



**Germany**

**Institute for Geoinformatics and Remote Sensing**

Designing and implementing a web-based Network Controlling System (NCS) for automated real time routing service over the web, based on open source technologies: a case study for Tehran

A thesis submitted to the  
Faculty of computer science and mathematics  
Of the  
University of Osnabrueck

In partial fulfillment of the requirement for the Doctorate degree  
(PhD) in Geoinformatics

Javad Sadidi  
2013  
Supervisor: Prof. Dr. Manfred Ehlers

## **Dedication**

To my wonderful wife and beloved parents

Despite the difficulties that they were suffering during my PhD

They continue to set an example of patience and human dignity

I give my thanks to Sepideh who never gave up to help and protect me during these years despite of all problems specially, donating the young time of her life that I should have spend with her for working on this thesis.

## Acknowledgement

All praises is due to God, who enabled me to complete my PhD as the highest official academic degree.

First and foremost, I would like to thank my supervisor Prof. Manfred Ehlers who I owed all my academic life to him. He accepted me as PhD student in his Institute and helped me much more I expected. During these years, he acts a moralist teacher as well as an academic teacher for me. He provided me with unflinching encouragement and supported me in different ways. He provided me with constant and valuable feedback and support. The guidance that he offered perfectly balanced freedom and motivation to explore and experiment on the one hand, and support for me in keeping on the right track on the other.

Maik Wannik the computer administrator of my institute to offer me the hardware needed for this research.

I would like to thanks all my colleagues in the institute for Geoinformatics and Remote sensing at university of Osnabrueck. Specially, Ihab Hijazi and Kai Behnke for introducing the basics of the research to me.

I am deeply indebted to my parents. Where I am today is no small part due to their encouragement and support, I thank my mother Batoul, my father Ali Akbar and also my brother and sisters for their constant prayers and good wishes.

I debt to my wife, Sepideh Abdolmaleki beloved teacher, has suffered homesick throughout my PhD. She has endured the distance between us during these years with patience.

I would like to thanks my best friends Dr.Mohammad Baaghideh and Morteza Nouri to protect me during these difficult years.

Finally, I would like to give my deep appreciation and thanks for all those wonderful people who have influenced and shaped my vision and ideas over the years and who are too numerous for me to list individually.



# Contents

<b>List of figures</b>	<b><a href="#">5</a></b>
<b>List of tables</b>	<b><a href="#">6</a></b>
<b>Abstract</b>	<b><a href="#">7</a></b>
<b>1 Introduction</b>	<b><a href="#">11</a></b>
1.1 Statement of the problem .....	<a href="#">13</a>
1.2 Objectives of the research .....	<a href="#">14</a>
1.3 Scientific contribution of the research /research process .....	<a href="#">15</a>
1.4 Organization of the thesis .....	<a href="#">16</a>
<b>2 State-of-the-art for web-based routing and navigation services</b>	<b><a href="#">17</a></b>
2.1 Google Maps direction API (Application Programming Interface).....	<a href="#">18</a>
2.2 TomTom .....	<a href="#">19</a>
2.3 OpenTripPlanner (OTP) .....	<a href="#">21</a>
2.4 MapQuest .....	<a href="#">21</a>
2.5 Travelling Salesman .....	<a href="#">22</a>
2.6 Open Source Routing Machine (OSRM) .....	<a href="#">23</a>
2.7 Literature review .....	<a href="#">24</a>
<b>3 Conceptual frame work</b>	<b><a href="#">27</a></b>
3.1 Foundations of the World Wide Web .....	<a href="#">27</a>

3.2	Web application technologies .....	<a href="#">28</a>
3.2.1	Server-side technologies .....	<a href="#">28</a>
3.2.1.1	Web application servers .....	<a href="#">28</a>
3.2.1.2	Server scripting languages .....	<a href="#">29</a>
3.2.2	Client/browser- side technologies .....	<a href="#">32</a>
3.2.3	Data exchange formats .....	<a href="#">35</a>
3.3	Web GIS .....	<a href="#">36</a>
3.4	Networks .....	<a href="#">37</a>
3.5	Routing services and the designed NCS .....	<a href="#">38</a>
3.6	Theory of routing algorithms used in this research .....	<a href="#">38</a>
3.6.1	Dijkstra .....	<a href="#">39</a>
3.6.2	Dijkstra for routing .....	<a href="#">41</a>
3.6.3	Shortest path AStar .....	<a href="#">41</a>
3.6.4	AStar for routing .....	<a href="#">42</a>
3.6.5	Shortest path ShootingStar .....	<a href="#">42</a>
<b>4</b>	<b>Methodology</b> .....	<a href="#">45</a>
4.1	Research material .....	<a href="#">45</a>
4.1.1	Platform .....	<a href="#">45</a>
4.1.2	Web Application server .....	<a href="#">47</a>
4.1.3	Data Base Management System .....	<a href="#">47</a>
4.1.4	Data .....	<a href="#">48</a>
4.2	Implementation of the simple routing algorithms over the web to find the best route by open source software (OSS) and Open Specifications (OS). .....	<a href="#">48</a>
4.2.1	Preliminary stage.....	<a href="#">49</a>
4.2.2	Browser client stage for the simple routing service .....	<a href="#">52</a>
4.2.2.1	Designing the preview .....	<a href="#">52</a>
4.2.2.2	Programming the designed preview on the web .....	<a href="#">52</a>
4.2.3	Server-side programming stage for the simple routing service .....	<a href="#">57</a>
4.3	Designing the real-time web-based NCS .....	<a href="#">64</a>
4.3.1	Introduction .....	<a href="#">64</a>

4.3.2	Designing the conceptual architecture of the NCS .....	<a href="#">66</a>
4.3.3	Programming and executing components of the designed architecture .....	<a href="#">70</a>
4.3.3.1	Programming the security protected administrator page .....	<a href="#">71</a>
4.3.3.2	Programming text live search engine for database changes ..	<a href="#">73</a>
4.3.3.3	Programming vectorial live search engine .....	<a href="#">75</a>
4.3.3.3.1	Vectorial search engine on the client-side .....	<a href="#">77</a>
4.3.3.3.2	Vectorial live search engine on the server-side...	<a href="#">77</a>
4.3.3.4	Closing process .....	<a href="#">80</a>
4.3.3.4.1	Closing process on the client-side .....	<a href="#">80</a>
4.3.3.4.2	Closing process on the server-side .....	<a href="#">81</a>
4.3.3.5	Opening process .....	<a href="#">82</a>
4.3.3.5.1	Opening process on the client-side .....	<a href="#">82</a>
4.3.3.5.2	Opening process on the server-side .....	<a href="#">82</a>
4.3.3.6	Sending and removing the vectorial warning .....	<a href="#">83</a>
4.3.3.6.1	Vectorial warning on the client-side .....	<a href="#">83</a>
4.3.3.6.2	Vectorial warning on the server-side .....	<a href="#">84</a>
4.4	Automatic live correction of the best found route .....	<a href="#">84</a>
4.4.1	Load Average parameter .....	<a href="#">85</a>
4.4.2	Live correction methods .....	<a href="#">86</a>
4.4.2.1	Submit-Correct method .....	<a href="#">86</a>
4.4.2.2	Interval -Execution method .....	<a href="#">87</a>
4.4.2.2.1	Programming Interval- Execution on client-side ..	<a href="#">87</a>
4.4.2.2.2	Programming Interval- Execution on server-side ..	<a href="#">89</a>
4.4.2.3	Check-Change-Correct method .....	<a href="#">89</a>
4.4.2.3.1	Programming Check-Change-Correct on the client-side .....	<a href="#">91</a>
4.4.2.3.2	Programming the Check-Change-Correct on the server-side .....	<a href="#">94</a>
4.4.3	Assessment of the correction methods and selecting the most efficient algorithm .....	<a href="#">95</a>

4.4.3.1	Monitoring the load average parameter for Interval-Execution method .....	<a href="#">95</a>
4.4.3.2	Monitoring the load average parameter for Check-Change-Correct method .....	<a href="#">98</a>
4.4.4	Comparison correction methods and selecting the best one .....	<a href="#">98</a>
4.5	Integrating all components together to serve the real time routing service over the World Wide Web .....	<a href="#">99</a>
<b>5</b>	<b>Results and Conclusion</b>	<a href="#">103</a>
5.1	Summary of the results and discussion .....	<a href="#">104</a>
5.1.1	Conceptual model of the NCS .....	<a href="#">104</a>
5.1.2	Description of the NCS .....	<a href="#">104</a>
5.1.3	Advantages of the NCS .....	<a href="#">106</a>
5.1.4	Evaluation between the methods for web- based live route correction and introducing a new web based live route correction method ..	<a href="#">107</a>
5.2	Conclusion .....	<a href="#">108</a>
5.3	Future work .....	<a href="#">110</a>
	<b>References</b> .....	<a href="#">111</a>



## List of figures

3.1	Web application technologies: web browsers communicate with the server and database via web programming languages. Data exchanges act as brokers .....	<a href="#">30</a>
3.2	Traditional model for web applications (left) compared to the Ajax model (right) (adopted from: <a href="http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications">http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications</a> ).....	<a href="#">34</a>
3.3	The research architecture as a webGIS system including the data, logical and presentation tiers .....	<a href="#">37</a>
3.4	Dijkstra's routing algorithm a network whose arcs have been labeled with distance between two nodes it is connecting .....	<a href="#">39</a>
3.5	Dijkstra algorithm selects the smallest candidate distance as a solved node .....	<a href="#">40</a>
3.6	Dijkstra algorithm, the stage just before the final stage .....	<a href="#">41</a>
4.1	Schematic preview of the simple routing service on the user browser side.....	<a href="#">54</a>
4.2	The result of implementing the simple routing service on the browser over the web .....	<a href="#">64</a>
4.3	Conceptual architecture of the NCS .....	<a href="#">69</a>
4.4	Entry page for the administrator on a Google Chrome browser .....	<a href="#">73</a>
4.5	Designed text live search engine .....	<a href="#">75</a>

4.6	Designed vectorial live search for the NCS .....	<a href="#">79</a>
4.7	Designed workflow diagram as conceptual architecture of Check-Change-Correct method .....	<a href="#">90</a>
4.8	The load average values graph for the correction methods.....	<a href="#">99</a>
4.9	Workflow diagram of the NCS to conduct the real-time routing service over the web .....	<a href="#">101</a>
4.10	The final web-based real-time routing service over the web.....	<a href="#">102</a>
5.1	load average values for the one minute period for 10 simultaneous users	<a href="#">108</a>

## **List of tables**

4.1	Load Average values for Interval-Execution .....	<a href="#">97</a>
4.2	Load Average values for Check-Change- Correct .....	<a href="#">97</a>

# Abstract

Nowadays, web-based applications are being developed every day. Web services play a main role in mutual information dissemination. Web-based routing services provide the possibility to use a routing service without needing to install professional spatial information software. Nor is it necessary to spend time gathering data, updating spatial data or running complicated algorithms. By using a web-based routing service, a client just opens a browser and defines the start and destination points. Then, the web-service can show the best route to the client. There are a number of free web-based routing services such as Google Maps, however, these services do not exploit a real time and dynamic routing services. The inability for current routing services to provide up to the second route information, offering automatic live correction with proprietary routing methods for classified users along with local database controlling facility for local administrators, can be a debilitating weakness for emergency and security personnel.

This research aims to develop current routing services to real time web-based service using open source technologies. The resulting program, Network Controlling System (NCS) updates and warns online users about closed or reopened parts of the network and corrects the found route whenever any update has happened. It is notable that all the mentioned processes happen automatically. NCS also exploits a live vectorial engine for segment controlling.

To implement and run the NCS, the following steps were taken:

- Running a simple routing service over the web according to the PgRouting project (FOSS4G2010) (<http://pgrouting.org/>) instruction
- Designing the conceptual architecture of the NCS
- Formulating the conceptual architecture to the programming codes

- Suggesting and evaluating new methods to offer real time capability to the routing system
- Integrating the NCS into the simple routing service
- Piloting the program in Tehran (capital of Iran)

After running a simple routing project without real time and controllable capabilities, the conceptual architecture of NCS was defined and programmed including the following components:

- A security protected administrator page as the entry web page for the administrator side.
- A text live search engine to search the target part of the network and submit updates.
- A vectorial search engine to find and locate the target part and its peripheral environment in larger scale of auxiliary information on other map servers like Google Maps/satellite/ terrain.
- Live vectorial warning to inform online users about closed parts and the ability to then re-open the after the closed street or segments.
- Live correction of the found route for online users. This facility would correct the found route whenever any update is happened for example a street or a subset of street is closed or reopened.

To program the conceptual architecture, the NCS was divided into two distinguishable parts: client-side and server-side. open source Technologies including open source Software and Open Standards web programming languages were used to program and run the model on the server. AJAX, JQuery, OpenLayers, GeoExt, ExtJS, JavaScript and HTML5 programming languages on the client-side along with PHP, SQL and PostGIS languages on the server-side were used to formulate the conceptual architecture of the NCS. GeoJSON and XML languages also work as exchange formats to act like brokers between the client and server-side. Ubuntu 12.04 (2013) was installed on the server as the platform and PostgreSQL server along with PostGIS and PgRouting extensions installed as relational database management system.

NCS integrated into the simple routing service offers designated real time service equipped with proprietary routing methods for classified users (from the simple routing service FOSSG2010 Pgrouting project), such as law officials and emergency personnel, who have authority to disregard traffic laws and restrictions.

To offer the routing live correction system in this research, three methods were designed, programmed and implemented. Then, a test was done to select the best method in which the least calculation capacity of the server is consumed. The evaluation results show that Check-Change-Correct method (designed and programmed for the first time in this research) imposes the least load average values to the server and subsequently, is able to serve more simultaneous users compare to other methods.

Finally, the research introduces a mobile controllable and real time routing service over the web, which can be used to decrease travelling time. This is very important in cities like Tehran, which suffers from a daily average of 182 accidents (<http://www.jamejamonline.ir/newstext.aspx>, 28/3/ 2012), voice and chemical pollutions.



# CHAPTER 1

## Introduction

The emergence of the World Wide Web has opened up a new world. Different types of information are being disseminated by transmission devices all over the world in a fraction of a second. A user in an unknown city can open a mobile device such as a cell phone and be able to use a routing service to find the best and shortest direction to the desired destination. Web services like these, in contrast to web sites which are usually isolated and closed to external software systems, are considered open in that they provide programming interfaces that can be accessed by other web applications. "Geospatial web Services leverage the power of GIS, programming components and the World Wide Web to serve an expanding variety of needs" (Fu & Sun, 2010). Some years ago, people believed that the information and data should be up to date but nowadays, the expectations and definitions have changed. Indeed, in the explosion of the information age, response must be up to second rather than merely up to date. Hence, the need arises for real time services.

"The Real-time web is a set of technologies and practices that enable users to receive information as soon as it is published by its authors, rather than requiring that they or their software check a source periodically for updates" ([http://en.wikipedia.org/wiki/Real-time\\_web](http://en.wikipedia.org/wiki/Real-time_web)).

Being real time in many affairs has a significant place in an individual's chance for success. As the Persian proverb goes:

"A stone thrown at the right time is better than gold given at the wrong time". In other words, by designing real time services like web-based routing service, the spent time can be saved.

Routing service is a branch of spatial web services and is used to find the best direction among the possible ways between the start and destination points based on routing algorithms. The algorithms used in this research are Dijkstra, AStar and ShootingStar.

This research aims to develop a current routing service for real time web-based services, using open source technologies. The resulting program, Network Controlling System (NCS) updates and warns online users about closed or reopened parts of the network and corrects the found route whenever any update has happened. It is notable that all the mentioned processes are designed to happen with few resources.

To program and implement a real time routing service over the web, the client-side programming languages, the server-side languages and the exchange programming scripts were necessary. We also needed a platform as operating system, web application server and a Relational Database Management System (RDBMS). Programming languages are considered as standards. The platform, web application server and RDBMS were used as software. This research was programmed and implemented using OS (open standards) and OSS (open source software). Open source technologies need more programming and subsequently are more complicated than license technologies. We have hidden the complexity of the system behind the service on the server and offer the user or administrator a friendly face. The system is designed so that a user and even an administrator of the system with only basic reading and writing literacy can use and control the network.

It is notable that designing a service over the web using open source technologies is much more complicated than license holder programs. For open source technologies, the programmer must do everything by writing commands and line-by-line programming and has to debug and run the written codes on the server to evaluate the practicality of the programs. Hence, to understand this research, a long learning curve in different programming standards including HTML, JavaScript, OpenLayers, GeoExt and Extjs in the client-side, PHP, SQL and PostGIS on the server-side, XML and GeoJSON as exchange formats have to be taken into account. The prerequisites of open source programming languages and open source technologies along with the conceptual basics of the research will be discussed in Chapter 3.



## 1.1 Statement of the problem

An integrated system obtained from the simple routing service with NCS, described in this research, offers a new and unique system for real time routing compare to current routing systems like Google Maps or TomTom. These routing services are being used for routing and car navigation, however, they are not real time services in the sense of an NCS. Another disadvantage of these systems is that the user does not have the possibility to select the proprietary routing algorithm. The third problem in the current systems is the lack of warning and up-to-second routing corrections while the user is on his/her way. In other words, when a problem occurs and a part of the network must be closed due to an accident, destruction, subsidence, landslide etc., there is no warning and real time correction facility. This is particularly dangerous for emergency personnel, who need to be timely and efficient. The correction system must be executed automatically and the system must be able to find a new alternative best route by considering the newly closed or reopened parts. Following is a summary of the disadvantages in the current routing services.

1- The current systems do not offer special routing algorithms for classified users. For example, police, emergency services or firefighters use the same route as normal users. Using the simple routing service equipped by NCS, gives the possibility to have a real time service (from NCS) with proprietary routing methods (from the simple routing service, FOSS4G2010 PgRouting workshop). For example, police and other officials have the authority to supersede traffic laws and take the shortest path in order to get to the target location as soon as possible. Public users have to respect traffic laws like turn restrictions, one-way routes etc. Hence, public users select heuristic methods to visit restriction like turning restriction, turning and traffic lights.

2- The current systems do not exploit a management system to control the network and keep it in real time. In current systems, the network data are imported into the database and clients use this data until new data is imported. Imagine an accident that happens in a street. The police would want to close a small part of the street until the problem is solved. After this, the closed part would be opened again. Using the designed system in this research (NCS), the police act as administrator and could open their mobile set on the accident scene,

close the street or a subset of it as a segment, solve the problem and then re-open the closed segment. The NCS is a proprietary system to control and manage the traffic by local officials. Current systems may offer a routing service in a city like Tehran but, for example, the police of Tehran are not entitled to access the Google database for control and management to close a segment for some minutes and reopen it.

3-Current systems do not exploit a live correction system. When the administrator closes a street or a special part of a street, online users have already selected their route. The NCS exploits a correction algorithm introduced for the first time in this research, in which all the found best routes by online users are corrected. The system removes the past route, calculates the new alternative direction by considering the closed parts and redraws the new route based on network updates like the parts closed by the administrator. This action is also repeated whenever the closed parts are opened.

