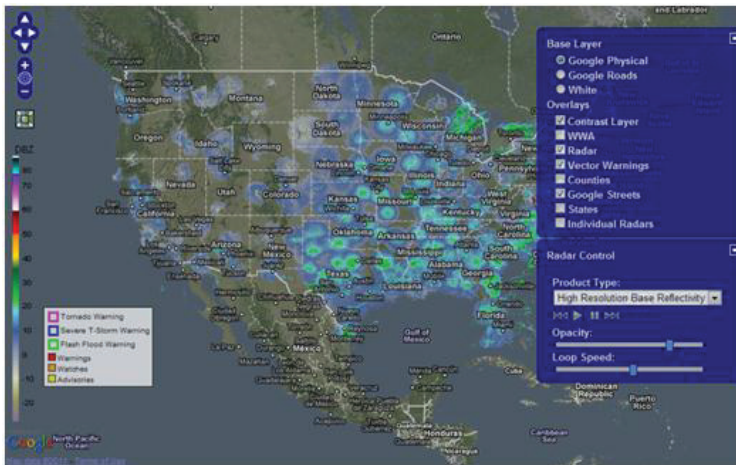


Figure4.3: Example of using animation in OpenLayers [URL 12]

#### 4.3.3. SIMILE and Timemap

SIMILE [URL 5] stands for Semantic Interoperability of Metadata and Information in unLike Environments. SIMILE is a joint project conducted by the MIT Libraries and MIT CSAIL. SIMILE aims at creating robust, open source tools that enable users to access, manage, visualize and reuse digital assets. The SIMILE project and its members are fully committed to the open source principles of software distribution and open development and for this reason it releases the created intellectual property (both software and reports) under a BSD license.

SIMILE is supported by AJAX (Asynchronous JavaScript and XML) which provides a capacity to get data from the server asynchronously without disturbing the displays and behaviours of the existing page. SIMILE Ajax is a JavaScript library which is slim, fast and can run on most of the recent browsers (e.g. Firefox, Opera, and Internet Explorer). The SIMILE project consists of 22 sub projects. SIMILE allows users to display data in different sizes and colours on the map. It allows users to interact with the map view (e.g. do zooming and panning). SIMILE allows users to do filtering, brushing and searching by using “exhibit facet” which enables the website’s authors to create dynamic exhibits of their collections without resorting to complex database and server-side technologies. In exhibit facet, the collections can be searched and browsed using faceted browsing. Figure4.4 shows an example of SIMILE Exhibit using JSON data type. SIMILE also provides the time plot and scatter plot. It provides a pop-up window for the attribute view. These views can be linked together. However, SIMILE does not support users in the animation function and displaying temporal data.

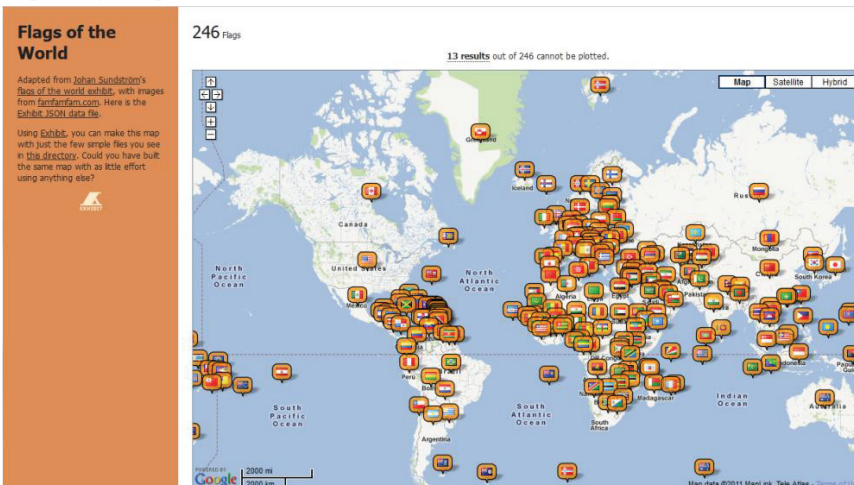


Figure4.4: Flags of the World map using SIMILE and JSON data type; Source: [URL 23]

Timemap [URL 6] is one of the SIMILE subprojects; it has two versions which can work similarly. They are Timemap Java and Timemap JavaScript (Timemap.js). Due to the limitation of time and the synchronization in web technology with the other libraries which will be used in the prototype, Timemap Java cannot be used in my research, only Timemap.js is examined in my research.

Timemap.js is an open source JavaScript library that allows users to combine Google Maps with SIMILE timeline. As a result, it provides functions from both Google Maps API and SIMILE timeline. Like other JavaScript libraries, timemap.js can run on most of the recent web browsers and on most of the recent operating systems. It allows users to display points, lines, polygons on the map and a timeline simultaneously. The temporal scale of the timeline view is linked with the map view. This library allows users to load multiple datasets in JSON, KML, Google Spreadsheet or GeoRSS format. Timemap.js only shows items in the visible range of the timeline on the map. Timemap.js also enables users to do filtering and displays different objects with different symbols and colours. Nevertheless, the filtering function that is provided by timemap.js is very simple. It does not allow users to filter several objects at the same time. Hence, if we want to do a real filter, this function needs to be improved. Although timemap.js does not support automatic animation, we can simulate animations by giving the option to move (pan) the timeline manually. In this case, the timeline can be considered as a linear temporal legend. Figure 4.5 shows an example of Timemap.js using KML.

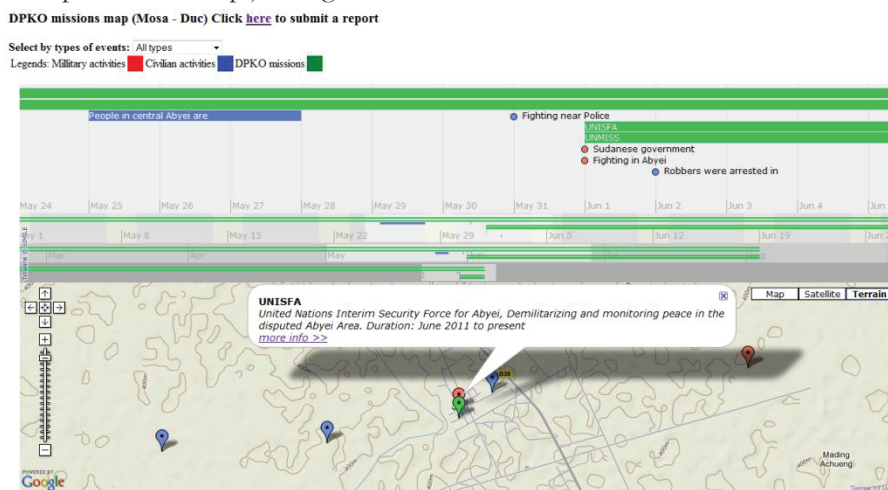


Figure 4.5: Department of Peacekeeping Operations (DPKO) missions map using Timemap API; Source: (DUC & MOSEME, 2011)

#### 4.3.4. GeoCommons API

The so-called GeoIQ platform powers the growing GeoCommons [URL 8] community of over 25,000 members who are actively creating and sharing thousands of datasets and maps across the world. With GeoCommons, anyone can contribute and share open data, easily build shareable maps and collaborate with others. One can also download the data shared/uploaded by the other users of GeoCommons.

GeoIQ [URL 13] is a geospatial data management, visualization and analysis platform providing collaborative, browser-based data analysis tools for use by both technical and non-technical users. GeoIQ helps users to quickly and intuitively make intelligent, data-driven decisions with no cumbersome training.

GeoCommons API allows users to display points, lines and polygons on the map. This library allows users to load multiple datasets which are uploaded on the GeoCommons website. GeoCommons offers mapping tools such as zooming and panning. It also provides filtering and searching tools. GeoCommons supports users to create animations of their map data.

GeoCommons also has another JavaScript library, namely Analysis API [URL 14], which allows users to do analysis online without any GIS desktop software. Analysis API can do some analysis such as

A prototype for the exploration of neogeographic point data with cartographic visualization tools

buffer,buffer\_intersect,sum,residual,difference,intersect,union,simplify,clip, spatial aggregation and cross dataset correlation. However, the current version of GeoCommons API and GeoCommons Analysis API need to use the data which are uploaded on the GeoCommons server. Therefore, since we want to create functions like submit reports and user management, etc... that need to use our own data server, it is impossible to use them in this research. Figure 4.6 shows an example of GeoCommons API.

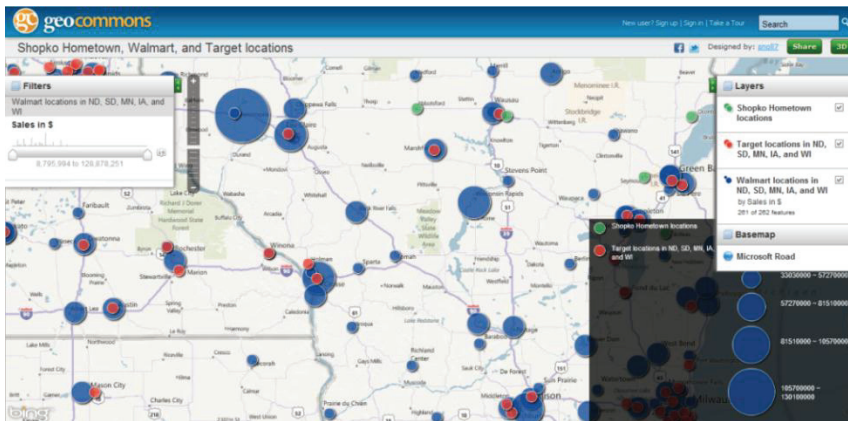


Figure 4.6: Map of Shopko Hometown, Walmart, and Target locations, using GeoCommons API; Source: [URL 15]

#### 4.3.5. Thematic Mapping API

Thematic Mapping API (TMA) [URL 24] is a new JavaScript library which allows us to create thematic maps from our own data source. It was created by SANDVIK (2008). He proposed a method of using the KML format to show data on thematic maps. This research shows us the way to use point, line and polygon neogeographic data as top layer of the maps by using TMA. This web visualization library was then upgraded to be able to offer users to show their own data from various formats (such as JSON, KML, KMZ, Google Spreadsheets, etc) on a base map of Google, Google Earth, OpenStreetMap, etc... TMA can run on most of the recent web browsers. However, for the TMA maps that use Google Earth API, we need to install the Google Earth plug-in for the web browser to be able to work. Because TMA uses Google Earth API which supports animation, it is possible to show the map with an animation if needed. In some examples of TMA, SANDVIK (2008) shows the possibility to do filtering. It allows users to create many types of thematic maps like choropleth maps, prism maps, proportional symbol maps, pie chart maps. With those various map types the different cartographic questions from the users can be answered. Nevertheless, TMA need to work with Google Earth plug-in installed on the computer. That means, the computer need to install Google Earth plug-in, that could be very uncomfortable because many computers cannot install more software due to the privacy of its organization, such as the computers in schools, or libraries, etc... Figure 4.7 is a screen dump of an example of Thematic Mapping API which uses KML file data, Google Earth API and provides a Control Panel to do filtering and control the visualization of the map on a Google Earth base map.



A prototype for the exploration of neogeographic point data with cartographic visualization tools

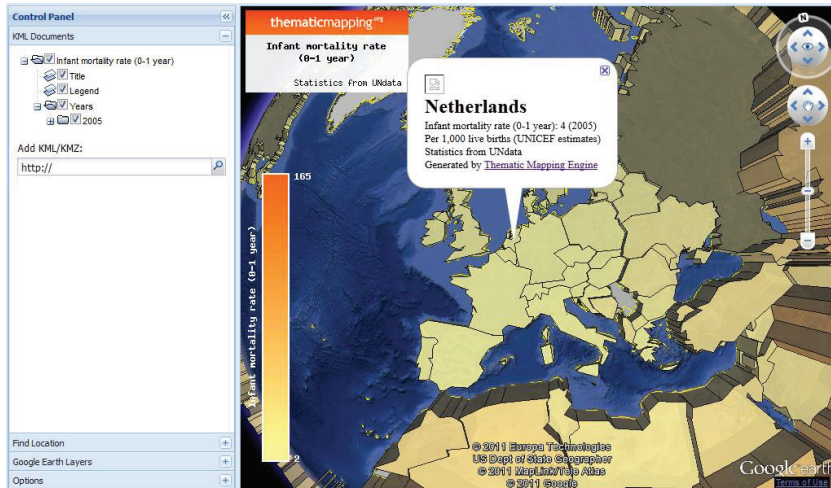


Figure 4.7: Infant mortality rate (0-1 year), statistics from UNdata, year 2005, using Thematic Mapping API; Source [URL 25]

#### 4.3.6. Google Chart Tools

Google Chart Tools (sometimes called Google Chart API)[URL 39] allows us to create charts and reporting applications over structured data and helps to integrate these directly into our website. They are exposed as JavaScript classes. Google Chart Tools provides a large number of well-designed chart types such as area charts, bar charts, scatter charts, tree map charts, etc. which can be easily customized to fit the look and feel of our website. Charts are rendered using HTML5/SVG technology to provide cross-browser compatibility and cross platform portability to iPhones, iPads, Android and Windows mobile phones. No plug-ins is needed to be able to run Google Chart Tools. All charts are populated with data using a common JavaScript DataTable class. Having a common data structure makes it easy to switch between chart types. It is able to do sorting, modifying, and filtering data with Google Chart Tools. We can populate a DataTable programmatically from data that are retrieved from our own database, or we can request data from a data provider that supports the Chart Tools Datasource Protocol. The Google Chart API itself does not provide the map view and temporal view. However, it will make a good combination with the other web visualization libraries to make the data more informative in the way that not only the numbers from the data are shown but also the charts made from them. Figure 4.8 shows an example of using Google Chart API to create a pie chart based on the Google Visualization DataTable.

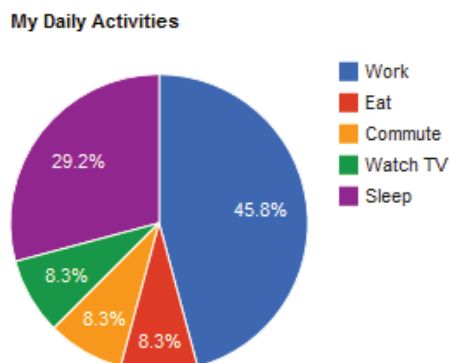


Figure 4.8: Example of using Google Chart API to create a pie chart; Source [URL 40]

#### 4.3.7. Summary of web visualization libraries

Each web visualization library above has its advantages and disadvantages (see Appendix B.1). There are some functions that one library supports but the other does not. Thus, one library can do some tasks very well but it is impossible for it to do other tasks. To solve this problem, I will use more than one web



visualization library to implement into my prototype. In this research, I will use the combination of web visualization libraries, Timemap API, SIMILE, Google Maps API and Google Chart API. The combination of different web visualization libraries will make possibility to execute more required tasks and gives more functions than using just one individual library.

#### **4.4. Summary**

In this chapter, we have discussed some basic concepts for the design and implementation of the prototype. Firstly, some of the basic cartographic design rules were reviewed. Later, some web visualization libraries were investigated in order to have more details of their characteristics, advantages and disadvantages. Finally, we have chosen the most suitable way to implement web visualization libraries into the prototype based on their characteristics and requirements mentioned in chapter 3. In the next chapter, the design and implementation of the prototype will be discussed step by step.

## 5. DESIGN AND IMPLEMENTATION OF THE PROTOTYPE

### 5.1. Introduction

In the last chapter, we have discussed the basic concepts for the design and implementation of the prototype. The cartographic design rules which are needed for the design of the prototype were mentioned first. After that, seven web visualization libraries have been investigated one by one. Finally, we have selected the most suitable way to implement webvisualization libraries into the prototype considering their characteristics, advantages, disadvantages and requirements of the prototype. In this chapter, we are first going to create the conceptual model of the prototype. After that, the work of design and implementation of the prototype will be done step by step. Firstly, the set-up of the data base will be described. Then, PHP and PostgreSQL will be mentioned as the server scripts and database server for the implementation of the prototype. After that, the required views and functions that were mentioned in chapter 3 will be implemented into the prototype. Finally, some output of the prototype will be described.

### 5.2. Conceptual model of the prototype

In this section, the conceptual model of my prototype is presented. A conceptual model is defined in [URL 23] as: "a type of diagram which shows a set of relationships between factors that are believed to impact or lead to a target condition; a diagram that defines theoretical entities, objects, or conditions of a system and the relationships between them". Figure 5.1 presents the conceptual model of the prototype and its components. The model shows the relationships from the database to the client/end users.

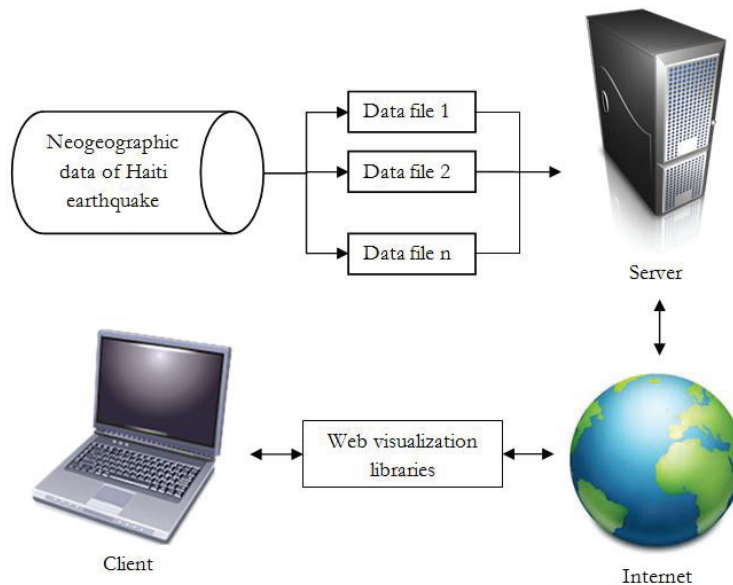


Figure 5.1: Conceptual model of visualizing neogeographic data set of Haiti earthquake using web visualization libraries

As mentioned in last chapter, the web visualization libraries that will be used to implement into the prototype support different file formats, such as KML, JSON, GeoRSS or Google Threadsheet. We can use different files to store data and show different data files at the same time by using the four web visualization libraries, Timemap API, SIMILE, Google Maps API and Google Chart API. Two of the main tasks of this prototype are to provide users with overview as well as detail-on-demand in one interactive map. At the low zoom levels, the data are shown to give users an overview. Combined with animations, an overview of the change (new reports appear) can be given to the users. At the high zoom levels, data about the details of each report can be shown. In some cases, at high zoom levels in some

specific areas (such as the places in which many victims gather to get support), the map displays may be still messy, because there are too many data to be shown. In this case, users may perform a function to get another data file that contains the declustered data of these areas.

Because different map types can be used for different purposes of the users, each map type will need data files that contain the data needed for that map type to represent them on the map. Moreover, as mentioned earlier, depending on different zoom levels, the data are shown differently; hence, in Figure 5.1, there are several data files that are generated from the database by using web programming. Via the server and the internet, those files will be transferred to the client and combined with the web visualization libraries for geovisualization to help users answer the questions related to what, where and when for different purposes. Those data files are dynamically generated whenever the database has changed (e.g. after an update of the database or when new data have been inserted into the database) or when the users want to use the clustering, filtering function, a new data file will be generated and transferred to the client. Next, those files are processed by Timemap API, Timeline and Google Maps API for the geovisualization. Finally, an application is created on the client side on the web browser for visual exploration of the Haiti earthquake neogeographic data.

### **5.3. Set up of the database**

According to the conceptual model (see Figure 5.1), the first item of the prototype is the database. It is one of the most important parts in my research. The case study data have been stored in a database management system (DBMS). Putting data in a DBMS makes the management job much easier. In addition, with DBMS, the data can be updated or modified by the users or volunteers by using a submit data web page and an input data web page. In this research, a PostgreSQL DBMS is used to store the Haiti earthquake neogeographic dataset.

#### **5.3.1. PostgreSQL database**

According to [URL 26], PostgreSQL is a powerful object-relational database system. It has most features that are present in large proprietary DBMSs, like transactions, sub selects, triggers, views, foreign key referential integrity, and sophisticated locking. It has native interfaces for ODBC, JDBC, .Net, C, C++, PHP, Perl, TCL, ECPG, Python, and Ruby. PostgreSQL is free software. It can work on all major operating systems, including Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), and Windows.

#### **5.3.2. Set up of the database**

Raw data that is downloaded from [URL 15] are in a comma-separated values (CSV) file. However, the file is sorted in the order of newest first while PostgreSQL will record the file in the opposite order. Thus the ID of newer events will be smaller than the older ones. I had to re-order the CSV file before importing it into PostgreSQL. After that, I wrote a short PostgreSQL script (see Appendix A.1) to import the data from the CSV file into PostgreSQL. Appendix C.1 shows the subset of the Haiti earthquake reports dataset inside PostgreSQL.

To make the data more informative, some information needs to be created. To create information about the location of reports based on the representative areas, I used two shapefiles that store data of 10 different departments (See first map in Appendix C.2) and 41 provinces (See second map in Appendix C.2) in Haiti, downloaded from [URL 27]. The file was then generalized in ArcGIS to simplify the shape of the departments in order to improve the speed of processing with the data later (third map and fourth maps in Appendix C.2 are respectively the results after generalization of first and second maps).

Later, I converted the maps of Haiti's departments and provinces into KML format. To convert .KML files to .CSV files format, I wrote two python programs(See Appendix A.5 and A.6) which produce .CSV files that can be easily imported into PostgreSQL by using PostgreSQL script (See Appendix A.2 and A.3).The datasets of the boundaries of 10 departments and 41 provinces of Haiti are shown inAppendix C.3 andAppendix C.4.

To make the work of analysismore accurate,such as calculating the density of reports submitted in Haiti, I created a network of grids based on the boundary maps above by using the Fishnet Data management tool in Arc Map 10.3. The tool allows us to create the grid network inwhich the number of rows and columns can be configured, the size of each gridcan also be managed, and the coordinates of four corners of the grids network can be set. I created a network grid that consists of 29 columns and 22 rows; each grid cell has a size of 0.1 by 0.1 degree (11.1km by 11.1km). After that, I removed the grids that are not covering Haiti by using the edit tool in ArcMap. The final grid network is shown in Appendix C.5.

Later, I converted the map of the grid network into KML format. To convert .KML files to .CSV files format, I wrote one more python program (see Appendix A.7) that uses the same concept as the ones used for KML files of Haiti'sdepartments and provinceswhich produce .CSV data files of the grid network. Then, the .CSV file is imported into PostgreSQL by using a PostgreSQL script (see Appendix A.4) that works similarly to the case of the Haiti Departments and Provinces .CSV data file.One more attribute is then added to the grid data that consists of the centers' coordinates for each grid cell in the network, which is necessary for the further work (positioning of point symbols). The dataset of the grid network is shown in Appendix C.6.

One of the required functions in my research is user management. This function is needed to control the accessibility of the users who can be separated into two groups: administrators and volunteers.Everybody can access to the prototype, however if they are not the existing users in the database, they can only view maps, submit reports on the prototype, etc... They cannot modify a report's detail and they are not able to see who the users of the prototype are. After logged in to the prototype, volunteers and administrators can do some jobs that are not available to the others. A volunteer can do some additional jobs such as changing details of reports, deleting reports and changing his own user information such as real name (the user's name cannot be changed), email, password. If you want to be a volunteer of the prototype, you can register, but you cannot log-in until your account is activated. At the highest level, administrators can create accounts for volunteers, change information of all users; the administrators can activate or deactivate a volunteer's account. For safety reasons, an administrator cannot deactivate his own account. Appendix C.7 shows a data table of users.

#### **5.4. PHP server script and the PostgreSQL database server**

In order to make the prototype to be a dynamic web site in which the users can do the work of data and user management, we need to use a server script to interact with the database server. In this research, I will use PHP as server script programming language and PostgreSQL as database server.

##### **5.4.1. PHP**

According to [URL 34], PHP stands for "PHP: Hypertext Preprocessor". It is a server side scripting language, like Microsoft's Active Server Page (ASP). It is open source software and widely used for making dynamic and interactive Web pages. PHP can be downloaded freely from the PHP official web page [URL 34]. It can run on most of the popular platforms, such as Windows, Linux, Mac OS, etc. PHP files can contain HTML and script inside. PHP file extensions are .php, .php3, .php4, .php5, .phps or .phtml. It supports many databases, such as PostgreSQL, MySQL, Oracle, etc. PHP can work with most of the recent servers (Apache, IIS, etc.).

#### 5.4.2. Apache

PHP requires a server application to transcript its code. In my thesis, I use the Apache server to run PHP. According to [URL 35], Apache is a powerful, flexible, HTTP/1.1 compliant web server. It supports the latest protocols(HTTP/1.1 (RFC2616)) and is highly configurable and extensible. Since Apache is opensource and free, users or developers can extend it by using the Apache module API. It can operate on Windows, Netware, Mac OS, and most versions of Unix, as well as several other operating systems.

#### 5.4.3. Using PHP to connect to the PostgreSQL database

To connect to the PostgreSQL database using PHP, a `pg_connect()` function with connection strings that consist of host name, name of the database, user and password was performed. The following codes describe how to connect to PostgreSQL using PHP. Those codes are stored in a file (namely `dconn.php`) which can be called in every page. If the connection's specification is changed, it will be easier to change the data connection in this file than to change it in all of the pages of the prototype.

```
$dbconn = pg_connect("host=itcnt07 dbname=haiti port=5432 user=hoang25746 password=zxcvbnm1")  
or die('Could not connect: ' . pg_last_error());
```

After using the connection to get data in the database, the `pg_close($dbconn)` function needs to be called at the end of the same web page to free the resource of the database server.

#### 5.4.4. Using PHP to create additional information for the original data

In order to make the data about the reports more informative, I created three more attributes in the table of reports, namely, `deptID`, `provID` and `gridID` which are the IDs of departments, provinces and grids from the three tables departments, provinces and grids. They will store the data for the IDs of the provinces, departments and grids from where the reports were sent. To fill in these three attributes, I needed to write a function that could check coordinates of the reports and whether the report is located inside a province/department/grid cell boundary. The function will input the ID of those provinces, departments and grid cells into the table of reports.

The basic algorithm to check whether the reports' coordinates are inside the province/department's boundary is named "ray-casting algorithm" [URL 48]. The method is testing how many times a ray, starting from the point and going in any fixed direction, intersects the edges of the polygon. If the point in question is not on the boundary of the polygon, the number of intersections is an even number if the point is outside, and it is odd if inside. The algorithm is based on a simple observation that if a point moves along a ray from infinity to the probe point and if it crosses the boundary of a polygon, possibly several times, then it alternately goes from the outside to inside, then from the inside to the outside, etc. As a result, after every two "border crossings" the moving point goes outside. Figure 5.2 shows the demonstration of the algorithm.

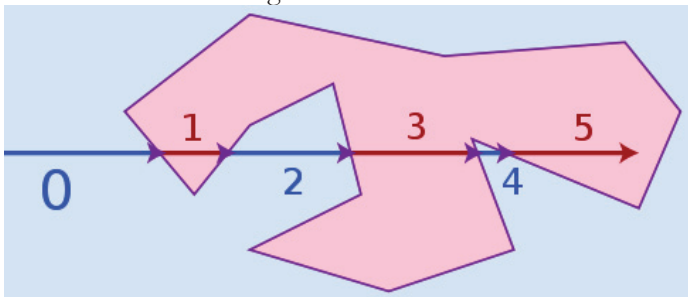


Figure 5.2: Ray casting algorithm's demonstration

Applying the "ray casting algorithm" to the case of the reports' coordinates as the points and the borders of the provinces as the polygons, we can check whether a report is inside a province or not. Because a province is inside a department, we can receive both the ID of the province and the department that covers the report.



