MSc. Computer Science for Communication Networks

Computing Project

Google Maps for Geo-localized Data

Authors:
Oscar Guillén
Patricia Reinoso

Adviser:
Éric Renault

June 22, 2017

Contents

Introduction

A smartphone application capable of capturing information about roads condition was created. It recognizes valuable information about pavement quality and pothole presence on the roads. At its current state, the application identifies these two features and stores them on a static XML file. However, a graphic representation of the data collected was lacking.

Google Maps is a map tool developed by Google. It provides an API that allows to represent geographic data in web applications. This tool was used to develop a simple web application to graphically represent the data obtained by the smartphone application. Given the fact that Google Maps uses KML files to represent geographic information, data treatment was needed to transform the XML file to the needed KML structure. Therefore, two important stages in the development process had to be considered: data processing and web application construction.

The goal of the project was to create a simple web application in order to graphically visualized the data captured by smartphone application, to master programming techniques learnt on previous modules and enforce autonomy and problem solving skills, including project management.

This report aims to explain the decisions made during the development process of the web application. First, the background, specifications and tools used are presented. Then, the development process, including all the stages, is explained. It is followed by the description of the performance tests executed, possible improvements and extensions, and conclusions.

Chapter 1

Google Maps for Geo-localized Data

1.1 Background

A geolocation-based smartphone application that allows to obtain information about roads condition was created. The final goal of the software is to display valuable information about the roads based on the data captured. Nevertheless, the information collected is stored using a static XML file that organizes the information but does not provide a mechanism to graphically visualize it. Thereby, data obtained had to be treated and represented on a simple web application using Google Maps.

1.2 Specification

1.2.1 Event types

Roads condition is described by 2 types of events: pavement quality and potholes presence, both are described below.

Pavement quality

Quality description of the road events are extended episodes that are captured during an undefined period of time. Irregularities on the pavement produce particular movements on the car (consequently on smartphones) that are captured during the displacement and that allow to determine the quality of the roads.

Quality description of the road events are identified on the XML data file by the element tag *<type 1>*

Potholes

Holes produced on pavement are known as potholes. They are very specific, high-intensity, independent events. This type of event is identified on the XML data file by the element tag *<type 2>*.

1.2.2 Event description

Both types of event are described on the XML data file using the corresponding tag by the following elements:

<ACCELERATION_LIST>

Represents the series of accelerations on the movements captured by the application. Each acceleration element (<ACCELERATION>) includes capture time, actual magnitude on the 3 axis (X, Y, Z) and accuracy. This attribute is essential for computing the intensity of the event.

<POSITION_LIST>

Represents the series of locations related to the event. Each position element (<POSITION>) includes capture time, coordinates (longitude and latitude) and accuracy. This attribute is crucial for the event representation on the map.

1.2.3 Event representation

Events that describe roads condition are displayed on a web application using Google Maps. In order to achieve that, the data source XML file was transformed into a KML file, which is a format used to specify geographic information on Google Maps. Also, each event is associated to a color to represent its intensity.

Pavement quality

They are represented on the map as paths. Therefore, each element on the position list is included on the representation. To compute the intensity of the event, an average of the accelerations is computed. The less the difference of the average with respect to the gravity, the better the quality of the road is.

Potholes

They are represented on the map as pin marks. Therefore, only the first element of the position list is included on the representation. The intensity of the event is computed based on the difference between the highest pick on the acceleration list and the gravity. The higher the difference, the higher the intensity of the event.

Considerations

On the file, many events that are located on the same position (or very closed given the accuracy matter) may be found. All of them are presented on the map. Events fusion is yet to be implemented and it is out of the scope of this project.

The transformation of the data is generated from a static XML sample file. It means that the map is not updated on real-time.

The map generated including the appropriate representation of the events can be visualized on a web browser.

1.3 Tools

The information obtained by the application was stored using a XML file, which established a particular structure for the events that had to be represented. Data had to be processed before being represented on a map.