4- The lack of a vectorial live search engine to search, display and locate a particular segment of a street is another disadvantage for the current systems. The vectorial live search engine enables the administrator and users to find a small segment and its peripheral environment on a map, satellite image and terrain like on Google maps/satellite/terrain. Without this service, finding and locating the small segments by the administrator is very time consuming and against the real time requirement.

## **1.2 Objectives of the research**

Recognizing the necessity of real-time and accessible services, this research has the following objectives:

1. Design of the conceptual NCS architecture and programming the NCS to develop a new, free, open source, web-based, real time routing service using open standards to offer a new controllable unique service different with the current systems.

2. Design of a vectorial live search engine over the web to find and locate small vectorial segments.
3. Development of correction methods and evaluation of these methods, design of a new efficient conceptual model, programming and executing the model (Check-Change-Correct method as the most efficient model) to implement the live correction for the calculated route.
4. Aggregate all parts of the system as a real time, manageable and rapid system by gathering all three before mentioned objectives to offer a very simple system for both users and administrator who controls the system. The designed system has to be so simple that everybody with just simple reading and writing literacy can use and even control the system. The controlling section of the system must be designed to be compatible with low speed internet and mobile devices. The administrator can control the network and offer a real time service even when he/she is in the street using a mobile device.
5. Enhance the efficiency of emergency services like police, emergency ambulances and fire fighters using integrated NCS with the simple routing service which offers different routing algorithms and live correction system for classified users including heuristic edge-based algorithms to simple algorithms that just search for the shortest path without considering restriction criteria.

### **1.3 Scientific contribution of the research /research process**

The research addresses micro-scale conceptual architectures for real time web-based routing services. More specially, it designs a new efficient model for live correction of routing algorithms to offer a light-weight real time service for online users. This issue is very important for serving spatial data in the web era because an integrated system contains spatial data processing, real time updating and correction systems and needs a huge level of processing capacity and a high speed internet connection. This research also contributes to design a vectorial live search engine for segment monitoring on the web.

First, a new efficient model for live correction is designed. This can be used for similar field of research based on networks analysis such as water network, pipelines, or facilities management.

Second, it develops a conceptual model for the design of a light segment controlling system for local administrators to give authority for closing and opening special segments of the target streets when a problem causes closing a part of a network.

Third, a vectorial live search engine is designed and programmed.

Finally, as a free and open source project, the theoretical framework is open and can be used for other research fields.

#### **1.4 Organization of the thesis**

Open source programming requires explicit system and programming knowledge which is to be part of the research. Hence, these developments have to be included in the Ph.D thesis. Chapter 1 gives a brief introduction into the research as well as the scientific relevance of the work. Chapter 2 provides a comparative study between current routing services and a review of similar projects. Chapter 3 is dedicated to conceptual issues including definitions and exploration into the different programming languages used to design, program and implement the research. Chapter 4 gives an overview of the methodology, programming and execution of the developed service. Chapter 5 presents results and discussion. This chapter compares the result of the research to the predefined objectives and discusses new frontiers for developing this research in the future.

# Chapter 2

## State-of-the-art for web-based routing and navigation services

Routing services are used to find the best route between places (considering distance and the cost). For a new and non-expert user, it is difficult to define which service is more suitable for a particular use, especially when the user is going to buy a service. According to Melby's definition (Melby, 2012), some criteria should be considered for evaluation of the efficiency in routing services as the following:

- Coverage area along with the accuracy
- Time of calculation
- Capacity to storage the routing data including maps
- User friendliness

The above mentioned factors apply for a general computer-based routing service and do not consider web-based systems. For web-based routing in addition to the mentioned criteria, the following items should be considered:

- Real time capability to inform users about the updates like temporary barred directions
- Auxiliary facilities to accelerate indicating the target parts for time saving and better decision making (for example, different types of search engines including text and vectorial search engines)
- More importantly a live correction algorithm to correct the found route for online users automatically whenever an update happened
- Large numbers of simultaneous users should be served by the system
- Response time to the requested tasks

There are a number of web-based routing systems including open source and commercial services growing to be compatible with mobile devices. Each of them contains particular advantages and disadvantages. This chapter intends to discuss the most commonly used web-based services with respect to their characteristics and disadvantages.

Nowadays, many routing services are web-based oriented because offering a real time service is only possible via an online network. The most known and commonly used routing services are discussed in the following:

## **2.1 Google Maps direction API (Application Programming Interface)**

"The Google Directions API is a service that calculates directions between locations using an HTTP request. You can search for directions for several modes of transportation, include transit, driving, walking or cycling. Directions may specify origins, destinations and waypoints either as text strings (e.g. "Chicago, IL" or "Darwin, NT, Australia") or as latitude/longitude coordinates"

(<https://developers.google.com/maps/documentation/directions/>). It is notable that the API can be embedded into other websites for developers without the need to go to the Google Maps website and set the target place every time. The Google Maps service contains a large number of advantages: hosted on the strongest servers distributed all over the world, high level security protected by skillful personnel, enhanced by satellite images, photos, weather and other useful auxiliary information. Nevertheless, the following deficiencies prevent Google Maps to be the high level practical model that can be used for Tehran:

1. Generally, Google direction service has been designed for static and not for real-time routing. For example, on daily average, at least 182 accidents happen in Tehran and police must close and reopen the related streets or segments. During this time, the users would be misled using a static system. In addition to accidents, other factors may influence on traffic controlling and force police to close a specific part of the network. Hence, the need arises to add a controllable real time service by officials like the police. Imagine that an accident has happened and police have to close a street and reopen it whenever the problem solved. Is there any connection between

the police of Tehran and Google's proprietary database to exercise control for a city like Tehran?

2. There is no module to serve proprietary routing methods for classified users. For example, police, emergency services and firemen who have authority to neglect the traffic laws and restrictions in order to save people's lives, need a different algorithm in which just the shortest path is considered and other parameters like traffic laws and restrictions are not considered. In contrast, non-authorized users have to respect all traffic laws and restrictions with a different algorithm.
3. A warning system to inform online users about the network updates like closed streets.
4. A user friendly and simple controlling service containing a route live correction system for local administrators to feed updates into the system. For example, a police present in an accident scene must be able to close the related street and reopen it whenever the problem is solved. Then, the system must be updated, a warning has to be sent for online users and the calculated route must be automatically corrected

Google Maps direction service is a free static routing service provided for almost all countries and is therefore applicable for large cities like Tehran. For some areas especially in the U.S, additional live traffic information is available but for Tehran this facility is not currently accessible.

## **2.2 TomTom**

TomTom NV (Euronext: TOM2) is a Dutch company offers navigation services for car, motorcycle and truck drivers. TomTom offers also real time traffic information to get the most accurate direction. "TomTom real time traffic creates a vivid picture of traffic conditions as they evolve, keeping drivers in control of their journeys" ([http://www.tomtom.com/en\\_gb/licensing/products/traffic/real-time-traffic/](http://www.tomtom.com/en_gb/licensing/products/traffic/real-time-traffic/)). According to the TomTom official website (<http://www.tomtom.com/lib/doc/licensing/RTTHT.US.pdf>), it offers three types of real time traffic information:

1. "HD Traffic: provides precise locations and delays caused by congestion on the road network, allowing routing programs to provide the fastest route based on actual current travel times".
2. "HD Flows: delivers a real time, detailed view of traffic speeds on the entire road network, designed for easy integration into traffic management systems or calculating current routing travel times".
3. "HD Route Time: is a turnkey solution providing highly accurate real time travel and delay times for a specific route either on a temporary basis or for permanent solutions".

The TomTom company claims that "Tom Tom provides the world's most accurate and most extensive coverage, and the highest update frequency for real time traffic information" ([http://www.tomtom.com/en\\_gb/licensing/products/traffic/real-time-traffic/](http://www.tomtom.com/en_gb/licensing/products/traffic/real-time-traffic/), 2013).

Compared to Google Maps direction service, TomTom navigation and routing service offers up-to the minute real time information. There are, however, still some disadvantages of the TomTom traffic system that this research is aiming to solve:

- TomTom does not cover Iran.
- Although it offers real time information and warnings about traffic conditions it does not contain a controlling system for each city that can be used by local administrators like the police
- It does not exploit a live correction system for online users when an administrator closes a part of the network because of an accident or other factors as well as when he reopens it
- TomTom does not offer a system that gives the possibility for different type of users to select proprietary routing methods. In TomTom, all type of users have to select the same routing algorithm



## 2.3 OpenTripPlanner (OTP)

OTP is an open source project mainly written in JavaScript and Java technologies. "OpenTripPlanner (OTP) provides an open source platform for multi-modal and multi-agency journey planning across systems of any size or complexity. OTP allows users to combine information about pedestrian, bike and transit journeys via a built-in web interface and for third-party applications by leveraging a public Application Programming Interface (API) " (<http://opentripplanner.com/vendors/>, 2013). Dijkstra and AStar routing algorithms are the core functions for finding the route but in large graphs, they run slowly (<http://wiki.openstreetmap.org/wiki/OpenTripPlanner>).

In 2008, Robert Geisberger from the University of Karlsruhe (Germany) represented a new method called Contraction Hierarchies, to speed-up on large graphs (Geisberger, 2008).

Multi-modal trips, travel time and elevation profile for biking are some of advantageous features offered by OTP (<http://wiki.openstreetmap.org/wiki/OpenTripPlanner>, 2013).

While the OTP routing service exploits a large variety of facilities, the following features not offered by OTP and are considered as OTP disadvantages (like TomTom and Google):

1. A controllable system for local officials like police for temporary closing or reopening a part of network
2. A live correction system to correct the best route found by online users whenever any update happens
3. A selective system that offers possibilities for different classified users to select proprietary routing methods.

## 2.4 MapQuest

MapQuest is similar to Google maps but it is based on OpenStreetMap data. "Using latitude/longitude pairs, request a multi-point route for up to 50 locations, receive maneuver narratives, road shield images, turn direction images, street names, static maneuver maps, route shapes in raw or compressed lat/long formats, and set options to avoid ferries, highways, toll roads, or even down to avoiding specific links in the road network. The route

can be retrieved as shortest, fastest, pedestrian or bicycle" (<http://developer.mapquest.com/web/products/open/directions-service>).

MapQuest can be embedded into proprietary websites as an Application Programming Interface (API). While MapQuest offers a free of charge web based routing services enhanced with leading application suitable for round trips, it does not offer the following services for clients (similar to the other discussed web-based routing services):

1. MapQuest uses OpenStreetMap database as a not real time system. Hence, MapQuest cannot be considered a real time service
2. There is no possibility to select proprietary routing algorithms for classified users
3. It lacks a controllable system for local administrators to offer temporary real time updates
4. It offers no live correction system to correct the calculated route for online users whenever an update happens

## **2.5 Travelling Salesman (TS)**

Travelling Salesman is a Java-based multi-platform routing and navigation program for OpenStreetMap data. Almost all parts of Travelling Salesman work as plug-in and can be replaced. GPS support, address support routing and navigation, map and map rendering, plug-in and tool-integration support are considered as offered features by Travelling Salesman (TS). Beginning with version 0.9.8, TS obeys turn restriction and national traffic rules when driving through/into other countries. Traffic Message Channel (TMC) is a work in progress to consider traffic obstructions in routing by TS project ([http://wiki.openstreetmap.org/wiki/Traveling\\_salesman](http://wiki.openstreetmap.org/wiki/Traveling_salesman)).

Although TS is being developed by TMC application and contains unique and interesting features but compared to the envisioned NCS (designed in this research for real time routing) it suffers from the following disadvantages:

1. This service must be installed as software on the client machine. Each time the client opens the service it must download the updates and map changes. To use TS in other

counties, firstly, the client has to download the maps which is usually a time consuming process especially, with low speed internet connections

2. The service does not prepare proprietary routing methods for classified users
3. There is no live correction system to correct the calculated route for online users
4. There exists no controlling system for local administrators
5. There is also no vectorial warning and vectorial search engine to inform users and locate the small-scale parts on a map or image server

## **2.6 Open Source Routing Machine (OSRM)**

The Open Source Routing Machine (OSRM) written in C++ is another open source application for large networks in continental scales. It uses Open Street Map data to implement routing ([http://en.wikipedia.org/wiki/Open\\_Source\\_Routing\\_Machine](http://en.wikipedia.org/wiki/Open_Source_Routing_Machine)). OSRM covers Iran territory and contains Tehran city for a high performance routing. HTML 5 provides the possibility to recognize the current location of clients. In spite of Travelling Salesman (TS) routing service, OSRM is a browser-based system in which updates are implemented on the server-side so that users do not have to worry about the latest updates. Although OSRM offers a high speed routing service with an almost worldwide coverage it does not offer the new features added in this research for a full real time service:

1. Vectorial warning and vectorial search engine to warn and locate the target part on a map/image server.
2. Live correction system
3. Controlling system for local administrators

## 2.7 Literature review

To date, a large number of projects have been carried out with open source technologies to create web-based spatial information systems. Venneman (2009) prepared a web-based OSS GIS. He used OpenLayer together with Mapbender, MapFish and Ka-Map software on the client-side with MapServer and GeoServer on the server-side to create interactive web-based geospatial web services. Raghavan et al. (2002) prepared a web system for landslide analysis and warning in Japan. They used OSS tools such as GRASS and PostgreSQL to prepare a web-based landslide analysis system for managers. Anderson and Moreno-Sanchez (2003) built a web-based spatial information system using Open Specification (OS) and OSS to prepare a web-based system for landuse planning. They showed that the use of OSS and OS has a number of benefits for web mapping, especially in terms of the costs. Roberts et al. (2009) designed a spatial information tool for land managers in Australia and tested it successfully with land managers and extension staff in Victoria, Australia. Their results are applicable in countries where there is a high level of farm computer ownership, where relevant spatial information is available in GIS format, where government makes spatial information available to the public, and where there is pressure for increased environmental awareness and improved decision making by land managers. Li et al. (2007) prepared a web-based OSS system for forest management. Using OSS and UMN MapServer, they prepared a web-based system to access spatial forest and travel information of Xixia County coming from two different data servers via a standard web browser to promote spatial forest information sharing. Miller et al. (2010) used FLOSS AJAX based framework (GeoExt, ExtJS and OpenLayers), WMS-C (TileCache, Mapnik) and WFS (FeatureServer) server architecture. They developed an original and efficient solution for visualization of a very large amount of spatial data on the web using open source technologies combining tiled raster maps and vector WFS data in a case study for Zagreb. They developed a service for fast web mapping solution for large scale city maps over the web. Lan et al. (2012) designed a framework to support monitoring of the web services status. Their designed framework effectively integrates geospatial web services with status interfaces and dynamic monitoring technology. Experiments show that this framework can effectively monitor execution information and remind users of bottlenecks thus providing the foundation for further

improvements in the model's execution efficiency. Chuanrong et al. (2005) create a web service for real time geospatial data sharing for time-critical application. They investigated the role of web Feature Services (WFS) and Web Map Services (WMS) to develop a solution for real-time geospatial data sharing at the feature level over the web. Their case study prototype has been implemented to query, extract, create, delete, update, and map geographic features stored in web-accessible and simple feature data stores for transportation emergency applications. Their results show that the OGC standards WFS and WMS play important roles in real-time geospatial data sharing and exchange from heterogeneous sources at the feature level for time-critical applications. WFS and WMS eliminate time-consuming data translation and facilitate reuse of existing geospatial data over the web.

However, the focus of above mentioned research lies predominantly on the creation of web-based geospatial services and rarely on real time services. They provide, however, a general framework for the NCS research. To our knowledge, there is no similar study on upgrading current routing services to real time services and providing live correction feeds for best routing over the web.



# CHAPTER 3

## Conceptual framework

Based on one of objectives of the research that is designing an easy-to-use NCS without the need to learn a complex program, the theoretical issues are explained here. It is a well-known fact that one needs complex programming to make software simple which usually is accompanied by a steep learning curve. This chapter presents the context of this research. It includes the conceptual framework that everybody has to know to understand the theory of the research and for implementing it on the web. It should be noted that only the concepts related to this research will be explained and non-related cases will be ignored. To achieve the objectives of this research, a long list of programming languages and web basics have to be learned and applied.

The Internet is defined as "a global computer network providing a variety of information and communication facilities, consisting of interconnected networks using standardized communication protocols" (<http://oxforddictionaries.com/definition/Internet>).

### 3.1 Foundations of the World Wide Web

HTTP (Hypertext Transfer Protocol), the URL (Uniform Resource Locator) and HTML (Hyper Text Markup Language) are three protocols of the web invented by Tim Berners-Lee, and are now international standards offered by W3C (Fu & Sun, 2010).

During this research, HTTP requests are used in text live search engine, vectorial search engine, and vectorial warning, whereas AJAX (Asynchronous JavaScript and XML) requests are used for database updating, implementing routing algorithms, live correction, etc. Generally, for all tasks that need to communicate with the server via the client-side, HTTP requests and, subsequently, HTTP responses are used.

URL (Uniform Resource Locator) is the web address and HTML can be considered a scripting language in the client-side.

### **3.2 Web application technologies (WAT)**

This section covers basic conceptual issues for the server-client relationship and deals with the implementation as an open source project on the web (figure 3.1). It has to be mentioned that describing all WATs is outside the scope of this research. Hence, only the technologies that are needed for programming and also implementation of the conceptual design are described.

#### **3.2.1 Server-side technologies**

As shown in figure 3.1, server-side technology is composed of two subdivision parts including web application servers and server scripting-programming languages as described in the following sections.

##### **3.2.1.1 Web application servers**

"Web application servers host the websites that the users can access and provide data for other applications. They are responsible for accepting HTTP requests from clients and serving clients with HTTP responses. Since they comply with HTTP specifications, web application servers know how to communicate with web clients" (Fu & Sun, 2010). PHP is also vastly used to communicate with servers. In this research, PHP and SQL were used to run SQL queries and to build the security protected web site for the administrator to control the network. Server scripting languages used in this research will be explained in details in web standards section. The most commonly used web application is Apache (<http://httpd.apache.org/>).

Apache Tomcat and IIS (Internet Information Services, formerly Internet Information Server) are the other two web application servers. The first one is an open source



programming of Java Servlet and JSP (Java Server Pages) and the latter one (IIS) is a web server application on Windows servers for .Net frame work.

### **3.2.1.2 Server scripting languages**

Server-side scripting languages are technologies to communicate with servers. These technologies run users requests on the server and deliver the result to the client-side. PHP, Java EE (Enterprise Edition), Servlet, JSP and JSF (Java Server Faces), ASP.Net, SQL (Structured Query Language) and .Net Mobile are the most commonly used server scripting languages. PHP and SQL are the famous scripts in open source era used to implement this research.