In the case of grids, because a grid cell is a square, it is faster and simpler to use only a comparison between a report's coordinates and two opposite corners of a grid cell. If a report's coordinates are in the range of the two corners' coordinates, the report is inside the grid cell, else it is located outside the grid cell. Figure 5.3 shows the demonstration of this algorithm.

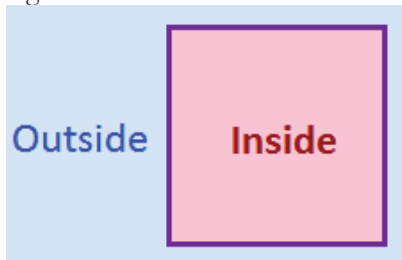


Figure 5.3: Demonstration of algorithm applied for grids

I created the PHP script (see Appendix A.8) to check and put the ID of the polygon in which they are into three new attributes in the table of reports by using the two algorithms. The result is shown in Appendix C.8. The script will be reused when the users submit new reports which need to be checked for their location in which province, department and grid cell.

#### 5.4.5. Using PHP to generate data files from the PostgreSQL database for visualizing neogeographic data

One of the data formats which can be used in many web visualization libraries is KML. However, the work of extracting a KML file from the PostgreSQL database server and transferring it to the client is a challenge, and it is unable to change the file's content if the data is updated. Hence, there is a need to use PHP to generate KML from the PostgreSQL database. The KML files that are generated have to be in the structure that is required by the web visualization libraries. The following lines describe a simple KML file:

---

```
<kml>
<Document>
<Placemark>
<name>Port-au-Prince air traffic control tower damaged - flights rerouted</name>
<TimeStamp><when>2010-01-12 04:08:00</when></TimeStamp>
<description>Some description</description>
<LookAt><coordinates>-72.292778,18.579721</coordinates></LookAt>
<Point><coordinates>-72.292778,18.579721</coordinates></Point>
<theme>I5</theme>
<tags> 5b. Structures a risque | Unstable Structure, Port-au-Prince, Ouest, </tags>
</Placemark>
</Document>
</kml>
```

---

In this research, the symbols are separated into three types: point symbols (as used in a nominal point symbol map), proportional point symbols (as used in proportional symbol maps), and area symbols (as used in choropleth maps).

For the point symbols (each representing one report), the reports in the database are categorized into 8 main categories and 51 small categories. Applying the cartographic design rules about graphic variables mentioned in chapter 4, in this point symbols map, 8 main categories are represented in equal importance, we need to use nominal representation for the point symbols. I used different images to differentiate the main categories in the map view. Figure 5.4-1 shows the lists of symbols for the 8 main categories. In the timeline, colour is used as a variable to differentiate between the main categories. The colours are the background colours of the images mentioned above, they are dark brown, blue, grey, black, grey brown, light brown, green and orange which are used for eight main categories from 1 to 8 respectively (see Figure 5.4-2).

For proportional point symbols which represent quantitative data for number of reports submitted, applying the cartographic design rules about graphic variables, the symbols are differentiated by using different sizes. To create these different symbols, I created a red-square icon in a fixed size. Later, I created a PHP script (see Appendix A.18) to change the size of that icon to different sizes that I need to use, I made the difference in size of two nearest symbols is 3 pixels(see example of proportional point symbols in Figure 5.4-3).

For area symbols which represent quantitative data for density of reports submitted, following the cartographic design rules in graphic variables, the symbols are differentiated by using different gray values (see example of area symbols in Figure 5.4-4)



Figure 5.4: Examples of symbols; 1: point symbols on mapview, 2: point symbols on timeline, 3: proportional point symbols, 4: area symbols

The identification of those symbols is stored in KML files within the tags `<theme></theme>`. To create these symbols, I created some JavaScript code to make themes in `timemap.js` files (see Appendix A.14) that contain the entire themes needed for the area and point symbols mentioned above.

In order to allow do filtering, the names of different categories (8 main categories and 51 small categories), departments and provinces are stored in KML files within the tags `<tags></tags>`. For storing the time, there are two types of times: time stamp and time span. A time stamp stores the data of events that occurred on a moment, while a time span stores the data of events that occurred in a period of time. Those types are selected depending on the objects' nature.

In the prototype, the two types point and area have to be shown on the map. For points, their coordinates are mentioned between the tags: `<Point><Coordinates>` and `</Coordinates></Point>`. For areas, their boundaries are mentioned inside the block of tags: `<MultiGeometry><Polygon><outerBoundaryIs><LinearRing><Coordinates>` and `</Coordinates></LinearRing></outerBoundaryIs></Polygon></MultiGeometry>`

The other pairs of tags: `<name></name>` and `<description></description>` mention the name and description of the event in the KML file.

In my research, 7 KML files are generated by using PHP scripts (see Appendix A.9 to A.13). They store data for reports' point symbols (used in nominal point symbol maps); departments, provinces and grid cells proportional point symbols (used in proportional symbol maps); departments, provinces and grid cells area symbols (used in choropleth maps). The polygon KML files of departments and provinces are also used as the boundary for the maps if the fill's opacity of their areas is set to 0%.

## 5.5. Implementation of the web visualization libraries

As mentioned earlier in chapter 4, I will use the combination of web visualization libraries, Timemap API, SIMILE, Google Maps API and Google Chart API. The combination of different web visualization libraries will make possibility to execute more required tasks and gives more functions than using just one individual library. In this section, firstly, I will improve some functions that are supported in those web visualization libraries. Then, for the required functions that are not supported in the four selected web

visualization libraries, I will create them by using PHP and JavaScripts. The improvement and creation of the required functions are mentioned as follows.

#### 5.5.1. Improving functionality of Timemap API

The original Timemap API allows users to combine Google Maps with SIMILE Timeline and timemap.js, hence Timemap API has functions that are supported by both Google Maps API and SIMILE Timeline. However, it has some limitations that need to be improved and fixed to meet the required functions in the exploration of neogeographic data. For example, Timemap API does not provide the real animation which can make the timeline move automatically. For the purpose of doing an overview of the data, the library needs to be improved.

Animation is a useful technique to emphasize changes and appearances and that is needed for getting an overview of the data. For example, it is helpful to show the changes of the total number of reports submitted in a province day by day during and after the earthquake. However, Timemap API does not provide the functions for animation such as play, pause, restart and control speed of animation. To show the animation, we need to move the timeline manually. In order to help users to show the animation easier and free them from unnecessary work (dragging the timeline) which make them cannot do the other work at the same time such as zooming and panning; there is a need to create the animation functions that make the timeline move automatically (play), pause, restart and make the ability to control the speed of animation. To deal with this challenge, I used the `setCenterVisibleDate()` function provided by Timemap API and JavaScript timing functions.

##### a. `setCenterVisibleDate()` function:

This function is used to move the timeline to a specific time stamp. For example:

```
t = new Date(2011,11,11);
setCenterVisibleDate(t);
```

After running this script, the timeline will move its center to 11/11/2011.

##### b. JavaScript timing functions:

JavaScript provides two functions in timing, namely, `setTimeout()` and `clearTimeout()`. `setTimeout()` is a function that executes the other function/code after a time interval. `clearTimeout()` clears the `setTimeout()`. For example:

```
time = 1000;
t=setTimeout("alert('Hello')",time);
clearTimeout(t);
```

After executing the first and second scripts, the message box “Hello” will appear after every second (1000 milliseconds). When we call the third script, the second script is disabled.

##### c. Creating speed and time step controller

One of the required tasks in animation is providing the ability for users to change the speed of animation. Using a slider is a comfortable way to control the speed of animation. I borrowed the code that was created by Erik Arvidsson [URL 41] to create the slider. The scripts in Appendix A.19 describe how to create a slider in the prototype.

Due to the temporal nature of the reports submitted, the period between the times two reports were submitted might be very short, or they may even have been submitted at the same time. Therefore, the speed of animation should not be too fast. I used two sliders to control the speed of animation. The first one controls the time step between two scenes; it has range from 1 to 60 minutes. The second one controls the speed of change between two scenes; it has range from 1 frame per second to 10 frames per second. For example, if users set the time interval step to 15 and the speed to 4 frames per second, the next scene will be the scene of 15 minutes after the current one, and in 1 second, the timeline will move for a period of 60 minutes.

##### d. Creating animation functions

Applying the functions mention above, I implemented the animation controller by creating a set of JavaScript functions: `play()`, `pause()`, `restart()`. The codes are shown in Appendix A.20.

In addition, sometimes the users want to focus on a specific date. They can do that by moving the timeline until the time they need is shown in the visible range of the timemap. However, if the time space between the current time and the wanted time is too long, it will take a while to move the timeline. Thus, a `gototime()` function is needed to let users jump to a specific time. The code of the `gototime()` function is shown in Appendix A.21.

By using animation functions and `gototime()` function, users can see the overview of the data in a animated way. With the Timemap API, the reports' distribution can be shown in mapview, when the users use animation function, the appearances of the new reports will be shown on the map. Users can also use `gototime()` function to see the data for the day they want to see. Those activities mentioned above are in the stage of overview in the Visual Information Seeking Mantra spread by SHNEIDERMAN (1996).

### 5.5.2. Creating zooming and filtering functions

According to the Visual Information Seeking Mantraspread by SHNEIDERMAN (1996), the next stage after overview is zoom and filter. However, Timemap API does not provide the ability to change the data for different zoom levels which is a need to satisfy one of the cartographic design rules about scale and resolution in map design. Moreover, although Timemap API support filtering, in the original version of Timemap API, the option of doing filtering is disabled. Hence, it needs to be changed to do the filtering function that is not available in the original version.

#### a. Manage zoom level by using Google Maps API

Some triggers are required to show different data at different zoom levels such as showing data for the whole of Haiti at a small zoom level, showing data for the departments in Haiti at middle zoom level and showing data for the provinces at big zoom level. The triggers perform a function whenever the zoom level of the map is changed. Google Maps API provided the `GEvent Listener` function that is suitable for this purpose. Whenever the map is in a certain range of zoom level, we can manage which KML files are shown, which files are hided. In addition, for different zoom levels, if we use the different symbols (in the case of proportional symbols map), we can also change suitable legends for them. The codes in Appendix A.22 shows how to config `GEvent Listener` function to make changes on data available and symbols available in the prototype.

#### b. Manage filtering functions

Timemap API has a loader file named `kml.js`. This is a loader class for loading KML files. By using this file, Timemap API supports all geometry data types (point, polyline, polygon, and overlay) and multiple geometry types. By default, Timemap API only loads some default tags in a KML file such as name, timespan, timestamp, theme, description, etc. In my research, the `<tags>` tag needs to be read and stored as a variable for the filtering function. The code below was added into `kml.js` to make the `<tags>` tag to be able to read and stored as variable "tags":

```
data.options.tags = getTagValue(pm, "tags");
```

After adding the code above, the filtering function based on the `<tags>` tag is available in `timemap.js`. I created several options for users to filter the data such as filter by category, filter by department and filter by provinces. After performing those functions, the `timemap.js` will eliminate from the user's view the items that do not meet the requirements. However, if the users want to see all the items again, a simple trick is needed. The codes of those functions are shown in Appendix A.23.

Next, two combo boxes were created to give users options to do filtering by the values of departments and provinces' names in the combo boxes. Those combo boxes were generated by using PHP scripts which connect to the database and get the values for the combo boxes. The code below is used to create a

combo box for departments' names. The one which is used to create a combo box for provinces' names has a similar structure:

For the case of categories: because it is difficult to show all 8 main categories and 51 small categories on one web page, I borrowed and changed an open source code to create a pop-up menu with JavaScript and CSS [URL 42] to show only the list of 8 main categories. Whenever users move the mouse pointer on a main category's name, the list of its sub-categories will be shown. The codes for the first category are shown in Appendix A.25; the other codes for other categories have a similar structure.

However, the filtering function of the Timemap API supports to filter only one tag. Hence, this Timemap API supported function cannot help users to discover correlations/conflicts between and in the data. In order to do this task, I created a filtering function by using PHP and PostgreSQL select query with different input parameters. By default, the map will show all of the data (all of the input parameters are blank). When users select some requirements, only the data which meet those requirements are shown. The requirements are time range, in which departments, in which provinces and in which categories. This function enables users to do filtering of multiple conditions. With this function users can look for the reports in some categories that are submitted in some provinces and in some departments in a specific time period. Appendix C.9 shows the set of filtering functions I created.

### 5.5.3. Designing the interface of the web pages

Timemap API is the combination of Google Maps API, SIMILE Timeline and timemap.js. Hence, to make use of those libraries, we have to include them in the web pages. There are several versions of SIMILE Timeline and timemap.js. In my research, I use SIMILE Timeline version 2.3.0 and timemap.js version 1.6. The steps of designing the interface of the web pages will be described below.

Firstly, we need to include the three web visualization libraries into the <head> tag of the pages. Before we initialize the timemap and timeline, we can define some themes for the bands that define how the bands can be shown. Next, we have to initialize the timemap and timeline; define the parameters which are needed for the timemap and timeline. Timemap.js requires defining the bands of the timeline to make the timeline able to start. The data of reports might be updated every minute. Hence, I use three bands on the timeline, they are HOUR, DAY and MONTH. They can show the temporal data in different scales.

Next, to set the start time of the timeline to the earliest incident's date and define which KML file will be visible at the start of the application, we need to add `datadisplyFunction()` function before closing `init()` function (function to initialize timemap and timeline):

Finally, the <div> tags that contain the timemap and timeline need to be defined in the body of the webpage to make timemap and timeline to be able to show. The codes to do put timemap and timeline into the prototype are shown in Appendix A.26.

The interface of the timeline is shown in Figure 5.5:

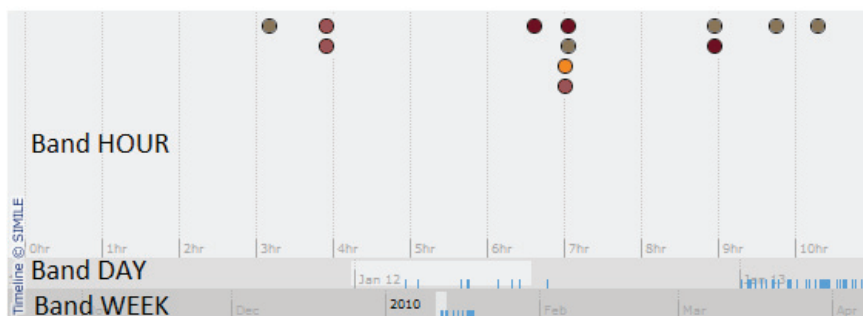


Figure 5.5: The timeline's interface



A prototype for the exploration of neogeographic point data with cartographic visualization tools

#### 5.5.4. Improving the infowindow

Detail-on-demand is the last stage of the Visual Information Seeking Mantraspread by SHNEIDERMAN (1996). Timemap API provides a pop-up window for users to see the detail of each item in the map view. In the original version of the Timemap API, the pop-up window shows all the information in the `<description>` tag (see Figure 5.6). All of the information is shown on the same line and if the string is too long, the string will jump to the next row. To make a better view for the users, the pop-up window needs to be improved. To solve this, I modified some codes in the `timemap.js` file and created some CSS codes to manage the interface of the infowindow (see Appendix A.27). The result after the modification is shown in Figure 5.7.

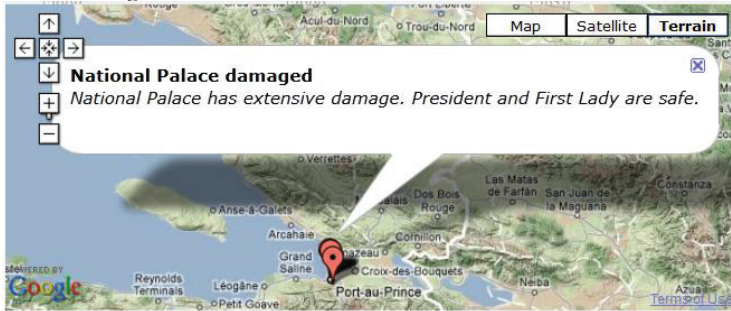


Figure 5.6: Original pop-up window

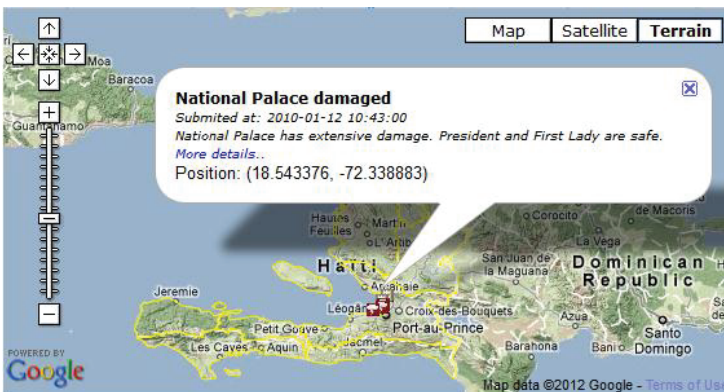


Figure 5.7: Improved pop-up window

#### 5.5.5. Using progressive.js to show huge amounts of data

Timemap API provides some types of data loaders, one of them is `progressive.js`. Because neogeographic data are usually extensive, if the timemap has to load all of the data at the beginning, the application could be extremely slow or unable to start. This JavaScript class allows the timemap to load data in a “progressive” way. That means, Timemap API does not need to load all of the data at the time it starts, the data can be loaded when users move the timeline to a period for which data are available but not loaded yet.

Firstly, we need to import the `progressive.js` file into the application, inside the pair of tags `<head></head>`:

```
<script type="text/javascript" src="../../loaders/progressive.js" ></script>
```

Then, the dataset in the `init()` function needs to be defined as shown in Appendix A.28. The KML file also needs to be defined to be able to load using a start and end condition, the codes can be found in Appendix A.9 and A.10.

#### 5.5.6. Using Google Chart API to discover correlations/conflicts

One of the tasks that were mentioned in Table 3.2 is “to discover correlations/conflicts”. To do this task, the data of different categories need to be shown together in order for the users to analyse the influences among them. By using Google Chart API, the selected data that users want to discover can be shown in a

diagram. In my prototype, I have chosen the pie chart to show those data. As mentioned in section 5.5.2, I created a filtering function that allows users to filter the data with more than one requirement. If users select more than one category in doing the filtering, a pie chart will appear in the infowindow to show the data of those categories. The codes that I used to show the data of the categories are inside Appendix A.9 to A.13. Figure 5.8 shows an example of an infowindow with a pie chart showing three categories.

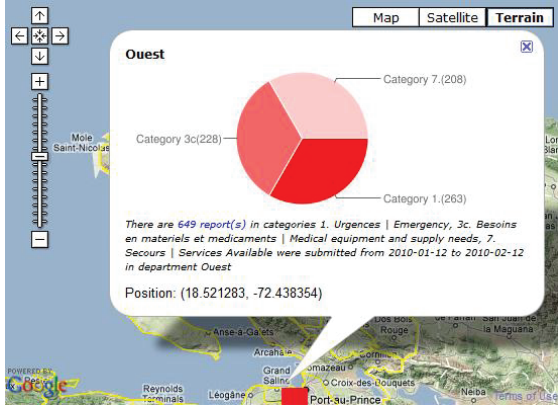


Figure 5.8: Infowindow with piechart

### 5.5.7. User management

As discussed earlier, user management is a required function in my research. The users are required to log-in to activate their authority. The record of their log-in needs to be stored in the time they are working with the prototype. PHP supports session variable to store information about, or change setting for a session of a user. Session variables hold information of one single user; they are available to all pages in one web application. In this research, I use session to do the works of user management such as storing user's information about real name, user name, user level and. The session support allows us to store data between requests in the `$_SESSION[]` superglobal array. When a visitor accesses a web page, PHP will check explicitly through `session_start()`, which is called on top of the web page, whether a specific session ID has been sent with the request. If this is the case, the earlier saved environment is recreated.

Applying session into my prototype, I created some web pages to do authority management:

#### a. Register/create user account page

This is the page to add a new user account into the database. The people who want to be volunteers of the prototype can use this page to create their account. However, their accounts cannot be accessed until the administrators activate them. Administrators of the prototype also can use this page to create new volunteers' accounts. The accounts which are created by administrators do not need to be activated before they can be accessed. To do the two different actions, a trick has been used in the codes below. There are two pages, a register/create form page and a registration page:

Figure 5.9 shows the interfaces of the register/create form page, the left one is the interface for people who want to be volunteers, the right one is the interface for administrators.

<p>Hello Guest!</p> <p><b>Register new volunteer's account</b></p> <p>Username <input type="text"/></p> <p>Real name <input type="text"/></p> <p>Email <input type="text"/></p> <p>Password <input type="password"/></p> <p>Retype password <input type="password"/></p> <p><input type="button" value="Register"/> <input type="button" value="Cancel"/></p>	<p>Hello administrator Hoàng Anh Đức!, logout</p> <p><b>Create new volunteer's account</b></p> <p>Username <input type="text"/></p> <p>Real name <input type="text"/></p> <p>Email <input type="text"/></p> <p>Password <input type="password"/></p> <p>Retype password <input type="password"/></p> <p><input type="button" value="Create"/> <input type="button" value="Cancel"/></p>
---	---

Figure 5.9: Registration pages

I used the function md5 (MD5 Message-Digest Algorithm) in the SQLquery to create a high security string for passwords. Even if somebody can access the database, he cannot know the password of users because the string is very hard to be decoded. For security reasons, it is impossible to register an administrator account from the prototype. I created some PHP codes for creating an administrator account which was used only once when I installed the database. The PHP codes to do the registration action and create a new administrator are shown in Appendix A.29.

#### b. Log-in page

A log-in page is the page to access an existing account and register a new session of a user. All of the session variables are registered here. Username and password are the inputs for the log-in page. I used AJAX (Asynchronous JavaScript and XML) to show the log-in page results in the index page. The PHP codes of the log-in page are shown in Appendix A.30. Figure 5.10 shows possible results after a log-in action, the left one shows an unsuccessful result, the right one shows a successful result.

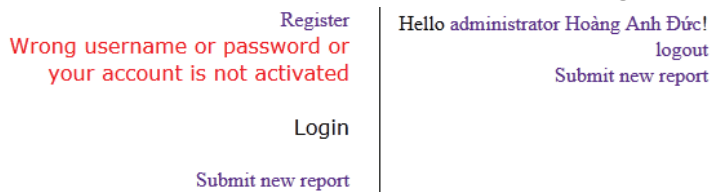


Figure 5.10: Log-in action results

After doing a successful log-in action, a user can access some additional functions based on their user level. A volunteer (user level is 2) can change his own information, modify the details of the reports including approving and verifying reports and is able to see quick report access links in the report's details page. An administrator (user level is 1) can do some more activities than a volunteer can do, such as activating/deactivating a volunteer's account, creating an activated volunteer's account, and seeing quick user access links in the user's details page.

#### c. User's details page

I created this page in order to help administrators and volunteers to see and change their information. By using the user's level, I separated the page's interface into two types: one is for volunteers, and the other one is for administrators. For the guest who accesses this page by accident, a sorry message will appear. The interfaces are shown in Appendix C.10, the left one is for volunteers, the middle one is for administrators and the right one is the message for guests.

#### d. Log-out page

After doing the works with the prototype, authenticated users need to stop their working session. By accessing the log-out page, all of the created sessions will be destroyed. In this page, I use the unset() function to do the job. The codes of the log-out page are shown below:

```
<?php
session_start();
if (isset($_SESSION['uid'])) {session_destroy();}
?>
```

#### 5.5.8. Submitting and updating reports

To maintain and develop the prototype, two functions of creating and updating reports are required. A submit report form is required for users of my prototype to input a new incident. An update report page is needed for administrators and volunteers to check whether the information in the reports is right or wrong. If the information is wrong, they can delete or just leave it in an unapproved state, else they can set the states of the report to be approved and verified.

In the two pages, I use a JavaScript function (see Appendix A.15) for using Google Maps API to show the map of Haiti. Users can use this map to search for their location. A JavaScript function (see Appendix

A.16) is used to get the coordinates of the place that was selected on the map. I created a JavaScript function (see Appendix A.17) to add and remove the categories of a report. After clicking the submit button, the coordinates of the report will be calculated by using the same algorithm of checking the location of a report (see section 5.4.4). The results, IDs for departments, provinces and grid cells the new report is in, will be stored in the database. If the new report is submitted outside Haiti, those IDs will be stored as NULL values. In the two pages, authority is also applied. On the submit report page, normal users can only submit unapproved reports without categorizing, while volunteers and administrators can submit approved reports that were categorized by them. On the report's details page, only volunteers and administrators can change the information, a guest cannot. The interfaces of the two pages are shown in Appendix C.11 and Appendix C.12. The left ones are the interfaces for unauthenticated users; the right ones are the interfaces for volunteers and administrators.

#### **5.5.9. Searching reports**

A required page in the prototype is a webpage that allows users to search for specific reports by using some requirements, such as reports containing a specific string in their names, locations and descriptions, reports that were submitted in a certain period of time or reports in specific departments, provinces and categories. After a user enters the requirements, a list of reports that meet the requirements will appear. By clicking on a report that is shown in the list, a frame that contains the information of this report will appear. Volunteers and administrators can use this frame to change the information of the report. Appendix C.13 shows the interface of the search report page.

Appendix C.14 shows the interface of the result after doing the search query in a search report page. If user accesses to the web page `reportlist.php` without any searching requirement, it will show the list of all reports submitted in Haiti.

#### **5.5.10. Changing the map types in the prototype**

The last required function of the prototype, as mentioned in Table 3.3, is "Select map types by purposes". According to Table 3.2, different geographic questions need to be answered by different types of maps. I created a home page that contains a list of possible geographic questions. Whenever a user clicks on a question in the list; a map which can answer that question will appear in the same page. To do this job, I created a frame below the questions list. A JavaScript function was applied to the questions' text to make each question link to one type of map. Some different questions can be solved by the same type of map; those questions' texts have the same link. But some others can be solved by more than one type of map; those questions get optional choices for the users to choose which type of map they want to use. The codes below are example of the JavaScript function I applied to the text of one of the questions to do the task "Select map types by purposes". Figure 5.11 shows the result after clicking on one of the questions.

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

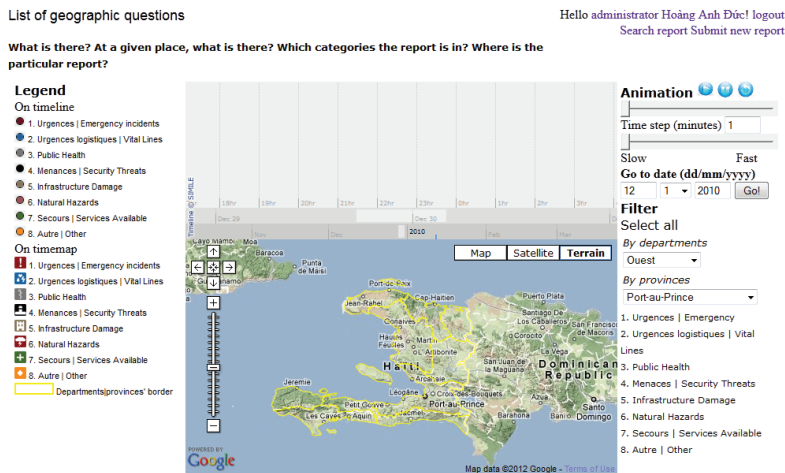


Figure 5.11: Interface of the prototype's home page, with nominal point symbols map

I created three types of maps: nominal point symbol maps, choropleth maps and proportional symbols maps. With different input parameters, the maps show different interfaces and results. I made a parameter for my prototype, named showpast. This parameter decides whether the map enables filtering by time or not. For the first option (with parameter showpast=1), all the data from the starting point to the current time of the timeline's topband. This option can be used to calculate the total number of reports submitted at that moment which is useful to establish trends. Especially when we use an animation function, the changes will be shown clearly. When we use the second option (with parameter showpast=0), only the events which happened in the time range inputted by the users can be loaded. This option is useful to estimate amounts in a specific period of time. Figure 5.25 shows the interface of the prototype showing a nominal point symbol map with the option showpast = 0.