1.3.1 Data processing

First, it was necessary to apply some transformations on the XML file and generate a KML data file. In order to accomplish it, the following tools were used:

- **XQuery:** is a language for querying, accessing and manipulating XML documents. This tool was used to access the list of accelerations of each event and calculate its intensity.

- **BaseX:** is a XQuery Processor that applies XQuery requests to XML files. Version 8.2.3 was used [1].

- **XSL (eXtensible Stylesheet Language):** is the language to specify transformations on XML files and it is capable of converting a given XML file into another XML document, such as KML. It was used to define the KML template.

- **xsltproc:** is a XSL processor that operates through the command line. It was used to apply the transformations specified on the XSL template on the XML file and obtain the KML needed.

1.3.2 Data representation

After generating the KML file, to represent data on the map the following tools were used:

- **KML:** is an XML notation created to represent geographic data using specific tag-based structures. KML files are used as source information in Google Maps and Google Earth [3].

- The web application was created using HTML, CSS and JavaScript.

- **Google Maps JavaScript API :** is an API provided by Google that supplies a set of functions and procedures to customize map visualization and access Google Map services on web applications [2]. Version 3.28 was used.

- **Google API Console:** used to register the project. It provides information, credentials and track of Google API usage.

- **Google API Key:** grants access to the Google Maps services. A generic unrestricted key was used for the development process.

- **Google Sites:** it was used to store all the files that needs to be publicly accessible. The URL of the site is: *https://sites.google.com/site/kmlfilestsp*

1.3.3 Project Management

- **Git:** the open-source, distributed version control system was used to storage and manage evolution of the application. Its branch structure was particularly useful to experiment various possible solutions and approaches of the implementation. Evolution of the project can be tracked on a GitHub repository [4].

- **Trello:** is a web application used to keep track of the progress and person responsible for each task.

1.4 Application Description

Acceleration Computation

The first step in data processing is computing the acceleration of the events to determine the intensity of each one of them. A script that uses XQuery tool was created for this purpose. Particularly, it queries the accelerations found on the original XML data file, performs some computations and returns a new XML file that includes the intensity of each event.

XLS Template

The desired format for the KML file was detailed using XSL language, which allows to create templates to specify transformations on XML files. The XSL file created has 3 templates. The first one defines the general structure and the style of the map (name, description, colors and other attributes). The other two allow to specify the coordinates and assign a given style to each event respectively.

The transformations specified on the XSL file are applied to the acceleration computation XML output file using xsltproc, a XSL processor. In this way, the KML file that is used for the representation on Google Maps is finally created. The arguments must be given in this very specific order and all of them are mandatory.

Shell Script

Given that XSL a is very limited language, it was not possible to use it to execute the calculations needed to determine the intensity of the events. Therefore, it was necessary to divide the data processing in two steps: acceleration computation and XSL transformation.

Consequently, a shell script was created to integrate them. This script receives four arguments: the XML data file, the XQuery file, the XSL template file and the desired name for the output XML file.

Application Registration and API Key Generation

Before creating the web application and the was necessary to register the application on the Google API Console and generate an API Key for authentication. This step allows to monitor the Google service usage.

The API key grants access to the Google Maps services. It was included on the HTML file as required. For the moment a generic unrestricted API Key was generated because the project is on development mode. This key can also be restricted to HTTP usage and it is even possible to specify the domains are able to use it.

Web Application

The web application is very simple and it consists on a HTML file, a JavaScript file and a CSS file. It uses the Google Maps JavaScript API, which facilitated the implementation. The application deploys the information contained in the KML file previously generated. One important requirement is that the KML file must be publicly allocated and accessible. Therefore, it was stored in a Google Site [5] to obtain a public URL. The reference to this URL is included on the JavaScript file.

1.5 Development Process

In this section it is detailed decisions made during the development process.

The project development was designed after making an analysis of the requirements. The development process aimed to successfully divide the responsibilities and activities among the members of the team. Furthermore, the task distribution and progress was monitored using the application Trello. Also, the expected time and the actual time of execution of all the task was tracked.