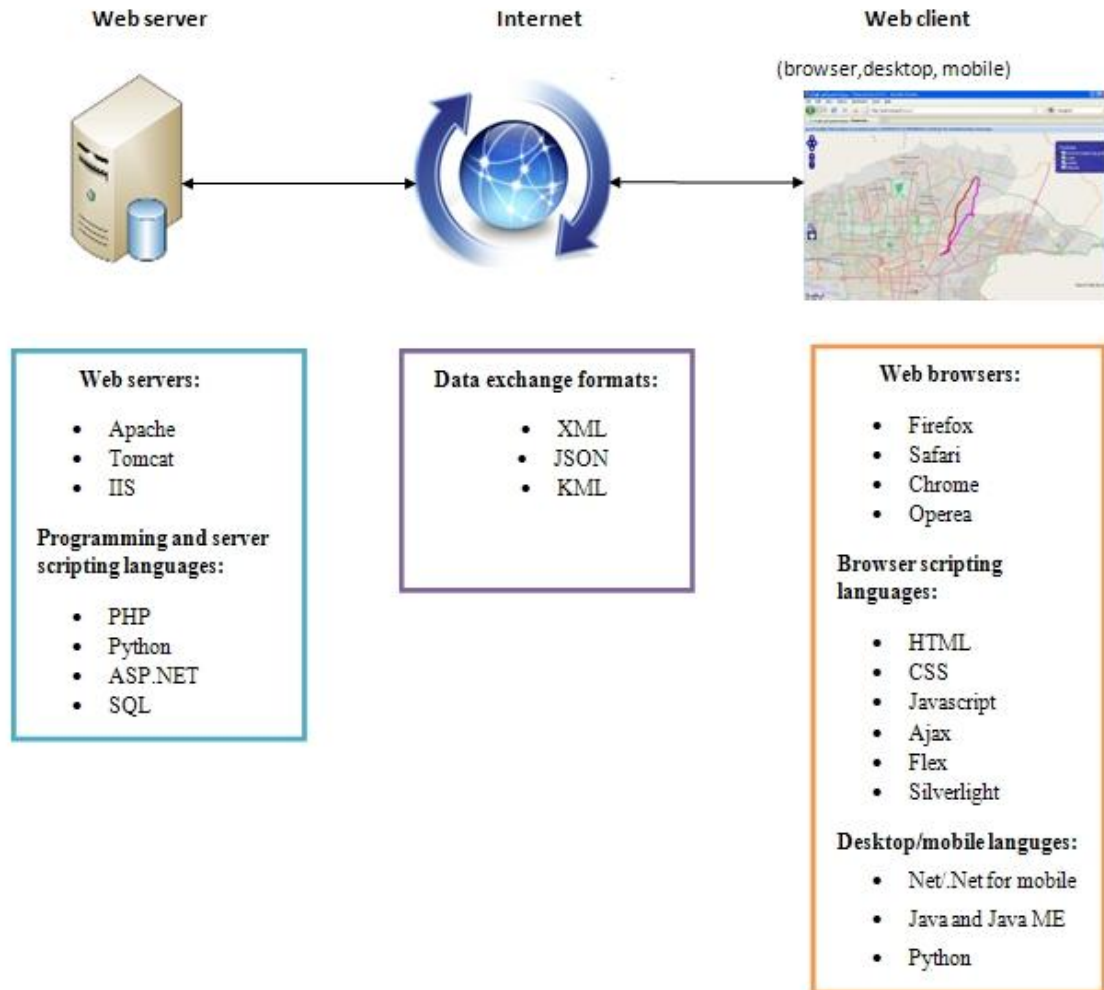


Figure 3.1: Web application technologies: web browsers communicate with the server and database via web programming languages. Data exchanges act as brokers (adopted from webGIS: principles and applications (p.29), Fu & Sun, 2010, ESRI press).

- **PHP** stands for Hypertext Preprocessor. It is a server-side scripting language, executed on the server and supports many databases like MySQL, Postgresql, or Oracle. PHP can be embedded inside HTML files. It is used to connect to databases, perform queries and secure them by a vast domain of capabilities such as PHP sessions. In this research, PHP is used to create a dynamic secure controlling system to manage the network, communicate with the database, performing queries inside PostgreSQL on the server and transforming the result into a GeoJSON vector layer. PHP is platform and web server application independent and could be installed and

used with them as an important extension. Combination of PHP and SQL-PostGIS offers great advantages in web-based spatial information systems like the present research. "The real power of PHP comes from its functions. It has more than 1000 built-in functions" ([http://www.w3schools.com/php/php\\_functions.asp](http://www.w3schools.com/php/php_functions.asp)). It should be noted that PHP covers a vast domain of functionality ranging from loops to functions and forms, sending emails, sessions and cookies to combine with other technologies like AJAX and SQL for connecting to databases. PHP was used in the following issues to design, program and implement the NCS:

- 1- Programming the security protected controlling and management service for the administrator.
- 2- Programming and running the routing function on the server and delivering the result as a GeoJSON vector layer whenever the client-side requests it. PHP implements this job by using SQL, PostGIS and GeoJSON programming languages. PHP puts the result inside the OpenLayers Store.
- 3- Querying the closed segments/streets and sending a vectorial warning in GeoJSON format. PHP implements this job by using SQL, PostGIS and GeoJSON programming languages.
- 4- PHP is used along with AJAX technology to program the text live search engine. This is used to search the street/segment names from the database, submit the found name to close or open a street/segment to update the database for offering a real time routing service.
- 5- PHP is also used to program a vectorial search engine. By this facility users and administrator are able to search, display and locate a segment as a subset of the target street along with its peripheral environment on map services like Google Maps.
- 6- PHP transmits a continuous time interval message to the AJAX-JQuery functions to announce the client-side about the latest updates done into the database. This process is used to offer live correction facility for the previous found route by online users. In fact, live correction facility programmed in this research as a part of NCS, corrects the found route and redraws the new best route for users when a street/segment is closed or opened by the administrator.

A detailed discussion of each PHP application in this research can be found in the next chapter. Figure 3.1 shows the connection between the client-side, server-side and exchange formats as brokers.

- **SQL** (structured query language) is for accessing and manipulating databases. SQL is an ANSI (American National Standard Institute) standard. SQL is widely used in this research to perform almost all PHP tasks related to the PostgreSQL database.

### 3.2.2 Client/browser-side technologies

"Web clients include a variety of programs and applications that can run inside or outside of the browser, including desktop and mobile applications" (Fu & Sun, 2010). The client-side languages used in this research are explained as the following:

- **HTML:** "HyperText Markup language: a set of standards, a variety of SGML (Standard Generalized Markup Language), used to tag the elements of a hypertext document. It is the standard protocol for formatting and displaying documents on the World Wide Web" (<http://dictionary.reference.com/browse/html>). The current version of HTML is HTML 5. One the most interesting APIs of HTML in spatial information era is the Geolocation API. This API is used to get the geographical position of a user. JavaScript has a vast domain variety of libraries including: OpenLayers, ExtJS and GeoExt.
- **JavaScript:** "is the world's most popular programming language" ([http://www.w3schools.com/js/js\\_intro.asp](http://www.w3schools.com/js/js_intro.asp)). JavaScript offers a large domain of capabilities from object orient programming to time interval functions. JavaScript also contains a large variety of libraries including: OpenLayers, ExtJS and GeoExt. JavaScript characteristics used in this research are the following:
  - JavaScript adds interactivity to HTML

- JavaScript can react to events. For example, in JavaScript, there is a function that provides a response when a user clicks on a certain text
  - JavaScript is used to secure web pages. In this research, for example, it was used to encrypt the routing page by preventing mouse right click. We will continue to use encryption functions while the PhD thesis has not been published
  - JavaScript is used to validate forms before submitting them to the server. This capability relieves the server from extra processing
  - JavaScript is used to perform codes according to the browsers. For this, it must have browser detection capability. In this research this capability was used to run live search with all major browsers plus the old versions of Internet Explorer
  - JavaScript is used to store and retrieve information on the system as cookies. For example, it can be used to store browser activities to keep track of websites that have been visited by users on this computer and the history of the uses.
- **OpenLayers:** according to OpenLayers official web site definition, "OpenLayers makes it easy to put a dynamic map in any web page. It can display map tiles and markers loaded from any source. OpenLayers has been developed to further the use of geographic information of all kinds. OpenLayers is completely free, open source JavaScript, released under the 2-clause BSD License" (<http://openlayers.org/>). OpenLayers is widely used in this research as a client-side programming languages. It has been used along with EXTJS and GeoEXT to prepare a rich set of functionalities to display and run the NCS over the web in the client browsers.
  - **ExtJS:** "ExtJS is one of the most popular JavaScript user interface framework" (<http://www.codeclouds.com/community/what-is-extjs-and-why-do-we-care/>). In this research it uses to create dynamic applications using technologies like AJAX.

- **GeoExt:** "GeoExt brings together the geospatial know how of OpenLayers with the dynamic user interface of Ext JS to help developers build powerful desktop style GIS apps on the web with JavaScript" ([http://www. geoext.org/](http://www.geoext.org/)). GeoEXT plays a key role to implement this research, especially the vectorial functionalities of GeoEXT provide the potential to draw and request vector formats on the browser.
- **AJAX:** Ajax stands for Asynchronous JavaScript and XML. "AJAX is the art of exchanging data with a server, and updating parts of a web page - without reloading the whole page" (<http://www.w3schools.com/ajax/default.asp>). Garrett (2005) has described the following technologies that are incorporated to create Ajax: HTML or XHTML and CSS displaying dynamically, XML for exchanging, and XSLT for asynchronous data retrieval using XMLHttpRequest and JavaScript brings all the before mentioned standards together in one package to consider as AJAX technology.

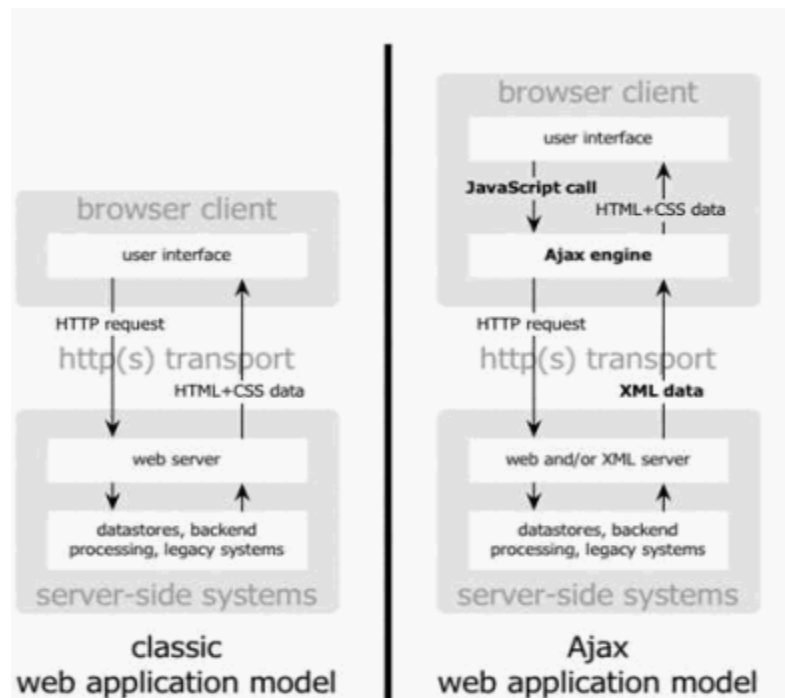


Figure 3.2: Traditional model for Web applications (left) compared to the Ajax model (right) (adopted from: <http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications>).

AJAX plays a key role in web-based real time systems like this research. As mentioned before, AJAX is able to send a request and display the result without refreshing the web

page. Hence, for live correction, sending warning to the users for closed or opened streets, text live search engine, vectorial search engine or, in general, for all real time systems AJAX is the best selection. For example, in this research AJAX is used to implement live applications like live search of street names, showing the closed parts of streets and removing opened parts as a live theme in GeoJSONformat without needing to refresh or reloading the web page.

- **JQuery:** "jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, JQuery has changed the way that millions of people write JavaScript" (<http://jquery.com/> ). To implement the live correction facility in this research, JQuery along with AJAX was used to check database updates and to request the new route for clients.

### 3.2.3 Data exchange formats

The main exchange formats used to implement this research are described as the following:

- **XML:** Extensible Markup Language "is for structure, transport and storage of data whereas HTML is for display" ([http://www.w3schools.com/xml/xml\\_what\\_is.asp](http://www.w3schools.com/xml/xml_what_is.asp)). For example, Open Street Map project uses XML to define the data structures, encoding relation, bounding box of the covered area, users, etc. In this research XML was used to define the structure, transfer and storage of data in the database. In fact, OSM data were stored in textual XML format and used to push and store the data into our database.
- **GeoJSON:** "is a format for encoding a variety of geographic data structures. A GeoJSON object may represent a geometry, a feature, or a collection of features. GeoJSON supports the following geometry types: point, linestring, polygon,

multiPoint, multiLineString, multiPolygon, and geometrycollection. Features in GeoJSON contain a geometry object and additional properties, and a feature collection represents a list of features" (Butler, 2008). In fact, GeoJSON is JSON that supports spatial data structures. GeoJSON is widely used in this research as the main exchange format. The server delivers vector format request in GeoJSON format. Since GeoJSON is used to deliver the calculated best route, vectorial warning, corrected route and the vectorial search engine results to the user browser, the conceptual concepts about the GeoJSON structure is explained in much more detail in the next chapter.

### **3.3 Web GIS**

"Web mapping is the process of designing, implementing, generating and delivering maps on the World Wide Web and its product. Web GIS is similar to web mapping but with an emphasis on analysis, processing of project specific geodata and exploratory aspects" (Fu & Sun 2010). A web GIS enriches a web project by adding GIS capabilities. Similar to other web GIS systems, the architecture of this research consists of three tiers including data, logical and presentation tiers (see figure 3.3).



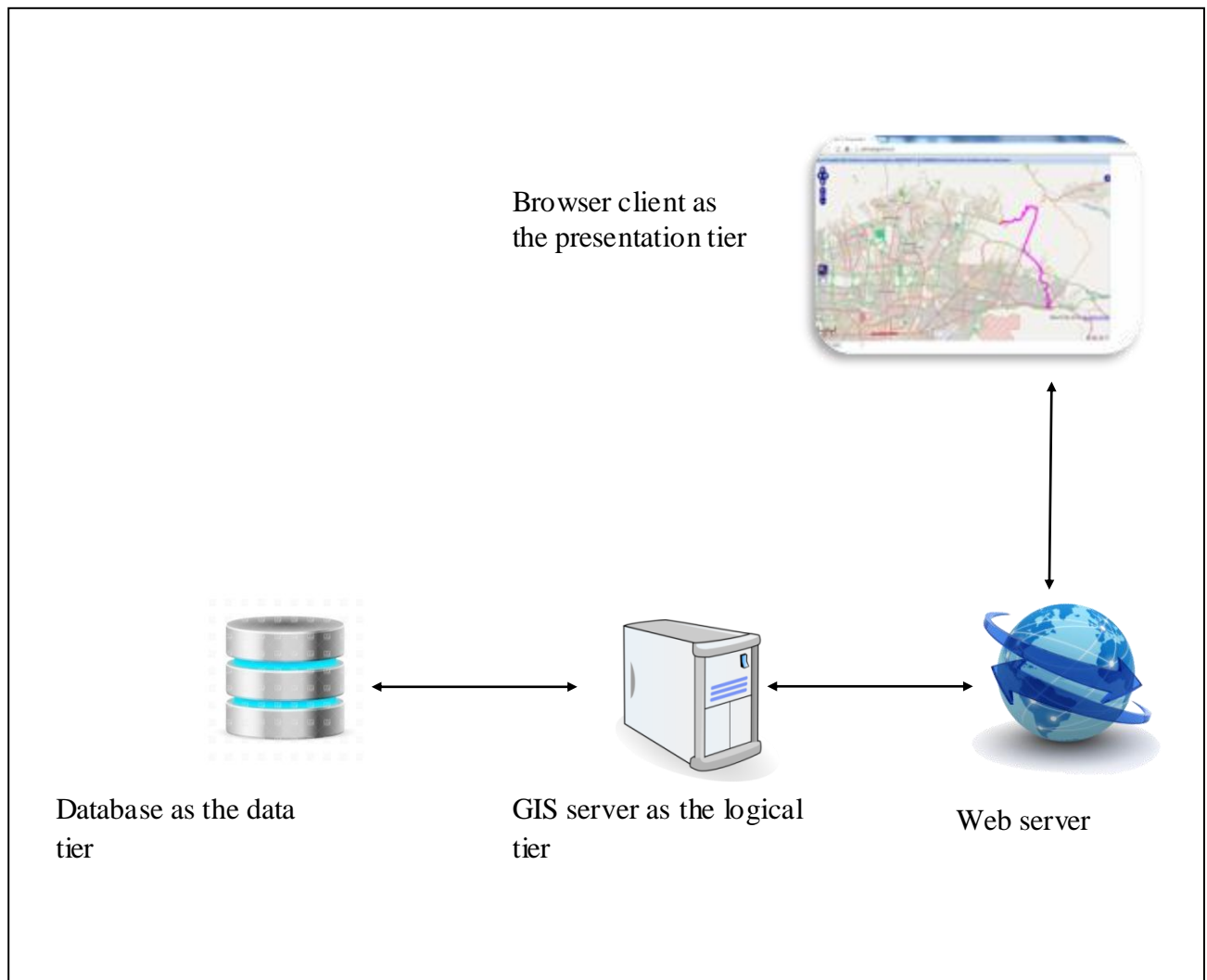


Figure 3.3: The research architecture as a web GIS system including the data, logical and presentation tiers

### 3.4 Networks

According to Fischer's (2003) definition, there are three types of networks described in the following:

- 1- Pure network: only topology and connections are included.
- 2- Flow network: represents topology and flow characteristics like path costs.

- 3- Transportation network: "is a flow network representing the movement of people, vehicles or goods".

Normally, network data are pure. Hence, to use the data within a database and implement network analysis capabilities like in this research, it is necessary to create topology for relationships by defining and introducing the network characteristics for a special task (here, for routing). By this action, we transform the pure data to a transportation network.

### **3.5 Routing services and the designed NCS**

Routing services can be implemented on the operating systems via special software on personal computers for a standalone user or implemented as a web service for a large number of users. This research addresses the latter case to offer a real time web routing service.

In fact, the core of this research is running a web service for real time routing and a system to control and manage this service (routing network) to gain real time capability. We have named it network controlling system (NCS). This research has concentrated on the design of an NCS with no learning curve for clients to have a real time routing web service. For background and understanding of the necessary concepts for this research, we will – at first – introduce routing methods followed by the employed programming languages and standards related to designing the NCS.

### **3.6 Theory of routing algorithms used in this research**

Before discussing routing algorithms, it is necessary to define two most common approaches to routing problems. Routing problems are classified in two approaches: the mathematical approach and heuristic approach (Hart & Nilsson, 1968).

"Heuristic refers to experience-based techniques for problem solving, learning, and discovery. Heuristic methods are used to speed up the process of finding a good enough solution where an exhaustive search is impractical" (<http://www.princeton.edu/~achaney/tmve/wiki100k/docs/Heuristic.html>). There are three

algorithms used in the simple routing service run in this research to find the best route: Dijkstra, AStar and ShootingStar. These algorithms can be selected by the client via a combo box. Since the current research is to develop and implement a routing service to find the best real-time route via the mentioned algorithm, algorithms play an important role in this research scenario. Hence, they are described in more detail in the following section.