In order to answer the questions like what the density/intensity of the reports in different areas is, I used choropleth maps to show the data. They also have two interfaces belonging to the option showpast: the additional parameter is showgrids (decide whether the map shows data by using the grid network or not). When we zoom in and out, the dataset will be shown differently, based on the zoom level. In large zoom levels, the map will show data of small areas (provinces); in smaller zoom levels, the map will show data of bigger areas (departments). When option showpast is 1, the animation function makes the polygons' colours change based on the value of the density of reports or the total number of reports submitted from the earliest day in the database to the current time on the topband. Figure 5.12 is the choropleth maps which based on polygons of departments and provinces of options showpast = 1 and showgrids = 0

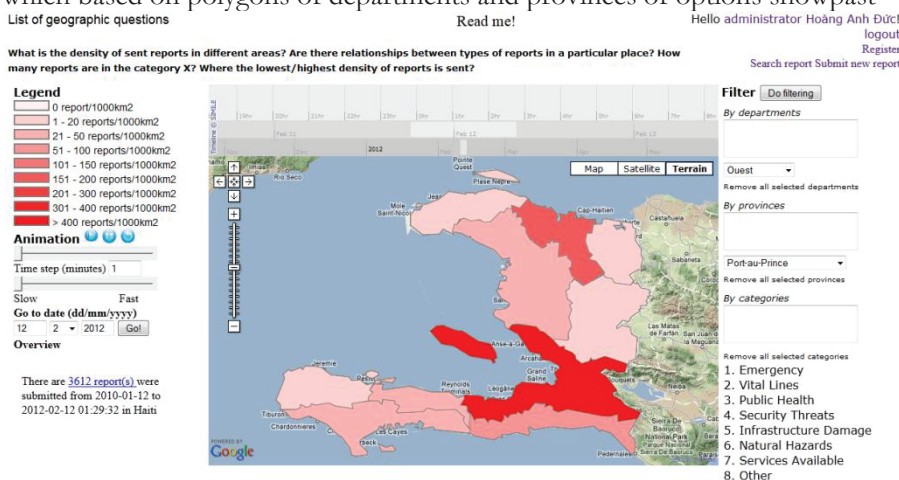


Figure 5.12: Interface of prototype showing choropleth map with options showpast=1 and showgrids=0



The data were shown for the polygons of departments and provinces. As the uneven geographic areas of the polygons may negatively influence the perception of the distribution of the neogeographic data, I used a grids network to separate the area of Haiti into small grid cells of which the areas are equal. Because the grid cell areas are much smaller than those of the departments and provinces, the classification interval that was used for departments and provinces to show the density of reports is not suitable. Hence, I had to use a different classification interval to show the density of reports when the data are shown for the grids network. Figure 5.13 presents the map showing the density of reports by using the grid network.

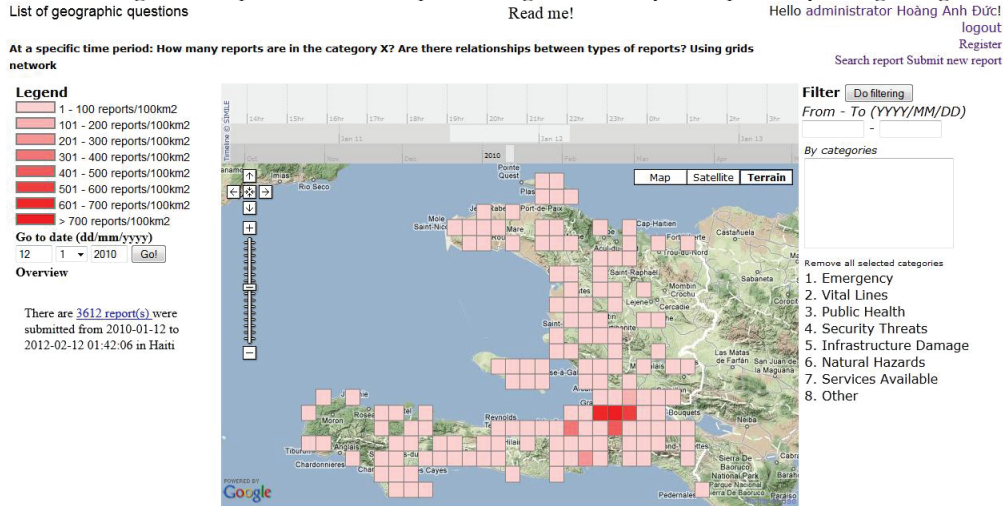


Figure 5.13: Interface of prototype showing choropleth map with grid network

The nominal point symbol map above can show the details of all reports. However, the map becomes cluttered if there are too many points on screen. To solve this problem, and to answer the other questions, the third type of map, namely, the proportional point symbol map is used. This type of map helps us to give the answer to questions such as “how many reports were submitted at a specific location and in a specific time period?” The maps in Figure 5.14 and Figure 5.15 and gives the answer of where and how many reports were submitted by showing different point symbol sizes at different places. Users can use the function filtering by time to specify the time period. For different zoom levels, the sizes of proportional symbols are changed automatically, the legends also change to fit the symbols on the map.

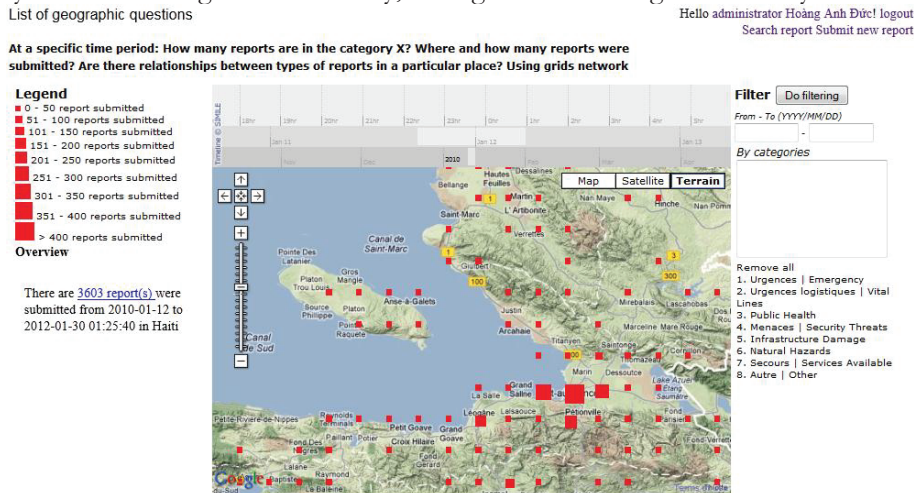


Figure 5.14: Interface of prototype showing proportional symbols map with grid network

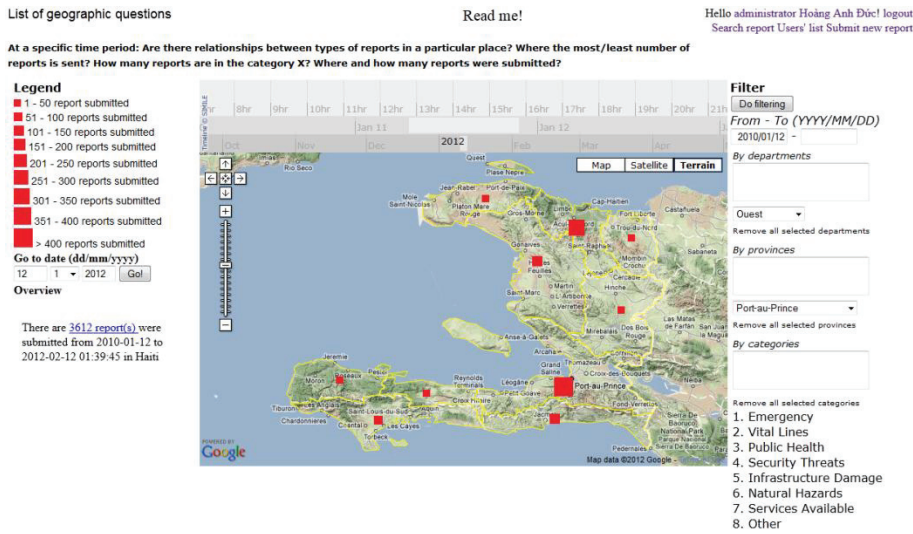


Figure 5.15: Interface of prototype showing proportional symbols map on departments

## 5.6. Summary

The implementation process of the prototype for the exploration of neogeographic data of the Haiti earthquake as a case was presented in this chapter. The implementation process was based on the tasks that were discussed in Chapter 3. Most of the views and functions that were discussed in Chapter 3 were implemented. Improvements of the functionalities as well as the GUI of the original selected web visualization libraries have been described as well. Some creations of new functionalities that are not supported by the four selected visualization libraries, Google Maps API, SIMILE, Timemap API and Google Chart API were described. Now all functionalities of the prototype should be tested and evaluated. The next chapter will describe the first evaluation of the prototype.

## 6. PROTOTYPE EVALUATION

### 6.1. Introduction

The previous chapter described the conceptual model, design and implementation of the prototype step by step. In this chapter, firstly, the usability testing of the prototype will be described. Then, the results of the test will be shown. Finally, some recommendations for the improvements of the prototype will be discussed.

### 6.2. Evaluation

#### 6.2.1. Objective of the evaluation

According to [URL 43] (the international standard ISO 9241-11 of the International Standardization Organization), usability is defined as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use”.

As mentioned earlier in chapter 3, the prototype in my research provides a map view, a temporal view (timeline), attribute view and animation with many functions such as search, zooming, filtering, user management, change map types, etc... Those views and functions are used by users to explore a neogeographic dataset of the Haiti earthquake of 2010. The main objective of a usability test of the developed prototype was to discover what works well, what has not been used and what needs to be improved. Normally, in a prototype design cycle there are several evaluations after some improvements have been made, but within this research only one initial evaluation has been executed and is the end phase that leads to conclusions and recommendations for future research.

#### 6.2.2. Selection of methods

There is several usability testing methods such as focus groups, thinking aloud, interviews, eye tracking, etc... In this research, to assess the prototype from the usability perspectives mentioned above, several methods are used together: interview, video observation, screen logging, thinking aloud, and eye tracking with the Tobii X60 eye-tracker and Tobii Studio software. All of the methods mentioned above are available in ITC's usability test lab.

The combination of various methods will provide different type of data such as eye movement data, voice, video, screen activities and answers to the questions. By analysing those data of different methods, we can have particular information about the usability issues of the prototype such as data about the satisfaction of users from some additional questions in the tasks, data about which part of the interface is the most attractive, which part is needed to be redesigned from eye tracking data or what functionalities we need to improve from the interview data, etc.

Moreover, when we use various usability test methods, if one of the research methods did not work out well, we have at least still the results of the other methods. For example, during the test, the participant might cover the eye tracker when he/she rises the handout to read and do the tasks at the same time. In this case, the data of eye tracking cannot be used but the other data from the screen logging and video recording may show that where he/she was looking at or doing with from the position of the mouse pointer on the screen.

#### 6.2.3. Describing methods

As mentioned above, in this research, the methodology is a combination of the following methods:

##### a. Think aloud protocol

According to DELIKOSTIDIS (2007), the think aloud protocol takes into account the investigation of what the users are thinking and doing when they are interacting with a design. If the participant is thinking aloud well, we can understand how he/she is performing the task, and how he/she is thinking about reaching his/her goal. We will hear about why he/she is doing the things he/she is doing, and the words he/she uses to describe it all. We also will get verbal feedback about how the participant feels about what is happening because he/she may say that he/she is frustrated, or annoyed or even happy. The main disadvantage of this method is the possible conflict between the participant's task execution and the verbal expression or description. Some participants do not want to talk aloud if they make mistakes. Some others do not want to talk too much or they prefer writing to talking. Thinking aloud may be unnatural and distracting to the users. There also some other information that are useful for the usability test, we need to use the other methods besides think aloud protocol to collect those information.

#### **b. Interview**

An interview is a usability testing method that involves the verbal interaction between the researcher and the participants for a specific intention. During the test, the output can be recorded by using voice recorder, or writing. It can be structured, following a strictly predefined frame of construction, content and questions; or be unstructured, in which the researcher can modify any part of the process according to the real situation during the test. The structured interviews are easier for analysis, as through their equable data results they allow for immediate comparative study. However, analysis of results can be problematic because the researcher's style and personality may affect the response of the participant.

#### **c. Video observation**

Video observation aims at capturing the participants' interaction with a particular design during testing process. During the test, a camera may record the participant's interactions, behaviour and/or facial expressions. However, it is difficult and time consuming to analyse the data especially if the test's duration is long. This method can support the other methods in usability testing when some strange activities happened, this could show the reason why those activities happened.

#### **e. Screen logging**

Screen logging is another method in usability testing which provides video data, used for the recording of timing and type of incidents that happen during the interaction with an interface design. This method is mainly applied for computer interfaces and software evaluations. By using this method, the required time for the completion of a task, or the errors that occurred during the usability testing of a design can be efficiently investigated. Disadvantages of this method are the possibility of problems issued by the excessive amounts of data to be analyzed. It may also need a special logging software customization in order to be used by a particular hardware system.

#### **f. Eye tracking**

POOLE & J.BALL (2004) defined eye tracking as "a technique whereby an individual's eye movements are measured so that the researcher knows both where a person is looking at any given time and the sequence in which their eyes are shifting from one location to another". The eye tracking method allows us to determine eye movements and eye-fixation patterns of a participant. Eye tracking can reveal information about human behaviours and/or usability problems which may not be obtained by conventional usability data collection methods (COOKE, 2005). However, this is an expensive usability testing method and it needs special software to analyse usability data.

#### **6.2.4. Software and hardware for the usability test**

In the usability testing lab, there is a usability testing system which allows us to do the wanted combination of usability testing methods. There is a camera for video observation of the face of the test person, a Tobii X60 eye-tracker [URL 49] which catches the eye movements and during the test the sound

of the thinking aloud and the interviews is recorded by using a microphone and the changes on the screen are logged. The software in the test computer is Tobii Studio 3.0.1 [URL 50] which helps us to record all of the data from screen logging, camera, microphone and eye tracker. With this system, we can make a usability test to collect usability testing data from various methods as mentioned above, think aloud, video observation, screen logging, interview and eye tracking. After the participants have done their tasks, a short interview will be conducted; the interviews are also recorded by Tobii Studio 3.0.1. The answers to the user satisfaction questions were collected from the answers on paper.

#### **6.2.5. Conducting the test**

##### **a. Objective and questions for testing the prototype**

The objective of the test was to evaluate the usability of the prototype for the exploration of neogeographic point data. The evaluation is based on the use of functions and views while the participants tried to do some given tasks and tried to find the answers to the questions posed in these tasks. Four tasks were conducted in the usability test and each task consisted of two small questions. Three tasks are organised according to the Visual Information Seeking Mantra that was discussed in section 3.3: overview, zoom and filter, and details-on-demand. One more task was used to test the usability of the data and user management functions of the prototype. The usability testing tasks and their questions are shown in Appendix D.2. The main tasks of the usability test and their questions are:

Task 1: Overview

Q1: Haiti is subdivided into Departments. Until now (06/02/2012), in which Department, were the lowest and the highest density of reports submitted: Artibonite [1], Centre [2] or Ouest [3]?

Q2: On which day were more reports submitted in the whole Haiti: 13/01/2010 or 14/01/2010?

Task 2: Zooming and filtering

Q1: List 2 grid ID's of two square symbols that had more than 200 reports submitted from 2010/01/12 to 2012/02/06?

Q2: Each Department in Haiti is subdivided into Provinces. What are the densities of reports submitted in the two provinces Léogâne and Port-au-Prince from 2010/01/12 to 2010/02/12?

Task 3: Details on demand

Q1: Where and when was the first report in category "8. Other" submitted? (Provide time, and coordinates)

Q2: Compare the number of reports submitted in categories "1. Emergency" and "7. Services Available" from 2010/01/12 to 2010/01/19 in department Ouest, which category has more reports? How many for each category?

Task 4: Data and user management

Q1: Log-in with the account Volunteer, password zxcvbnm1, try to submit a new report. What is the ID number of the report that you submitted?

Q2: In the user's details page, you can access to the page that shows the list of users. See in the page, how many users are there in the list?

##### **b. Test procedure**

Firstly, I sent emails to the PhD students in the Geo-Information Processing Department, of the Faculty ITC of the University of Twente to ask whether they wanted to participate in the usability test. Before the actual tests began, two pilot tests were conducted to estimate the time needed for a test and to discover any difficulties with the task description and execution. Then, we created a schedule for the test, and arranged the participants' names into 6 time slots of the schedule based on the available time of each participant. Some personal information of the participants are mentioned in Appendix D.3



For the usability testing, I installed the database on a database server of the ITC local network (itcnt07) and installed an Apache server on my laptop. I bought the domain name www.hoang25746.com to link to the IP address of my laptop and put the prototype onto the ITC local network. Hence, everybody who is working inside the ITC network can access my website through the domain name www.hoang25746.com. In the usability testing lab's computer, we needed to run Tobii 3.0.1 and change the default web browser to Firefox, because the prototype does not work well with Internet Explorer.

The usability testing session was executed on the 7<sup>th</sup> February 2012 and in the morning of 8<sup>th</sup> February 2012. Each participant did the test individually. The average time spent by a test participant was about 1 hour and 15 minutes. During the test, some participant did not talk out loud; I have to remind them to talk what they were thinking. The eye tracking data for one participant could not be used because the system could not recognize this participant's eyes. The test plan and the interview questions are shown in Appendix D.1. The whole test contains 10 steps:

- 1) Welcome the participants and give an introduction of myself, the usability test, the prototype as well as the software and hardware in the lab.
- 2) Give the participant an introduction for the tasks.
- 3) Ask the participant if he/she has any question about the rule he/she has to play in the test and about the introduction.
- 4) Do the calibration for the eye tracking system.
- 5) Participants have about 5-10 minutes to get familiar with the prototype.
- 6) Participants execute the tasks to answer the questions. During the test, participants could move their head a little bit. However, they could not move their body or leave their chair and could not press Esc which will make the system stop. When the participants are looking for the answer for each question, they will say aloud what they are thinking and what the answer for each question is.
- 7) The data about eye movement, video, screen logging and the voice of the participants are recorded.
- 8) After answering all the questions in the tasks, participants will answer some questions about the ease of doing the tasks as well as their satisfaction of the prototype.
- 9) After the participants finish the tasks and questions, a short interview is conducted to collect data from participants about what the problems of the prototype are, and what improvements are needed.
- 10) Farewell the participants.

### **6.3. Results**

#### **a. Effectiveness**

Effectiveness of the prototype can be accessed by the correctness of answering the questions in the tasks. For privacy reasons, the participants got a random number which is different from the order in the test's schedule. Appendix D.4 shows the summary of the correctness of the answers that were provided by the 6 participants. All of the participants gave a wrong answer for question 2 of task 2 and the last question was answered correctly by only half of the participants. The reasons why the participants gave wrong answers are:

- For question 2 of task 2, after I analysed the data of where the participants were looking at and from what they talked as well as the position of mouse pointer on the screen and the numbers they put, the reason was found. 3 participants used the total number of reports submitted in the two provinces Léogâne and Port-au-Prince to answer the question of density of the reports. The others used the density of sent reports in the two provinces to answer. However, they input the wrong time period for the filtering function.

A prototype for the exploration of neogeographic point data with cartographic visualization tools

- For the last question was answered correctly by only half of the participants because the ones who provided the wrong answer looked at the ID of the last user to put into the answer instead of the total number of users (see Figure 6.1). During the time I checked the prototype I created 3 users and deleted 2 of them and that made the ID of the last user greater than the total number of users.

Hello volunteer Participant! [logout](#)  
[Home](#) [Search report](#) [Submit report](#) [Reports' list](#)

Showing **11 users in 1 pages** Correct answer

1 pages: [First](#) [Last](#)

ID	User's name	User's email	User's real name	User's level
1	Administrator	ducha.hung@gmail.com	Hoàng Anh Đức	Administrator
2	Volunteer	participant@itc.nl	Participant	Volunteer
5	Hari	dhonjuh@yahoo.com	Hari Krishna Dhonju	Volunteer
7	yeayallig	bogale26283@itc.nl	Yetnayet A. Bogale	Volunteer
8	CodeRed	odeyemi07463@itc.nl	Chris Odeyemi	Volunteer
9	eden	eden25571@itc.nl	ugyen eden	Volunteer
10	dube26865	dube26865@itc.nl	Timothy	Volunteer
11	susheel	susheel dangol@gmail.com	susheel	Volunteer
12	baral26430	baral.tikaram@gmail.com	tikaram	Volunteer
13	Swamy	bmdevswamy@hotmail.com	Mahadevaswamy.B	Volunteer
14	abrha	abrha26285@itc.nl	kiiflom	Volunteer

Where most of the participants looked at

Username Volunteer

Real name

Email

Password

Retype password

Figure 6.1: Screen shot of users' list page

## b. Efficiency

Efficiency refers to the amount of endeavour that users need to achieve their purposes. In this research, I provided participants questions about the difficulty of answering the questions in the tasks (see Appendix G.2). The participants who need more time to do a task filled hard or very hard to those questions, in contrary, the persons who have done the task quickly answered those questions as easy or very easy. The levels of difficulty for each question of the tasks (as mentioned in section 6.2.5) are rated from 1 to 5 (see Appendix D.5). With the questions of task 4 (data and user management), the average score is 3.67, the lowest average score is 3 for task 1 (overview), task 3 (detail on demand) and task 2 (zoom and filtering) are in the middle with the average score are 3.42 and 3.08 respectively That means, the tasks are easier and easier for the participants to do after they have done the previous ones. The reason might be that the participants did not have enough time to get familiar with the prototype; after getting familiar with the prototype by doing the first tasks, the participants found it less difficult to do the later tasks. One other reason is according to their background and experience with interactive maps and timeline as well as animation. For whom who has worked with interactive maps, timeline, animation, and has good knowledge about cartographic can do the tasks faster than the others. The overall score for the whole test is 3.3.

## c. Users' satisfaction

Satisfaction is the opinion of the users about the prototype: is the prototype good or poor. After finishing all the tasks, a short question at the end of the questionnaire was used to get information about their satisfaction with the prototype as follows:

Please rate your satisfaction of the prototype?

Very poor	Poor	Normal	Good	Very good
-----------	------	--------	------	-----------

The satisfaction levels are classified into 5 levels. The average score of the satisfaction level of all participants is 4.17 (see

AppendixD.6). The average score reflects the satisfaction of the users with the prototype.

## d. Eye tracking data

As mentioned above eye tracking data can give information that the other usability testing methods cannot give. There are several outputs that can be extracted from the eye tracking data, they are scan path/gaze plot, hot spot/heat map and area of interest. Scan path/gaze plot is a consequence of fixations and saccades. A scan path can be known as a "view route". A hot spot/heat map is based on the summary of

gaze positions gained from multiple sessions and users. An area of interest analysis allows to define areas and to compare eye-tracking data for those areas, e.g. number of fixations, fixation duration etc.

Tobii 3.0.1 software can create all of the three outputs above. Unfortunately, it supports directly creating those outputs only for testing web applications using Internet Explorer as the web browser. In the usability test, Firefox was used as the web browser, hence Tobii 3.0.1 could not export the outputs of scan path/gaze plot and hot spot/heat map directly. In stead, ordinary screen logging had to be used in Tobii Studio and because the prototype is designed as a prototype in which the users can select different maps for different purposes, each map may have a different design. Hence, the area of interest cannot provide accurate data for the whole time of the test. Moreover, the participants selected different maps many times during the test. For that reason, if we want to make an analysis of area of interest, we had to split the recordings into scenes for small periods of time. This work is time consuming and due to a lack of time, I did not use the area of interest for the analysis of the eye tracking data. The only data that we could use from the eye tracking data collected is viewing one by one where test persons looked at when answering the questions and executing the tasks. That information has been used to give reasons why the participants answer wrong some questions in the tasks. Figure 6.2 shows one example of eye-tracking data that gives reason why participants answer question 2 of task 2 incorrectly.

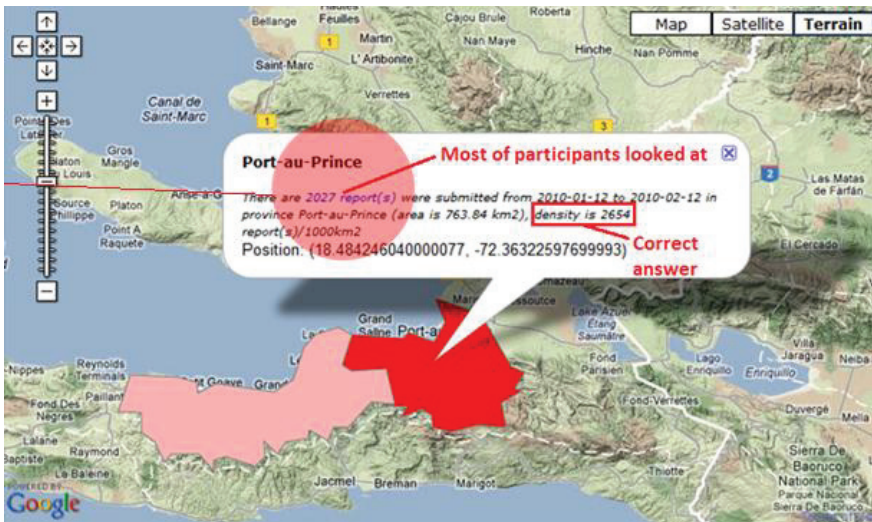


Figure 6.2: Screen shot of one participant who gave wrong answer in question 2 of task 2

#### e. Video recordings and screen logging data

As mentioned before, if the data of one method could not be used by unexpected problem, we can use the other methods to get some additional information and other information that are needed for usability test. Given some problems with the eye tracking system, this proved to be useful as follows. With the overview task, I expected that the participants would use the animation function or go to the date function to find the answers. However, from the screen logging it became clear that some participants did not use animation functions; they only used the “go to date” function or tried to drag the timeline to the date they wanted. The reason for this issue might be because I asked the question to compare the two days (13/01/2010 and 14/01/2010) that near the earliest day of the data (12/01/2010), hence it is easy for them to drag the timeline. The participants who used the animation were not familiar with the function of change steps and speed of animation. Thus, some of them thought that the timeline did not move. In fact, it was moving at steps of one minute per 2 seconds which is very small in comparison with the range of more than 14 hours on the timeline.

Next, in question 2 of the task zooming and filtering, some participants who did not read the instruction carefully had to try to do this question at least 3 times. This was because after the filtering function was

run for this question, a map showing department which contain the two provinces Léogâne and Port-au-Prince appeared instead of the map showing two provinces. That made them confused and thought that they used the wrong map.