First, the requirements were established and analyzed to start developing the activities. An initial meeting was needed to analyze the problem and to establish the first decisions. Then, meetings were scheduled every 2 weeks to discuss about progress and propose new implementation approaches.

Several approaches were considering when implementing the application. The evolution of the project can be found into a repository. Using the version control system, Git, allowed to experiment different implementation approaches taking advantage of the branch structure. The impractical proposals remain as experimental ones within experimental branches that never merge into the master branch.

The development process was divided into main activities as follow:

XML File Analysis

The construction of the web application was executed based on a XML data file provided by the smartphone application developer. It was necessary to dedicate a meeting to discuss the file structure, and establish the representation for each type of event and the computations needed for the color representation. Nonetheless, these aspects were discussed and improved during the development cycle.

KML generation

An initial approach was reached after the XML file analysis. *xsltproc* was selected as the XML file processor for the project. Thus, the XSL template was built aiming to transform XML into KML. This first release was developed without an analysis of the acceleration information. However, the location and placemark representation of each type of event was sucessfully done using this tool. This process was divided into: study of KML files and building of XLS templates.

Acceleration analysis

After the KML experiments, it was necessary to start analyzing and computing acceleration from event types by solving the project. Thus, some applications were tried as XQuery processor, but at the end BaseX was selected. After this process, it was necessary to redefine the XLS templates to include the intensity of the events and the color representation. The acceleration computation for each event is note the most appropriate one given the fact that real data was missing.

Web application construction

The web application construction task includes the creation of the appropriate HTML, CSS and JavaScript files, the registration of the application on Google API Console, API key generation, and including the appropriate link of the KML file generated into the application (which was the most challenging part of the task because several approaches were considered).

Testing

Software performance analysis was needed and executed given the possible amount of data that it the application may have to handle and process. In consequence, it was introduced a software testing cycle aiming to perform a complete analysis of the software performance based on KML file generation. More details are included in the section 1.6.

Documentation

Documentation is an important part of software development, therefore, some time was dedicated to this task. More details can be found on section 1.7.

1.5.1 Time-Tracking

Expected time and actual time taken on each task were tracked. The expected values were inferred by the team members, however, in some cases the difference between the time values may be considerable because it took the expected results were not accomplished and other the developing had to continue using another approach. The time-tracking values are presented on the table 1.1.

1.6 Performance Testing

Only performance testing of the application has been considered. In this case, randoms event types were generated to measure how long the application takes to create the KML file. It was divided into three classes of test suites. The goal has been finding the behavior of the application performance considering each event type both separately and together. It has been timed the KML file generation to make it possible. Thus, it has been tested using:

Table 1.1: Time-Tracking

Activity	**Expected Value**	**Real Value**
XML Analysis	2h	4h
KML First Analysis	2h	2h
XSL Templates Creation	8h	10h
Acceleration Computation	8h	18h
Building Web Application	12h	17h
API Registration and Key generation	1h	1h
Testing	10h	18h
Documentation	10h	10h

- Only type-1 event.

- Only type-2 event.

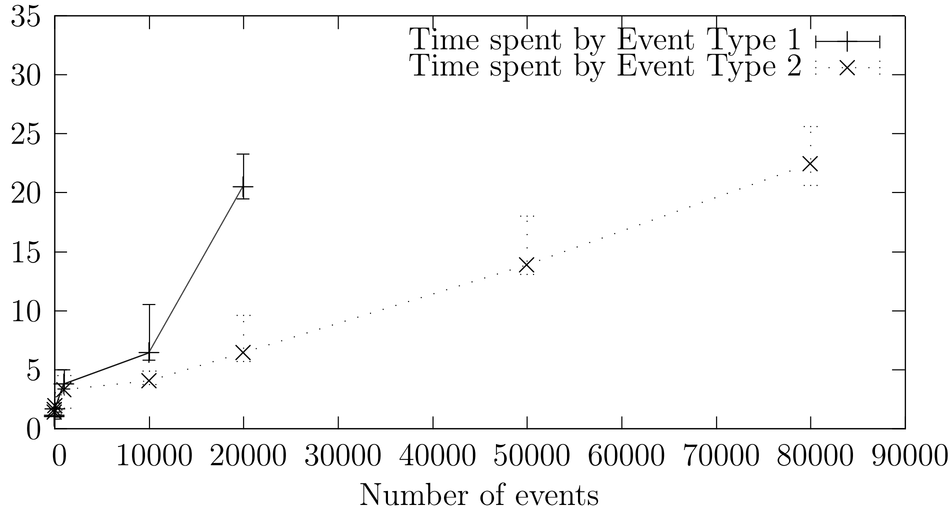
- Both event types.