### 3.6.1 Dijkstra

This algorithm was invented by Edsger Dijkstra in 1956. This algorithm considers  $n$  points and creates the minimum cost tree that provides the minimum distance between points (Dijkstra, 1959).

To find the shortest path between start and destination points (as shown in the figure 3.4), the following steps are implemented:

- 1- Once the start and destination point are selected (source (start) point is O and destination is T), it assumes that all points as unsolved.

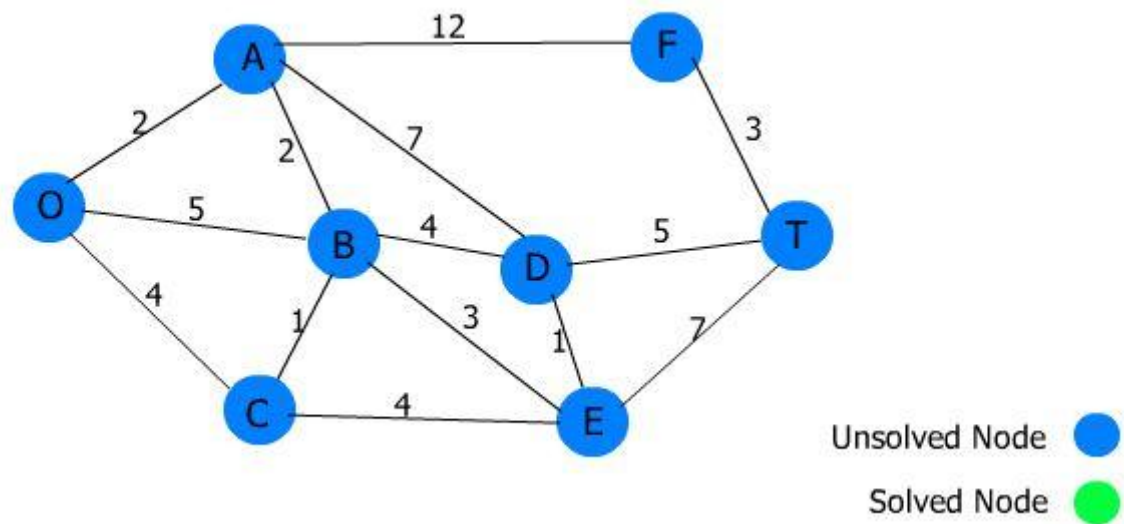


Figure 3.4: Dijkstra's routing algorithm, a network whose arcs have been labeled with distance between two nodes it is connecting (adopted from <http://www.sce.carleton.ca/faculty/chinneck/po.html>).

- 2- In the next step, it initializes the start point as solved and labels it zero because its distance from origin is zero.
- 3- It identifies all unsolved points connected to any solved nodes and for each connection of the solved or unsolved node, it calculates the candidate distance (as shown in figure 3.5) and selects the shortest candidate distance, sets the point related to it as a solved node (candidate distance is: distance to the solved node + length of arc).

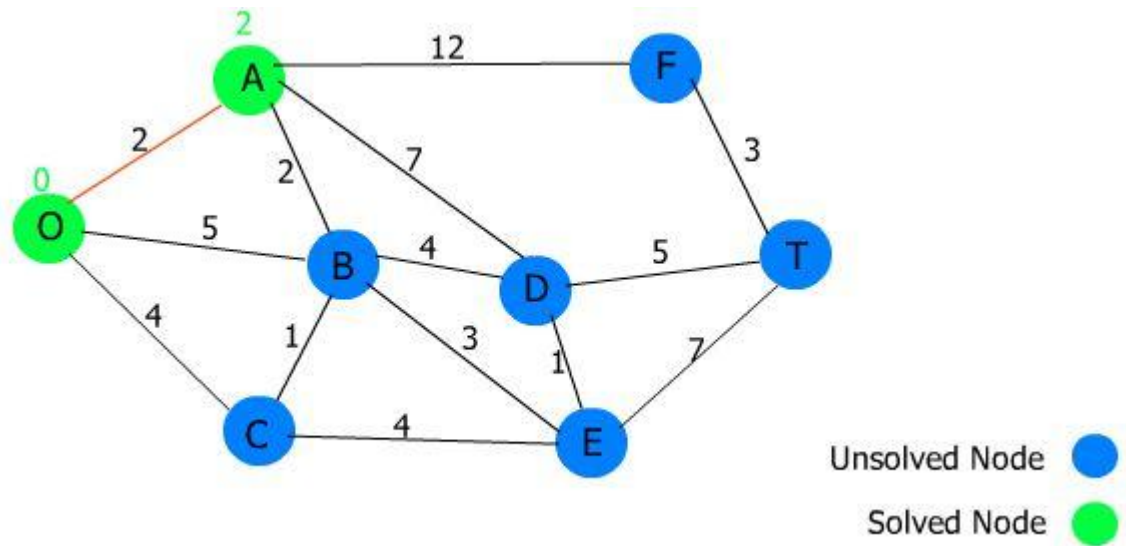


Figure3.5: The Dijkstra algorithm selects the smallest candidate distance as a solved node (adopted from <http://www.sce.carleton.ca/faculty/chinneck/po.html>).

- 4- It identifies all unsolved nodes directly connected to a solved node and calculates the candidate distance for all connections and selects the shortest candidate distance as a solved node, like in the former stage. If there are two or more candidate distances with the same length, one will be selected arbitrarily.
- 5- The process will be continued until the destination point is reached (figure 3.6 shows the stage just before the final step to get to the destination).

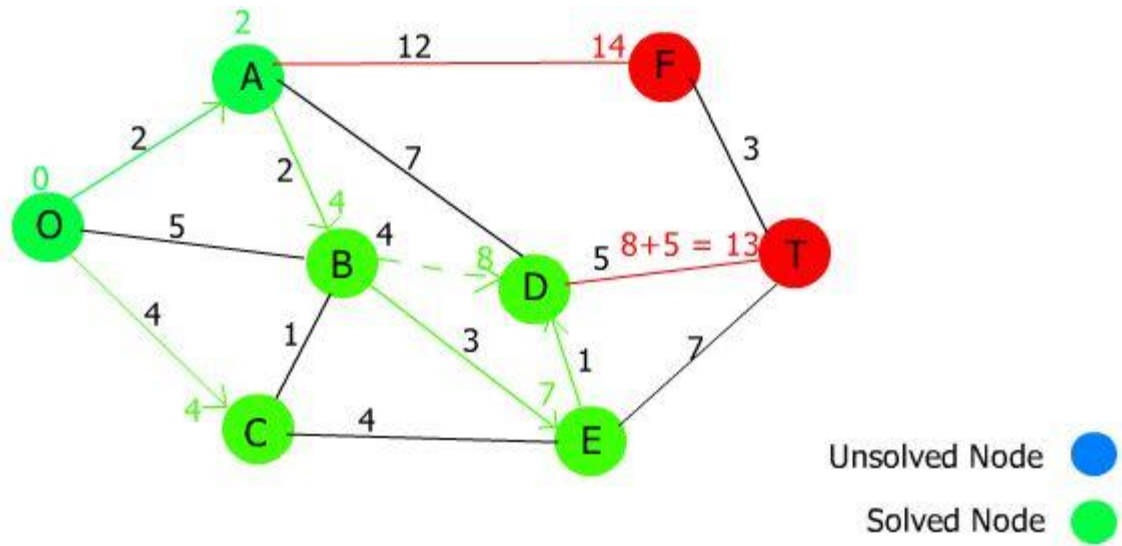


Figure 3.6: Dijkstra algorithm: the stage just before the final step (adopted from <http://www.sce.carleton.ca/faculty/chinneck/po.html>).

### 3.6.2 Dijkstra for routing

"The Dijkstra algorithm was the first algorithm implemented in PgRouting. It does not require other attributes than source and target ID, attributes and costs. It can distinguish between directed and undirected graphs" (<http://www.pgRouting.org/docs>, 2011).

Dijkstra has the core and wrapper function for routing. Core function is the base of each algorithm and can be enriched by the wrapper functions. Wrapper functions enhance the core functions with transformations, bounding box limitations, etc. It should be noted that Dijkstra's algorithm is a mathematical approach and not a heuristic algorithm.

### 3.6.3 Shortest path AStar

"AStar is the most popular choice for path finding, because it's fairly flexible and can be used in a wide range of contexts. AStar is like other graph-searching algorithms in that it can potentially search a huge area of the map. It's like Dijkstra's algorithm in that it can be used to find a shortest path. It's like Greedy Best-First-Search in that it can use a heuristic to

guide itself. In the simple case, it is as fast as Best-First-Search" (<http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html>).

The AStar algorithm was introduced by Hart, Nilsson and Raphael (1968). "It uses a distance-plus-cost heuristic function (usually denoted  $f(x)$ ) to determine the order in which the search visits nodes in the tree. The distance-plus-cost heuristic is a sum of two functions: the path-cost function, which is the cost from the starting node to the current node (usually denoted  $g(x)$ ) and an admissible "heuristic estimate" of the distance to the goal (usually denoted  $h(x)$ )" (<https://code.google.com/p/astar-search/>).

Chertok (2010) explained AStar method in detail at the following web address (<http://www.untoldentertainment.com/blog/introduction-to-a-a-star-pathfinding-in-actionscript-3-as3-2>).

#### **3.6.4 AStar for routing**

"AStar algorithm is another well-known routing algorithm. It adds geographical information to source and target of each network link. This enables the routing query to prefer links which are closer to the target of the shortest path search" ([http://workshop.pgrouting.org/chapters/shortest\\_path.html#shortest-path-a](http://workshop.pgrouting.org/chapters/shortest_path.html#shortest-path-a), 2013).

The difference between Dijkstra and AStar in path routing is that Dijkstra uses just the real cost (for example the distance) between the nodes while AStar uses two factors: the real cost and the approximate cost from each node to destination point because it is a heuristic function.

AStar exploits a cost function to define the difficulty of passing a node and time required to reach to a node (<http://theory.stanford.edu/~amitp/GameProgramming/MovingObstacles.html>).

#### **3.6.5 Shortest path ShootingStar**

"The special feature of ShootingStar is link to link (edge based), not node based as Dijkstra and AStar algorithms do. This makes it possible to define relations between links for

example, and to solve some other vertex-based algorithm issues like "parallel links", which have same source and target but different costs" ([www.pgrouting.org](http://www.pgrouting.org)).

For routing, ShootingStar algorithm also uses two new parameters to extend its efficiency: Rule parameter that describes a rule for turning restriction and To\_cost parameter that shows the cost of a restricted passage can be very high in a case of turn restriction or comparable with an edge cost in a case of traffic light). To run ShootingStar algorithm we have to assign an edge to edge passage cost (as a special attribute of our adjacent edges structure or as a cost of the line graph) which represents any kind of limitations or penalties for moving from one edge to another like turning restrictions or traffic lights ([www.pgrouting.org](http://www.pgrouting.org)).

Like the Dijkstra and AStar algorithms, ShootingStar contains core and wrapper functions.

In this research, Dijkstra, AStar and ShootingStar algorithms have been used and implemented (in the simple routing service) as a selective combo box to provide possibilities for classified users to give select proprietary routing algorithms. Using the combo box enables police, emergency services, firemen and all users who have authority to neglect the traffic restrictions to select a shortest path not available to regular users. In contrast, they have to select the method in which all type of restrictions are considered and applied to define the cost of each edge in routing.





# CHAPTER 4

## Methodology

This chapter represents the methodology of the research that was used to design, program and implement the NCS as a controllable, manageable and real-time web-based routing service. The methodology is based on the concepts presented in chapter 3. In this chapter, we explain how the conceptual basics including geospatial web standards, programming languages, database technologies, and server and web application technologies are combined to create a real-time service in the web era. This chapter explains the NCS step by step from installing server, web application server, and a simple routing service to the live text engine. It also explicates the vectorial search engine for real-time web service, warning messages and live correction of the calculated best route. These research components are explained in detailed in terms of the technical programming aspect.

### 4.1 Research material

Research material includes open source software, web application server, open specifications components, RDBMS, and data.

#### 4.1.1 Platform

Ubuntu Server 12.04 (the latest stable long term support (LTS) version of Ubuntu Server) is used as the platform of this research. We used the latest release of Ubuntu Server as the server platform because there are some reasons that convinced us to select it:

- 1- This research is based on open source software and Ubuntu Server is an open source technology.
- 2- Ubuntu is free of charge.
- 3- A very efficient and skillful support team exists to update releases in the future.
- 4- It provides minimum system requirements in terms of hardware like RAM (Random Access Memory), disk space, and graphic card. Renting or acquiring a server can be done at minimum costs.
- 5- The Linux operating system family is more secure than Windows. For example, last year 48,000 new virus signatures were documented for Windows, compared to 40 for Linux  
(<http://www.desktoplinux.com/articles/AT5836989728.html?regen=yes&display=yes>).
- 6- There is no requirement of restarting the computer when the configuration changed or new programs are installed. In server versions this capability is very useful because at the restarting time all users are disconnected from the servers. This server may be a part of distributed computing like in large-scale projects that use many servers and any interruption affects all results at the respective time. Using Linux servers like Ubuntu prevents such disadvantages. Some Windows configuration changes require a reboot and may take a long time. For example if we use a Windows server, during the reboot time all users are disconnected from the real-time service. Almost all Linux configuration changes can be done without affecting unrelated services and without the need to reboot. The clients can use the system and changes can happen simultaneously.
- 7- Linux is multiplatform and can work with wide a variety of processors and machine architectures from Macintosh to a mainframe. In contrast, Windows is limited to Intel and Intel compatible processors.

## **4.1. 2 Web application server**

The Apache web server is used to host websites. Apache along with PHP5 and PHP-PgSQL were installed on the server. The latter is used to communicate with the SQL database (e.g. PostgreSQL).

## **4.1.3 Database management system**

PostgreSQL is a cross platform RDBMS that can be used for almost all operating systems including Linux, Windows, Solaris, MacOS and Free BSD. Besides the advantages mentioned above, Linux has PostgreSQL already in its software package section. PostgreSQL contains a large variety of extensions to enhance its capability in special fields. It includes extensions from import/export to spatial extensions like PostGIS. The main extensions of PostgreSQL in geospatial era are the followings:

- 1- PostGIS "adds support for geographic objects to the PostgreSQL object-relational database. Actually, PostGIS "spatially enables" the PostgreSQL server, allowing it to be used as a backend spatial database for geographic information systems (GIS), much like ESRI's SDE or Oracle's Spatial extension. PostGIS follows the OpenGIS "Simple Features Specification for SQL" and has been certified as compliant with the "Types and Functions" profile" (<http://www.postgresql.org/download/products/6-postgresql-extensions/>).
- 2- PgRouting enhances the PostGIS/PostgreSQL geospatial capabilities to offer geospatial routing functions. Using PgRouting extension enables PostgreSQL to run routing algorithms including Dijkstra, AStar and ShootingStar as will be explained in detail in this chapter.
- 3- PostPic "is an extension for PostgreSQL that enables image processing inside the database, like PostGIS does for spatial data. It adds the new 'image' type to the SQL,

and several functions to process images and to extract their attributes" (<http://www.postgresql.org>, 2012).

#### **4.1.4 Data**

The data used for this research was downloaded from OSM website. The format of the data is in XML. The structure of data is as the following:

- An XML suffix defines the character encoding as UTF8 (UCS (Universal Character Set) Transformation Format—8-bit) and version 1.0.
- An OSM element containing the version of API (features used) and the OSM generator to conform to XML data.
- Blocks of nodes containing: id of node, latitude and longitude of the node, time stamp, user, user ID and other auxiliary data for each node.
- Blocks of ways containing id, time stamp, related node to this way as references.
- Blocks of relations containing the reference members.

#### **4.2 Implementation of the simple routing algorithms over the web to find the best route by open source software and open specifications**

To implement a routing service for street networks on the web by open source Software and Open Specifications, many stages needed to be programmed and implemented. In this research, we have categorized the stages as the following:

- Preliminary stage
- Browser client programming
- Server-side programming
- Integrating the whole service components and testing for probable errors
- Debugging and running the service over the web

Here, we try to show step by step on how to proceed but it should be noted that Pgrouting is an open source system and volunteer forum for this reason. Everybody can participate in this project, develop, use and transform it to his/her interest to match and comply for special goals. In this research, we use the Pgrouting project FOSS4G2010 workshop instruction (<http://pgrouting.org/>, FOSS4G2010 workshop) to implement the simple routing service (offers proprietary routing algorithms) and develop it for our specific goals to offer a real time service. The NCS is a quite new project programmed as a real time web service for the first time to give real time routing capability for users.

#### **4.2.1 Preliminary stage**

This stage includes setting up the necessary materials and data like open source applications, enriching them by spatial analysis capabilities, data importing, creating topology and so on. Within this context, the following steps were performed:

- 1- Installing Ubuntu server 12.04 64-bit and the necessary components:

The Linux server is easy to install. Everybody can refer to the Ubuntu web site and download and install it. The minimum system requirements for Ubuntu Server 64-bit are 128 MB of RAM and 1 GB of Disk space. The 64-bit edition needs AMD64 and Intel x64 including AMD Opteron and Intel EM64T Xeon.

The following components are installed on Ubuntu Server to access and communicate with the server:

- SSH (Secure Shell) server: installing this component enables the administrators to access the server, edit files/folders and manipulate the server via a secure protocol by SSH client.
- SSH client: this program is installed on the computers so that the administrator can access the server via this computer. There are some software packages that can act as

SSH client. We selected Putty for this job. It should be noted that the SSH protocol is only command based and works across platforms.

- FTP server: File Transfer Protocol (FTP) is installed on the server to transfer and share files over the web by using FTP client. Proftpd software was used as FTP server in this research.
- FTP client: the protocol used to connect to the server, upload, download and manage files on the server. Filezilla client software is used as FTP client in this research.
- Editor: editor software is used to write and edit codes on the server. After installing the FTP server, it is possible to set up an editor on the administrator computer to access the server by FTP protocol for writing/editing codes and files/folders. Pspad editor is the software to do this job efficiently. In fact, this open source software is an HTML, PHP, JavaScript, XML and other web standards editor with extra functions and facilities.

## 2- Installing the web application server

Apache with PHP5 and PHP-mysql extensions was installed for this research.

## 3- Installing PostgreSQL and complementary extensions for routing

PostgreSQL is a Relational Database Management System for hosting the database and for running a large variety of functions from a simple *create table* function to complicated spatial and routing functions. For this research, PostgreSQL was enriched with the proper extensions to serve spatial and network analysis functions. Therefore, PostgreSQL8.4 along with the following extensions was installed:

- PostGIS to add support for geospatial data into PostgreSQL and to turn it from a pure database to a spatial database for GIS.
  - PgRouting to add the capability of running different routing algorithms within PostgreSQL/PostGIS.
- ## 4- Installing OSM2pgRouting tool to import OpenStreetMap data into the pgRouting database

It is a command-based tool to automatically generate the routing network topology

and to create tables for feature types and road classes. Usually, downloaded OSM data contain more than what is needed for routing. Hence, it is imported by OSM2pgRouting tool based on a configuration XML file. OSM data as an XML file consists of three major feature types: nodes, ways and relations as mentioned in the data section of this chapter. By using a configuration XML file, OSM2pgRouting tool only imports node, ways and classes that are defined in the configuration file used for routing

- 5- Creating a template database into PostgreSQL to import the network data
- 6- Creating the procedural language
- 7- Adding the PostGIS functions into the created database including Postgis.sql and spatial\_ref\_sys.sql to support geometry
- 8- Adding pgRouting functions including the following:
  - Core functions to add three main routing algorithms function (Dijkstra, AStar and Shooting Star)
  - Wrapper functions to extend the core functions with transformations, bounding boxes limitations, etc. Wrapper functions improve the performance of the system especially in large data sets. For example, it is possible to limit the search area by defining a bounding box.
  - Topology function to import network topology into the created database
- 9- Importing OSM data into the PostgreSQL database and creating topology. This stage imports the OSM data into the created database based on the configuration XML file and creates topology of the network.