By watching the screen logging, the reason why a participant gave a wrong answer to question 2 of task 1 and one other test person could not finish this question might be that they did not use the middle band to see the overview. According to the position of the mouse pointer, the position of the two timeline's bands the thinking aloud recordings, it seems that they consider that the top band is the same band as the middle band. One used the number of reports sent within some hours of one day to compare with the other period in the same day. The other felt too difficult to compare between the two days; this participant then got stuck on selecting other questions to solve this question.

#### **f. Think aloud data**

During the test, participants were asked to speak loudly what they were thinking. However, by some reasons given earlier, some of the participants did not talk loudly during the test; instead of that, they wrote on the paper of tasks their answers and also gave some comments by writing. For the participants who did the work of thinking aloud well, the data is significant for revealing the problems they met and reasons why they answered the questions wrong.

From the recording, some participants said that it was hard to read the list of geographic questions to do the tasks and they did not know that they should click on the questions to make the maps appear. A participant said that she thought the topband and middle band show the same data, this was the reason why she answered the question 2 in the overview task incorrectly. Two participants said that they expected after they submitted a new report in question 1 of task 4, there should be a message that informs them they submitted successfully.

Besides that, during the test, the participants gave advices to improve the design and functionalities of the prototype. All of them said that the list of geographic question should be redesigned. Two of them said that the departments and provinces should have labels. A participant suggested that the time format in the prototype needs to be united.

#### **g. Interview data**

At the end of each test, a short interview was conducted. From the voice recorded in the interviews, I found that there are a lot of problems in with the design of the GUI of the prototype. Again, all of the participants recommended that the interface of the list of questions needs to be changed. Some of the participants complained about the mouse pointer's icon when the mouse pointer was moving on the clickable areas. They also recommended that the legends need to be redesigned and that the information window which shows the pie chart needs to be improved. One more suggestion was to add some confirmative information after users finished updating a report, their detail or after they submitted a new report to let them know whether the activities are finished or not.

### **6.4. Summary about the usability of the prototype**

After the usability test, it was proved that the prototype is helpful for the exploration of neogeographic point data. The prototype provides the tools that allow users to explore the original point data by showing their information such as location, description, categories, and department/province they are located in. Furthermore, the prototype allows users to estimate the amounts and quantify spatial anomalies which are helpful for the authorities (a United Nation staff for example) to know where the victims need them most, where the most emergency situation located, etc... The prototype also can help the users to discover correlations/conflicts that is helpful for the work of analysis data. The functions of data and user management will be very helpful if the prototype is applied into the other dataset and/or the other

purposes. From the answers of question about satisfaction of the test participants with the prototype, the prototype is rated as good and helpful for the users.

### **6.5. Limitations of the test**

After finishing the usability test, I realized some limitations of the test procedure as well as limitations of the eye tracking system.

Firstly, during the first test, my laptop was disconnected from the internet for a while and that made the test had stopped for a while. After the laptop was reconnected to the internet, the prototype worked well as usual. Next, the Tobii 3.0.1 does not support using Firefox as the web browser. Hence, the eye tracking data outputs for path/gaze plot and hot spot/heat map are unable to be generated directly. The purpose of the test was not testing how fast it is to get familiar with the prototype. Since the prototype has some functions that needed time to get familiar with, it could have been better if participants would have been given more time to become familiar with the prototype before the real test.

Last but not least, the instructions for the tasks are not clear enough for the test participants. If the instructions are separated into smaller steps, the participants could follow the steps easier and they might do the tasks faster than usual.

### **6.6. Limitations of the prototype**

The first limitation is related to the design of the list of geographic questions. The distances between the questions are quite narrow; there are also too many questions shown. Hence, during the usability test, it was hard for the users to read all of the questions.

The second limitation is related to the number of classes that were used in classification of the data. I used 9 classes to classify the total number of reports and the density of reports in different areas. Hence, the colours of classes are hard to be differentiated.

Thirdly, according to the advices from the participants in the usability test, the design needs to be improved in some parts of the map. When they used the filtering function for more than one category, the design of the pie chart for showing the number of reports in each category made them confused because the total number of reports for all of the selected categories is greater than the number of reports.

Next, there is a lack of confirmative information. Users were confused because no message was shown to them after they had done an activity.

Another limitation is related to the speed of the website: the speed of the timeline is faster than that of the map. If we use the animation function in some maps, the data on the map were loaded after a few seconds, when the timeline is moving.

Finally, the prototype was developed and tested only on two web browsers, Mozilla Firefox and Opera. Hence, the filtering functions might not work in other web browsers, because some JavaScript functions that were used to select the requirements for the filtering function cannot work on the other web browsers.

### **6.7. Recommendation for the improvement of the prototype**

After the usability test, there are some problem of the prototype that can be solved by improving some functionalities and design of the prototype. Firstly, the list of geographic questions can be replaced by a multiple choices form. For different tasks, the users just need to know what they require from the map, and choose what they need from the form and then the map that provides those requirements will appear. For example, when a user needs a map that can show the density of reports in a specific time period in different areas (departments and provinces), they can select the map with three options: show density,



show specific time period and show by departments or provinces. This work can be done by using JavaScript with a technique similar to the technique used for the list of geographic questions.

Secondly, the problem with the colours of classes in the maps can be solved by using a smaller number of classes or a wider range of colours. One other suggestion is combining PHP and JavaScript on the Timemap API. I found that we can move the JavaScript codes that used to create themes in the map from the timemap.js file to the PHP file in which the map is shown. Hence, with the help of PHP, we can provide the users the ability to select colours for the legends and the number of classes that they want to use in the classification of the data.

Next, to improve the problem of the lack of confirmative information, each time the users submit a report or update their information, we can use some messages to inform users that they have done these activities successfully.

For the problem of the web browsers' compatibility, we can use some replacements for the functions that are not supported in the other web browsers and use JavaScript navigator object [URL 44] to recognize which web browser is used and apply the correct codes for different web browsers.

Furthermore, the timeline can be improved by giving the option for users to change the time interval of the top band of the timeline in the point symbols map. In some cases, users may want to see reports that were sent in a longer time period or they want to display all the reports sent in a very short period. PHP can be used together with JavaScript to meet this requirement.

The search function can be improved by providing the ability to search the reports in a shorter time period such as the ability to search the reports that were sent within a few minutes instead of within a few days now. Using the "AND" operator to search on the reports' titles and descriptions can be improve, only the "OR" operator is used in the search function of the prototype now. That will help the users to search more easily all what they need.

## **6.8. Summary**

In this chapter, the usability testing of the prototype was described. Usability testing was conducted using think aloud, video observation, screen logging, interviews and eye tracking with the help of Tobii 3.0.1 software to record and analyse the data. The assessment of the effectiveness and the efficiency of the prototype as well as the user's satisfaction were mentioned in this chapter. Furthermore, from eye tracking data, audio/video recorded, screen logging and interviews, some limitations of the prototype were revealed such as problem with the design of the list of geographic questions or the labels of areas on the map need to be developed. Finally, some summary and recommendation for the improvement of the prototype were mentioned. The next chapter will present the conclusions, some limitations of the prototype as well as the recommendations for further research.

## 7. CONCLUSION AND RECOMMENDATIONS

### 7.1. Conclusion

The main objective of this research was to develop a prototype to explore neogeographic point data on top of already existing and freely available base maps, by using web visualization libraries and web programming and by applying cartographic design rules to meet various users' purposes. The main objective can be achieved by answering all the research questions for the five sub-objectives presented in Chapter 1. Before answering these questions, we have reviewed the terminologies that surround the concept of neogeography and cartographic visualization through literatures to have a general view of neogeography, neogeographic data, and their sources, how they are analysed and visualized by using cartographic visualization and neogeography maps in chapter 2.

Thereafter, the research questions per sub-objective were answered as follows:

1. To provide an overview of the different uses of neogeographic point data maps by formulating the geographic questions users may want to have answers to.
  - What are the questions users want to have answers to?

For different purposes of exploration of neogeographic data, the users need to have answers to various numbers of geographic questions. The questions users want to have answers to were mentioned in chapter 3 as a list of geographic questions in Table 3.2.

- What are the tasks related to these questions?

The required tasks which were created based on the Visual Information Seeking Mantra proposed by SHNEIDERMAN (1996) and the questions users want to have answer to are shown in Table 3.3.

2. To link these questions to specific cartographic visualization solutions.
  - What cartographic visualization solutions can be used to answer these questions?

Cartographic design rules result in various visualization solutions (map types) that can answer these questions. It was investigated which visualization libraries could provide these solutions.

3. To gain an overview of relevant existing open source web visualization libraries for visualizing neogeographic data
  - Which web visualization libraries already exist and what are their characteristics?

Many web visualization libraries that can be used for cartographic visualization are available on the internet. However, since time was limited, we had to focus on a subset, which consisted of seven web visualization libraries, namely Google Maps JavaScript API, OpenLayers API, SIMILE, Timemap API, GeoCommons API, Thematic maps API and Google Chart API. They were investigated one by one in chapter 4 to see their characteristics, advantages as well as disadvantages.

- What are the criteria for choosing a suitable web visualization library for the purpose of this research?

Based on the required views and functions discussed in chapter 3, some criteria to choose the suitable web visualization libraries to implement into the prototype were described in chapter 4. Then, these criteria which are shown in Table 4.2 were used to select the categories to use in the thesis.

A prototype for the exploration of neogeographic point data with cartographic visualization tools

- Which web visualization library is most suitable for the purpose of this research?

After investigating the seven web visualization libraries, I decided to use some of the seven web visualization libraries together: they are Timemap API, SIMILE, Google Maps API and Google Chart API. Hence, problems due to a lack of functionalities when we use only one web visualization library can be solved.

4. To design a Graphical User Interface considering the required views and functions for visualization of neogeographic data
- Which ways of cartographic visualization are required?

The required ways of cartographic visualization were reviewed in section 2.5, in which I have chosen the way to use cartographic visualization in my thesis.

- Which functionalities are required?

The required functionalities were listed based on the required views and tasks. Those functionalities were mentioned in Table 3.3 in chapter 3. Based on those required functions, different web visualization libraries which support those functions will be used in the prototype.

- How to design and implement a research prototype as a prototype, taking user tasks and data characteristics into account?

The process how to design and implement the prototype was described in chapter 5 step by step. Firstly, the set-up of the data base were described. Then, PHP and PostgreSQL were mentioned as the server scripts and database server for the implementation of the prototype. After that, the required views and functions mentioned in chapter 3 were implemented into the prototype. Finally, some outputs of the prototype were described.

5. To evaluate the prototype
- How to evaluate the prototype?

A usability testing was conducted in chapter 6 by using a combination of different methods: think aloud, screen logging, video observation, interviews and eye tracking.

- What are the outcomes of the evaluation of the prototype?

The test described the assessment of the prototype from the view of its effectiveness and satisfaction perspective. Furthermore, from eyetracking data and video observation data, some limitations of the prototype were revealed such as problem with function design or the issue of using labels in the map need to be developed. From the usability test, some of limitations of the prototype were found. They were mentioned, and then be expressed to find the causes of the limitations in section 6.6.

## **7.2. Recommendations for further research**

The main research problems were tackled. However, some features of the prototype need to be developed. Some other suggestions are:

- Use overlay to show labels for departments and provinces in the country. It helps the users to know more easily which department or province they are looking at.
- Change the icon of the mouse pointer when the users move it to clickable areas on the website.
- Further testing for the usability is necessary to fully evaluate the functionalities of the system. In the usability testing section of this research, I gave some tasks for users to find answers to. To answer those questions, users did not have to use all the provided functions. Hence, further usability testing is recommended.

- Last but not least, this research is applied to the case study of the Haiti earthquake of 2010. The earthquake has ended but we can use this prototype to apply for other purposes such as collecting transportation reports about accidents and transportation problems, which is necessary and useful in the case of Vietnam. Moreover, if another natural hazard occurs, we can easily apply this prototype to that area by updating the boundary data and change the file dconn.php to redirect the data connection to the new database.

## LIST OF REFERENCES

---

1. BAILEY, T., & GATRELL, A. (1995). *Interactive spatial data analysis*. Harlow: Longman Higher.
2. BROOK, J., CARLSSON, C., & PETERS, N. J. (2001). *Reclaiming San Francisco: History, Politics, Culture (A City Lights Anthology)* (3rd edition ed.): City Lights Publishers.
3. COOKE, L. (2005). Eye Tracking: How it works and how it relates to usability Vol. 52.(pp. 456-463). Retrieved from [http://www.wcupa.edu/academics/sch\\_cas.eng/faculty/documents/TC\\_eyetracking\\_art.pdf](http://www.wcupa.edu/academics/sch_cas.eng/faculty/documents/TC_eyetracking_art.pdf)
4. COOTE, A., & RACKHAM, L. (2008). *Neogeographic data quality – is it an issue?* Paper presented at the AGI Conference 2008, UK. [http://www.consultingwhere.com/resources/Neogeography+Data+Quality+-+is+it+an+issue+-+V1\\_1.pdf](http://www.consultingwhere.com/resources/Neogeography+Data+Quality+-+is+it+an+issue+-+V1_1.pdf)
5. CRAMPTON, J. W. (2001). Maps as social constructions: Power, communication, and visualization. *Progress in Human Geography* 25, 235-52.
6. D'LOG. (2008). A short enquiry into the origins and uses of the term “neogeography” Retrieved 10th, November 2011, from <http://www.d-log.info/on-neogeography.pdf>
7. DAGOGNET, F. (1977). Une epistemologie de l'espace concret: neo-geographie (Problemes et controverses): Librairie Philosophique J. Vrin.
8. DAS, T., & KRAAK, M. J. (2011a). *Does Neogeography need designed maps?* Paper presented at the 25th International Cartographic Conference, Paris.
9. DAS, T., & KRAAK, M. J. (2011b). *Geovisualization in Neogeography?* Paper presented at the Linking Geovisualization with Spatial Analysis and Modeling, Hamburg, Germany.
10. DELIKOSTIDIS, I. (2007). *Methods and techniques for field - based usability testing of mobile geo - applications*. ITC, Enschede. Retrieved from [http://www.itc.nl/library/papers\\_2007/msc/gfm/delikostidis.pdf](http://www.itc.nl/library/papers_2007/msc/gfm/delikostidis.pdf)
11. DORLING, D., & FAIRBAIRN, D. (1997). *Mapping : ways of representing the world*. S.l.: Addison Wesley Longman.
12. DUC, H. A., & MOSEME, M. (2011). *Using timeline map to inform the public about Peacekeeping missions and conflicts Case of Sudan*. Module report, University of Twente, ITC.
13. EERI. (2010a). The Mw 7.0 Haiti Earthquake of January 12, 2010: Report #1 Retrieved 5th September 2011, 2011, from [http://www.eeri.org/site/images/eeri\\_newsletter/2010\\_pdf/Haiti\\_Rpt\\_1.pdf](http://www.eeri.org/site/images/eeri_newsletter/2010_pdf/Haiti_Rpt_1.pdf)
14. EERI. (2010b). The Mw 7.0 Haiti Earthquake of January 12, 2010: Report #2 Retrieved 5th September 2011, 2011, from [http://www.eeri.org/site/images/eeri\\_newsletter/2010\\_pdf/Haiti\\_Rpt\\_2.pdf](http://www.eeri.org/site/images/eeri_newsletter/2010_pdf/Haiti_Rpt_2.pdf)
15. GIBIN, M., SINGLETON, A., MILTON, R., MATEOS, P., & LONGLEY, P. (2008). An Exploratory Cartographic Visualisation of London through the Google Maps API. *Applied Spatial Analysis and Policy*, 1(2), 85-97. doi: 10.1007/s12061-008-9005-5
16. GOODCHILD, M. (2007). Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69(4), 211-221. doi: 10.1007/s10708-007-9111-y
17. HUDSON-SMITH, A., CROOKS, A., GIBIN, M., MILTON, R., & BATTY, M. (2009). NeoGeography and Web 2.0: concepts, tools and applications. *Journal of Location Based Services*, 3(2), 118-145. doi: 10.1080/17489720902950366
18. HUISMAN, O., & DE BY, R. A. (2009). *Principles of geographic information systems : an introductory textbook* (Fourth edition ed. Vol. 1). Enschede: ITC.



19. KRAAK, M. J., & BROWN, A. (2001). *Web cartography: developments and prospects*: Taylor & Francis.
20. KRAAK, M. J., & ORMELING, F. (2009). *Cartography: visualization of geospatial data*. GB: Pearson Education.
21. MACEACHREN, A. M. (1995). *How maps work : representation, visualization and design*. New York etc.: The Guildford Press.
22. MLAY, J. (2010). *Map design in a neogeography environment*. University of Twente Faculty of Geo-Information and Earth Observation ITC, Enschede.
23. PEUQUET, D. J. (1994). It's about time : a conceptual framework for the representation of temporal dynamics in geographic information systems. *In: Annals of the Association of American Geographers*, 84(1994)3, pp. 441-461.
24. POOLE, A., & J.BALL, L. (2004). Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future Prospects Retrieved 29th, June, 2011, from <http://www.alexpoole.info/blog/wp-content/uploads/2010/02/PooleBall-EyeTracking.pdf>
25. ROBINSON, A. H., SALE, R. D., & MORRISON, J. L. (1995). *Elements of cartography* (Sixth ed.). New York etc.: Wiley & Sons.
26. SANDVIK, B. (2008). *Using KML for Thematic Mapping*. Msc, University of Edinburgh. Retrieved from [http://thematicmapping.org/downloads/Using\\_KML\\_for\\_Thematic\\_Mapping.pdf](http://thematicmapping.org/downloads/Using_KML_for_Thematic_Mapping.pdf)
27. SHNEIDERMAN, B. (1996). eyes have it : a task by data type taxonomy for information visualizations. *In: 1996 IEEE Symposium on Visual Languages, September 03-06, 1996, Boulder, Colorado*, pp. 336-343.
28. SLOCUM, T. A., MCMASTER, R. B., KESSLER, F. C., & HOWARD, H. H. (2009). *Thematic cartography and geovisualization* (Third edition ed.). Upper Saddle River: Prentice Hall.
29. TURNER, A. J. (2006). Introduction to Neogeography. Retrieved from <http://brainoff.com/iac2009/IntroductionToNeogeography.pdf>
30. VAN ELZAKKER, C. P. J. M., ORMELING, F. J., & KRAAK, M. J. (2004). *use of maps in the exploration of geographic data*. 116, University of Utrecht, Utrecht. Retrieved from [http://www.itc.nl/library/Papers\\_2004/phd/vanelzakker.pdf](http://www.itc.nl/library/Papers_2004/phd/vanelzakker.pdf)

## LIST OF URL'S

---

1. Map of newspaper food columns in the United States; Access date: 15th July 2011; Available from: <http://www.allmyfeatures.com/amf/foodmap/googlefoodmap.htm>
2. Google Maps; Access date: 17th, August 2011; Available from: <http://maps.google.com/>
3. Google MAP API. 2009; Access date: 17th, August 2011; Available from: <http://code.google.com/apis/maps/>
4. OPENLAYERS; Access date: 17th, August 2011; Available from: <http://www.openlayers.org/>
5. SIMILE; Access date: 6th, June 2011; Available from: <http://code.google.com/p/simile-widgets/>
6. TIMEMAP; Access date: 6th, June 2011; Available from: <http://code.google.com/p/timemap/>
7. ESRI Javascript API; Access date: 15th, July 2011; Available from: <http://help.arcgis.com/en/webapi/javascript/arcgis/>
8. GeoCommons API; Access date: 17th, August 2011; Available from: [http://geocommons.com/help/javascript\\_API/](http://geocommons.com/help/javascript_API/)
9. Website of NAVTEQ, the Leading Global Provider of Digital Map Data; Access date: 17th, August 2011; Available from: <http://www.navteq.com>
10. Real Madrid FC's fans map; Access date: 22th, August 2011; Available from: <http://geocommons.com/maps/96236>
11. Google markermanager; Access date: 29<sup>th</sup>, September 2011; Available from: <http://google-maps-utility-library-v3.googlecode.com/svn/trunk/markermanager/>
12. National weather service 2011; Access date: 29<sup>th</sup>, September 2011; Available from: <http://radar.srh.noaa.gov/>
13. GeoIQ website; Access date: 20<sup>th</sup>, September 2011; Available from: <http://www.geoiq.com/>
14. Map of Shopko Hometown, Walmart, and Target locations; Access date: 29<sup>th</sup>, September 2011; Available from: <http://geocommons.com/maps/93857>
15. haiti.ushahidi.com. (2010). Crisis Map of Haiti Retrieved 4th September, 2011, from <http://haiti.ushahidi.com/>
16. WikiMapia website; Access date: 10<sup>th</sup>, October 2011; Available from: <http://wikimapia.org>
17. Flickr map website; Access date: 10<sup>th</sup>, October 2011; Available from: <http://www.flickr.com/map/>
18. Google MyMaps; Access date: 10<sup>th</sup>, October 2011; Available from: <http://maps.google.com/maps/mm>
19. OpenStreetMap website; Access date: 11<sup>th</sup>, October 2011; Available from: <http://www.openstreetmap.org/>
20. Microsoft Bing Maps website; Access date: 11<sup>th</sup>, October 2011; Available from: <http://www.bing.com/maps/>
21. Map my tracks website; Access date: 17<sup>th</sup>, November 2011; Available from: <http://www.mapmytracks.com/>
22. Function of Client and Server environment. [cited 2011 25/11]; Available from: [http://www.computerfreetips.com/Database\\_tips/sql\\_server/Administration/sql\\_server\\_data.html](http://www.computerfreetips.com/Database_tips/sql_server/Administration/sql_server_data.html)
23. Conceptual model definition. 2011 [cited 2011 09/12]; Available from: <http://dictionary.reference.com/browse/conceptual+model>
24. Thematic Mapping website; Access date: 20<sup>th</sup>, October 2011; Available from: <http://thematicmapping.org/api/>
25. Infant mortality rate (0-1 year) of the world, year 2005; Access date: 23<sup>th</sup>, October 2011; Available from: <http://thematicmapping.org/ext-js-google-earth-api/example.html>
26. PostgreSQL website; Accessdate: 30<sup>th</sup>, November 2011; Available from: <http://www.postgresql.org/>
27. Haiti boundaries map; Accessdate: 12<sup>th</sup>, December 2011; Available from: [http://cegrp.cga.harvard.edu/files/haiti/haiti\\_admin\\_boundaries.zip](http://cegrp.cga.harvard.edu/files/haiti/haiti_admin_boundaries.zip)

28. London Profiler Website; Access date: 20<sup>th</sup>, September 2011; Available from:  
<http://www.londonprofiler.org/>
29. Hankhonio's tracks map on Map My Tracks; Access date: 25<sup>th</sup>, December 2011; Available from:  
<http://www.mapmytracks.com/explore/activity/359178>
30. Map of America's food newspapers, designed by Tanmoy Das; Access date: 14<sup>th</sup>, August 2011; Available from: <http://geonetwork.itc.nl/zanzibar/index3.html>
31. Haiti earthquake in 2010; Access date: 14<sup>th</sup>, August 2011; Available from:  
[http://en.wikipedia.org/wiki/Haiti\\_earthquake\\_2010](http://en.wikipedia.org/wiki/Haiti_earthquake_2010)
32. An example of a riding bike's track in gpx format; Access date: 1<sup>st</sup>, January 2012; Available from:  
<http://www.mountainbikebill.com/images/Trails/AZ08/SouthMountainTelegraphUpperNational.gpx>
33. Nestoria search engine for finding UK property; Access date: 20<sup>th</sup>, September 2011; Available from:  
<http://www.nestoria.co.uk/>
34. PHP official website; Access date: 1<sup>st</sup>, January 2012; Available from: [www.php.net/](http://www.php.net/)
35. Apache Software Foundation website; Access date: 1<sup>st</sup>, January 2012; Available from:  
[www.apache.org/](http://www.apache.org/)
36. Web widget for visualizing Temporal Data; Access date: 1<sup>st</sup>, January 2012; Available from:  
<http://www.simile-widgets.org/timeline/>
37. Map of United State's food newspapers; Access date: 14<sup>th</sup>, August 2011; Available from:  
<http://www.allmyfeatures.com/amf/foodmap/googlefoodmap.htm>
38. San Diego Wildfires 2007 interactive map; Access date: 12<sup>th</sup>, January 2012; Available from:  
<http://www.signonsandiego.com/firemap/>
39. Google ChartTools/Google Chart API; Access date: 14<sup>th</sup>, January 2012; Available from:  
<http://code.google.com/apis/chart/>
40. Google Chart Tools, visualization by using Pie Chart; Access date: 14<sup>th</sup>, January 2012; Available from:  
<http://code.google.com/apis/chart/interactive/docs/gallery/piechart.html>
41. Slider (WebFX); Access date: 1<sup>st</sup>, January 2012; Available from:  
<http://webfx.eae.net/dhtml/slider/slider.html>
42. Simple drop down menu; Access date: 2<sup>nd</sup>, January 2012; Available from:  
[http://javascript-array.com/scripts/simple\\_drop\\_down\\_menu/](http://javascript-array.com/scripts/simple_drop_down_menu/)
43. Definition of usability; Access date 29<sup>th</sup>, January 2012; Available from:  
[http://www.usabilitynet.org/management/b\\_what.htm](http://www.usabilitynet.org/management/b_what.htm)
44. Navigator object on W3schools.com; Access date 8<sup>th</sup>, February 2012; Available from:  
[http://www.w3schools.com/jsref/obj\\_navigator.asp](http://www.w3schools.com/jsref/obj_navigator.asp)
45. Definition of resolution; Access date 8<sup>th</sup>, February 2012; Available from:  
<http://ezinearticles.com/?Definition-of-Resolution&id=163162>
46. Random weather map, using Google Maps API and mapmarker.js; Access date 8<sup>th</sup>, February 2012; Available from:  
[http://google-maps-utility-library-v3.googlecode.com/svn/tags/markermanager/1.0/examples/weather\\_map.html](http://google-maps-utility-library-v3.googlecode.com/svn/tags/markermanager/1.0/examples/weather_map.html)
47. BSD License definition; Access date 10<sup>th</sup>, February 2012; Available from:  
<http://www.linfo.org/bsdlicense.html>
48. Ray-casting Algorithm; Access date 1<sup>st</sup>, January 2012; Available from:  
[http://rosettacode.org/wiki/Ray-casting\\_algorithm](http://rosettacode.org/wiki/Ray-casting_algorithm)
49. Tobii X60 eye-tracker website; Access date 10<sup>th</sup>, February 2012; Available from:  
<http://www.tobii.com/en/eye-tracking-research/global/products/hardware/tobii-x60x120-eye-tracker/>
50. Tobii Study eye-tracking software; Access date 10<sup>th</sup>, February 2010; Available from:  
<http://www.tobii.com/en/eye-tracking-research/global/products/software/tobii-studio-analysis-software/>

## APPENDICES

### APPENDIX A: Scripts and codes

Appendix A.1 PostgreSQL script that import data from .CSV file to the PostgreSQL database (for reports' data)

```
CREATE table haiti."reports" (ID bigint, INCIDENT_TITLE varchar, INCIDENT_DATE timestamp
without time zone, LOCATION varchar, DESCRIPTION varchar, CATEGORY varchar, LATITUDE double
precision, LONGITUDE double precision, APPROVED boolean, VERIFIED boolean);
copy haiti."Reports" from 'C:/haiti_reports.CSV' DELIMITERS ',' CSV;
```

Appendix A.2 PostgreSQL script that import data from .CSV file to the PostgreSQL database (for departments' data)

```
create table haiti."departments" (deptID integer, deptname varchar, boundary1 varchar,
boundary2 varchar, boundary3 varchar, area double precision);
copy haiti."departments" from 'C:/Data/haiti_border_departments.CSV' DELIMITERS ',' CSV;
```