For test suite performance samples, it was timed 100 times the KML file generation to obtain a better approximation of the behavior considering correct data randomly generated for the specific number of event type per sample. Then, it was calculated the mean of the 100 random samples, and the minimum and maximum values, to deploy the final behavior results by deploying the points generated and the confidence intervals per sample. Moreover, it is necessary to consider some technical details about the machine which was used to time the KML file generation:

- 512 MB of Virtual Memory (assigned by BaseX)

- CPU @ 2.50GHz x 4

Graph 1.1: Pothole and Paths Performance Testing



In the Graph 1.1, it was deployed the performance testing of KML files generation with event type 1 and event type 2 separately aiming to make a comparison between both behaviors.

The samples used were from 1 sample to 20000 samples per each event type. It was calculated the time spent generating the final KML file. It can be seen almost the same behavior per each event type, but it is clear that increase slower than event type 1. This behavior is given by computing more accelerations.

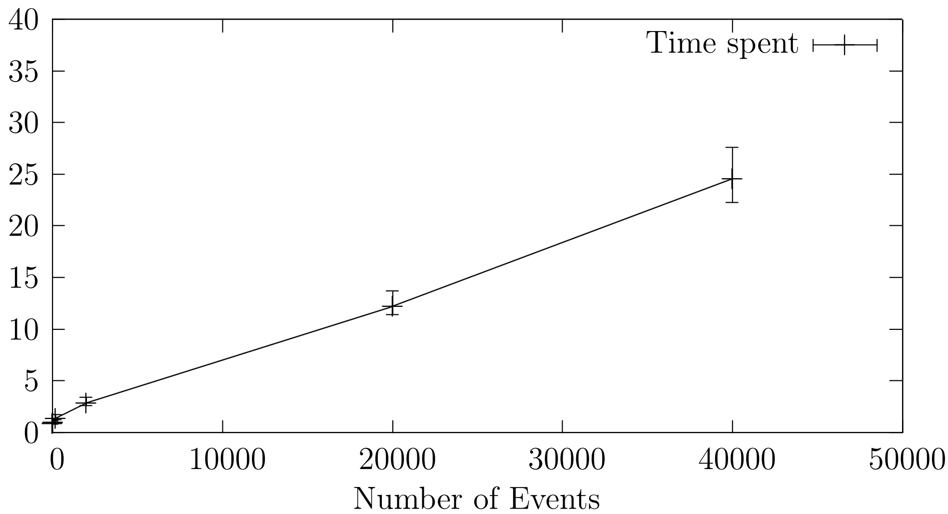
It was proved 50000 and 100000 samples for type 1 and type 2 respectively and the time increased strongly. However, 50000 samples do not return the KML files for event type 1. It was the same behavior for type 2 but using 100000 samples. It was returned an out of main memory message for both cases. Out of main memory message is reached when the virtual memory is completely used.

Thus, it is hard to compare in this case which one is less efficient than the other. But, we can induce using the out of memory error that paths analysis uses more memory than paths analysis.