## **4.2.2 Browser client stage for the simple routing service**

Client programming is for designing and dissemination the browser face, accessible facilities and tools for the service over the web. Attractive appearance, adequacy of needed tools, supported functions in a particular field, ease of use and appreciation of non-expert users, are some of the characteristics of an efficient service/site over the web. Generally, users judge about the capability and efficiency of the web service based on the above factors. To program and run a simple routing service, the browser client stage is divided into two main steps:

### **4.2.2.1 Designing the preview**

The preview user interface of the routing service on the browser is designed based on what a user expect from a routing service. As shown in figure 4.1, a user needs some tools and functions to run the routing service and to get the result on his/her browser. In this research, the offered tools and functions include: Zoom in/out, move up/down, layer switcher, start and destination points, bounding box selector, on-mouse-over event to report real coordinates, combo box routing algorithm selector, routing result as a vector line, and background image of the study area.

### **4.2.2. 2 Programming the designed preview on the web**

After designing the service interface on a paper as a draft, it is the time to implement it by programming scenarios on the web server. It is possible to use any editor like Notepad or WordPad and copy the file on the server via SSH protocols and Linux server editors like Nano. Nevertheless, the more practical way is installing a standard web editor that contains extra tools and facilities to connect to the server via FTP. PsPad editor is an open source standard web editor and was therefore used in this research. The most parts of this stage are in JavaScript. GeoExt along with OpenLayers and ExtJS libraries of JavaScript on the client-side have been used by the PgRouting project (FOSS4G2010) to disseminate the



project to users. In fact, GeoEXT gets the spatial support from OpenLayers and offers dynamic capabilities from ExtJS.

To program the preview on the client side the following components has been used:

- 1- JavaScript, Ext, OpenLayer, GeoExt and Proj4js libraries along with CSS were included inside the <script> tag of JavaScript located through the head tag of the HTML page which is supposed to represent the project to users.
- 2- Projection transformation variables were employed to transform geographic coordinates to the Mercator projection.
- 3- OpenLayer's Control.DrawFeatures functions were included to draw the start and destination points on the browser.
- 4- OpenLayers Control.DragFeatures function was used to add the drop and drag capabilities to the user browser. By means of this function, users can move points to another location as new start or destination points.

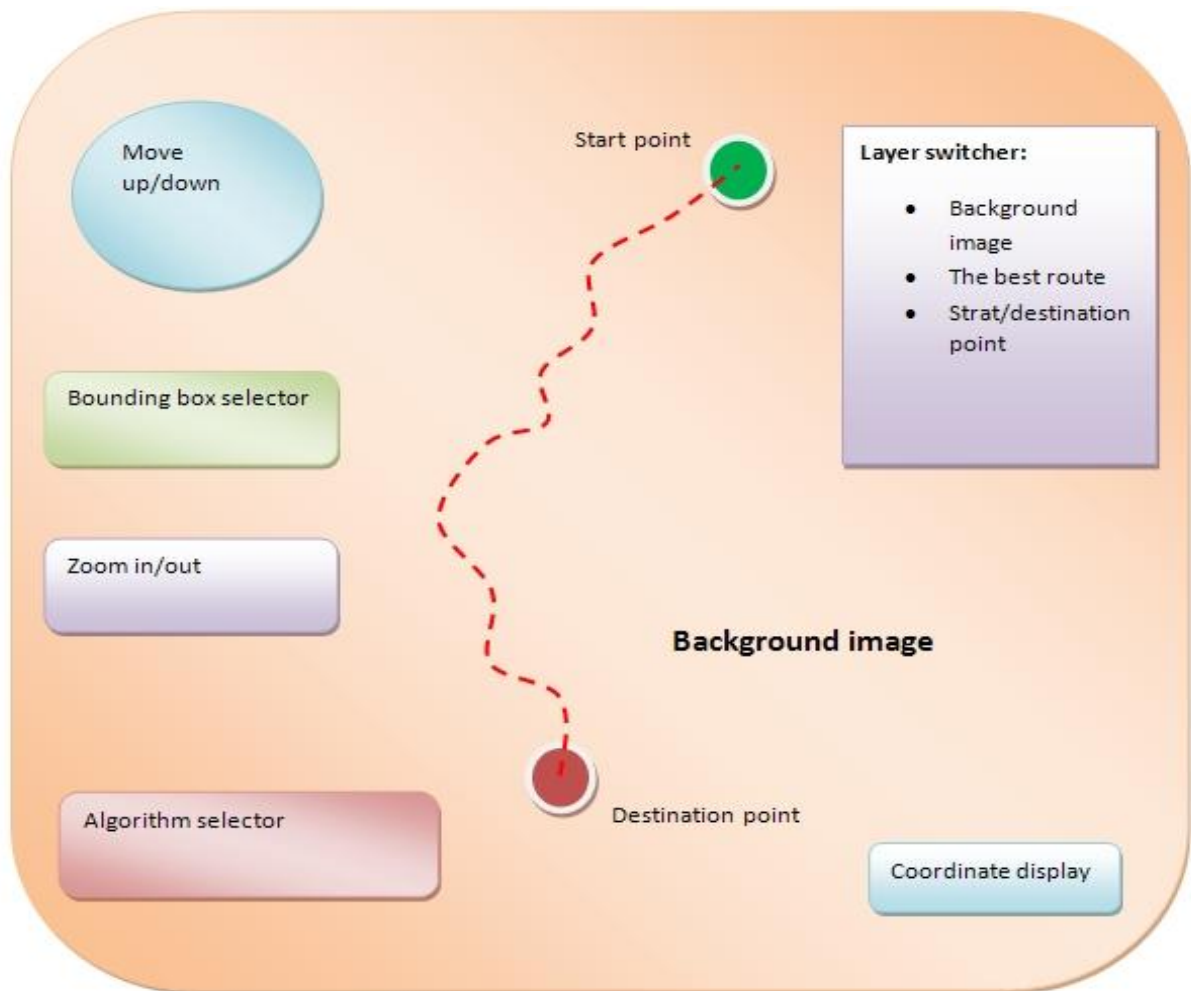


Figure 4.1 : Schematic preview of the simple routing service on the user browser side

- 5- Openlayers.Layer.Vector layer variable defines the vector features that can be drawn with the related geometry and attributes.
- 6- A stylish Openlayer's Layer.Vector layer variable was utilized to draw the calculated best route on the user browser in vectorial line format.
- 7- An OpenLayer's Map object was defined to add the two previous variables on the map through the browser.

8- A map panel was created to display the background image layer. In contrast to the last layers, it is a WMS layer.

9- Map control functions were added including the following:

- Draw and drag points control
- Layer switch control
- Mouse toolbar control
- Mouse position in decimal degree with a mouse-on-over event
- Scale line control

10- A FeatureStore variable has been added to execute the PHP file written to find the best route between the start and destination point.

In GeoExt programming, a feature Store contains:

- Store: is simply a container to store information into which we can push data and get it back to the client-side.
- Layer: the FeatureStore will automatically draw the GeoJSON result received from the broker file (in this research a PHP file) of the server into the layer. This layer was created and styled before.
- Field: lists all the attributes returned along with the geometry. The PHP file returns the segment length.
- Proxy: the proxy item specifies where the data should be taken from (a HTTP server). The proxy type is `GeoExt.data.ProtocolProxy`. This class connects the ExtJS world (the Store) with the OpenLayers world (the protocol object).

- Protocol: this OpenLayers component is able to make HTTP requests to a URL (PHP script) and to parse the response (format option). By adding the `internalProjection` and `externalProjection` option, reprojection is done to comply to the background WMS and the vector layer because the route may come from the server in a different spatial reference system.

The PHP file has to be connected to the database, executes the queries by Pgrouting and delivers the result in GeoJSON format. Subsequently, the created Store puts the Geojson vector line inside the defined layer as an OpenLayer vectorial layer to display the best route, transforms Geographic to Mercator projection and displays it on the client browser.

11- Create a combo box with `Ext.form.ComboBox` function to offer a selective box between different routing algorithms for classified users. It contains Dijkstra, AStar and ShootingStar algorithms. For example police and emergency fleet use different routing algorithms than normal users because they just look for the fastest way to get to the destination without adhering to respective travel regulations such as turn restrictions, traffic lights or one-way streets:

12- Secure the web service by JavaScript codes. It maintains the web service free from penetration and source code theft. In fact, it blocks any right click event.

13- Embed trigger functions to recall the web service for special events.

In web programming, triggers are used to re-execute a command or function when something happens. For example, if the start or destination changes, the routing algorithm must be executed again and subsequently, the new route will be calculated and drawn. In the simple routing service, the following triggers were used:

- Draw trigger: executes the web service when the start and destination points are drawn
- Move trigger: re-executes the web service (i.e. the routing services to find the best route) when either the start or destination point is moved.

- Combo box trigger: to offer arbitrary mode of routing algorithms depending on the user class. The combo box also acts as a trigger. When the user selects another method, the web service would be executed again on the server and deliver the newly calculated route to the online user. In this research, police, emergency personnel and fireman use the Dijkstra algorithm as a vertex-based and not a heuristic method, whereas regular users makes use of the ShootingStar as an edge-based heuristic algorithm. There is also another option to use the AStar algorithm as a vertex-based heuristic method.

14- Add Pgrouting function: in fact, this function is the key function for routing. It calls the web service through the defined Store argument:

15- Integrate the full source code with additional codes to give executable capability in the browser client-side.

#### **4.2.3 Server-side programming stage for the simple routing service**

By server-side programming languages, we can connect to the database and execute functions by the server and then deliver the result to the client-side. This is what we need to connect to the PostgreSQL database, execute the scripts by parsing the written codes, get the result, wrap it in spatial or another formats and finally, and deliver it to the client-side. In the current research, we use PHP and MySQL along with AJAX and JQuery as server-side programming languages. GeoJSON is used as the data exchange format to retrieve the vectorial results on the browser side.

To implement the server-side stage for finding the best route, the following steps have been done (Server-side scripting for designing the NCS will be explained in the related sections):

1- Define the database connection codes: this is the first step when we want to work with a database system. To connect to a database the following parameters must be defined:

- Database name: it must be defined when the raw data is imported to the RDBMS. It also must be enhanced by PostGIS/Pgrouting extensions to get spatial capabilities.
- Host name: is the server name or server IP address in which the database is located.
- User name: user name of the database owner.
- Port: number of the opened port for communication.
- Table: contains tabular imported data

2- Retrieving the start and destination point coordinates:

```
$start = split(' ', $_REQUEST['startpoint']);
$startPoint = array($start[0], $start[1]);
$end = split(' ', $_REQUEST['finalpoint']);
$endPoint = array($end[0], $end[1]);
```

3- Finding the nearest edge to the start and destination point and keeping it into a temporary array.

When a user draws the start or destination point on the browser, the system should find the nearest edges to them and start calculating the requested route. Normally, when a user clicks on a point in the browser, it is not exactly coincident with the start or end vertex of an edge. To solve this problem, the system must find the nearest vertex to the point and considers it as the start or destination point. For ShootingStar as an edge-based algorithm, that is enough to find the nearest edge, but for Dijkstra and AStar as vertex-based algorithms the system can select either the start or end of an edge. To implement this step, FindNearestEdge function is used to find the nearest edge to the start and destination point:

To find the nearest edge, the following steps have been programmed:

- Defining the function based on the coordinate system
- Connecting to the database by PHP-MySQL programming
- Getting a PostGIS query to calculate the distance of each edge to the drawn point
- Ordering the calculated distances
- Running the query to find the nearest edge
- Closing the connection, putting the query result in PHP arrays, returning and keeping it to be used in the next step

4- Selecting the routing algorithm, opening a connection to the database and running the query.

The Switch statement is used to have selecting possibility among the routing algorithms. The statement uses the request function to get the selected algorithm from the HTML page on the user's browser. With this, the user can select the appropriate routing methods.

RT function is one of PostGIS functions to convert the query results in vector formats like KML or GeoJSON. It uses the start and destination point variables and PgRouting to run the requested method. The results will be kept as a new vector layer and need to contain both geometry and id columns of the database on the server.

5- Returning the query result as a vector layer on the browser

At this stage, we need GeoJSON programming to return the result of the executed query in the previous stage in GeoJSON format and, finally, to display the vector layer on the browser of users. Since this stage is very important to have the result on the browser as the final stage, the written codes are explained in more conceptual details. Explaining the whole GeoJSON programming is beyond the scope of this research, although, we try to explain as much as needed to understand just the framework of the current step. The following codes are needed to return the query result on GeoJSON and send it to the client-side:

```

<?php

    $geojson = array(
        'type' => 'FeatureCollection',
        'features' => array()
    );

    while($sedge=pg_fetch_assoc($query)) {

        $feature = array(
            'type' => 'Feature',
            'geometry' => json_decode($sedge['geojson'], true),
            'crs' => array(
                'type' => 'EPSG',
                'properties' => array('code' => '4326')
            ),
            'properties' => array(
                'id' => $sedge['id'],
                'length' => $sedge['length']
            )
        );

        array_push($geojson['features'], $feature);
    }

    pg_close($dbcon);

    header('Content-type: application/json',true);
    echo json_encode($geojson);

?>

```

Firstly, the Geojson variable is declared. "GeoJSON is a format for encoding a variety of geographic data structures" (<http://geojson.org/>). To explain the above codes, we need to refer some basics of GeoJSON programming language. For deep understanding of these codes, we refer to the GeoJSON documentation along with PHP-SQL and PostGIS from their official websites.

Using GeoJSON interchanging programming, the representation of the following objects is possible (<http://geojson.org/>): Geometry (Point, LineString, Polygon, MultiPoint, MultiLineString and MultiPolygon), Feature (Geometry with other attributes), Collection of Geometries and Collection of Features.



"A complete GeoJSON data structure is always an object (in JSON terms). In GeoJSON, an object consists of a collection of name/value pairs also called members. For each member, the name is always a string. Member values are string, number, object, array or literals: true, false, and null" (Schaub & etc. 2008).

As above written GeoJSON codes show, firstly, a GeoJSON variable has been declared. Its type is an array of FeatureCollection object. A FeatureCollection object must have a member with the name feature contained the related value as an array.

It creates a GeoJSON arrays used in the next step to put the edge variable into it. Now, we add the edge variable also created in the before stages into the GeoJSON variable. For this stage, GeoJSON programming along with PHP and SQL languages are required.

As seen a While loop with `pg_fetch_assoc` function used to push the result of the routing query to the GeoJSON array. `Pg_fetch_assoc` function returns an associative array that corresponds to the fetched row.

As mentioned before, the power of PHP comes from its functions. This is why PHP programming is recommended for such type of researches. It contains a large variety of functions specially, in arrays case. To be more theoretical acquaintance with how implementation of this step, some basics of PHP array functions is necessary.

"There are specific database handling functions for populating arrays from database queries, and several functions return arrays" (<http://www.php.net/manual/en/intro.array.php>, 2012). To return a query result as a line in vector format like GeoJSON, we need to use Associative Arrays in PHP. According to W3schools definition, there are three kinds of arrays ([http://www.w3schools.com/php/php\\_arrays.asp](http://www.w3schools.com/php/php_arrays.asp)):

- "Numeric (indexed) array : An array with a numeric index: A numeric array stores each array element with a numeric index
- Associative array : An array where each ID key is associated with a value
- Multidimensional array : An array containing one or more arrays"

Using associative arrays, we can use the values as keys and assign values to them (key=>value) like the above codes. In a multidimensional array, each element in the main array can also be an array and each element in the sub-array can be an array, and so on.

Here, we defined an array of Feature object to add the edge variable. A Feature object must have a member with name 'property' and its related value is an array of ID and Length columns for the relevant row into PostgreSQL database. When a Feature variable is declared includes array of Features, for each Feature inside this array, Crs, Type, and Properties name with the related array values are defined. We know that each row as a street line contains more than one edge (segment). This is why we use array for all values of the member names.

Up to now, it was explained how the server calculates the first corresponded row as a GeoJSON vector feature what is the first part (edge) of the best route between the start and destination points.

Now, by using the `array_push` function, the created feature is added to the Feature Collection array:

```
array_push($geojson['features'], $feature);
```

To implement for the rest edges, the While loop function is used to keep continue including all edges into the array while all of them are calculated in the routing algorithm. Then, the connection is closed (`pg_close($dbcon);`) and finally, the routing result is returned on the browser in GeoJSON format.

#### 1- Registering a host name and domain name on the web:

At this stage, the Institute for Geoinformatics and Remote Sensing at University of Osnabrück offered a server. Then, Ubuntu server 12.04 (the latest version at the current time) along with all necessary programs was installed.

#### 2- Transferring all files and folders to the server and running it to serve the service over the web (figure 4.2).

The final result of the simple web routing service for finding the best route over the web is shown in figure 4.2. As seen in this figure, the web service is also ready to use for classified

users. If the user is policeman, fireman, ambulance driver, or everybody who has the necessary authority to ignore driving laws can choose 'police, emergency, fireman' from the combo box items. This item just considers the shortest direction and does not consider "no turn" restriction and other traffic and driving laws. If you are a normal citizen, you must select 'Common users' because for normal users respecting to the traffic laws is necessary.