Appendix A.3 PostgreSQL script that import data from .CSV file to the PostgreSQL database (for provinces' data)

```
create table haiti."provinces" (provID integer, deptname varchar, provname varchar, boundary1
varchar, boundary2 varchar, boundary3 varchar, area double precision);
copy haiti."province" from 'C:/Data/haiti_border_provinces.CSV' DELIMITERS ',' CSV;
```

Appendix A.4 PostgreSQL script that import data from .CSV file to the PostgreSQL database (for grids' data)

```
create table haiti."provinces" (gridID integer, center varchar, border varchar);
copy haiti."province" from 'C:/Data/haiti_border_provinces.CSV' DELIMITERS ',' CSV;
```

Appendix A.5 Python source code to convert .KML file format to .CSV file format (for data of departments)

```
#Import required modules
import string
import sys
#define default files
inputf = "doc.kml"
output = "haiti_border_departments.CSV"
#define global variables
NAME = ""
BORDER = ""
#Main function
def kml2csv(inputf1,output):
# get polygon for Haiti
    found = 0
    start = 0
    try:
        fin = open(inputf)
        text = ""
    except IOError:
        print "Could not open file" + inputf
    else:
        for line in fin.readlines():
            if line.find("<Placemark") > -1 :
                start = start + 1
            if start > 0:
                #input name of department
                nstart = line.find("<name>")
                nend = line.find("</name>")
                if nstart > -1:
```

```

        pcount = 0
        boundary = ["", "", ""]
        NAME = line[(nstart+6):(nend)]
    #input area
    areastart = line.find("<td>F_AREA</td>")
    if areastart>-1:
        counttd = 0
        found = 1
    if found == 1:
        areastartcpy = line.find("<td>")
        areaendcpy = line.find("</td>")
        if areastartcpy >-1:
            counttd = counttd + 1
            if ((areastartcpy >-1) and (counttd == 2)):
                areaid = line[(areastartcpy+4):areaendcpy]
                found = 0
    pstart = line.find("<outerBoundaryIs><LinearRing><coordinates>")
    pend = line.find("</coordinates></LinearRing></outerBoundaryIs>")
    if pstart >-1:
        BORDER = line[(pstart + 42):pend]
        BORDER = BORDER.replace(", ", " ")
        boundary[pcount] = BORDER
        pcount = pcount + 1
    if line.find("</Placemark>") > -1 :
        text = text + NAME
        for i in range(0,3):
            text = text + "," + boundary[i]
        text = text + "," + areaid + "\n"

    fin.close()
    #define output file
    fout = open(output,"w")
    fout.write(text)
    fout.close()
kml2csv(inputf,output)

```

#### Appendix A.6 Python source code to convert .KML file format to .CSV file format (for data of province)

```

#Import required modules
import string
import sys
#define default files
inputf = "doc.kml"
output = "haiti_border_province.CSV"
#define global variables
NAME = ""
BORDER = ""
#Main function
def kml2csv(inputf1,output):
    # get polygon for Haiti
    found = 0
    start = 0
    try:
        fin = open(inputf)
        text = ""
    except IOError:
        print "Could not open file" + inputf
    else:
        for line in fin.readlines():
            if line.find("<Placemark>") > -1 :
                start = start + 1
            if start > 0:
                #input name of department
                nstart = line.find("<name>")

```



```

nend = line.find("</name>")
if nstart >-1:
    pcount = 0
    boundary = ["", "", ""]
    NAME = line[(nstart+6):(nend)]
#input name of province
adm2start = line.find("<td>ADM2</td>")
if adm2start >-1:
    counttd = 0
    found = 1
    if found == 1:
        adm2startcpy = line.find("<td>")
        adm2endcpy = line.find("</td>")
        if adm2startcpy >-1:
            counttd = counttd + 1
            if ((adm2startcpy >-1) and (counttd == 2)):
                adm2id = line[(adm2startcpy+4):adm2endcpy]
                found = 0
#input area
areastart = line.find("<td>F_AREA</td>")
if areastart>-1:
    counttd = 0
    found = 1
    if found == 1:
        areastartcpy = line.find("<td>")
        areaendcpy = line.find("</td>")
        if areastartcpy >-1:
            counttd = counttd + 1
            if ((areastartcpy >-1) and (counttd == 2)):
                areaid = line[(areastartcpy+4):areaendcpy]
                found = 0
        pstart = line.find("<outerBoundaryIs><LinearRing><coordinates>")
        pend = line.find("</coordinates></LinearRing></outerBoundaryIs>")
        if pstart >-1:
            BORDER = line[(pstart + 42):pend]
            BORDER = BORDER.replace(", ", " ")
            boundary[pcount] = BORDER
            pcount = pcount + 1
        if line.find("</Placemark>") > -1 :
            text = text + NAME + "," + adm2id
            for i in range(0,3):
                text = text + "," + boundary[i]
            text = text + "\n"
    fin.close()
#define output file
fout = open(output,"w")
fout.write(text)
fout.close()
kml2csv(inputf,output)

```

#### Appendix A.7 Python source code to convert .KML file format to .CSV file format (for data of grids)

```

#Import required modules
import string
import sys
#define default files
inputf = "grid_haiti.kml"
output = "grid_haiti.CSV"
#define global variables
NAME = ""
BORDER = ""
#Main function
def kml2csv(inputf1,output):

```

```
# get polygon for Haiti
found = 0
start = 0
try:
    fin = open(inputf)
    text = ""
except IOError:
    print "Could not open file" + inputf
else:
    for line in fin.readlines():
        if line.find("<Placemark") > -1 :
            start = start + 1
        if start > 0:
            #input name of department
            nstart = line.find("<name>")
            nend = line.find("</name>")
            if nstart > -1:
                NAME = str(start)
            pstart = line.find("<outerBoundaryIs><LinearRing><coordinates>")
            pend = line.find("</coordinates></LinearRing></outerBoundaryIs>")
            if pstart > -1:
                BORDER = line[(pstart + 42):pend]
                BORDER = BORDER.replace(", ", " ")
                text = text + NAME + ",," + BORDER + "\n"
    fin.close()
    #define output file
    fout = open(output, "w")
    fout.write(text)
    fout.close()
kml2csv(inputf, output)
```

Appendix A.8 PHP script to check the coordinates of reports in side which polygons and input the ID of department, province and grid they are in into database.

```
<?php
include("dconn.php");
class pointLocation {
var $pointOnvt = true; // Check if the point sits exactly on one of the vertices
function pointInPolygon($point, $polygon, $pointOnvt = true) {
$this->pointOnvt = $pointOnvt;
// Transform string coordinates into arrays with x and y values
$point = $this->pointStringToCoordinates($point);
$vertices = array();
    foreach ($polygon as $vt) {$vertices[] = $this->pointStringToCoordinates($vt); }
// Check if the point sits exactly on a vt
if ($this->pointOnvt == true and $this->pointOnvt($point, $vertices) == true) {return 1;}
    $intersections = 0;
$vertices_count = count($vertices);
for ($i=1; $i < $vertices_count; $i++) {
$vt1 = $vertices[$i-1];
$vt2 = $vertices[$i];
//Check if the point is on the left of the vertex and the horizontal line from it to the right
side cut the vertex
if (($point['y'] >= min($vt1['y'], $vt2['y'])) and ($point['y'] <= max($vt1['y'], $vt2['y']))
and ($point['x'] <= max($vt1['x'], $vt2['x']))){ $intersections++;}
}
// If the number of edges we passed through is even, then it's in the polygon.
```

```

    if ($intersections % 2 != 0) {return 1;} else {return 0;}
}

function pointOnvt($point, $vertices) {
foreach($vertices as $vt) {
if ($point == $vt) {return true;}
}
}

function pointStringToCoordinates($pointString) {
$coordinates = explode(" ", $pointString);
return array("x" => $coordinates[0], "y" => $coordinates[1]);}
}

//Initialize new class pointLocation
$pointLocation = new pointLocation();
//Input points from database of reports
$result = pg_exec($dbconn, "select * from haiti.\"reports\"")
or die('Could not access to table: ' . pg_last_error());
$numrows = pg_numrows($result);
$result1 = pg_exec($dbconn, "select * from haiti.\"provinces\" INNER JOIN
haiti.\"departments\" ON haiti.\"provinces\".deptname = haiti.\"departments\".deptname")
or die('Could not access to table: ' . pg_last_error());
$numrows1 = pg_numrows($result1);
// Loop on rows in the result set of reports
for($ri = 0; $ri < $numrow; $ri++) {
$row = pg_fetch_array($result, $ri);
    $ID = $row[0]; $Lat = $row[6]; $Lon = $row[7]; $point = "$Lon $Lat";
for($di = 0; $di < $numrows1; $di++){
$row1 = pg_fetch_array($result1, $di);
    $provID = $row1[0]; $deptID = $row1[6];
    $Border1 = $row1[3]; $Border2 = $row1[4]; $Border3 = $row1[5];
    //Change format of border in the data to the format in the check function
$Border1 = str_replace(" 0 ", " ", $Border1);
$Border1 = str_replace(" 0", "", $Border1);
$Border1 = substr($Border1,1);
$Border2 = str_replace(" 0 ", " ", $Border2);
$Border2 = str_replace(" 0", "", $Border2);
$Border2 = substr($Border2,1);
$Border3 = str_replace(" 0 ", " ", $Border3);
$Border3 = str_replace(" 0", "", $Border3);
$Border3 = substr($Border3,1);
$array1 = explode(' ', $Border1);
$array2 = explode(' ', $Border2);
$array3 = explode(' ', $Border3);
if(($pointLocation->pointInPolygon($point, $array1)==1)||($pointLocation->pointInPolygon($point, $array2)==1)||($pointLocation->pointInPolygon($point, $array3)==1)){
$sql_update = "UPDATE haiti.\"reports\" SET \"deptID\" = '$deptID', \"provID\" = '$provID'
WHERE id = '$ID';";
$result2 = pg_query($dbconn, $sql_update)
    Or die("Error in SQL query: " . pg_last_error());}
}

```

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
break;
}
}
}
$result1 = pg_exec($dbconn, "select * from haiti.\"grids\"")
or die('Could not access to table: ' . pg_last_error());
$numrows1 = pg_numrows($result1);
// Loop on rows in the set of reports
for($ri = 0; $ri < $numrow; $ri++) {
$row = pg_fetch_array($result, $ri);$ID = $row[0];$Lat = $row[6];$Lon = $row[7];
for($gi = 0; $gi < $numrows1; $gi++){
$row1 = pg_fetch_array($result1, $gi);
$gridID = $row1[0];
$Border1 = $row1[2];
$Border1 = str_replace(" 0 ", " ", $Border1);
$Border1 = str_replace(" 0", "", $Border1);
$array1 = explode(' ', $Border1);
    //Set the box of grid by using 4 values:
$maxlat = -9999;$maxlon = -9999;$minlat = 9999;$minlon = 9999;
//Loop on rows in the set of grids' boundary's vertices
for ($pi=1; $pi<5; $pi++){
$array2 = explode(' ', $array1[$pi]);
if ($maxlat<$array2[1]){$maxlat = $array2[1];}
if ($maxlon < $array2[0]){$maxlon = $array2[0];}
if ($minlat > $array2[1]){$minlat = $array2[1];}
if ($minlon > $array2[0]){$minlon = $array2[0];}
}
if (($Lat>=$minlat and $Lat<=$maxlat) and ($Lon>=$minlon and $Lon<=$maxlon)){
$sql_update = "UPDATE haiti.\"reports\" SET \"gridID\" = '$gridID' WHERE id = '$ID'";
$result2 = pg_query($dbconn, $sql_update)
    or die("Error in SQL query: " . pg_last_error());
}
break;
}
}
}
pg_free_result($result);
pg_free_result($result1);
pg_close($dbconn);
?>
```

### Appendix A.9PHP script to export data from PostgreSQL server to KML files, for reports in points map

```
<?php
include("dconn.php");
date_default_timezone_set('Europe/Amsterdam');
$start = "";
$end = "";
if (isset($_GET['start'])){    $start = $_GET['start'];}
else{    $start = date('m/d/Y G:i:s', PHP_INT_MAX + 1);}
if (isset($_GET['end'])){    $end = $_GET['end'];}
else{    $end = date("Y-m-d");}
```

```

if (isset($_GET['dept'])) { $deptID = $_GET['dept']; }
else { $deptID = ""; }
if (isset($_GET['prov'])) { $provID = $_GET['prov']; }
else { $provID = ""; }
$sql = "SELECT DISTINCT ON (incident_date, incident_title, description) id, incident_title,
incident_date, description, longitude, latitude, \"provID\", \"deptID\", category FROM
haiti.\"reports\" WHERE incident_date < '$end' and incident_date > '$start' and approved =
'TRUE' ORDER BY incident_date, incident_title, description, id desc";
$result = pg_exec($dbconn, $sql)
or die('Could not access to table: ' . pg_last_error());
echo "<?xml version='1.0' encoding='UTF-8'?><kml><Document>";
while($row = pg_fetch_array($result)) {
$id = $row['id'];
$cat = $row['category']; $cat = str_replace(' ', '', $cat);
$cat = substr($cat, 0, strlen($cat)-1); $arr = explode(',', $cat);
for ($i=0; $i<sizeof($arr); $i++) { $arr[$i] = substr($arr[$i], 0, 1); }
$checkcat = "";
for ($i=0; $i<sizeof($arr); $i++) {
if (substr_count($checkcat, $arr[$i]) == 0) { $checkcat = $checkcat . $arr[$i] . ","; }
}
$array = explode(',', substr($checkcat, 0, strlen($checkcat)-1));
if (strlen($row['provID']) > 0) {
$sql = "SELECT provname FROM haiti.\"provinces\" WHERE \"provID\" = ".$row['provID'];
$result1 = pg_exec($dbconn, $sql)
or die('Could not access to table: ' . pg_last_error());
$row1 = pg_fetch_row($result1);
$provname = $row1[0];
} else { $provname = ""; }
if (strlen($row['deptID']) > 0) {
$sql = "SELECT deptname FROM haiti.\"departments\" WHERE \"deptID\" = ".$row['deptID'];
$result2 = pg_exec($dbconn, $sql)
or die('Could not access to table: ' . pg_last_error());
$row2 = pg_fetch_row($result2);
$deptname = $row2[0];
} else { $deptname = ""; }
for ($i=0; $i<sizeof($array); $i++) {
switch (substr($array[$i], 0, 1)) {
case
"1": echo
"<Placemark><Point><coordinates>".$row['longitude'].",".$row['latitude']. "</coordinates></Poin
t>
<name>".$row['incident_title']. "</name>
<TimeStamp><when>".$row['incident_date']. "</when></TimeStamp>
<description>&lt;b>Submitted at:</b> ".$row['incident_date']. " &lt;br>&lt;b>In
categories:</b>&lt;u>
".$row['category']. "&lt;/u>&lt;br>&lt;b>Description:</b>".substr($row['descript
ion'], 0, 200). " &lt;a href = &quot;reportdetail.php?id=".$row['id']. "&quot; target =
&quot;_blank&quot; &gt;More details..&lt;/a></description>
<LookAt><coordinates>".$row['longitude'].",".$row['latitude']. "</coordinates></LookAt>
<theme>";
if ($_GET['zoomlevel'] == 1) { echo "I1"; }
if ($_GET['zoomlevel'] == 2) { echo "I12"; }
if ($_GET['zoomlevel'] == 3) { echo "I13"; }
echo "</theme>
<tags> ".$row['category']. $provname. ", ".$deptname. ", </tags></Placemark>";
break;
//Codes for case 2 to case 8 are similar to case 1.
}
}
}
echo "</Document>\n</kml>";
pg_free_result($result);
pg_close($dbconn);
?>

```



Appendix A.10 PHP script to export data from PostgreSQL server to KML files, for departments in choropleth map, structure for provinces is similar.

```
<?php
include("dconn.php");
date_default_timezone_set('Europe/Amsterdam');
$start = "";
$end = "";
if ($_GET['start'] != ""){$start = $_GET['start'];}
else{$start = date("Y-m-d h:i:s",strtotime("2010-01-12 00:00:00"));}
if ($_GET['end'] != ""){$end = $_GET['end'];}
else{$end = date("Y-m-d h:i:s",strtotime("now"));}
if(isset($_GET['showpast'])){ $showpast = $_GET['showpast'];
}else{$showpast = 1;}
if(isset($_GET['cat'])){
    $cat = $_GET['cat']; $cat = str_replace("'", "&quot;", $cat); $cat0 = $_GET['cat'];
}else{ $cat = ""; $cat0=""; }
if(isset($_GET['deptid'])){ $deptid = $_GET['deptid']; $deptid0 = $_GET['deptid'];
}else{ $deptid = ""; $deptid0 = "";}
if(isset($_GET['provid'])){ $provid = $_GET['provid']; $provid0 = $_GET['provid'];
}else{ $provid = ""; $provid0 = "";}
if(isset($_GET['gridid'])){ $gridid = $_GET['gridid'];
}else{ $gridid = "";}
if(isset($_GET['dens'])){ $dens = $_GET['dens'];
}else{ $dens = "";}
$sql="";//start sql string
$sqlstart = "SELECT DISTINCT haiti.\"departments\".deptid as
deptid,haiti.\"departments\".deptname as deptname, haiti.\"departments\".boundary1 as
boundary1, haiti.\"departments\".boundary2 as boundary2,haiti.\"departments\".boundary3 as
boundary3,haiti.\"departments\".area as area FROM haiti.\"provinces\" INNER JOIN
haiti.\"departments\" ON haiti.\"provinces\".deptname = haiti.\"departments\".deptname ";
$sql1 = "";
if($provid!=""){//For provinces query
    $where = "";
    $provid = substr($provid,0,strlen($provid)-1);
    $array = explode(',', $provid);
    for($i=0;$i<sizeof($array);$i++){
        if($i==0){ $where = $where." provid=\".$array[$i].\" ";
        }else{ $where = $where." OR provid=\".$array[$i].\" "; }
    }
    $sql1 = $sql1." ". $where;
}else{ $sql1 = "";}
$sql2 = "";
if($deptid!=""){//For departments query
    $where = "";
    $deptid = substr($deptid,0,strlen($deptid)-1);
    $array = explode(',', $deptid);
    for($i=0;$i<sizeof($array);$i++){
        if($i==0){ $where = $where." deptid=\".$array[$i].\" ";
        }else{ $where = $where." OR deptid=\".$array[$i].\" ";}
    }
    $sql2 = $sql2." ". $where;
}else{ $sql2 = "";}
if ($sql1!=""){
    if($sql2!=""){ $sql = $sqlstart. " WHERE ($sql1 OR $sql2)";
    }else{ $sql = $sqlstart. " WHERE ($sql1)";}
}else{
    if($sql2!=""){ $sql = $sqlstart. " WHERE ($sql2)";
    }else{$sql = $sqlstart;}
}
$result = pg_exec($dbconn, $sql)
    or die('Could not access to table: ' . pg_last_error());
echo "<?xml version='1.0' encoding='UTF-8'?><kml><Document>";
```

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
while($row = pg_fetch_array($result)){
    //Calculate number of incident in each departments
    if ($_GET['showpast']==1){
        $sql = "SELECT count(*) FROM haiti.\"reports\" where \"deptID\" = ".$row['deptid']." and
incident_date < '". $end."' ";
    }else{
        $sql = "SELECT count(*) FROM haiti.\"reports\" where \"deptID\" = ".$row['deptid']." and
incident_date < '". $end."' and incident_date > '". $start."'";
        if($cat!=""){
            $where = "";
            $cat = substr($cat,0,strlen($cat)-2);
            $array = explode(' ', $cat);
            for($i=0;$i<sizeof($array);$i++){
                if($i==0){ $where = " category LIKE '%".$array[$i]."% " ";
                }else{ $where = $where." OR category LIKE '%".$array[$i]."% " ";
            }
        }
        $where = " AND (".$where.")";
        $sql = $sql.$where;
    }
    $result1 = pg_exec($dbconn, $sql)
    or die('Could not access to table: ' . pg_last_error());
    $row2 = pg_fetch_row($result1);
    $rptnum = $row2[0];
    $density = round((1000*$rptnum/$row['area']),0);
    $Border1 = "";$Border2 = "";$Border3 = "";
    //For polygon 1
    $Border1 = str_replace(" ", "", $row['boundary1']);
    $Border1 = str_replace(",0,", " ", $Border1);
    $Border1 = substr($Border1, 1, strlen($Border1)-3);
    $bound = str_replace(" 0 ", " ", $row['boundary1']);
    $bound = str_replace(" 0", "", $bound);
    $bound = str_replace(", ", " ", $bound);
    $bound = substr($bound, 1, strlen($bound)-1);
    $array = explode(' ', $bound);
    $averlat = 0;$averlon = 0;
    $Slat = 0;$Slon = 0;
    for ($i=0; $i<sizeof($array); $i=$i+2){$Slat = $Slat + doubleval($array[$i]);}
    for ($i=1; $i<sizeof($array); $i=$i+2){$Slon = $Slon + doubleval($array[$i]);}
    $averlat = doubleval(2*$Slat/sizeof($array));
    $averlon = doubleval(2*$Slon/sizeof($array));
    //If a department has 2 polygons and if a department has 3 polygons, codes similar to 1
    $averlat = ($averlat + doubleval(2*$Slat/sizeof($array)))/2;
    $averlon = ($averlon + doubleval(2*$Slon/sizeof($array)))/2;
}
$catarray[][]="";
if($cat!=""){
    $where = "";
    $array = explode(' ', $cat);
    for($i=0;$i<sizeof($array);$i++){
        $where = " AND category LIKE '%".$array[$i]."% " ";
        $result2 = pg_exec($dbconn, $sql.$where)
        or die('Could not access to table: ' . pg_last_error());
        $row3 = pg_fetch_row($result2);
        $catarray[$i][0]=$row3[0];
        $catarray[$i][1]=$array[$i];
    }
}
echo "<Placemark><name>".$row['deptname']."</name>";
if($showpast==1){
    echo "<TimeSpan><start>2010-01-12</start><end>$end</end></TimeSpan>
<description>";
    if($cat!=""){
        $array = explode(' ', $cat);
        if(sizeof($array)>=2){//Show pie chart to compare data of different categories
```

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
echo "<img src='http://chart.apis.google.com/chart?cht=p&chd=t:";
for($i=0;$i<sizeof($array);$i++){
if($i!=sizeof($array)-1){echo $catarray[$i][0].",";
    }else{echo $catarray[$i][0];}
}
echo "&chs=350x150&chf=bg,s,ffffff00&chco=FF0000&chl=";
for($i=0;$i<sizeof($array);$i++){
if($i!=sizeof($array)-1){
echo "Category%20".substr($catarray[$i][1],0,2).(".$catarray[$i][0].")%7C";
    }else{
        echo "Category%20".substr($catarray[$i][1],0,2).(".$catarray[$i][0].");}
}
echo "'&gt;";}
}
$cat = str_replace("'", "", $cat);
echo "<br />There are <a target = "_blank" href =
"reportslist.php?cat=$cat0&gridid=$gridid&start=$start&end=$end&showpast=
$showpast&depid=".$row['depid'].","& $rptnum report(s) </a>";
if ($cat!=""){echo "in categories ".$cat;}
echo " were submitted from 2010-01-12 00:00:00 to $end in department ".$row['deptname']."
(area is ".round($row['area'],2)." km2), density is $density report(s)/1000km2</description>";
}else{
echo "<TimeSpan><start>$start</start><end>$end</end></TimeSpan>
<description>";
    //The codes to show the pie chart is same to the one above
}
echo "<br />There are <a target = "_blank" href =
"reportslist.php?cat=$cat0&gridid=$gridid&start=$start&end=$end&showpast=
$showpast&depid=".$row['depid'].","& $rptnum report(s) </a>";
if ($cat!=""){echo "in categories ".$cat;}
echo " were submitted from $start to $end in department ".$row['deptname']." (area is
".round($row['area'],2)." km2), density is $density report(s)/1000km2</description>";}
echo"<theme>";
if($dens==1){
    if ($density==0) {echo "theme1";
} elseif ($density>0 and $density<=20) {echo "theme2";
} elseif ($density>20 and $density<=50) {echo "theme3";
} elseif ($density>50 and $density<=100) {echo "theme4";
} elseif ($density>100 and $density<=150) {echo "theme5";
} elseif ($density>150 and $density<=200) {echo "theme6";
} elseif ($density>200 and $density<=300) {echo "theme7";
} elseif ($density>300 and $density<=400) {echo "theme8";
} elseif ($density>400) {echo "theme9";}
}else{
if ($rptnum>0 and $rptnum<=50) {echo "theme2";
} elseif ($rptnum>50 and $rptnum<=100) {echo "theme3";
} elseif ($rptnum>100 and $rptnum<=150) {echo "theme4";
} elseif ($rptnum>150 and $rptnum<=200) {echo "theme5";
} elseif ($rptnum>200 and $rptnum<=250) {echo "theme6";
} elseif ($rptnum>250 and $rptnum<=300) {echo "theme7";
} elseif ($rptnum>300 and $rptnum<=400) {echo "theme8";
} elseif ($rptnum>400) {echo "theme9";
} elseif ($rptnum==0) {echo "theme1";}
}
echo "</theme><tags>".$row['deptname'].",</tags>
<LookAt>";
if ($row['depid']== 1){echo "-72.438354,18.521283";
}else if ($row['depid']== 7){
    echo "-73.795166,18.245003";
}else if ($row['depid']== 8){
    echo "-74.080811,18.529096";
}else if ($row['depid']== 10){
    echo "-72.982178,19.818390";
```

```

}else{echo "$averlat,$averlon";}
echo "</LookAt><MultiGeometry><Polygon><outerBoundaryIs>
<LinearRing><coordinates>".$Border1."</coordinates></LinearRing>
</outerBoundaryIs></Polygon>";
if ($Border2!=""){
echo "<Polygon><outerBoundaryIs><LinearRing><coordinates>".$Border2."</coordinates>
</LinearRing></outerBoundaryIs></Polygon>";
}
if ($Border3!=""){
echo "<Polygon><outerBoundaryIs><LinearRing><coordinates>".$Border3."</coordinates>
</LinearRing></outerBoundaryIs></Polygon>";
}
echo "</MultiGeometry></Placemark>";
}
pg_free_result($result);
pg_free_result($result1);
echo "</Document></kml>";
pg_close($dbconn);
?>

```

Appendix A.11 PHP script to export data from PostgreSQL server to KML files for departments in proportional symbols map, files' structure for provinces and for the whole Haiti is similar.