Table 1.2: Integration Performance Test Suites

	Samples
(a)	1 of type 1 and 1 of type 2
(b)	10 of type 10 and 1 of type 2
(c)	100 of type 1 and 100 of type 2
(d)	1000 of type 1 and 1000 of type 2
(e)	10000 of type 1 and 10000 of type 2
(f)	20000 of type 1 and 20000 of type 2
(g)	50000 of type 1 and 50000 of type 2

Graph 1.1: Pothole and Paths Performance Testing



In the table 1.2 and Graph 1.2, it can be seen the generation of KML files with both event types together. It means pothole intensity and path quality samples. It is indeed the software performance. Out of main memory message is reached when the virtual memory is completely used.

The samples used were from 1 sample to 50000 samples per each event type. It was calculated the time spent generating the final KML file. It can be seen the same behavior that it has been found in the first graph. There is a faster increase when it is proved with 20000 samples. In this case, there was a problem with 50000 samples per event type where it was returned the out of main memory message. It is important to remind that the error appeared for 50000 samples of event type 1 in the previous test suite. Thus, this error result was expected.

The out of main memory message indicates a fail error. Because it cannot continue analyzing the XML file, needed of memory, and then it cannot generate the next files that must be used to generate the final KML file. The out of main memory messages denotes the need for more capacity, which is proportional to need more resources.

1.7 Documentation

The documentation of the project is located on the directory *Documents*. It includes the initial specifications of the project, the documentation for JavaScript code generated automatically with JsDoc tool, and the user manual created with the goal of guiding the user on the execution of the application and the developer on the generation of an appropriate KML for the map deployment, or in case of improvement or extension of the project.

1.8 Limitations

During the development of the application, some limitations were encountered. The data provided was insufficient. It only included some potholes and the paths were too short. Therefore it was necessary to generate own data to visualize the work executed.

In addition, given the absence of real data, we ignored real behavior of the events. Therefore calculations on the intensity of the events may no be the most appropriate and needs to be improved.

1.9 Future Work

This application has great potential for improvements and extension. Some of them are presented below.

The computation of the intensity of the event is really basic. For example, since the calculation of the quality of the road is based on an average value, in the case of a very long path where only a fragment has a very poor quality, it will be misrepresented on the map. It is necessary to work with real data in order to adapt the computation to its real behavior.

The interface of the web application is really simple. It can be improved to upgrade the user experience. In addition, some features such as allowing the user to specify a specific location on the map can be included.

Rather than generating a single KML file with all the events, it could be possible to generate several ones according to some predefined regions. In this way, only the files needed may be display on the web application given the user requests.

In addition, more testing can be done to application to verify the placemarks locations and to verify the computation of the intensity of the events.

One goal would be that the web application updates on real time when the smartphone application users add roads condition data.

Conclusion

We presented and detailed the software development of a geo-localized data computation project. This project provides a solution for the graphic representation of data obtained by an application that captures roads condition.

The focus of this project was the data transformation and computation, therefore the web application developed is very basic and it is only used to easily deploy the analyzed data.

We enhanced our software development skills. It was performed well-documented praxis, testing skills and coding abilities. Moreover, the team work was involved. The activity management and meetings with the adviser were essential in development cycle. Even if we had meetings and an adviser we had the autonomy to find proper solutions to the problem and to learn how to implement them.

Data limitation was a important issue in order to deal with real-world behavior. The given data that was expected to be computed was not complete enough. Furthermore, the data analysis, computation and testing process were affected by trying to find useful cases. Nevertheless, this project aims to become accessible to future works. Hence, it is opened to be improved.

Bibliography

[1] BaseX Documentation. http://docs.basex.org/wiki/Main_Page.

[2] Google. Google Maps JavaScript API Documentation. https://developers.google.com/maps/documentation/javascript/.

[3] Google. KML Documentation. https://developers.google.com/kml/documentation/.

[4] O. Guillén and P. Reinoso. Geolocalized-Data GitHub Repository. https://github.com/patriciareinoso/Geolocalized-Data.

[5] O. Guillén and P. Reinoso. Google Maps for Geolocalized Data Google Site. https://sites.google.com/site/kmlfilestsp.