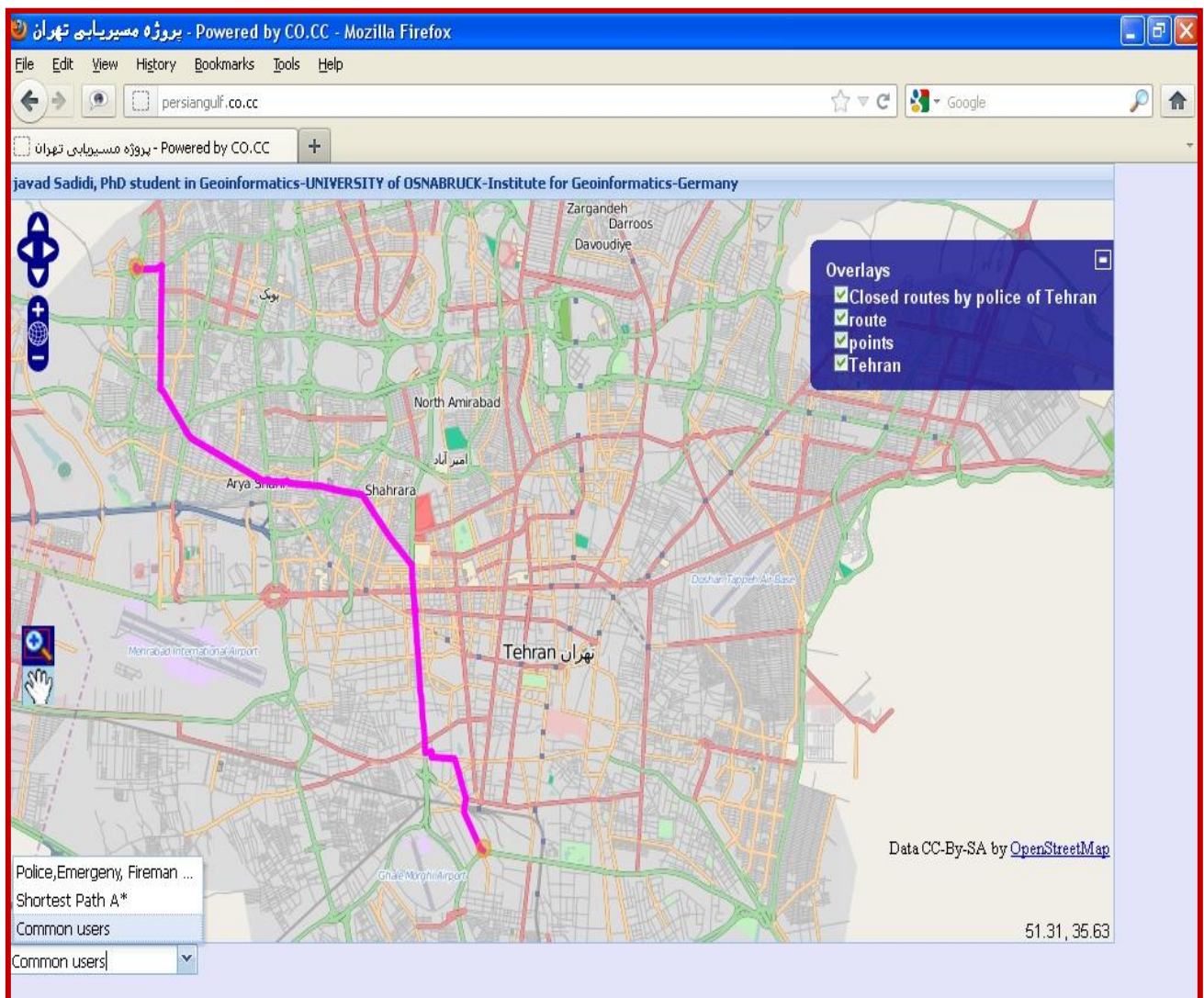


Figure 4.2: The result of implementing the simple routing service for the case study area on the browser over the Web.

### 4.3 Designing the real time web-based network controlling system (NCS)

#### 4.3.1 Introduction

Being real time in cases related to spatial information systems reduces the reaction time for spatial phenomena like natural disasters. Large cities have a large number of streets and roads. Real-time finding of the best route inside such cities is complicate complex tasks but also very helpful for citizens and relevant organizations. In large cities like Tehran people

waste their time in heavy traffic and polluted places instead of their home or office. Due to this traffic, police, emergency services, firemen, and other organizations are not able to get to target places on time. If a real-time system could be used, however, the travelling time can decrease significantly so that lives of injured people can be saved and monetary losses be avoided. For regular users, it can also save travel time and therefore movement costs.

To control the target network, the administrator (police or other officials) has an AJAX live search system to find the street name stored in the database in order to close the specific section of the network. Once the administrator closes a street, the designed NCS displays the closed streets on the user browser and recalculates the best route by considering the applied changes. This is done without refreshing or reloading the page. Once the problem is solved, the administrator opens the closed segment and the new best route between the start and destination point is recalculated. According to the objectives of the research, we designed it so simple that inexpert personnel can use it on their mobile phones. The designed system is also compatible for mobile devices and low speed internet connections even by SIM card connections. In fact, the complex side of the project is implemented on the server and a very simple visual interface is offered to the administrators and other users.

To implement the above tasks to be accessible on the web, a network controlling system (NCS) was designed based on open source technologies. Using the NCS, the simple routing service was upgraded to the new real-time version. Designing the NCS for crowded cities like Tehran is a significant step to help people to save the time, human life and promoting the prosperity index.

To implement the NCS the following steps were taken:

- Needs assessment: this stages includes an estimation for the needed hardware, consultation with experienced persons to realize the needed modules and determination of the necessary programming languages to implement the NCS
- Design of the conceptual architecture of the system on paper
- Translation of the conceptual architecture from paper to computer language via programming languages as a new web service
- Execution of the system and debugging probable errors

- Testing advantages, disadvantages and capabilities of the system
- Integration of the NCS with the simple routing system of PgRouting project (FOSS4G 2010) to offer real-time routing capability
- Evaluation of the designed system to run the final version

#### **4.3.2 Designing the conceptual architecture of the NCS**

For implementing a service, it is better to draw its components on a paper to imagine the relationships between the components. By this, the conceptual framework of the system can be formed and the necessary programs and standards can be identified. Designing the conceptual model by drawing it on the paper and defining the elements and relations between them, leads us to a photographic understanding of the system and subsequently, re-defining the efficient programs and standards to implement the project. To design the NCS, the following tasks according to the main objectives of the research were considered:

- 1- In normal situations the clients use their browsers to find the best route designed via the simple routing service. It is notable, however, that in cities like Tehran, this simple service is almost never efficient because more than one street or road are closed or opened during a couple of minutes by police. On average, 182 accidents happen in Tehran every day (<http://www.jamejamonline.ir/newstext.aspx>, 28/3/2012). Hence, a real time routing service over the web is very necessary and the static routing systems do not cover routing requests in such cities. If we imagine 80% of accidents happen in active hours of a day (6-20 h), the routing service needs to be updated on average 20 times per hour. It is clear that for each street closing process, opening process and subsequently, a route correction has to happen. With NCS, the update process as a real-time system happens once the change (closing or opening a street) has taken place.
- 2- A password protected service for the official administrator who has authority to control the network.
- 3- By entering user name and password on the mobile or on the computer of the administrator, the service connects to the server and is ready to do predefined tasks

to control the network. Number of administrators must be unlimited because in the same time, many policemen are in the streets through the city. If something happens and they need to close the street, they can decide to do it immediately and prevent wasting the time of users, emergency services, police, firemen etc. This facility gives the real-time possibility to the system instead of calling to the control center and requests the personnel in charge to do it. This is more secure, faster and cheaper because there is no need to hire other personnel in the control center. Each administrator (here: police) who is present in the city can close and open the special network elements (here: streets and segments).

- 4- The service must use a text live search engine to connect to the database, find and select the target street to close or open it. Writing the full name of a street also takes the time. The administrator may write the name wrong and receive an error message. This happens especially for long names. By using the designed live search engine, the administrator is able to find and apply a street name for closing or reopening in a couple of seconds.
- 5- The live search engine must be able to be supported by all major browsers even in old versions and specially, in mobile devices browsers.
- 6- The live search engine must support Persian language.
- 7- The service must draw the current closed streets as a vector layer on the user browser with no refresh or reloading the routing web page. The service also should send the closed streets to the browsers for all online users as soon as the administrators do it and warn the users for changes.
- 8- The service must remove a vector layer as a closed street from the users browsers once the administrators open one or a number of streets, this job must also be done automatically without the need to refresh or reload the routing page. By this capability, users can watch some streets drawn as closed and continue once the streets open.
- 9- The service must correct the current best route immediately when a street is closed or opened. It has to recalculate the new best route for each user and must submit the new best route to his/her browser in real time. Different routes have to be calculated

and sent simultaneously to many users because each user has independent start and destination points may be using different routing algorithms.

10- The service must be able to close and reopen a special segment of a street or road as well. Many streets are longer than what the administrator wants to close. In this case, a subset of the respective street is closed instead of the whole street. To assist the administrator, the NCS service exploits a vectorial live search engine to locate and draw the closed segments of the target street(s).

11- One of objectives of the research is designing the service for low literacy personnel. Hence, in the research, we aim to design it very simple for both user and administrator. Complex and programming aspects have to be hidden.

Based on above concerns, the conceptual architecture of the NCS was designed on paper as seen in figure 4.3.



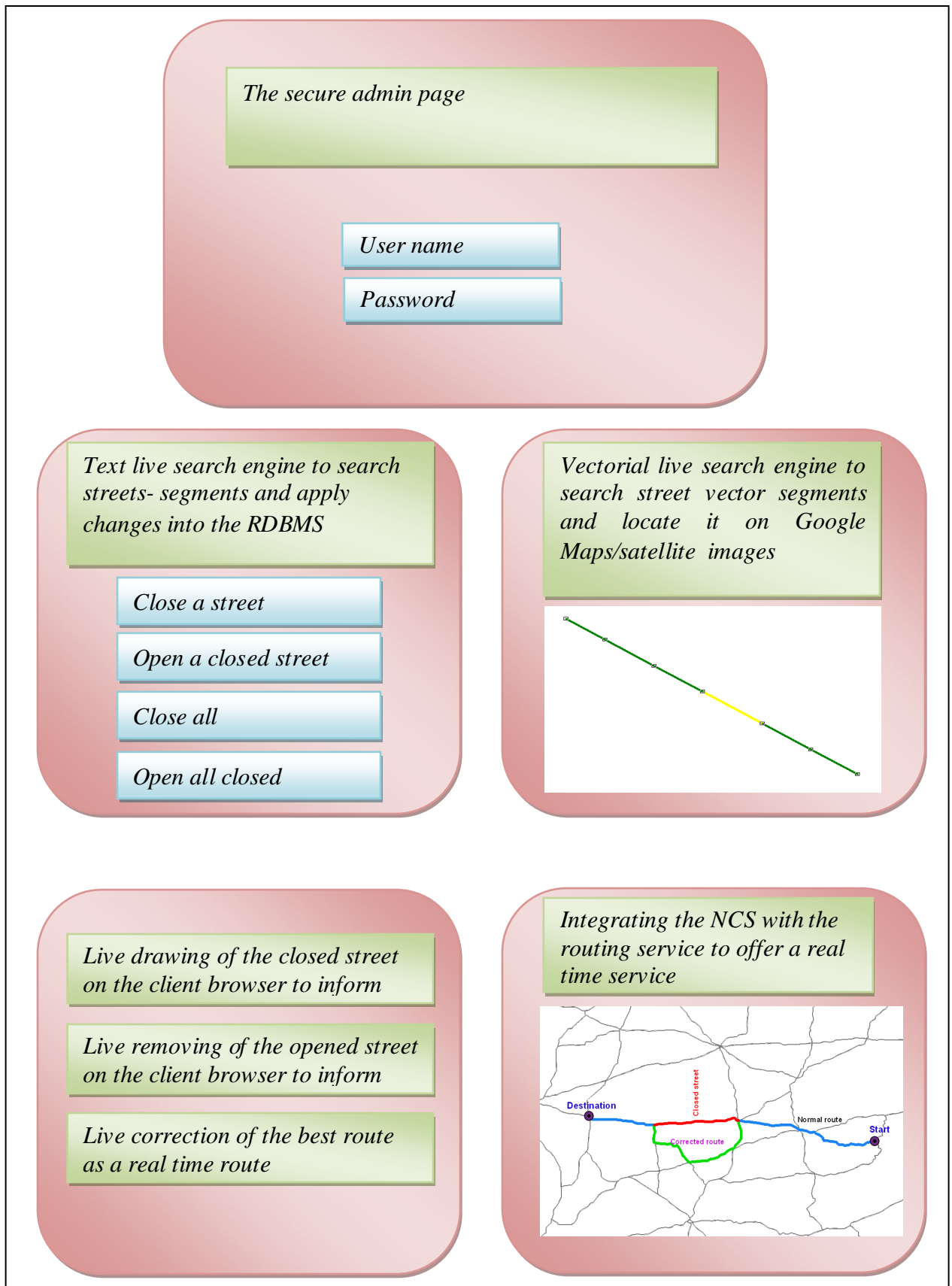


Figure 4.3: Conceptual architecture of the NCS

#### **4.3.3 Programming and executing components of the designed architecture**

As mentioned before, this research aims to design the NCS for upgrading the simple routing system to a real-time routing service over the web. Hence, a complicated programming scenario is needed to cover this vast research frontier about translating the designed architecture on the paper (the conceptual architecture) to computer codes language, executing the written codes, debugging the system and implementing the assessment stage to consider the efficiency of the different programming scenarios. There are different programming scenarios to implement the NCS but capacity consuming scenarios impose a futile loading average to the server. Subsequently, the server may not be able to response quickly enough to the user requests. On the other side, another programming scenario may run faster with less calculations and the server is able to response to more simultaneous users. The efficiency assessment stage for spatial information systems is indispensable because compared to usual services, spatial information analyses impose more calculations and loading values for the server. For web services like the designed NCS, the response time depends on the load average factor and is very important. Any delay of response defeats the purpose of a real-time system. This issue is a vital factor for organizations like police, fire fighters and emergency fleet because they depend on a real time system to save lives and avert damages.

Imagine that an emergency ambulance is carrying a sick person to the hospital. In its way an accident happens and police close the route, The NCS sends a warning, corrects the route immediately and suggests the new direction but the server is not able to a real-time response. It may even be overloaded and crashed because of a high number of users. What would happen? The system is not able to apply new changes and even the closed street(s) would not be shown. The emergency vehicle enters the closed street, gets stuck in the traffic jam and vital time will be loosed due to an inefficient programming scenario. Therefore, the need for efficiency assessment arises when we consider the spatial information analysis need for real-case scenarios. If we add real-time capability to spatial information systems, we impose more calculations and heavier loading factors to the server because a real-time system needs to be updated frequently. For this research, we also have investigated different

programming scenarios for testing the efficiency and elected the best one to be implemented in the NCS for best real time service.

It should be noted that some issues of web programming are beyond the scope of this research but it is necessary to know their relevance to this research. For example, from a security point of view, web security standard are issues that require further research to avoid security breaches.

According to the conceptual architecture of the NCS, it was divided into two separate items, the client-side that runs on browsers like Google Chrome, Mozilla, Opera, etc and the server side. For both items, there are special programming languages to implement the model. The designed and programmed NCS is implemented and tested on a server that belongs to Institute for Geoinformatics and Remote Sensing of the University of Osnabrueck. In fact, by designing and implementing the new service, we have enhanced current routing services like Google Maps, TomTom and other available services to real-time systems that can be controlled by an administrator with only simple writing and reading skills. In the following, the NCS components are discussed individually, although they work together as an integrated system.

It is evident, that the reliability of the designed service depends on the administrator side and any errors in this issue can provide wrong results to the users and, subsequently, render the designed service useless. Hence, the administrator side must be secure and impenetrable by unauthorized persons. Imagine that a hacker infiltrates the designed NCS and closes or opens a street. The hacker is going to disrupt the system deliberately. What will happen? Users believe to have found the best route encounter a traffic jam or an emergency team or hospital vehicle gets misled into a closed street. These problems have to be considered by any system efficiency analysis.

#### **4.3.3.1 Programming the security protected administrator page**

The administrator page is protected by username and password. It is notable that this page has been added to the administrator side programs by PHP Include function. To design the administrator page, it is necessary to use some functions for providing an efficient and secure capability to it but detail explanation of all functions and programmed lines is beyond

the scope of this research. Here, we have tried to represent a compact and clear description for better understanding. open source oriented research projects are more complicated than license holder systems in terms of knowledge of language and programs issues to understand the conceptual frame work of the research.

To program the administrator page, firstly, an assosiative array was defined to store the username and password as login information. The Isset function is used to define logout time. This is important when administrator leaves the mobile set inside the city or loses it. This function redirects the page after predefined time and subsequently, for entering to the administrator page, the username and password are needed twice. HTML forms along with PHP are used to create username and password boxes and submit button. It is notable that all these codes are contained by ShowLoginPasswordProtect function. Figure 4.4 shows the entry page for the administrator on Google Chrome browser.

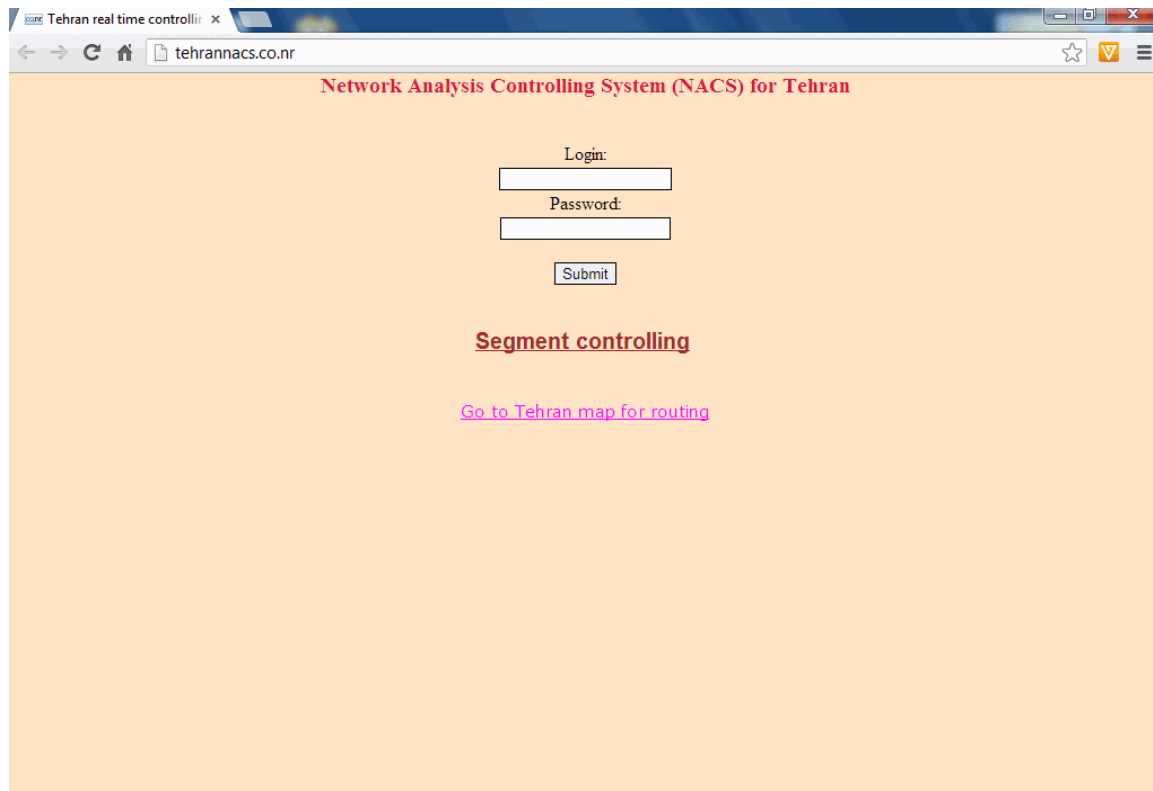


Figure 4.4: Entry page for the administrator on a Google Chrome browser

#### 4.3.3.2 Programming text live search engine for database changes

Usually, for real time systems like what is designed in this research, a constant connection between the administrator/client and a relational database system on the server is needed because the database is always updated and used. In fact, fluctuation behavior is intrinsic of real-time systems and, subsequently, real time databases. The programmed text live search engine is used to search, select and submit a street or segment name for applying the necessary information whenever the administrator is going to update the database on the server (close or open actions for a street/segment). The update process will be explained in the following.