```

<?php
include("dconn.php");
date_default_timezone_set('Europe/Amsterdam');
$start = "";
$end = "";
if ($_GET['start'] != ""){$start = $_GET['start'];}
else{$start = date("Y-m-d h:i:s",strtotime("2010-01-12 00:00:00"));}
if ($_GET['end'] != ""){$end = $_GET['end'];}
else{$end = date("Y-m-d h:i:s",strtotime("now"));}
if(isset($_GET['showpast'])){ $showpast = $_GET['showpast'];}
else{ $showpast = 1;}
if(isset($_GET['cat'])){
    $cat = $_GET['cat'];
    $cat = str_replace("'", "&quot;", $cat);
    $cat0 = $_GET['cat'];
}else{ $cat = ""; $cat0="";}
if(isset($_GET['deptid'])){
    $deptid = $_GET['deptid'];
    $deptid0 = $_GET['deptid'];
}else{ $deptid = ""; $deptid0 = "";}
if(isset($_GET['provid'])){
    $provid = $_GET['provid'];
    $provid0 = $_GET['provid'];
}else{ $provid = ""; $provid0 = "";}
if(isset($_GET['gridid'])){
    $gridid = $_GET['gridid'];
}else{ $gridid = "";}
$sql="";//start sql string
$sqlstart = "SELECT DISTINCT haiti.\"departments\".deptid as
deptid,haiti.\"departments\".deptname as deptname, haiti.\"departments\".boundary1 as
boundary1, haiti.\"departments\".boundary2 as boundary2,haiti.\"departments\".boundary3 as
boundary3 FROM haiti.\"provinces\" INNER JOIN haiti.\"departments\" ON
haiti.\"provinces\".deptname = haiti.\"departments\".deptname ";
$sql1 = "";
//For provinces query
if($provid!=""){
    $where = "";
    $provid = substr($provid,0,strlen($provid)-1);
    $array = explode(',', $provid);
    for($i=0;$i<sizeof($array);$i++){

```

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
        if($i==0){ $where = $where." provid=".$array[$i]." ";
        }else{$where = $where." OR provid=".$array[$i]." ";}
    }
    $sql1 = $sql1." ". $where;
}else{$sql1 = "";}
$sql2 = "";
if($deptid!=""){          //For departments query
    $where = "";
    $deptid = substr($deptid,0,strlen($deptid)-1);
    $array = explode(',', $deptid);
    for($i=0;$i<sizeof($array);$i++){
        if($i==0){ $where = $where." deptid=".$array[$i]." ";
        }else{$where = $where." OR deptid=".$array[$i]." ";}
    }
    $sql2 = $sql2." ". $where;
}else{$sql2 = "";}
if ($sql1!=""){
    if($sql2!=""){
        $sql = $sqlstart. " WHERE ($sql1 OR $sql2)";
    }else{$sql = $sqlstart. " WHERE ($sql1)";}
}else{
    if($sql2!=""){
        $sql = $sqlstart. " WHERE ($sql2)";
    }else{$sql = $sqlstart;}
}
$result = pg_exec($dbconn, $sql)
    or die('Could not access to table: ' . pg_last_error());
echo "<?xml version='1.0' encoding='UTF-8'?><kml><Document>";
while($row = pg_fetch_array($result)){
    //Calculate number of incident in each departments and incident_date > ".$start."
    if ($_GET['showpast']==1){
        $sql = "SELECT count(*) FROM haiti.\"reports\" where \"deptID\" = ".$row['deptid']." and
incident_date < ".$end." ";
    }else{
        $sql = "SELECT count(*) FROM haiti.\"reports\" where \"deptID\" = ".$row['deptid']." and
incident_date < ".$end." and incident_date > ".$start."";}
    $sql1 = $sql;
    if($cat!=""){
        $where = "";
        $cat = substr($cat,0,strlen($cat)-2);
        $array = explode(' ', $cat);
        for($i=0;$i<sizeof($array);$i++){
            if($i==0){$where = " category LIKE '%".$array[$i]."% " ";
            }else{$where = $where." OR category LIKE '%".$array[$i]."% " ";}
        }
        $where = " AND (\".$where.\")";
        $sql = $sql.$where;
    }
    $result1 = pg_exec($dbconn, $sql)
        or die('Could not access to table: ' . pg_last_error());
    $row2 = pg_fetch_row($result1);
    $rptnum = $row2[0];
    $Border1 = "";
    $Border2 = "";
    $Border3 = "";
    //For polygon 1
    $Border1 = str_replace(" ", " ", $row['boundary1']);
    $Border1 = str_replace(",0,", " ", $Border1);
    $Border1 = substr($Border1, 1, strlen($Border1)-3);
    $bound = str_replace(" 0 ", " ", $row['boundary1']);
    $bound = str_replace(" 0", " ", $bound);
    $bound = str_replace(", ", " ", $bound);
    $bound = substr($bound, 1, strlen($bound)-1);
```



## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
$array = explode(' ', $bound);
$averlat = 0; $averlon = 0;
$slat = 0; $slon = 0;
for ($i=0; $i<sizeof($array); $i=$i+2){$slat = $slat + doubleval($array[$i]);}
for ($i=1; $i<sizeof($array); $i=$i+2){$slon = $slon + doubleval($array[$i]);}

$averlat = doubleval(2*$slat/sizeof($array));
$averlon = doubleval(2*$slon/sizeof($array));
//If a department has 2 polygons and department has 3 polygons, codes are the similar to 1
}
if ($rptnum>0){
$catarray[][]="";
if($cat!=""){
$where = "";
$array = explode(' ', $cat);
for($i=0;$i<sizeof($array);$i++){
$where = " AND category LIKE '%" . $array[$i] . "%' ";
$result2 = pg_exec($dbconn, $sql.$where)
or die('Could not access to table: ' . pg_last_error());
$row3 = pg_fetch_row($result2);
$catarray[$i][0]=$row3[0];
$catarray[$i][1]=$array[$i];
}
}
echo "<Placemark><name>".$row['deptname']."</name>";
if($showpast==1){
$start = date('Y-m-d', strtotime($end) - 86400);
echo "<TimeSpan><start>$start</start><end>$end</end></TimeSpan>
<description>";
if($cat!=""){
$array = explode(' ', $cat);
if(sizeof($array)>=2){
echo "<img src='http://chart.apis.google.com/chart?cht=p&chd=t:";
for($i=0;$i<sizeof($array);$i++){
if($i!=sizeof($array)-1){echo $catarray[$i][0].",";
}else{echo $catarray[$i][0];}
}
echo "&chs=350x150&chf=bg,s,ffffff00&chco=FF0000&chl=";
for($i=0;$i<sizeof($array);$i++){
if($i!=sizeof($array)-1){
echo "Category%20".substr($catarray[$i][1],0,2).".".$catarray[$i][0]."%7C";
}else{
echo "Category%20".substr($catarray[$i][1],0,2).".".$catarray[$i][0].".";};
}
echo "'&gt;";
}
}
$cat = str_replace(' ', '"', $cat);
echo "<br />There are <a target = "_blank" href =
"reportslist.php?cat=$cat0&gridid=$gridid&start=2010-01-
12&end=$end&showpast=$showpast&deptid=".$row['deptid'].","&gt; $rptnum
report(s) </a>";
if ($cat!=""){echo "in categories ".$cat0;}
echo " were submitted from 2010-01-12 00:00:00 to $end in department
".$row['deptname']."</description>";
}else{
//In case show past = 0, codes for showing pie chart are similar to the case showpast = 1
$cat = str_replace(' ', '"', $cat);
echo "<br />There are <a target = "_blank" href =
"reportslist.php?cat=$cat0&gridid=$gridid&start=$start&end=$end&showpast=
$showpast&deptid=".$row['deptid'].","&gt; $rptnum report(s) </a>";
if ($cat!=""){echo "in categories ".$cat0;}
echo "were submitted from $start to $end in department ".$row['deptname']."</description>";
}
```

```

}
echo"<theme>";
if ($rptnum>0 and $rptnum<=50) {echo "protheme1";
} elseif ($rptnum>50 and $rptnum<=100) {echo "protheme2";
} elseif ($rptnum>100 and $rptnum<=150) {echo "protheme3";
} elseif ($rptnum>150 and $rptnum<=200) {echo "protheme4";
} elseif ($rptnum>200 and $rptnum<=250) {echo "protheme5";
} elseif ($rptnum>250 and $rptnum<=300) {echo "protheme6";
} elseif ($rptnum>300 and $rptnum<=350) {echo "protheme7";
} elseif ($rptnum>350 and $rptnum<=400) {echo "protheme8";
} elseif ($rptnum>400) {echo "protheme9";}
echo "</theme><tags>".$row['deptname'].",</tags>
<Point><coordinates>";
if ($row['deptid']== 1){echo "-72.438354,18.521283";
}else if ($row['deptid']== 7){echo "-73.795166,18.245003";
}else if ($row['deptid']== 8){echo "-74.080811,18.529096";
}else if ($row['deptid']== 10){echo "-72.982178,19.818390";
}else{echo "$averlat,$averlon";}
echo "</coordinates></Point></Placemark>";
}
}
pg_free_result($result);
pg_free_result($result1);
echo "</Document></kml>";
pg_close($dbconn);
?>

```

## Appendix A.12 PHP scripts to export data from PostgreSQL server to KML file, for the grids network in choropleth map

```

<?php
include("dconn.php");
date_default_timezone_set('Europe/Amsterdam');
$start = "";
$end = "";
if ($_GET['start'] != ""){$start = $_GET['start'];}
else{ $start = date("Y-m-d h:i:s",strtotime("2010-01-12 00:00:00"));}
if ($_GET['end'] != ""){$end = $_GET['end'];}
else{$end = date("Y-m-d h:i:s",strtotime("now"));}
if(isset($_GET['showpast'])){ $showpast = $_GET['showpast'];
}else{ $showpast = 1;}
if(isset($_GET['cat'])){
$cat = $_GET['cat'];
$cat = str_replace("'", "&quot;", $cat);
$cat0 = $_GET['cat'];
}else{ $cat = ""; $cat0="";}
if(isset($_GET['deptid'])){ $deptid = $_GET['deptid']; $deptid0 = $_GET['deptid'];
}else{ $deptid = ""; $deptid0 = "";}
if(isset($_GET['provid'])){ $provid = $_GET['provid']; $provid0 = $_GET['provid'];
}else{ $provid = ""; $provid0 = "";}
if(isset($_GET['gridid'])){ $gridid = $_GET['gridid'];
}else{ $gridid = "";}
if(isset($_GET['dens'])){ $dens = $_GET['dens'];
}else{ $dens = "";}
$sql = "SELECT * FROM haiti.\"grids\"";
$result = pg_exec($dbconn, $sql)
or die('Could not access to table: ' . pg_last_error());
echo "<?xml version='1.0' encoding='UTF-8'?><kml><Document>";
while($row = pg_fetch_array($result)){
$sql = "SELECT count(*) FROM haiti.\"reports\" WHERE \"gridID\" = ".$row['gridid']." AND
incident_date > '$start'";
if($end!=""){ $sql = $sql." AND incident_date < '$end'";}
if($cat!=""){

```

```

$where = "";
$cat = substr($cat,0,strlen($cat));
$array = explode(' ', $cat);
for($i=0;$i<sizeof($array);$i++){
if($i==0){$where = " category LIKE '%" . $array[$i] . "%' ";
}else{$where = $where . " OR category LIKE '%" . $array[$i] . "%' ";}
}
$where = " AND (" . $where . ")";
$sql = $sql . $where;
}
$result1 = pg_exec($dbconn, $sql)
or die('Could not access to table: ' . pg_last_error());
$row1 = pg_fetch_row($result1);
$rptnum = $row1[0];
$density= round((100*$rptnum/123.21),0);
$center = $row1[1];
$center = str_replace(' 0', "", $center);
$center = str_replace(' ', "", $center);
$center = substr($center, 1, strlen($center)-1);
$border = str_replace(" ", "", $row[2]);
$border = str_replace(",0,", " ", $border);
$border = substr($border, 1, strlen($border)-3);
if ($rptnum>0){
if($cat!=""){
$where = "";
$array = explode(' ', $cat);
for($i=0;$i<sizeof($array);$i++){
$where = " AND category LIKE '%" . $array[$i] . "%' ";
$result2 = pg_exec($dbconn, $sql . $where)
or die('Could not access to table: ' . pg_last_error());
$row3 = pg_fetch_row($result2);
$catarray[$i][0]=$row3[0];
$catarray[$i][1]=$array[$i];
}
}
echo "<Placemark><name>Grid ID: ".$row['gridid']. "</name>";
if($showpast==1){echo "<TimeSpan><start>2010-01-12</start><end></end></TimeSpan>";
<description>";
if($cat!=""){
$array = explode(' ', $cat);
if(sizeof($array)>=2){//Display pie chart for comparison of more than 2 categories
echo "<img src='http://chart.apis.google.com/chart?cht=p&chd=t:";
for($i=0;$i<sizeof($array);$i++){
if($i!=sizeof($array)-1){echo $catarray[$i][0].",";
}else{echo $catarray[$i][0];}
}
echo "&chs=350x150&chf=bg,s,ffffff00&chco=FF0000&chl=";
for($i=0;$i<sizeof($array);$i++){
if($i!=sizeof($array)-1){
echo "Category%20".substr($catarray[$i][1],0,2). "(" . $catarray[$i][0] . "%7C";
}else{echo "Category%20".substr($catarray[$i][1],0,2). "(" . $catarray[$i][0] . "%";}
}
echo "'&gt;";
}
}
$cat = str_replace(' ', "", $cat);
echo "<br />There are <a target = "_blank" href = "reportslist.php?cat=$cat0&gridid=".$row['gridid']."&start=$start&end=$end&showpast=$showpast&deptid=$deptid0&provid=$provid0"&gt; $rptnum report(s)</a>";
if ($cat!=""){echo "in categories ".$cat0;}
echo " were submitted from 2010-01-12 to $end here, the density is $density report(s)/100km2</description>";

```

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
}else{echo "<TimeSpan><start>$start</start><end>$end</end></TimeSpan>
<description>";
    //Codes to display pie chart for comparison of galleries are similar to codes above
$cat = str_replace("'", "", $cat);
echo "<br />There are <a target = "_blank" href =
"reportslist.php?cat=$cat0&gridid=".$row['gridid']."&start=$start&end=$end&am
p;showpast=$showpast&deptid=$deptid0&provid=$provid0">> $rptnum report(s)
</a>";
if ($cat!=""){echo "in categories ".$cat0;}
echo " were submitted from $start to $end here, the density is $density
report(s)/100km2</description>";}
echo "<theme>";
if($dens==1){
if ($density==0) {echo "theme1";
} elseif ($density>0 and $density<=100) {echo "theme2";
} elseif ($density>100 and $density<=200) {echo "theme3";
} elseif ($density>200 and $density<=300) {echo "theme4";
} elseif ($density>300 and $density<=400) {echo "theme5";
} elseif ($density>400 and $density<=500) {echo "theme6";
} elseif ($density>500 and $density<=600) {echo "theme7";
} elseif ($density>600 and $density<=700) {echo "theme8";
} elseif ($density>700) {echo "theme9";}
}else{
if ($rptnum>0 and $rptnum<=50) {echo "theme2";
} elseif ($rptnum>50 and $rptnum<=100) {echo "theme3";
} elseif ($rptnum>100 and $rptnum<=150) {echo "theme4";
} elseif ($rptnum>150 and $rptnum<=200) {echo "theme5";
} elseif ($rptnum>200 and $rptnum<=250) {echo "theme6";
} elseif ($rptnum>250 and $rptnum<=300) {echo "theme7";
} elseif ($rptnum>300 and $rptnum<=400) {echo "theme8";
} elseif ($rptnum>400) {echo "theme9";
} elseif ($rptnum==0) {echo "theme1";}
}
echo
"</theme><MultiGeometry><Polygon><outerBoundaryIs>LinearRing<coordinates>".$Border."</coordin
ates></LinearRing></outerBoundaryIs></Polygon></MultiGeometry>
<LookAt>$center</LookAt></Placemark>";
}
}
pg_free_result($result);
echo "</Document></kml>";
pg_close($dbconn);
?>
```

### Appendix A.13 PHP scripts to export data from PostgreSQL server to KML file, for the grids network in proportional symbols map

```
<?php
include("dconn.php");
date_default_timezone_set('Europe/Amsterdam');
$start = "";
$end = "";
if ($_GET['start'] != ""){$start = $_GET['start'];}
else{$start = date("Y-m-d h:i:s",strtotime("2010-01-12 00:00:00"));}
if ($_GET['end'] != ""){$end = $_GET['end'];}
else{$end = date("Y-m-d h:i:s",strtotime("now"));}
if(isset($_GET['showpast'])){ $showpast = $_GET['showpast'];
}else{$showpast = 1;}
if(isset($_GET['cat'])){
    $cat = $_GET['cat'];
    $cat = str_replace("'", "&quot;", $cat);
    $cat0 = $_GET['cat'];
}else{ $cat = ""; $cat0="";}
}
```

```

if(isset($_GET['deptid'])) {
    $deptid = $_GET['deptid'];
    $deptid0 = $_GET['deptid'];
} else { $deptid = ""; $deptid0 = ""; }
if(isset($_GET['provid'])) {
    $provid = $_GET['provid'];
    $provid0 = $_GET['provid'];
} else { $provid = ""; $provid0 = ""; }
if(isset($_GET['gridid'])) {
    $gridid = $_GET['gridid'];
} else { $gridid = ""; }
if(isset($_GET['dens'])) { $dens = $_GET['dens']; }
else { $dens = ""; }
$sql = "SELECT * FROM haiti.\"grids\"";
$result = pg_exec($dbconn, $sql)
    or die('Could not access to table: ' . pg_last_error());
echo "<?xml version='1.0' encoding='UTF-8'?><kml><Document>";
while($row = pg_fetch_array($result)) {
    $sql = "SELECT count(*) FROM haiti.\"reports\" WHERE \"gridID\" = \".$row['gridid'].\" AND
incident_date > '$start' AND incident_date < '$end'";
    if($end!="") { $sql = $sql." AND incident_date < '$end'"; }
    if($cat!="") {
        $where = "";
        $cat = substr($cat,0,strlen($cat));
        $array = explode(' ', $cat);
        for($i=0;$i<sizeof($array);$i++){
            if($i==0) { $where = " category LIKE '%".$array[$i]."% " "; }
            else { $where = $where." OR category LIKE '%".$array[$i]."% " "; }
        }
        $where = " AND (\".$where.\")";
        $sql = $sql.$where;
    }
    $result1 = pg_exec($dbconn, $sql)
        or die('Could not access to table: ' . pg_last_error());
    $row1 = pg_fetch_row($result1);
    $rptnum = $row1[0];
    $center = $row1[1];
    $center = str_replace(' 0', "", $center);
    $center = str_replace(' ', "", $center);
    $center = substr($center, 1, strlen($center)-1);
    $border = str_replace(" ", "", $row1[2]);
    $border = str_replace(",0", " ", $border);
    $border = substr($border, 1, strlen($border)-3);
    if ($rptnum>0) {
        if($cat!="") {
            $where = "";
            $array = explode(' ', $cat);
            for($i=0;$i<sizeof($array);$i++){
                $where = " AND category LIKE '%".$array[$i]."% " ";
                $result2 = pg_exec($dbconn, $sql.$where)
                    or die('Could not access to table: ' . pg_last_error());
                $row3 = pg_fetch_row($result2);
                $catarray[$i][0]=$row3[0];
                $catarray[$i][1]=$array[$i];
            }
        }
        echo "<Placemark> <name>Grid ID: \".$row['gridid'].\"</name>
<TimeSpan><start>2010-01-12</start><end></end></TimeSpan><description>";
        if($cat!="") {
            $array = explode(' ', $cat);
            if(sizeof($array)>=2) {
                echo "&lt;img src='http://chart.apis.google.com/chart?cht=p&chd=t:";
                for($i=0;$i<sizeof($array);$i++){

```



## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
        if($i!=sizeof($array)-1){ echo $catarray[$i][0].",";
        }else{ echo $catarray[$i][0]; }
    }
    echo "&chs=350x150&chf=bg,s,ffffff00&chco=FF0000&chl=";
    for($i=0;$i<sizeof($array);$i++){
        if($i!=sizeof($array)-1){
            echo "Category%20".substr($catarray[$i][1],0,2).(".$catarray[$i][0].")%7C";
        }else{ echo "Category%20".substr($catarray[$i][1],0,2).(".$catarray[$i][0].")";;}
    }
    echo "<br>";
}
$cat = str_replace("'", "", $cat);
echo "<br> />There are <a target = "_blank" href =
'&quot;reportslist.php?cat=$cat0&gridid=".$row['gridid']."&start=$start&end=$end&
showpast=$showpast&deptid=$deptid0&provid=$provid0&quot;>& $rptnum report(s)
</a>";
if ($cat!=""){echo "in categories ".$cat;}
echo " were submitted here from $start to $end</description><theme>";
if ($_GET['zoomlevel']==1){//GMap Zoomlevel 1-6
if ($rptnum>=0 and $rptnum<=200) {echo "protheme1";
} elseif ($rptnum>200 and $rptnum<=400) {echo "protheme2";
} elseif ($rptnum>400) {echo "protheme3";}
}
if ($_GET['zoomlevel']==2){//GMap Zoomlevel 6-8
if ($rptnum>0 and $rptnum<=100) {echo "protheme2";
} elseif ($rptnum>100 and $rptnum<=200) {echo "protheme3";
} elseif ($rptnum>200 and $rptnum<=300) {echo "protheme4";
} elseif ($rptnum>300 and $rptnum<=400) {echo "protheme5";
} elseif ($rptnum>400 and $rptnum<=500) {echo "protheme6";
} elseif ($rptnum>500) {echo "protheme7";}
}
if ($_GET['zoomlevel']==3){//GMap Zoomlevel 8-10
if ($rptnum>=0 and $rptnum<=50) {echo "protheme3";
} elseif ($rptnum>50 and $rptnum<=100) {echo "protheme4";
} elseif ($rptnum>100 and $rptnum<=150) {echo "protheme5";
} elseif ($rptnum>150 and $rptnum<=200) {echo "protheme6";
} elseif ($rptnum>200 and $rptnum<=250) {echo "protheme7";
} elseif ($rptnum>250 and $rptnum<=300) {echo "protheme8";
} elseif ($rptnum>300 and $rptnum<=350) {echo "protheme9";
} elseif ($rptnum>350 and $rptnum<=400) {echo "protheme10";
} elseif ($rptnum>400) {echo "protheme11";}
}
if ($_GET['zoomlevel']==4){//GMap Zoomlevel >10
if ($rptnum>=0 and $rptnum<=50) {echo "protheme4";
} elseif ($rptnum>50 and $rptnum<=100) {echo "protheme5";
} elseif ($rptnum>100 and $rptnum<=150) {echo "protheme6";
} elseif ($rptnum>150 and $rptnum<=200) {echo "protheme7";
} elseif ($rptnum>200 and $rptnum<=250) {echo "protheme8";
} elseif ($rptnum>250 and $rptnum<=300) {echo "protheme9";
} elseif ($rptnum>300 and $rptnum<=350) {echo "protheme10";
} elseif ($rptnum>350 and $rptnum<=400) {echo "protheme11";
} elseif ($rptnum>400) {echo "protheme12";}
}
echo "</theme><Point><coordinates>$center</coordinates></Point>";}
echo "</Placemark>";
}
}
pg_free_result($result);
echo "</Document></kml>";
pg_close($dbconn);
?>
```

#### Appendix A.14 Define themes for point, proportional point and polygon symbols

//For 8 point symbols of main categories:

```
I1: new TimeMapTheme({
  iconImage:      "../images/emergency.png",
  color:          "#6F1625",
  lineOpacity:    1,
  lineWeight:     2,
  fillOpacity:    0.9,
  eventTextColor: null,
  eventIconPath:  "timemap/images/",
  eventIconImage: "I1.png",
  classicTape:    false,
}),
```

The 7 other point symbols are similarly defined, 8 different colours were used, and 8 different icon images in timeline and timemap were used.

//For proportional point symbols

```
prothemel: new TimeMapTheme({
  iconImage:      "../images/image_points.php?size=3&fname=squarepoint.png",
  color:          "#FFF0F0",
  lineOpacity:    1,
  lineWeight:     3,
  fillOpacity:    0.9,
  eventTextColor: null,
  eventIconPath:  "timemap/images/",
  classicTape:    false,
  polygonLineColor: "#808080",
  polygonLineWeight: 1,
  polygonLineOpacity: 1,
})
```

The 11 others proportional point symbols are similarly defined, size for iconImage from 3-25.

//For border and the other polygon symbols

```
border: new TimeMapTheme({
  color:          "#FF0000",
  lineOpacity:    1,
  lineWeight:     3,
  fillOpacity:    0,
  eventTextColor: null,
  eventIconPath:  "timemap/images/",
  classicTape:    false,
  polygonLineColor: "#FFFF00",
  polygonLineWeight: 1,
  polygonLineOpacity: 1
}),
themel: new TimeMapTheme({
  color:          "#FFF0F0",
  lineOpacity:    1,
  lineWeight:     3,
  fillOpacity:    0.9,
  eventTextColor: null,
  eventIconPath:  "timemap/images/",
  classicTape:    false,
  polygonLineColor: "#808080",
  polygonLineWeight: 1,
  polygonLineOpacity: 1,
}),
```

The 8 other polygon symbols are defined similarly, background colours' range from #FFF0F0 to \$FF0000.

#### Appendix A.15 JavaScripts to load Google Maps in submit reports and reports' details pages

```
function load_Gmap() {
  if (GBrowserIsCompatible()) {
```

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
var map = new GMap2(document.getElementById("Gmap"));
map.addControl(new GSmallMapControl());
map.addControl(new GMapTypeControl());
map.enableDoubleClickZoom();
map.enableScrollWheelZoom();
map.enableContinuousZoom();
var center = new GLatLng(18.773557,-72.317368);//Lat lon coordinates of Haiti's center
map.setCenter(center, 7);
geocoder = new GClientGeocoder();
var marker = new GMarker(center, {draggable: true});
map.addOverlay(marker);
//Get coordinates after drag marker
GEvent.addListener(marker, "dragend", function() {
    var point = marker.getPosition();
    map.panTo(point);
    document.getElementById("rptlatlon").value = point.lat().toFixed(6)+' '+
point.lng().toFixed(6);
});
//Get coordinates after move the map
GEvent.addListener(map, "moveend", function() {
    map.clearOverlays();
    var center = map.getCenter();
    var marker = new GMarker(center, {draggable: true});
    map.addOverlay(marker);
    document.getElementById("rptlatlon").value = center.lat().toFixed(6)+' '+
center.lng().toFixed(6);
    GEvent.addListener(marker, "dragend", function() {
        var point = marker.getPosition();
        map.panTo(point);
        document.getElementById("rptlatlon").value = point.lat().toFixed(6)+' '+
point.lng().toFixed(6);
    });
});
}
```

### Appendix A.16 JavaScripts codes to search location in submit reports and reports' details pages

```
function gotoAddress(address) {
    var map = new GMap2(document.getElementById("Gmap"));
    map.addControl(new GSmallMapControl());
    map.addControl(new GMapTypeControl());
    map.enableDoubleClickZoom();
    map.enableScrollWheelZoom();
    map.enableContinuousZoom();
    if (geocoder) {
        geocoder.getLatLng(address, function(point) {
            if (!point) {
                alert(address + " not found");
            } else {
                document.getElementById("rptlatlon").value = point.lat().toFixed(6)+' '+
point.lng().toFixed(6);
                map.clearOverlays();
                map.setCenter(point, 14);
                var marker = new GMarker(point, {draggable: true});
                map.addOverlay(marker);
                GEvent.addListener(marker, "dragend", function() {
                    var pt = marker.getPosition();
                    map.panTo(pt);
                    document.getElementById("rptlatlon").value = pt.lat().toFixed(6)+' '+
pt.lng().toFixed(6);
                });
            }
        });
    }
    GEvent.addListener(map, "moveend", function() {
```

```
map.clearOverlays();
var center = map.getCenter();
var marker = new GMarker(center, {draggable: true});
map.addOverlay(marker);
document.getElementById("rptlatlon").value = center.lat().toFixed(6)+' '+
center.lng().toFixed(6);
GEvent.addListener(marker, "dragend", function() {
var pt = marker.getPosition();
map.panTo(pt);
document.getElementById("rptlatlon").value = pt.lat().toFixed(6)+' '+ pt.lng().toFixed(6);
});
});
}
});
}
document.getElementById("rptlocation").value = document.getElementById("address").value;
}
```

#### Appendix A.17 Function to add/remove items in report's details, submit new report and search report pages

```
//Add category and choose the existing category to delete it
function setSelectedCat(ul) {
    text = document.getElementById('rptcatid').value;
    if (text.indexOf(ul.text)>=0){
        deltext = ul.text + ", ";
        text = text.replace(deltext,"");
        document.getElementById('rptcatid').value = text;
    }else{ document.getElementById('rptcatid').value += ul.text + ", ";}
};

//Add department and choose the existing department to delete it, created similarly for
add/remove province in search report's page.
function setSelectedDept(sl) {
var selObj = document.getElementById('sldeptid');
var slid;
for (i=0; i<selObj.options.length; i++) {
    if (selObj.options[i].selected) {
        slid = i+1;break;}
}
text = document.getElementById('deptname').value;
id = document.getElementById('deptid').value;
if (text.indexOf(sl.value)>=0){
    deltext = sl.value + ", ";
    delid = slid+", ";
    $("#sldeptid").val([]);
    text = text.replace(deltext,"");
    id = id.replace(delid,"");
    document.getElementById('deptname').value = text;
    document.getElementById('deptid').value = id;
}else{
    document.getElementById('deptname').value += sl.value + ", ";
    document.getElementById('deptid').value += slid + ", ";}
};
```

#### Appendix A.18 PHP scripts to do resize the images in the prototype

```
// Define file and new size
$size = $_GET['size'];
$percent = $size/10;
$shape = $_GET['shape'];
$filename = 'red-'. $shape. '.png';
// Content type
header('Content-Type: image/jpeg');
// Get new sizes
```

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
list($width, $height) = getimagesize($filename);
$newwidth = $width * $percent/100;
$newheight = $height * $percent/100;
// Load
$thumb = imagecreatetruecolor($newwidth, $newheight);
$source = imagecreatefrompng($filename);
// Resize
imagecopyresized($thumb, $source, 0, 0, 0, 0, $newwidth, $newheight, $width, $height);
// Output
imagejpeg($thumb);
```

### Appendix A.19 Scripts to create a slider in the prototype

```
<div class="slider" id="slider-1"><input class="slider-input" id="slider-input-1"/></div>
<script type="text/javascript">
var s = new Slider(document.getElementById("slider-1"), document.getElementById("slider-input-1"));
s.setMaximum(60);
s.setMinimum(1);
s.onchange = function () {
    document.getElementById("h-value").value = s.getValue();
    document.getElementById("h-min").value = s.getMinimum();
    document.getElementById("h-max").value = s.getMaximum();
    s.setValue(s.getValue());
    s.setMinimum(s.getMinimum());
    s.setMaximum(s.getMaximum());
};
s.setValue(30);
window.onresize = function () {
    s.calculate();
};
</script>
```

### Appendix A.20 JavaScripts for set of animation functions

```
function play(){// play animation
    d1= tm.timeline.getBand(0).getCenterVisibleDate().getTime();
    step=s.getValue();
    time=s3.getValue();
    time=100+19*(100-time);
    //If step = 1 or 60, the time value will be changed 60000 or 3600000 milliseconds after
    //executing play() function
    d1 = d1 + step*60*1000;
    d = new Date(d1);
    enddate = new Date();
    if (d>enddate) { //restart the animation if the timeline goes to the current date
        restart();
    }else{
        tm.timeline.getBand(0).setCenterVisibleDate(d);
        //If time=2000 or 100 play()function will execute 0.5 or 10 times per second
        t=setTimeout("play()",time);
    }
};
function pause(){clearTimeout(t);};// pause animation
function restart(){// restart animation
    dreset = tm.eventSource.getEarliestDate();
    tm.timeline.getBand(0).setCenterVisibleDate(dreset);
};
```

### Appendix A.21 JavaScripts function jump to specific time

```
function gototime(){
    gotime=
    Date(document.getElementById('goyear').value,document.getElementById('gomonth').value-
    1,document.getElementById('goday').value);
    new
```



```

tm.timeline.getBand(0).setCenterVisibleDate(gotime);
tm.filter('map');
tm.filter('timeline');
tm.timeline.layout();
};

```

### Appendix A.22 JavaScripts codes to manage zoom level by using Google Maps API

```

GEvent.addListener(tm.map, "zoomend", function(oldLevel, newLevel) {
    if (newLevel>10){
        tm.each(function() { //Zoomlevel 4
            tm.datasets.Reports4.visible = true;
            tm.datasets.Reports3.visible = false;
            tm.datasets.Reports2.visible = false;
            tm.datasets.Reports1.visible = false;
        });
        document.getElementById("zoomlevel1").innerHTML = "";
        document.getElementById("zoomlevel2").innerHTML = "";
        document.getElementById("zoomlevel3").innerHTML = "";
        document.getElementById("zoomlevel4").innerHTML =
            "<img src='../images/image_points.php?size=9&fname=squarepoint.png' /> 0 - 50 report submitted<br /><img src='../images/image_points.php?size=11&fname=squarepoint.png' /> 51 - 100 reports submitted<br /><img src='../images/image_points.php?size=13&fname=squarepoint.png' /> 101 - 150 reports submitted<br /><img src='../images/image_points.php?size=15&fname=squarepoint.png' /> 151 - 200 reports submitted<br /><img src='../images/image_points.php?size=17&fname=squarepoint.png' /> 201 - 250 reports submitted<br /><img src='../images/image_points.php?size=19&fname=squarepoint.png' /> 251 - 300 reports submitted<br /><img src='../images/image_points.php?size=21&fname=squarepoint.png' /> 301 - 350 reports submitted<br /><img src='../images/image_points.php?size=23&fname=squarepoint.png' /> 351 - 400 reports submitted<br /><img src='../images/image_points.php?size=25&fname=squarepoint.png' />> 400 reports submitted<br />";
        tm.filter("map");
        tm.filter("timeline");
        tm.timeline.layout();
    }
});
//The codes for the others range of zoom levels is structured similarly to the codes above.
});

```

### Appendix A.23 JavaScripts codes of filter functions

```

function setSelectedTag(selectedtags) { // onChange handler for filtering from select menu
    window.selectedTag = selectedtags.value;
    // run filters
    tm.filter('map');
    tm.filter('timeline');
    // update the timeline
    tm.timeline.layout();
};

function setSelectedTagUl(ul) { // onClick handler for filtering from list menu
    window.selectedTag = ul.text;
    // run filters
    tm.filter('map');
    tm.filter('timeline');
    // update timeline
    tm.timeline.layout();
};

//Select all items
function setSelectAll() {
    window.selectedTag = "";
    // run filters
    tm.filter('map');
    tm.filter('timeline');
    // update timeline
}

```

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

```
tm.timeline.layout();
};
```

### Appendix A.24 PHP codes to create combo box for departments' names and provinces' name

```
$result = pg_exec($dbconn, "SELECT DISTINCT deptname FROM haiti.\"departments\" ORDER BY
deptname ASC");
or die('Could not access to table: ' . pg_last_error());
echo "<form method=post><select name='sldept' onchange='setSelectedTag(this);'>";
echo "<option selected value='' onclick = 'setSelectAll()'>All departments</option>";
while ($row = pg_fetch_array($result)) {
echo "<option value=\"".$row['deptname']."\">".$row['deptname']."</option>";
}
echo "</select></form>";
```

### Appendix A.25 Codes to show popup list of categories

```
<a onclick = "setSelectAll()">All categories</a><br />
<a class = "mcat1" onmouseover="mopen('m1')" onmouseout="mcloseTime()" onclick =
"setSelectedTagUl(this)">1. Emergency</a>
<div class = "mcatdiv" id="m1" onmouseover="mcancelcloseTime()" onmouseout="mcloseTime()">
<a onclick = setSelectedTagUl(this) >1d. Fire</a><br />
<a onclick = setSelectedTagUl(this) >1c. People trapped</a><br />
<a onclick = setSelectedTagUl(this) >1b. Medical Emergency</a><br />
<a onclick = setSelectedTagUl(this) >1a. Highly vulnerable</a>
</div>
```

### Appendix A.26 Scripts to do put timemap and timeline into the prototype

```
<script type="text/javascript"
src="http://maps.google.com/maps?file=api&v=2&key=AIzaSyArgnxFEgzUg1FCDWNUoZNP_WNE8XYyt2Y"></s
cript>
<script type="text/javascript" src="../../timeline_2.3.0/src/webapp/api/timeline-
api.js"></script>
<script type="text/javascript" src="js/timemap.js"></script>
// Define one of the band's themes namely bandthemeH (theme for band named HOUR)
var bandthemeH = Timeline.ClassicTheme.create();
bandthemeH.event.track.gap = 1;
bandthemeH.event.tape.height = 1;
// Initialize the timemap and timeline
tm = TimeMap.init({
mapId: "map", // Id of map div element (required)
timelineId: "timeline", // Id of timeline div element (required)
options: { eventIconPath: "../images/" },
datasets: [{
{
id: "Reports1",
title: "Departments KML Dataset",
type: "kml",
options: {
// Data to be loaded in KML from a remote URL
url: "data/kml_provinces_pointsmaph.php?start=&end=&prov=&dept=&isborder=1"
}
},
//Band information of the band HOUR, the other bands are defined in similar structure.
bandInfo: [{
width: "50%", //Define the height of band
intervalUnit: Timeline.DateTime.HOUR, //Define the type of band, the firstband is also called
topband
intervalPixels: 50, //Define the interval inside the band
showEventText: false, //false means the events' text are not shown on timeline
overview: true, //true means the events' icons are shown in tiny size and unable
to click on them
theme: bandthemeH //Define the theme of band
}],
```

```
//dataDisplayedFunction to show the data on the timemap
dataDisplayedFunction: function(tm) {
    tm.datasets.Reports1.visible = true;
    tm.datasets.Reports2.visible = false;
    start_date=tm.eventSource.getEarliestDate();
    tm.timeline.getBand(0).setCenterVisibleDate(start_date);
tm.filter("map");
tm.filter("timeline");
tm.timeline.layout();
}
//Codes to define timemap and timeline in the prototype
<div id="timeline"></div>
<div id="map"></div>
```

#### Appendix A.27 Codes to change interface of infowindow in the prototype

```
//JavaScript codes
TimeMapItem.openInfoWindowBasic = function() {
    var html = this.opts.infoHtml;
    // create content for info window if none is provided
    if (html === "") {
        html = '<div class="infotitle">' + this.opts.title + '</div>';
        if (this.opts.description !== ""){
            des=this.opts.description;
            description=des.split(";");
            de=0;
            while (de < description.length){
                html += '<div class="infodesc">' + description[de] + '</div>';
                de+=1;}
            }
        lo=this.getInfoPoint();
        html += '<div class="info"> Position: ' + lo + '</div>';
    }
}
//CSS codes
div.infotitle { font-size: 12px; font-weight: bold; width: 400px;}
div.infodesc { font-size: 10px; font-style: italic; width: 400px;}
div.info { font-family: Arial; font-size: 13px; font-style: bold; width: 400px;}
```

#### Appendix A.28 JavaScripts codes in init() function for showing data by using progressive.js

```
datasets: [{
    id: "Reports1",
    title: "Progressive departments KML Dataset",
    //Define the type of data loader
    type: "progressive",
    options: {
        // Data to be loaded in KML from a remote URL
        loader: new TimeMap.loaders.kml({
            // url with start/end placeholders
            url: "data/kml_reports_pointsmap.php?&start=[start]&end=[end]"}),
        //Define the start date of data will be loaded
        start: "2010-01-12",
        // lower cutoff date for data
        dataMinDate: "2010-01-12",
        // 12 hours in milliseconds, the data will be loaded within each 12 hours period
        interval: 43200000,
        // function to turn date into string appropriate for service
        formatDate: function(d) { return TimeMap.util.formatDate(d, 3);}
    }
},
```

#### Appendix A.29 PHP codes to register a new user.