In this research, the search engine must possess field restricted search capability to search and select a street based on its name. It must be programmed so that it starts the search process when the administrator hits a letter on the keyboard. The search engine starts

searching into the name field of the database on the server and suggests the streets name which contains the written letter alphabetically. If the administrator leaves out a letter or misplace them, the search engine will show nothing because the record name in the database and what is written by the administrator must be the same but when the administrator hits the first letter of a street name or a segment, the search engine will suggest the whole name and, subsequently, the administrator can copy it in the submit box to close or re-open a street or segment. The text live search engine is able to perform the following tasks for this research:

- Close: to close a street/segment
- Open: to open a closed street/segment
- Open all: to open all closed streets/segments at once
- Close all: to close all streets/segment at once for blocking the service

The implementation of the above mentioned tasks in terms of technical and programming aspects will be explained in a later section. Here, however, it has to be mentioned that for 'Open' and 'Close' tasks, there is a live search box to search and select the requested street. As shown in figure 4.5, the text live search engine connects to the fields which contain street names and suggests the streets that include the entered letter on the search box when the administrator hits a letter on keyboard (see figure 4.5). The NCS also exploits a vectorial live search engine to discover and locate a special segment as a small part of the target street. To show a special part of a street as a segment, an integrated cooperation between the text live search engine and the vectorial live search engine is necessary. The text live search engine finds and updates the target segment and the live vectorial search engine delivers the name from the text live engine. Then it finds the segment in the database and creates an array, and puts the segment edges inside the array. It is then delivered as GeoJSON vector layer to the browser, where it is located based on its geographic coordinates. Zooming in on the segment, redirections and its peripheral environment is performed on Google Maps. It should be noted that all these processes are done automatically by the live vectorial live search engine. This engine provides the possibility to see and manage even a small part of the street on Google Maps, so it can be considered and its peripheral environment can be viewed for better decision-making because the administrator can focus on the part of the target street that must be closed or opened.



Figure 4.5: The designed text live search engine

As seen in figure 4.5, HTML forms are used to create boxes for live search. One of the vital aspects in programming live search engines is their compatibility with various browsers and versions of them. In this research, the programmed codes are compatible with IE, Firefox, Chrome, Opera and Safari.

#### 4.3.3.3 Programming vectorial live search engine

The vectorial live search engine was designed to select and manage a special segment as a subset of a street instead of the whole line, based on its spatial characteristics. This capability of the NCS is used when the administrator is going to select and manage (open/close) a requested segment of a line. For example, an accident happens in the middle

of a street and police must select and close just a small segment of this street. It is not necessary to close the whole street because the segments from the start point to the closed segment and from the closed segment to the destination point can be used and connected to alternative segments of the neighboring streets. The vectorial live search program in NCS draws the requested segment and merges the requested segments to form a special street. The designed vectorial search is connected to Google Maps, GoogleEarth and GoogleTerrain to present the selected segment and its peripheral environment on maps, satellite images, volunteer photos, together with weather etc. As shown in figure 4.6, the administrator is able to select and locate a special segment in the database by clicking on the browser (shown in blue color). This helps to recognize which part of the target street is closed and, by clicking on the embedded link, the related environment along with other auxiliary information like weather, photos, satellite image, terrain and map can be monitored.

As shown in figure 4.6, the programmed vectorial live search engine prepares an efficient environment for the administrator to manage and control a special part of a street instead of the whole street. In this system, each street is composed of number of segments and the administrator can control the routing process by using segments. Without the vectorial search engine, it is not possible to find and locate a segment because segments are not as enough as long to see their location and adjacent peripheral environment like the big streets. After selecting and applying the requested segment, the NCS excludes the target segment as a part of the street and finds the alternative route based on neighboring segments. This solution remains but it is temporary and whenever the administrator opens the segment, the system includes it again to find the best route. How to program the live search engine is discussed directly but performing the live search results including opening and closing a segment of the target street will be discussed in detail later in this section.

To program the vectorial search engine Open Layers, ExtJs, GeoExt, PHP, SQL-PostGIS, JavaScript and HTML are used. For better understanding, client and server programming side steps are discussed separately.



#### **4.3.3.3.1 Vectorial live search engine on the client-side**

OpenLayers, JavaScript, ExtJS, GeoExt and HTML (as described in chapter 3), are standards and programs on the client-side used in this research to create a vectorial search engine. As seen in figure 4.6, vectorial engine on the client-side includes a background image such as Google Maps or GoogleSatellite along with other auxiliary information, a vector layer to represent the closed segment, a number of panels including Map, Store, Grid and main panel and also map controls like zoom, pan, etc. once the administrator closes a segment by text engine, vectorial engine represents the segment as a vector layer on the browser. To embed Google services as background layers in the client-side, application program interfaces (API) are used. "Google Maps has a wide array of APIs that let you embed the functionality of Google Maps into your own website and applications, and overlay your own data on top of them" (<https://developers.google.com/maps/documentation/>, 2012). According to Google Maps website description (<https://developers.google.com/maps/documentation/>), there are different types of Google Maps APIs including: Maps JavaScript, Maps image, Maps for business, Google Earth, web Services and Places APIs. To represent the segment name and highlighting the segment whenever the administrator clicks over it and also the closed segment as a vector layer on the browser, firstly, GeoExt store panel, grid panel, main panel variables were defined and in the next step, a line format vector layer to put the query result as a GeoJSON layer for displaying on the browser, the feature store for binding it to the line vector layer and a grid panel configured with the feature store and zooming to a group of selection were created using GeoExt, Extjs and JavaScript programming languages. Map controls, map and display projections were defined by OpenLayers language.

#### **4.3.3.3.2 Vectorial search engine on the server-side**

As mentioned before, server scripting languages are required to communicate with the server and to perform calculations, to execute the functions and subsequently to deliver the results to the client-side. PHP and SQL-PostGIS are used to implement the server-side tasks. To program the vectorial engine on the server-side, database connection parameters are defined to get access to the PostgreSQL database. Once the connection string is run, a

PostGIS query is programmed to select the closed segment by the administrator. Then, a FeatureCollectionArray as a GeoJSON layer is defined to store the PostGIS query result. Finally, the selected segments are added to the GeoJSON layer to deliver to the client-side. In the last stage, attributes fields of the vector layer like name, length etc represented on the client-side are defined and the PHP connection is closed.

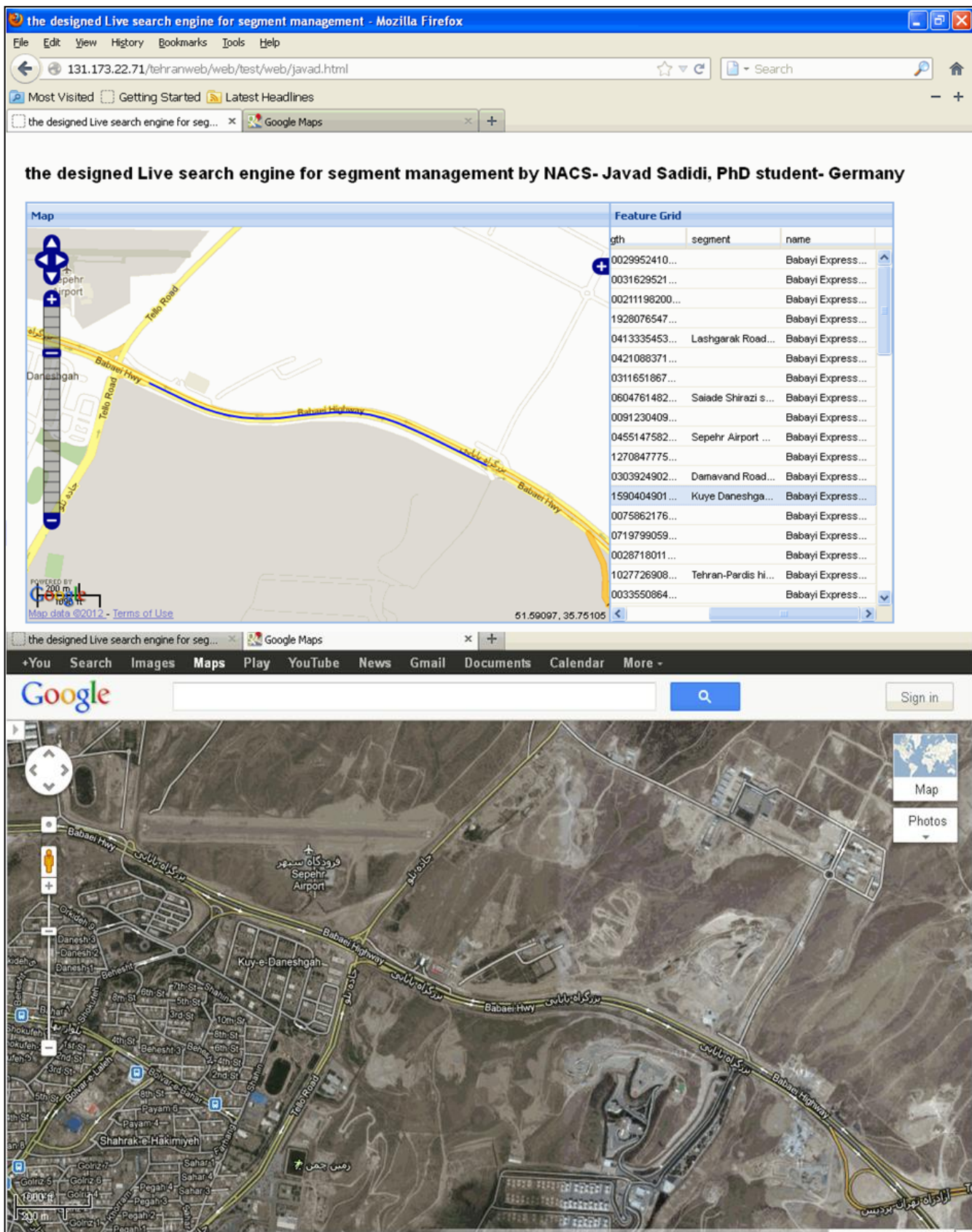


Figure 4.6: Designed vectorial live search for the NCS. The administrator is able to search, select and manage a segment as a part of the target street (top) and to consider the selected segment in Google Maps/Google Earth to have a better view of the target's neighboring areas (bottom).

It has to be noted that by the designed vectorial search engine, the administrator is able to find a closed target segment from a street. As seen in figure 4.6, if the administrator clicks on a special segment name provided by the predefined topology, the designed vectorial search engine finds the segment as a GeoJSON vector layer and locates it on Google Maps. Without the vectorial engine, it would be very difficult and time consuming for the administrator to find the segment on the map; time is the least thing that one can waste in a real time system.

#### **4.3.3.4 Closing process**

This process allows the administrator to close a special street or a subset of a street and prevents the routing service to select it as part of routing calculation. In fact, the closing process prevents the server to use the target street/segment in the routing process so long as it has not been opened. For example, in a real time routing system a user has selected a route from his starting to his destination point when an accident happens. The system has to be programmed in a way that it must close the selected street/segment, send a warning to the user browser (which street/segment is closed), remove the current route, calculate the new route, send it to online users and display it on the user browsers. As done before, the client and the server-side programming processes are separated although, they subsequently work together as an integrated system. The client-side sends a request to the server-side and the server responds. All these processes are done with the assumption that the administrator has selected the target street. For the segment as a subset of the target street, the vectorial search engine helps to find and locate it. The administrator finds the target street/segment using the text live search engine selects the target street/segment and, finally, submits the target/segment to be closed.

##### **4.3.3.4.1 Closing process on the client-side**

The closing process on the client-side includes the administrator and the user side. The administrator side searches the name of the street/segment, finds it and submits its name to the server-side. The client-side uses a 'submit' button to send the found name via the text

live search engine to the server-side. The action of the HTML form would be the execution of the PHP file on the server. It gets the street/segment name by JavaScript Onkeyup. The Onkeyup is a JavaScript event that occurs when the user releases a key on the keyboard. By the `<input type="text" onkeyup="*Function()">`, it is possible to execute a function when the user releases a key. By executing the function that is programmed in the text live search engine, the HTML form gets the name of the target street/segment. The client-side gets access to the PHP file and sends the name using the 'Post' method of HTTP request. The rest of the process is done on the server-side and will be explained in the server-side section.

#### **4.3.3.4.2 Closing process on the server-side**

Briefly, in terms of technical aspect to implement the closing process on the server-side, the administrator transmits the name of target segment to the server by submitting it to a target file (PHP file) on the server-side. The PHP file gets the name and connects to the RDBMS. Once the connection is opened, a query is made to change the status of the target segment from open to close situation. After changing the status, another query is implemented to add up one number to the ID column for each segment. This is necessary for a real-time routing service because the comparison of ID column in specific time intervals shows the database changes. After implementing the previous steps, another query is made to create or replace a View inside the RDBMS. It is clear that for the first time a new View is created and the following times it is replaced. Once the View is created, all the records with 'close' status are addressed to this View and the remaining records are submitted to the created View with not 'close' situation (open). Finally, all the above mentioned queries are executed downward and the connection is closed.

To implement the close process, the administrator can open his/her mobile and close a street or a segment of a street in a couple of seconds. Subsequently, the RDBMS is updated as explained before, a warning message of a closed street or segment in GeoJSON vector format is sent to the user browsers, the new route based on the new conditions that has been sent by the administrator is calculated and the corrected route is sent to the user browsers in GeoJSON vector layer.

#### **4.3.3.5 Opening process**

The opening process is done when a problem that caused the closing of a street or a part of street (segment) has been solved. At this time, the administrator decides to reopen the target street/segment so that it can be used again. In this case, the administrator uses the text live search engine analog to the closing process and submits the name of the target street/segment to the server via a request. The server runs a process to include the previously excluded part for routing. Once the opening process is done, the server includes the opened part in the routing function, removes the live drawing of the closed part (removes the vector layer from user browser), corrects the routing function and sends the corrected route to the user browsers in vector layer format. Like the closing, the opening process contains the client and server-side.

##### **4.3.3.5.1 Opening process on the client-side**

The opening process is done to open a street or segment that was previously closed. The administrator accesses the password protected web site, uses the text live search engine to retrieve the name of the target street/segment, submits it and the other tasks are done by the server-side. To avoid rehashing and overstating, the explanation of the client-side is withdrawn because it is like the closing process and the only difference is that the administrator presses the 'open' button instead of the 'close' button.

##### **4.3.3.5.2 Opening process on the server-side**

The opening process on the server-side is like the closing process. The only difference is that the situation of the segment is set to 'open', the 'open' records are addressed to the 'open' View and the remaining records are addressed to the 'close' view in RDBMS.

#### **4.3.3.6 Sending and removing the vectorial warning**

This part of NCS was designed to inform users once a street/segment is closed by sending the street/segment to the user browser in GeoJSON vector format. The NCS corrects the previously selected route for users when a street/segment is either closed or opened. Without this module, however, the users do not know which street/segment has been closed. The users are even unaware if the whole street or a small subset of a street is closed. The system was designed in a way that warnings are constantly updated and opened parts are removed automatically from the user's browser.

Description of sending and removing vectorial warning like other parts in this chapter has been divided into the client and server stages.

##### **4.3.3.6.1 Vectorial warning on the client-side**

On the client-side, the vectorial warning including characteristics, strategies, map styles, target file on the server-side, format, vector type and displayed name of the vectorial warning are defined in the OpenLayers programming language. The strategy of the layer is defined so that the user browser checks the server updates every  $n$  seconds, draws the newly closed layer and removes the opened streets/segments from the client browsers.

In the written codes on the client-side, a variable is defined to store the closed parts of the network in vector format with a refresh rate of  $n$  seconds. The style of the map including color and width is declared. An HTTP protocol sends a request to the PHP file on the server-side to calculate and send the newly closed street/segments in GeoJSON format. By connecting to the PHP file, the client-side gets the closed parts from the updated database every  $n$  seconds and displays them. To display the variable as a layer on the browser, it must be included in OpenLayers maps.

#### **4.3.3.6.2 Vectorial warning on the server-side**

The server-side retrieves the closed parts whenever the client-side connects and requests it. The server-side delivers the result in GeoJSON vector format to the client-side for displaying.

To program the vectorial warning on the server-side, the database connection parameters to define database name, port number, table name, username/password of the database are set. A variable is defined to declare the SQL/PostGIS query. By variable definition, the 'close' View in the RDBMS (created or replaced whenever the administrator closes or opens a street/segment) is selected as GeoJSON layer and a GeoJSON array is created to return the query result that previously done. In the next step, the closed segments ('close' View) are returned to the GeoJSON array. To return the segments to the GeoJSON array, a While loop and Pg\_fetch\_assoc functions from PHP and PostgreSQL programming are used, and each feature array is added in FeatureCollectionArray to aggregate all edges in a GeoJSON layer. Finally, the database connection is closed and the result of all above mentioned process is delivered in GeoJSON vector format to the client-side (user's browser).

#### **4.4 Automatic live correction of the best found route**

An automatic correction process happens when the user makes use of the real time routing service and has found the best route between the start and destination points and the administrator closes or opens a part of the network. At this time, the system sends a warning as a vector layer and corrects the route automatically. This facility in real-time services prevents the users from wasting time by driving into a closed section of the network. As mentioned before, this process requires a large number of calculations on the server-side. Hence, to program this part of research, different programming scenarios were tested and the best one with the least amount of calculations and the minimum response time was selected for implementation.

As a first step, an investigation was carried out to identify different programming scenarios on the client and server-side using open source technologies. Secondly, the different possible ways were programmed and executed on the server and client-side and their



performances were assessed. In the assessment stage, two parameters were considered: the number of simultaneous users that can be served by the system and the respective response times. In UNIX-based platforms, these parameters could be provided by the 'load average' parameter.

One of objectives in this research is the assessment of the different methods and their effects on the load average in real time routing services. In this part of the research, the possible programming scenarios are investigated in order to maintain a real-time system and select the method in which the minimum load average be achieved.

#### **4.4.1 Load Average parameter**

"In UNIX computing, the system load is a measure of the amount of computational work that a computer system performs. The load average represents the average system loads over a period of time" ([http://en.wikipedia.org/wiki/Load\\_computing](http://en.wikipedia.org/wiki/Load_computing), 2012). On Linux operating systems the load average is a criterion to define the server capability to answer the simultaneous requested calculations. "The performance benefits of Load balancing are indeed strongly dependent upon the Load index. Load indices based on resource queue lengths are found to perform better than those based on resource utilization" (Ferrari & Zhou, 1988). Load Average is measured during the last one, five and fifteen minute-periods. The load averages greater than means that all the calculation requested from the server can't be done immediately and some of them would be done with delay. For multi-CPU systems the real load average values represented by the system must be divided by the number of CPUs and then compare to 1 (<http://blog.scoutapp.com/articles/2009/07/31/understanding-load-averages>).