```
<?php
$username = $_GET['username'];
```

```

$realname=$_GET['realname'];
$email=$_GET['email'];
$password = $_GET['pwd'];
if(isset($_GET['activated'])){$activated=$_GET['activated'];}
}else{$activated="";}
if($activated!=""){
    $sql = "INSERT INTO haiti.\"users\" (uid,username,password, email,
realname, activated, ulevel) VALUES((SELECT max(uid) FROM haiti.\"users\")+1, '$username',
md5('$password'), '$email', '$realname', TRUE, 2)";
}else{$sql = "INSERT INTO haiti.\"users\" (uid,username,password, email, realname,
activated, ulevel) VALUES((SELECT max(uid) FROM haiti.\"users\")+1, '$username',
md5('$password'), '$email', '$realname', FALSE, 2)";}
$result = pg_query($dbconn, $sql)
or die('There is an error: ' . pg_last_error());
?>
//Code to create Administrator
<?php
$sql = "INSERT INTO haiti.\"users\" (uid,username,password, email, realname, activated,
ulevel) VALUES((SELECT max(uid) FROM haiti.\"users\")+1, 'Administrator', md5('zxcvbnm1'),
'ducha.hung@gmail.com', 'Hoàng Anh Đức', TRUE, 1)";
$result = pg_query($dbconn, $sql)
or die('There is an error: ' . pg_last_error());
?>
Appendix A.30 PHP codes for log-in page
<?php
$sql="SELECT * FROM haiti.users WHERE username = '$_GET['user'].'" and \"password\" =
md5('$_GET['pwd'].'"') and activated = TRUE";
$result = pg_exec($dbconn, $sql)
or die('Could not access to table: ' . pg_last_error());
$row = pg_fetch_row($result);
if ($row!=""){
    echo "Hello <a href = 'userdetail.php?id=".$_row[0]."' target = '_blank'>".$_row[4]."</a>,"
<a href = 'logout.php'> logout</a>";
}else{
    echo "<div id='alert'><font color = red>Wrong username or password or your account is not
activated</font></div><br /><a id = 'showloginbox' onclick = 'showLogin()'>Login</a>";}
$_SESSION['uid']=$row[0];$_SESSION['uname']=$row[1];
$_SESSION['realname']=$row[4];$_SESSION['ulevel']=$row[6];
?>

```

## APPENDIX B: Tables in the research

Appendix B.1: Advantages and disadvantages of 7 investigated web visualization libraries

	Web libraries	Advantages	Disadvantages
1	Google Maps JavaScript API	<ul style="list-style-type: none"> <li>- Provides Google base map with various functions for mapping</li> <li>- Supports most recent web browser.</li> <li>- Can overlay data from KML, KMZ and GeoRSS.</li> </ul>	<ul style="list-style-type: none"> <li>- Does not support animation and timeline</li> </ul>
2	OpenLayers API	<ul style="list-style-type: none"> <li>- Can load map data from many sources.</li> <li>- Users can use all the tools on all the data sources.</li> <li>- Provides interactive mapping functions</li> <li>- Provides animations as a sequence of layers or images</li> </ul>	<ul style="list-style-type: none"> <li>- For each animation step, a layer class need to be created.</li> <li>- Does not provide temporal brushing for large time-series</li> </ul>
3	SIMILE	<ul style="list-style-type: none"> <li>- Slim, fast and can run on most of the recent browsers</li> <li>- Allows users to do filtering, brushing and searching</li> <li>- Provides time plot and scatter plot</li> <li>- Provides pop-up window</li> </ul>	<ul style="list-style-type: none"> <li>- Does not support animation and displaying temporal data.</li> </ul>
4	Timemap	<ul style="list-style-type: none"> <li>- Allows users to combine Google Maps API with SIMILE timeline</li> <li>- Provides both functions of Google Maps API and SIMILE timeline.</li> <li>- Allows users to load data in multiple datasets.</li> </ul>	<ul style="list-style-type: none"> <li>- Does not allow users to filter several objects at the same time.</li> <li>- Does not support automatic animation</li> </ul>
5	GeoCommons API	<ul style="list-style-type: none"> <li>- Allows users to display points, lines, and polygons on the map.</li> <li>- Allows users to load multiple datasets.</li> <li>- Offers mapping tools like zooming, panning, etc...</li> <li>- Provides filtering and searching tools.</li> <li>- Allows users to do analysis online such as buffer, sum, intersect, etc...</li> </ul>	<ul style="list-style-type: none"> <li>- Must use GeoCommons server to store data and use data uploaded on GeoCommons server.</li> </ul>
6	Thematic maps API	<ul style="list-style-type: none"> <li>- Allows us to create thematic maps such as choropleth maps, prism maps, proportional symbol maps, pie chart maps, etc... from our own data</li> <li>- Can show data from various formats such as JSON, KML, KMZ, Google Spreadsheets, etc... on base map of Google, Google Earth, OpenStreetMap, etc...</li> </ul>	<ul style="list-style-type: none"> <li>- Must install Google Earth API if users want to use animation and showing temporal data by using Google Earth base map.</li> </ul>
7	Google Chart API	<ul style="list-style-type: none"> <li>- Allows us to create charts and reports</li> <li>- Can be integrated directly into our website</li> <li>- No plug-in is needed</li> <li>- Able to do sorting, modifying, and filtering</li> </ul>	<ul style="list-style-type: none"> <li>- Does not provide map view and temporal view</li> </ul>

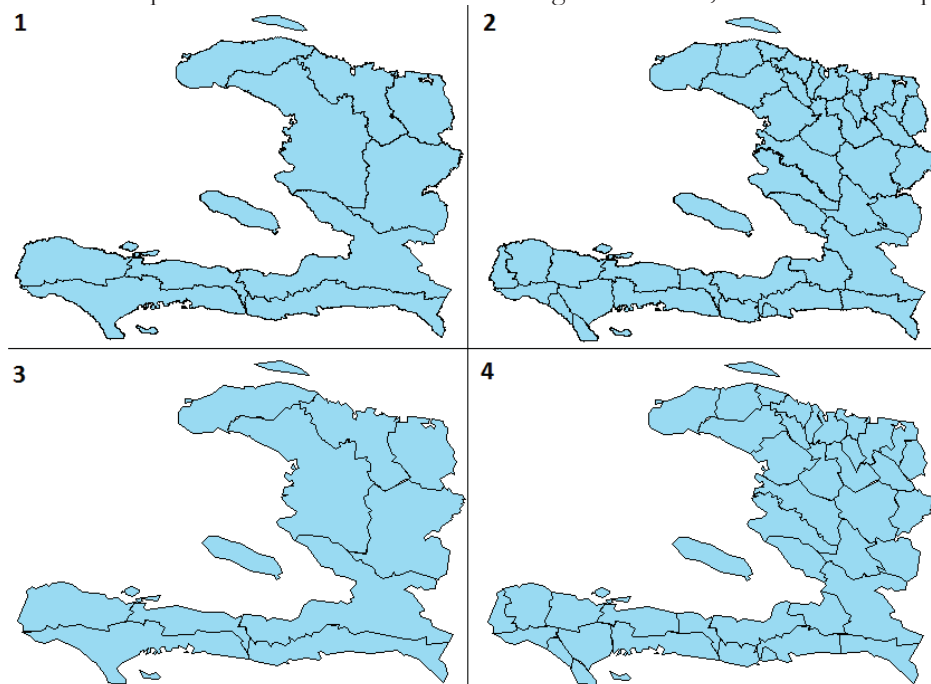


## APPENDIX C: Figures in the research

Appendix C.1: Haiti earthquake reports dataset in PostgreSQL

	id bigint	incident_title character va	incident_date timestamp wi	location character va	description character va	category character va	latitude double pre	longitude double pre	approved boolean	verified boolean
1	1	Port-au-Prince a	2010-01-12 04:0	Airport, Port-au	NBC reports Por	5b. Structures a	18.579721	-72.292778	TRUE	FALSE
2	2	Magnitude 7.0,	2010-01-12 04:5	Chauflard	Location: 18.45	6c. Seisme et re	18.465589	-72.412499	TRUE	FALSE
3	3	Haiti: Earthquak	2010-01-12 04:5	17km south-wes	This report was	6c. Seisme et re	18.484819	-72.336731	TRUE	FALSE
4	4	Your Haiti earth	2010-01-12 07:5	port-au-prince	Photos from the	1. Urgences   Er	18.539269	-72.336408	TRUE	FALSE
5	5	Post-earthquake	2010-01-12 07:5	Port-Au-Prince	RT @caribnews:	8. Autre   Other	18.508912	-72.325058	TRUE	FALSE
6	6	Photos haiti ear	2010-01-12 08:0	Port-au-Prince	RT @LATimesPh	1. Urgences   Er	18.539269	-72.336408	TRUE	FALSE
7	7	Collapsed Buildi	2010-01-12 09:5	Delmas 19, Rue	My mother is pa	5a. Structure ef	18.539269	-72.336408	TRUE	FALSE
8	8	National Palace	2010-01-12 10:4	National Palace	National Palace	5a. Structure ef	18.543376	-72.338883	TRUE	TRUE
9	9	Eye-Witness Re	2010-01-12 11:1	Ave Christophe	Tequila Minsky,	5a. Structure ef	18.538328	-72.333067	TRUE	FALSE
10	10	Urgent need of	2010-01-12 13:0	Saint Martin bek	[6011] english:	2a. Penurie d'ali	18.55225	-72.33293	TRUE	FALSE
11	11	Urgent need of	2010-01-12 13:0	Saint Martin bek	[6011] english:	2a. Penurie d'ali	18.55225	-72.33293	TRUE	FALSE
12	12	Hôpital Albert Sc	2010-01-13 01:0	Deschapelles, H	Dr Michele Barry	7d. Services de	19.083333	-72.5	TRUE	TRUE
13	13	Is Dr. Jean Pap	2010-01-13 01:2	port of port-au-	anyone know if	5a. Structure ef	18.529096	-72.350464	TRUE	FALSE
14	14	Trapped in build	2010-01-13 01:3	Mont Joli-Turges	To anyone in th	1c. Personnes p	18.5352	-72.332954	TRUE	FALSE
15	15	Looking for 3 An	2010-01-13 01:3	Port Au Prince, I	My family and I	8e. Nouvelles de	18.539269	-72.336408	TRUE	FALSE

Appendix C.2: Haiti boundary map, 1 and 3 are 10 departments of Haiti before and after generalization; 2 and 4 are 41 provinces of Haiti before and after generalization, shown in Arc Map 10.3.



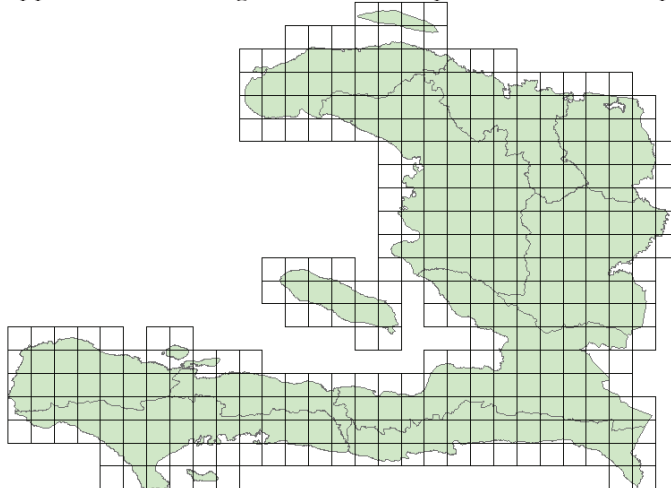
Appendix C.3: Data table for departments in Haiti

	deptid bigint	deptname character v	boundary1 character v	boundary2 character v	boundary3 character v	area double precis
1	1	Ouest	-73.2047498	-72.6269573		5258.270133
2	2	Sud-Est	-72.4513520			2144.901285
3	3	Nord	-72.5552689			2230.914186
4	4	Nord-Est	-71.9614620			1725.765507
5	5	Artibonite	-72.6957164			5145.446736
6	6	Centre	-71.9840254			3659.201469
7	7	Sud	-73.7037005	-73.3926438	-73.6635463	2789.917956
8	8	Grand-Anse	-73.8089162	-73.7549026	-74.1943904	2018.746566
9	9	Nippes	-73.5673537			1280.730987
10	10	Nord-Ouest	-72.8163661	-72.7964546		2223.632475

Appendix C.4: Data table for provinces in Haiti

	provid bigint	deptname character	provname character varyin	boundary1 character vari	boundary2 character vari	boundary3 character vari	area double prec
1	41	Nord-Ouest	Môle Saint Nicolas	-73.158740771			1174.400757
2	40	Nord-Ouest	Saint Louis du Nord	-72.748182906			195.546591
3	39	Nord-Ouest	Port-de-Paix	-72.816366194	-72.796454615		853.672806
4	38	Nippes	Anse à Veau	-73.567353765			896.291145
5	37	Nippes	Miragoâne	-73.181616302			384.439842
6	36	Grand-Anse	Corail	-73.808916233	-74.077317310	-73.754902647	823.695813
7	35	Grand-Anse	Anse D'Ainault	-74.384723747			341.267058
8	34	Grand-Anse	Jérémie	-74.194390483			853.771374
9	33	Sud	Chardonnières	-74.076882487			401.972625
10	32	Sud	Coteaux	-74.035479231			186.958854
11	31	Sud	Aquin	-73.392643835	-73.663546354		1107.497727
12	30	Sud	Port-Salut	-73.872906415			186.675471
13	29	Sud	Cayes	-73.703700522	-73.785601670		906.837921
14	28	Centre	Cerca La Source	-71.748632791			633.447252
15	27	Centre	Lascahobas	-71.834821484			763.125777
16	26	Centre	Mirebalais	-72.221742914			901.182582
17	25	Centre	Hinche	-71.984025479			1361.445858
18	24	Artibonite	Marmelade	-72.364279238			757.347228
19	23	Artibonite	Dessalines	-72.739568272			1192.832973
20	22	Artibonite	Saint-Marc	-72.745920116			1111.267953

Appendix C.5: Haiti grids network map, shown in Arc Map 10.3.



Appendix C.6: Data table for grids network based on the borders of Haiti

	gridid bigint	center character varying	border character varying
1	1	-74.0303763309999 18.07205718100005 C	-74.08037633099985 18.02205718100015
2	2	-73.93037633099993 18.07205718100005	-73.98037633099995 18.02205718100015
3	3	-73.83037633099991 18.07205718100005	-73.88037633099998 18.02205718100015
4	4	-73.73037633099989 18.07205718100005	-73.78037633099996 18.02205718100015
5	5	-73.63037633099992 18.07205718100005	-73.68037633099993 18.02205718100015
6	6	-73.5303763309999 18.07205718100005 C	-73.5803763309998 18.02205718100015 C
7	7	-71.83037633099997 18.07205718100005	-71.88037633099992 18.02205718100015
8	8	-71.73037633099995 18.07205718100005	-71.7803763309999 18.02205718100015 C
9	9	-74.0303763309999 18.17205718099996 C	-74.08037633099985 18.12205718100012
10	10	-73.93037633099993 18.17205718099996	-73.98037633099995 18.12205718100012
11	11	-73.83037633099991 18.17205718099996	-73.88037633099998 18.12205718100012
12	12	-73.73037633099989 18.17205718099996	-73.78037633099996 18.12205718100012
13	13	-73.63037633099992 18.17205718099996	-73.68037633099993 18.12205718100012
14	14	-73.5303763309999 18.17205718099996 C	-73.5803763309998 18.12205718100012 C
15	15	-73.43037633099988 18.17205718099996	-73.48037633099995 18.12205718100012
16	16	-73.33037633099991 18.17205718099996	-73.38037633099998 18.12205718100012
17	17	-73.23037633099989 18.17205718099996	-73.28037633099996 18.12205718100012
18	18	-73.13037633099987 18.17205718099996	-73.18037633099993 18.12205718100012
19	19	-73.0303763309999 18.17205718099996 C	-73.08037633099991 18.12205718100012
20	20	-72.93037633099988 18.17205718099996	-72.98037633099995 18.12205718100012

Appendix C.7: Data table for users

uid bigint	username character varying	password character varying	email character varying	realname character varying	activated boolean	ulevel integer
1	Administrator	bd3856affd9e5e3	ducha.hung@gmail.com	Hoàng Anh Đức	TRUE	1
2	Volunteer	bd3856affd9e5e3	participant@itc.nl	Participant	TRUE	2
5	Hari	5353a483948381a	dhonjuh@yahoo.com	Hari Krishna Dhonju	TRUE	2
7	yeayallig	c56e1d4df09230a	bogale26283@itc.nl	Yetnayet A. Bogale	TRUE	2
8	CodeRed	10aa393bc40283f	odeyemi07463@itc.nl	Chris Odeyemi	TRUE	2
9	eden	7c52abe8932ad2f	eden25571@itc.nl	ugyen eden	TRUE	2
10	dube26865	084f9e1ad260ea5	dube26865@itc.nl	Timothy	TRUE	2
11	susheel	81ccfd19206edd4	susheeldangol@gmail.com	susheel	TRUE	2
12	baral26430	a1a7b6a11392cec	baral.tikaram@gmail.com	tkaram	TRUE	2
13	Swamy	fc6896432e1773	bmdevswamy@hotmail.com	Mahadevaswamy.B	TRUE	2
14	abrha	e18c9b8a6cc18e0	abrha26285@itc.nl	kiflom	TRUE	2

Appendix C.8: Data table for reports after applying the check and input PHP script for allocating IDs

id bigint	incident_title character varying	incident_date timestamp without time zone	location character varying	description character varying	category character varying	latitude double precision	longitude double precision	approved boolean	verified boolean	deptID bigint	provID bigint	gridID bigint
1	Port-au-Prince a	2010-01-12 04:00	Airport, Port-au	NBC reports Por	5b. Structures a	18.579721	-72.292778	TRUE	FALSE	1	1	130
2	Magnitude 7.0,	2010-01-12 04:5	Chauflard	Location: 18.45	6c. Seisme et re	18.465589	-72.412499	TRUE	FALSE	1	1	109
3	Haiti: Earthquak	2010-01-12 04:5	17km south-wes	This report was	6c. Seisme et re	18.484819	-72.336731	TRUE	FALSE	1	1	110
4	Your Haiti earth	2010-01-12 07:3	port-au-prince	Photos from the	1. Urgences   Er	18.539269	-72.336408	TRUE	FALSE	3	12	130
5	Post-earthquake	2010-01-12 07:5	Port-Au-Prince	RT @caribnews:	8. Autre   Other	18.508912	-72.325058	TRUE	FALSE	1	1	110
6	Photos haiti ear	2010-01-12 08:0	Port-au-Prince	RT @LATimesPh	1. Urgences   Er	18.539269	-72.336408	TRUE	FALSE	3	12	130
7	Collapsed Buildi	2010-01-12 09:5	Delmas 19, Rue	My mother is pa	5a. Structure ef	18.539269	-72.336408	TRUE	FALSE	3	12	130
8	National Palace	2010-01-12 10:4	National Palace	National Palace	5a. Structure ef	18.543376	-72.338883	TRUE	TRUE	3	12	130
9	Eye-Witness Re	2010-01-12 11:1	Ave Christophe	Tequila Minsky,	5a. Structure ef	18.538328	-72.333067	TRUE	FALSE	1	1	130
10	Urgent need of	2010-01-12 13:0	Saint Martin bel	[6011] english:	2a. Penurie d'ali	18.55225	-72.33293	TRUE	FALSE	1	1	130
11	Urgent need of	2010-01-12 13:0	Saint Martin bel	[6011] english:	2a. Penurie d'ali	18.55225	-72.33293	TRUE	FALSE	1	1	130
12	Hôpital Albert S	2010-01-13 01:0	Deschapelles, H	Dr Michele Barry	7d. Services de	19.083333	-72.5	TRUE	TRUE	5	22	203
13	Is Dr. Jean Pap	2010-01-13 01:2	port of port-au-	anyone know if	5a. Structure ef	18.529096	-72.350464	TRUE	FALSE	1	1	130
14	Trapped in build	2010-01-13 01:3	Mont Joli-Turges	To anyone in th	1c. Personnes p	18.5352	-72.332954	TRUE	FALSE	1	1	130
15	Looking for 3 An	2010-01-13 01:5	Port Au Prince, I	My family and I	8e. Nouvelles de	18.539269	-72.336408	TRUE	FALSE	3	12	130

Appendix C.9: Set of filtering functions in the prototype

**Filter**

From - To (YYYY/MM/DD)

2010-01-12 - 2010-01-13

**By departments**

Ouest, Sud-Est, Nord, Nord-Est,

Nord-Est

Remove all selected departments

**By provinces**

Port-au-Prince, Croix des Bouquets, Baint,

Remove all selected provinces

**By categories**

1d. Incendie | Fire, 8. Autre | Other, 6. Natural Hazards,

Remove all

1. Urgences | Emergency

2. Urgences logistiques | Vital Lines

3. Public Health

4. Menaces | Security Threats

5. Infrastructure Damage

6. Natural Hazards

7. Secours | Services Available

8. Autre | Other

Appendix

Hello volunteer Hoàng Anh Đức! logout  
Home

#### User's details

Username  
Real name  
Email  
Password  
Retype password

C.10:

#### User's

Hello administrator Hoàng Anh Đức! logout  
Home Create a user account

#### User's details

< Previous

Username

Real name

Email

Password

Retype password

Activated

#### details

#### page

Sorry Guest, you are not allowed to access to this page!

Click [here](#) to go to the home page

## Appendix C.11: Submit report page

Hello Guest! Click here if you want to go to the home page

### Submit new report


Title

Date time Date: 2012-1-25 Time: 16:55:56

Location Latitude, Longitude: 18.773557, 72.317368

Description

Submit Cancel



Hello administrator Hoàng Anh Đức! logout

### Submit new report

Title

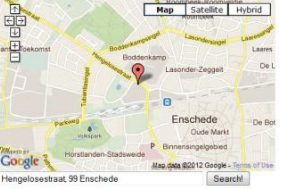
Date time Date: 2012-1-21 Time: 16:56:28

Location Latitude, Longitude: 52.223808, 6.885967

Description: This is a test description

Category: 8. Autre | Other

Submit Cancel



## Appendix C.12: Report's details page

Hello Guest! Click here if you want to go to the home page

### Report details

ID: 3604

Title: TEST

Date time: 2012-01-25 16:39:49

Location: Hengelosestraat 99 Enschede

Latitude, Longitude: 52.223808, 6.885967


Description: This is a test description

Category: 8. Autre | Other

Department: This report is not inside Haiti

Province: This report is not inside Haiti

Close



Hello volunteer Hoàng Anh Đức! logout

### Report details

ID: 3603

Title: missing person in Haiti earthquake

Date time: 2011-09-07 01:51:00

Location: Haiti

Latitude, Longitude: 18.546674, 72.323894

Description: We are yet to find Mr Raymond, his wife and 9 children, they lived in Spain but was on holidays in Haiti during the earthquake

Category: 6. Natural Hazards


Department: Overseas

Province: Port-au-Prince

Approved: True

Verified: True

Close



## Appendix C.13: Search report page

Hello administrator Hoàng Anh Đức! logout

Home Submit report Users' list

### Search report

Title/description: Use OR to search multiple title/description

Start and end date: Start (YYYY/MM/DD): 2010/01/12 End (YYYY/MM/DD): 2010/01/19

Select department to add: Nord

Select the existing one to remove: Remove all selected departments

Select province to add: Léogâne

Select the existing one to remove: Remove all selected provinces

Select category to add: Remove all selected categories

Select the existing one to remove: 1. Urgences | Emergency  
2. Urgences logistiques | Vital Lines  
3. Public Health  
4. Menaces | Security Threats  
5. Infrastructure Damage  
6. Natural Hazards  
7. Secours | Services Available  
8. Autre | Other

Approved: True ☒ False ☐

Verified: True ☐ False ☐

Search Close

Nord,

Port-au-Prince, Léogâne,

1. Urgences | Emergency, 7. Secours | Services Available, 3c. Besoins en matériels et médicaments | Medical equipment and supply needs,

## Appendix C.14: Searching results/reports list page

Hello administrator Hoàng Anh Đức! [logout](#)  
[Home](#) [Search](#) [report](#) [Submit](#) [report](#)

Showing 239 reports in 12 pages

12 pages: < 1 [2] [3] [4] [5] > >

ID	Report's title and description	Time submitted	Location
4	<b>Your Haiti earthquake photos - Washington Post</b> Photos from the Washington Post	2010-01-12 07:35:00	port-au-prince
6	<b>Photos haiti earthquake</b> RT @LATimesPhotos: PHOTOS Haiti earthquake http://bit.ly/79QDS0 #Haiti #HaitiQuake	2010-01-12 08:01:00	Port-au-Prince
20	<b>Border road down, Looting started in Pau-P</b> I received this report at 6 pm EST, from relatives now in Petion Villa, Haiti, relayed by phone to my cousin Clive d'Adesky in Miami. FROM WHAT WE	2010-01-13 03:03:00	border crossing, airport Port-au-Prince
22	<b>Parliament has collapsed</b> "President Rene Preval (...) told the Miami Herald on Wednesday that: "Parliament has collapsed. The tax office has collapsed. Schools have	2010-01-13 03:56:00	Port-au-Prince
23	<b>The tax office has collapsed</b> "President Rene Preval (...) told the Miami Herald on Wednesday that: "Parliament has collapsed. The tax office has collapsed. Schools have	2010-01-13 04:00:00	Port-au-Prince
36	<b>American Citizen broken leg - Please Help</b> Distress in Haiti (American Citizen) Please send rescue!!! Citizen is Dieula Mc Nally address Delmas 19 disuencie Rue Jeune #15 Her leg is broken sh	2010-01-13 06:20:00	Port-au-Prince
49	<b>Looking For Grand-parents</b> Looking for Emmanuella Dasque, Joseph Gabriel Dasque. US citizens. Live @ Delmas 41 # 18bis. We haven't been able to get in contact with them for 2 d	2010-01-13 07:34:00	Delmas, Haiti
51	<b>HELP ME FIND MY SISTER NADIA CADET AND HER FAMILY</b> If anyone is listening in Haiti right now please do go to: 7 VILLAGE ULDECA, DELMAS 33, PORT-AU-PRINCE My sister NADIA CADET, husband FRANTZ CADET a	2010-01-13 07:58:00	Delmas 73

23

**Title** The tax office has collapsed

**Date time** Date 2010-01-13 Time 4 : 0

**Location** Port-au-Prince

**Latitude, Longitude** 18.539167,-72.335

**Description** "President Rene Preval (...) told the Miami Herald on Wednesday that: "Parliament has collapsed. The tax office has collapsed. Schools have collapsed. Hospitals have collapsed."

**Category** 1. Urgences | Emergency, Sa. Structure effondres | Collapsed structure, 2. Urgences logistiques | Vital Lines, 3. Public Health, 4. Menaces | Security Threats, 5. Infrastructure Damage, 6. Natural Hazards, 7. Secours | Services Available, 8. Autre | Other

**Department** Ouest

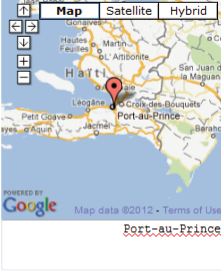
**Province** Port-au-Prince

**Approved** True ☒ False ☐

**Verified** True ☐ False ☒

[Modify](#) [Delete](#)

**Map** **Satellite** **Hybrid**



Map data ©2012 • Terms of Use

**Port-Au-Prince**

[Search!](#)



## **APPENDIX D: Usability test plan, tasks, questions and results**

### Appendix D.1 Usability test plan

#### **Information needs:**

The prototype in my research provides a map view, a temporal view (timeline), attribute view and animation with many functions such as search, zooming, filtering, user management, change map types, etc. Those views and functions are used by users to explore neogeographic data of the Haiti earthquake 2010. The main objective of the usability test is to discover what works well, what has not been used and what needs to be improved. The test will look at the problems the participants face, how fast it is to finish the tasks and how they interact with the GUI. After the test, I will be able to see from the results how they used the prototype to do the tasks, how many of them easily completed the tasks, and who did not or had difficulties to do the tasks. Moreover, from the problems they experienced, I can see what functions in the prototype could not be used or were difficult to use to do the tasks. After the test, I will give the participants a short interview about the problems they encountered.

#### **Test setup:**

##### ***Prior to arrival of test persons***

- Put the plug in the wall socket and switch on the computer
- Log on with username & password of the Research Lab
- Make sure the tasks are ready for the test person.
- Make sure the seat is in a correct position (in front of the computer)

##### ***Arrival of the test person***

- Welcome test person
- Introduce myself
- Introduce the test and equipment used.

Greet the test person and introduce myself.

Good morning/afternoon, welcome to the usability test, my name is Duc. Thank you for your participating in the usability test of my thesis's implementation. Please feel relaxed and sit down as I explain the test to you.

The purpose of today's test is to discover what works well, what has not been used and what needs to be improved in the implementation product of my MSc thesis research. There are four tasks which you will have to perform on the computer you see here and for each task there are instructions which you will have to follow.

The test is a combination of five techniques: screen logging, thinking aloud, eye tracking, video recording and an interview. As you perform the task, the computer will be recording the actions you take on the screen, and you are expected to say out loud what you are thinking, looking at and doing as you perform the tasks. You don't have to be tensed: no mark is graded for this test and your personal information is a secret after the test, just relax and be natural. During the test, this device will be tracking your eyes' movements; still there is no need to be tensed. It's just another way of collecting the data. After the test we will take a few minutes to talk about my product in the form of an interview.

Please remember that no grade is applied for this test. Instead, I'm learning from you. If things don't work out well, it's not your fault, and if you have questions please be free to ask (but don't move out of your seat).

- Turn on the laptop, turn on WAMP server, and put the prototype online.
- Set the home page of Mozilla Firefox in the test computer to website [www.hoang25746.com](http://www.hoang25746.com)



- Give the task sheet to the test person to read the Introduction to the task description, whilst I execute the steps below. Encourage the test person to think aloud already when reading the Introduction (as a kind of practicing).
- Double click on the Tobii Studio icon on the desktop
- Click “Yes” (when asked “Do you want to allow ...”)
- Choose your project Hoang25746 from the project library window
- Click “Open”
- Click on “Start Recording”
- Select the name of the participant from the list that appears
- Now, the calibration starts and for this part I will be assisted by Willy Kock or Corné van Elzakker: below are the steps they will follow:
  - Click “Continue”
  - An image appears with 2 white circles, which should be somewhere in the centre of the black rectangle. If not: ask the test person to move forward or backward or change the height of the chair.
  - Ask the test person to focus on and follow the black dot inside the red circles that are showing up. Let them keep their head still and only move the eyes.
  - Click “Start”.
  - Judge the results and recalibrate, if necessary.
  - If the results are acceptable: click “Accept”.
- Tell the test persons to put the task sheet in front of them, but don’t cover the view of the eye tracker to their eyes. And very important, tell them to not press Esc during the test.
- Once again: remind the test person of thinking aloud.
- Start timing the test.
- Now the test can really start > click “Start Recording”.
- The prototype interface appears.
- Ask the test person to execute the tasks, whilst thinking aloud (remind them during the test).
- When the test person finishes the tasks, carry on with the interview (do not stop recording).

### Interview

After the test in the interview I will ask the questions below, and some additional questions may arise depending on how they respond.

#### Task 1

1. Do you have any difficulty when you do the overview task?
2. Were there any other functions you needed to use to do the task but could not find?
3. Is there any improvement needed for overview functions?

#### Task 2

1. Do you have any difficulty when you do the zooming and filtering task?
2. Were there any other functions you needed to use to do the task but could not find?
3. Is there any improvement needed for zooming and filtering function?

#### Task 3

1. Do you have any difficulty when you do the detail-on-demand task?
2. Were there any other functions you needed to use to do the task but could not find?
3. Is there any improvement needed for detail-on-demand functions?

#### Task 4

1. Do you have any difficulty when you do the data and user management task?

## A prototype for the exploration of neogeographic point data with cartographic visualization tools

2. Were there any other functions you needed to use to do the task but could not find?
3. Is there any improvement needed for data and user management functions?
- When the interview is done: press the “Esc” button on the keyboard. Wait until the “Closing in x seconds” has disappeared.
- Thank the participant for a job well done, for their coming and participating in the test.

### ***What’s next?***

#### *Prepare for the next test*

- Close all the tabs opened by the test person.
- Open the home page of the prototype.

#### *or Go through the work done*

- Look at the recordings by going to the “Replay” tab.

#### *or End the session*

- Close the Tobii Studio programme (e.g. by clicking the white cross on red background in the right hand top corner).
- Shut down Windows and PC.
- Take the black plug out of the wall socket.
- Return the test room’s key.

## Appendix D.2 Usability test’s tasks

### **Introduction:**

Hello and welcome to the usability test for the implementation product of the MSc thesis research on “A Prototype for the exploration of neogeographic point data with cartographic visualization tools”. Supposedly, you are a United Nation’s staff who has to work with the data from reports from SMS, blog posts, email, etc... that have been sent from the victims and volunteers of earthquake that occurred in Haiti in 2010. The contents of the reports were grouped into 8 main categories and 51 sub categories (see in the categories’ list paper). A report may be considered into more than one category; it depends on the information given in the description of the report. Let’s assume that you have to analyse the information from the total number/density of reports and their details to send the support forces to places where they are needed.

Is it clear to you that what role you have to play? Do you have any questions?

Now you can open the prototype, have a look at the “read me” page to see how the maps look like, what are the components of the maps, and what the functions are supported in the maps.

Now you are going to do 4 tasks, named by functions you need in each task: overview, zooming and filtering, detail on demand, and data and user management. Before you do each task, remember to read the instruction, it will make you do the particular task easier. The tasks are on the next pages.

### **Tasks for the usability test**

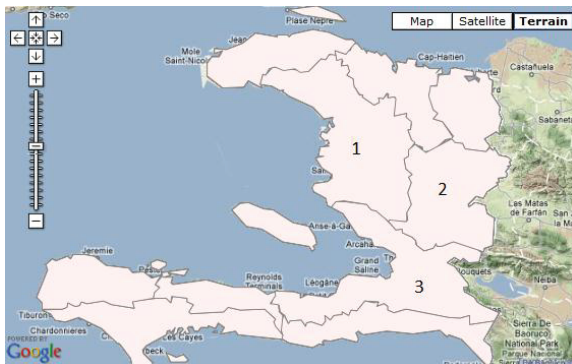
**Remember, read the instruction carefully before you start each task.**

#### **Task 1: Overview**

Q1: Haiti is subdivided into Departments. Until now (06/02/2012), in which Department, were the lowest and the highest density of reports submitted: Artibonite [1], Centre [2] or Ouest [3]?

**Instruction:** Move the mouse pointer to the text “List of geographic questions”; a list of questions will appear. Select the question that relates to the purpose of estimation amount for density of reports and can show the lowest and the highest density of sent reports. Go to the date (06/02/2012), look at areas’

colours, compare to the legend, you know where the lowest and the highest density of reports submitted were. Then, you click on the one which has lowest or highest density of reports, fill the name into the answer below.



Highest:

Lowest:

Was it easy to do this question?	Very easy	Easy	Normal	Difficult	Very difficult
----------------------------------	-----------	------	--------	-----------	----------------

Q2: On which day were more reports submitted in the whole Haiti: 13/01/2010 or 14/01/2010?

**Instruction:** Go to the list of question, select the question that can show the overview of the data. You can use the animation function to make the timeline runs to the two needed days or go to the two days by using “go to date” function. On the timeline, you can see the overview of the whole day in the middle time band. Select the date below which you think has more reports submitted.

☐ 13/01/2010      ☐ 14/01/2010

Was it easy to do this question?	Very easy	Easy	Normal	Difficult	Very difficult
----------------------------------	-----------	------	--------	-----------	----------------

### Task 2: Zooming and filtering

Q1: List 2 grid ID's of two square symbols that had more than 200 reports submitted from 2010/01/12 to 2012/02/06?

**Instruction:** Go to the list of question, select the question that leads to the map of grids network, and provide the answers to the question how many reports were submitted at a specific time period. Now you will see a grids network map. Compare the symbols' sizes to the sizes on the legend. You can zoom in to make the symbols bigger. Click on 2 symbols whose sizes are in the range of 201 and above. Write grid ID's of them into the answer.

Grid ID:                      and

Was it easy to do this question?	Very easy	Easy	Normal	Difficult	Very difficult
----------------------------------	-----------	------	--------	-----------	----------------

Q2: Each Department in Haiti is subdivided into Provinces. What are the densities of reports submitted in the two provinces Léogâne and Port-au-Prince from 2010/01/12 to 2010/02/12?

**Instruction:** Go to the list of question, select the question that related to the density of reports submitted at a specific period of time. Now you will see a map of Haiti's departments. Look at the filter functions on the right, input the requirements of the data about range of time and provinces by filling in the time period and clicking on the name of provinces. After clicking “Do filtering”, the map of the department where the two provinces is in will appear. Zoom in, the map of selected provinces will appear; click on the provinces' area. Fill in the numbers you see into the answers below.

Density of reports in Léogâne:                      and Port-au-Prince:

Was it easy to do this question?	Very easy	Easy	Normal	Difficult	Very difficult
----------------------------------	-----------	------	--------	-----------	----------------

### Task 3: Details on demand

Q1: Where and when was the first report in category “8. Other” submitted? (Provide time, and coordinates)

**Instruction:** Go to the list of questions; select the question that can provide the answer for which categories the report is in, when and where the report was submitted. Now you will see a map of the reports. Look at the legend of category “8. Other” and find the first one on the timeline. Click on that

A prototype for the exploration of neogeographic point data with cartographic visualization tools

symbol, write their details about time and position from report's information window into the answer below.

Time:

Coordinates:

Was it easy to do this question?	Very easy	Easy	Normal	Difficult	Very difficult
----------------------------------	-----------	------	--------	-----------	----------------

Q2: Compare the number of reports submitted in categories “1. Emergency” and “7. Services Available” from 2010/01/12 to 2010/01/19 in department Ouest, which category has more reports? How many for each category?

**Instruction:** Go to the list of question, select the question that can provide the answers for how many reports were submitted at a specific period of time, and the relationship between types of reports. Now you will see a map of the reports in different areas. Input the requirements for the data of time, departments and categories of the reports, and do filtering. Click on the department Ouest's point symbol which is the only one on the map. Now you will see a pie chart showing the information you need. Write them down into the answer below.

“1. Emergency” > = < “7. Services Available”

Number of reports for “1. Emergency”:

and “7. Services Available”:

Was it easy to do this question?	Very easy	Easy	Normal	Difficult	Very difficult
----------------------------------	-----------	------	--------	-----------	----------------

#### Task 4: Data and user management

Q1. Now, look at the upper-right corner of the home page, log-in with the account “Volunteer”, password “zxcvbnm1”. Try to submit a new report. After submitted, search for it again by using the detail you have just given. There might be a list of reports that have the same details as the report you submitted, click on the report that you want to see the details to review it. Check the box “verified” which is located at the end of report's details form, and modify it if that report is yours. What is the ID number of the report that you submitted?

Report's ID:

Was it easy to do this question?	Very easy	Easy	Normal	Difficult	Very difficult
----------------------------------	-----------	------	--------	-----------	----------------

Q2. Logged in as a volunteer, you can change the account's real name, email and password if you want. In the user's details page, you can access to the page that shows the list of users. See in the page, how many users are there in the list? Go to home page, click on “log out”. Now you are the guest of the prototype.

Number of users:

Was it easy to do this question?	Very easy	Easy	Normal	Difficult	Very difficult
----------------------------------	-----------	------	--------	-----------	----------------

Please rate your satisfaction of the prototype?

Very poor	Poor	Normal	Good	Very good
-----------	------	--------	------	-----------

#### Appendix D.3 Table of participants' personal information

Participant	Background	Experience with interactive maps	Experience with timeline and/or animation
1 (Female)	Temporal generalization	Use Google Maps in daily life	Work with timeline conceptually
2 (Female)	Computer science (BSc) Geoinformatics (MSc)	4-5 years	Modest
3 (Male)	Computer scientist by training	3 years	Poor
4 (Male)	- 6 year experience in .NET development and ASP.NET - Very good knowledge about development using JAVA, C++, J2ME. - Very good for ArcGis and ArcObject	Have worked with Google Maps for 2 years. Last project with Google Maps API was in Nov/2011	Poor
5 (Male)	No information	No information	No information
6 (Female)	Cartographer	Have worked since appearing until now	Modest

Appendix D.4: Summary of the correctness of the answers provided by six participants

		Person 1	Person 2	Person 3	Person 4	Person 5	Person 6	Overall	Average
Task 1	Q 1	R	R	R	R	R	R	100%	83.4%
	Q 2	R	R	W	R	R	W	66.7%	
Task 2	Q 1	R	R	R	R	R	R	100%	50%
	Q 2	W	W	W	W	W	W	0%	
Task 3	Q 1	R	R	R	R	R	R	100%	91.6%
	Q 2	R	W	R	R	R	R	83.3%	
Task 4	Q 1	R	R	R	R	R	R	100%	75%
	Q 2	R	R	W	W	R	W	50%	
Over all		87.5%	75%	62.5%	75%	87.5%	62.5%		

Appendix D.5: Summary of difficulty level experienced by participants (1-very difficult; 2-difficult; 3-normal; 4-easy; 5-very easy)

Test person	Overview		Zoom and filter		Detail on demand		Data and user management	
1	Average: 4		Average: 3		Average: 4		Average: 5	
	Q1: 5	Q2: 3	Q1: 4	Q2: 2	Q1: 4	Q2: 4	Q1: 5	Q2: 5
2	Average: 2.5		Average: 4		Average: 4.5		Average: 3.5	
	Q1: 3	Q2: 2	Q1: 4	Q2: 4	Q1: 5	Q2: 4	Q1: 3	Q2: 4
3	Average: 1.5		Average: 3		Average: 3.5		Average: 3.5	
	Q1: 2	Q2: 1	Q1: 4	Q2: 2	Q1: 2	Q2: 5	Q1: 3	Q2: 4
4	Average: 3.5		Average: 2.5		Average: 2		Average: 2	
	Q1: 4	Q2: 3	Q1: 2	Q2: 3	Q1: 2	Q2: 2	Q1: 1	Q2: 3
5	Average: 3.5		Average: 3		Average: 4		Average: 5	
	Q1: 4	Q2: 3	Q1: 3	Q2: 3	Q1: 5	Q2: 3	Q1: 5	Q2: 5
6	Average: 3		Average: 3		Average: 2.5		Average: 3	
	Q1: 3	Q2: 3	Q1: 3	Q2: 3	Q1: 3	Q2: 2	Q1: 3	Q2: 3
Overall	Average: 3		Average: 3.08		Average: 3.42		Average: 3.67	
	Q1: 3.5	Q2: 2.5	Q1: 3.33	Q2: 2.83	Q1: 3.5	Q2: 3.33	Q1: 3.33	Q2: 4

Appendix D.6: Summary of user's satisfaction (1- very poor; 2- poor; 3-normal; 4-good; 5-very good)

Test person	Satisfaction level
Test person 1	4
Test person 2	4
Test person 3	5
Test person 4	4
Test person 5	5
Test person 6	3
Overall	4.17