Assessment and monitoring of the load average on the server is very important and a critical parameter in real time systems. As mentioned before, spatial analyses are computationally expensive and subsequently cause a high load average. In spatial information based web services, is a vital point to keep the load average below 1. Spatial analysis requests a large of number of calculations at any particular time. Hence, it needs a high capacity processor on the server-side and efficient algorithms to do the job with the least amount of computations.

#### **4.4.2 Live correction methods**

As shown above, the best programming scenario is the method with the least amount of calculations and subsequently the smallest load average is imposed on the server.

To figure out the possible programming scenarios, this part of the thesis work suggests on a model in which the following tasks can be implemented whenever the administrator updates the database system:

1. The model must consider the new updates that happen in the database. The new closed or opened subsets of the network (streets or segments) must be counted.
2. A vectorial warning is sent to the online clients to inform about the newly closed or opened parts.
3. The model must retrieve the coordinates of the start and destination points, transform them to the database spatial reference system and sends them to the server
4. The model must define the triggers that request the server to re-execute the routing function. Triggers may be run by the user, administrator or automatically by the system depending on the model.
5. The model must calculate the updated best route based on the new conditions, remove the previous route and redraw the corrected route for online users.

The suggested model for this step of the research must be able to implement the above mentioned conditions. Hence, the research focuses on selecting and programming possible scenarios by open source web programming languages, assesses the programmed models and selects the best one in which the smallest load average can be achieved. At this stage, an investigation was carried out to determine the possible ways to correct selected routes. The following models were designed and programmed:

##### **4.4.2.1 Submit-correct method**

With this method, a new route is calculated and sent to users every time the administrator closes or opens a part of network. To implement this method, the server must save the latest

start and destination point coordinates for each user. Whenever the administrator updates the network, it must retrieve the latest start and destination points, calculates the new route and sends the route back to each user browser.

This method imposes a high load average on the server because it has to instantly calculate and return the new route to all users simultaneously. All calculation and sending processes take place in a fraction of a second because once the administrator presses the submit button, the server has to calculate the new routes for all users simultaneously and to return the respective results to clients. Moreover, the user browsers have to constantly send the start and destination point coordinates to the server and the server must remove the previous coordinates for each user and save the new ones because each user may select different start and destination points. Hence, the server gets involved in a demanding cycle. Regardless of other aspects, the server is overloaded many times due to the fact that all calculation have to be performed on the server at the same time. The main disadvantage of this method is that the server must run all processes simultaneously and parallel and then has to stay idle until the administrator submits a new closing or opening process.

#### **4.4.2.2 Interval-Execution method**

This method forces the routing process to execute in constant time intervals. In this method each user browser sends a request to the server to calculate the new route and to deliver it to the client browser every  $n$  seconds. The following sections describe the implementation of this method.

##### **4.4.2.2.1 Programming Interval-Execution method on client-side**

To program this model on the client-side the following steps were done:

- 1- Define a three argument function as the routing function. The routing function includes three arguments: Store, Layer and Method. These arguments as the function variables implement the necessary tasks for the function. The function removes the previous route if there is a new one, transforms the geometries of the start and destination point to the

geometry database, loads the store and gets the method from the client combo box to send it to the server. The server uses the selected method to perform the routing process. As mentioned before, the routing functions contain three arguments: Store, Layer and Method as explained here:

- Store: creates a store as a `GeoExt.data.FeatureStore` to connect to the server and request it to query the routing process. The Store connects to the PHP file on the server using HTTP protocol, gets the result in GeoJSON format, converts the coordinates and puts the result by specific fields into the layer argument.
- Layer: to display the start and destination point layer: an OpenLayers vector point.
- Method: the Method argument gets the value of the Method variable from combo box selected by the user via the browser (when the routing function is executed, it gets the value of the Method variable for the Method argument).

2- Assign the value of the each argument to the relevant variables. By this stage, the JavaScript functions know how the variable values have been calculated when the routing function is called. By calculating the variable values assigned to the Store, Layer and Method arguments, the routing function can be calculated whenever recalled.

3- Recalling the routing function by Interval-Execution method:

In the first step, the routing function is defined by the three arguments and in the second step, the related variables are defined. In the third step, the routing function is recalled using pre-calculated arguments (`Store`, `points_layer`, `Method.getValue()`) in particular time interval methods.

```
setInterval( function() {  
    routing(store, points_layer, method.getValue());  
}, n );
```

#### **4.4.2.2 Programming Interval-Execution method on server-side**

To request the server for calculating the new route and deliver it to the client-side in GeoJSON vector format, a PHP file with PHP, SQL and PostGIS scripting languages is used to query the requested routing algorithm, to convert it to GeoJSON and to deliver it to the created Store. Then the Store wraps the result inside the line vector layer in OpenLayers to draw it on the user browsers. This step is quite the same as on the server-side programming stage to implement a simple routing service.

#### **4.4.2.3 Check-Change-Correct method**

As a first step, a request is sent by the client to the server to realize if any updates have happened since the latest check. If any update has taken place, another request is sent to the server to calculate the new route and deliver it to the client-side. To implement this method an attempt was done to consider the most efficient way for programming and implementing this method. After a number of tests on both the server and the client-side and after a thorough evaluation of the possible scenarios, conceptual architecture of the method was designed (Figure 4.7).

The process of programming this method consisted of six stages. To perform a practical way to run this architecture over the web, an integrated AJAX (Asynchronous JavaScript and XML) and JQuery approach is needed. AJAX and JQuery are the programming languages to perform such type of tasks with AJAX, JQuery, JavaScript, GeoEXT, OpenLayers, HTML5 and EXT on the client-side, and PHP, SQL and PostGIS on the server-side. XML and GeoJSON were used as exchange formats to program and implement the Check-Change-Correct method on the Linux server platform. Basis for implementing the conceptual framework, in-depth knowledge about JavaScript functions, JQuery including functions, call back functions, Deferred Objects like Deferred.done to add handlers was needed. Apart from these, to program the conceptual architecture we also need deep understanding of PHP-SQL, PostGIS and specially, GeoJSON programming languages.

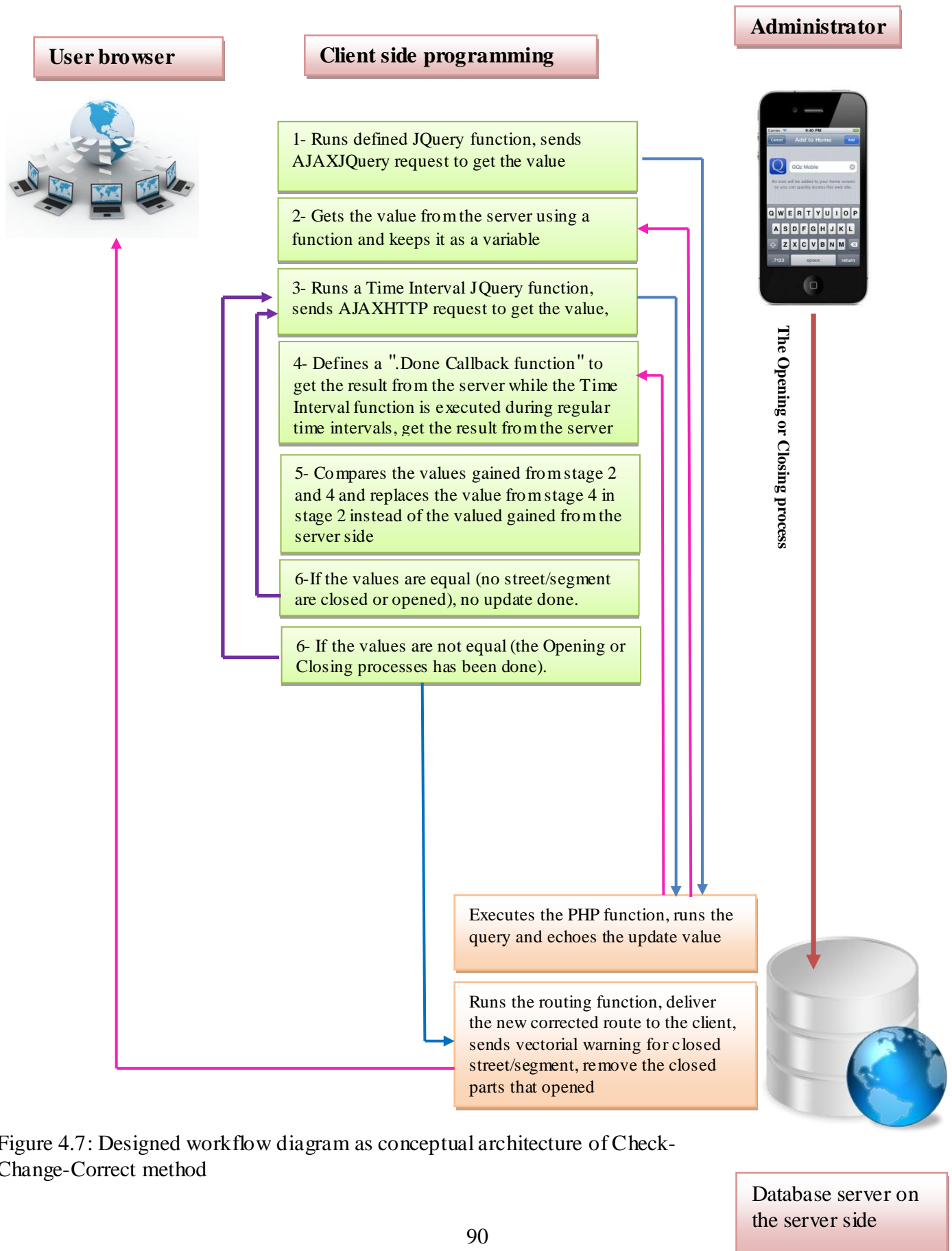


Figure 4.7: Designed workflow diagram as conceptual architecture of Check-Change-Correct method

#### 4.4.2.3.1 Programming Check-Change-Correct method on the client-side

The essential programming languages needed to implement this method were mentioned before. To program the method on both the client and the server-side, we take the workflow diagram as the conceptual architecture. To program the client-side, the following steps were taken:

- 1- Load the update value from the server using HTTP request

```
function calculate()
{
    var id;
    $.ajax({ type: "GET",
    url: "testrealtime.php",
    2async : false,
    success: function(result) { id = result; }
    });
    return id;
}
var x= calculate();
```

As seen in above codes, a JavaScript function is defined to connect to get the ID column value and Store it in a predefined variable via calling the predefined function. This action is done by a PHP target file on the server-side using AJAXHTTP request (GET). Once the user opens the browser to use the routing service, this stage is implemented by the browser to store the situation of the database as id value on the client-side. During the next steps this value is constantly changed in specific time intervals with the new values to warn the user browser about the database updates. If the value that is returned from the function is identical to the new value, no update has happened, otherwise, an update has happened by the administrator and the browser has to request the new route from the server-side.

- 2- Call the defined function and store the returned value

After loading the returned value from the server by the JavaScript function (calculate), it is the time to call the function. Here, the function is called and stored as a global function. It must be mentioned that this process is done just once while the user keeps the browser open. If the user opens a new tab or refreshes the web page, the function would be executed again to get the update value and store it as a global JavaScript variable.

```
var x= calculate();
```

### 3- Define a time interval function

At this stage, a time interval function is defined. This function is executed at the specified time intervals.

```
setInterval(function() {  
  
$.ajax({  
  
url: "/*.php",  
})
```

There are two types of timing event functions in JavaScript: SetTimeout and SetInterval. SetTimeout "executes a function once after waiting a specified number of milliseconds" while the SetInterval method waits a specified number of milliseconds, and then continues to execute the function, once at every given time interval ([http://www.w3schools.com/js/js\\_timing.asp](http://www.w3schools.com/js/js_timing.asp)). Here, the function retrieves the update value from the server by AJAXHTTP request.

### 4- Add a handler to be called when the Deferred object is resolved (adding callback function) inside the TimeInterval function

```
.done(function(result from server) {
```

Since the programmed codes at stage2 returns JQuery XHR(XMLHttpRequest) object derived from a Deferred object( refer to <http://api.jquery.com/category/deferred-object/>), it



is possible to add a callback function by the .done method. This could be a single or array of functions called when the Deferred object is solved. Once the Deferred object is solved, the .done callback function/s would be called. Here, the callback function contains an argument (result from the server). When the AJAX request is performed, the result comes from the server. Then the .done callback function is called and the returned value from the server is assigned to its argument as update value.

5- Add conditional express to detect updates and call the routing function

```

        if(result from server > x)
        {
            pgrouting(store, points_layer, method.getValue());

```

If clause is located inside the .done callback function to compare the returned value coming from the JavaScript function (stage 1) with the value returned from the callback function. If the value from the callback function is greater than the value from JavaScript, the updating process has been initiated by the administrator. It means that a street or a segment has been closed or opened. Hence, the routing function must be re-executed to remove the current route and replace it with the corrected one.

Routing functions were discussed in detail in the previous sections.

6- Assign the value returned from the callback function to variable x instead of the JavaScript function

```

x = result from server;

```

This process is done inside the callback function and assigns the returned value from the callback function to the variable x. By this process, the callback saves the update value for the next time the TimeInterval function is executed after n milliseconds to compare with the update value that comes from the callback function. Without this step, the If clause statement would always compares the update value gained from the callback with the JavaScript function value. It was mentioned that we need the JavaScript just for the first time of the TimeInterval function execution. It happens when the client opens the browser. For the next executions of the TimeInterval, the value returned from the callback function in

the previous execution is used. It gives possibility to recognize updates that happen into the RDBMS on the server.

7- Close the TimeInterval function

`}, n);`

At this stage, the interval time is defined. It initiates an execution of the TimeInterval function every n milliseconds. This gives the possibility to check the updates uninterrupted from the server.

#### **4.4.2.3.2 Programming Check-Change-Correct method on the server-side**

Programming of this method on the server-side consists of three different parts: a script for returning the update value, a script for the routing function and the sending of vectorial warnings to the clients. The routing function and vectorial warnings have been explained before. Here, the written script to get the update value from the RDBMS is explained.

The ID column in the database on the server-side is updated once the administrator closes or opens a part of the network. A counter has been programmed in both the opening and closing process to keep the ID column update. When the administrator updates the database, the counter adds up one to the ID column. By this stage, the client-side detects if any updates have happened.

On the server-side a PHP function is defined to return the value of ID column to the client-side whenever is called by the user's browser. Inside the function an SQL query is executed to connect to the RDBMS and gets the ID value. A variable is also defined to store the ID value of the first record from an array of equal ID values. Finally, the predefined function returns the variable value to the client-side.

#### **4.4.3 Assessment of the correction methods and selecting the most efficient algorithm**

As mentioned before, the load average is a metric to evaluate the server performance when two or more algorithms can perform the same job. In this research, the load average parameter is used to test the efficiency of each correction method. Hence, we calculated the load average for the Interval-Execution method and the Check-Change-Correct method. The submit-correct method was omitted from the assessment process because it imposes that all updating and correction processes for all online users have to be performed simultaneously at once on the server. To select the best method, the load average for each method was monitored during 1, 5 and 15 minutes time intervals for a specific configuration that was available for the tests. The method with the minimum load average is selected as the best. The server hardware specification is as following:

- Processor: Intel(R) Core™ 2 CPU 6300@ 1.86GHZ 1.87GHZ
- RAM: 3 GB
- Platform: Linux
- Operating system: Ubuntu 12,04 server
- System type: 64bit

##### **4.4.3.1 Monitoring the load average parameter for Interval-Execution method**

The load average is shown by the ‘Top’ command on Linux servers. It was calculated for 1, 5 and 15 minutes. To evaluate this method we need a group of users. To monitor the Interval-Execution load average values, 10 users were selected. The users entered the web service website ([WWW.persiangulf.co.nr](http://WWW.persiangulf.co.nr)) and started using the real-time routing service simultaneously. For monotony in results, the users also did the same things for the Check-Change-Correct method. Each user requested the same routes in terms of the start and destination points and the same number of requests for both methods.

The load average values for the server used for this research while it is almost idle are 0, 0.1 and 0.05 respectively.

After 15 minutes, the load average values were recorded (Table 4.1). For more simplicity, the values are multiplied by 100 to remove decimal values. As mentioned before the used server has two processors hence, the loading capacity for the server is 200. It means that values between 0-200 indicate that there are no backups for the requested calculations and all requests can be done immediately. For values above 200, the requests must be queued and answered with delay depending on the load average value. Greater load average values cause longer delay in response.

A real-time system like the one to be designed and implemented in this research, needs load average values smaller than 200. As seen in table 4.1, the Interval-Execution method yields almost all values greater than 200. This method can be used with 1 or 2 users but not in the real world.

Table 4.1- Load Average values for Interval-Execution

time minutes	1 min	5 min	15 min
0	0	1	5
5	582	294	118
6	574	332	139
7	565	375	166
8	501	391	184
9	438	394	198
10	325	373	203
11	173	322	196
12	120	278	109
13	130	251	185
14	307	276	198
15	406	309	214
16	451	342	231
17	626	412	262
18	534	426	277
19	634	479	305
20	536	482	317
21	669	527	342
22	610	531	355
23	681	576	383
24	731	608	405
25	766	637	428
26	876	690	458
27	913	735	488
28	903	765	514
29	901	790	538
30	897	809	560
31	901	825	582
32	917	844	603
33	879	846	619
34	829	838	630
35	869	846	645
36	895	858	663
37	833	847	671
38	695	810	670
39	841	836	688
40	483	742	665
41	301	640	635
42	221	553	605
43	146	473	574

Table 4.2- Load Average values for Check-Change- Correct

time minutes	1 min	5 min	15 min
0	1	7	7
5	117	58	25
6	88	61	29
7	64	58	30
8	36	52	30
9	33	50	30
10	125	71	39
11	105	76	43
12	69	73	43
13	30	61	41
14	15	51	39
15	11	43	37
16	17	40	37
17	34	42	37
18	30	40	37
19	24	37	37
20	41	40	38
21	27	37	37
22	64	51	42
23	81	59	45
24	135	80	53
25	96	80	55
26	94	83	58
27	71	79	58
28	58	75	58
29	43	69	57
30	29	61	56
31	40	60	56
32	58	63	57
33	51	61	57
34	30	54	55
35	50	57	56
36	52	57	56
37	38	54	55
38	31	50	54
39	67	57	56
40	60	57	57
41	51	57	57
42	52	56	57
43	81	63	59

#### **4.4.3. 2 Monitoring the Load Average parameter for Check-Change-Correct method**

For a proper result, the testing conditions for both methods are the same in terms of number of simultaneous users, actions done by users on the same server. During the test time, the administrator was closing or opening the same segments as with the previous method. Table 4.2 presents the load average values for the Check-Change-Correct method with values that are a lot smaller than the ones in table 4.1. By using this newly developed and implemented method, the system can be true real-time.

#### **4.4.4 Comparison correction methods and selection of the best**

Comparing the load average values in tables 4.1 and 4.2, it is easy to realize difference between the methods. As mentioned before, the submit-correct method is the worst in terms of load average because it imposes all processes to be done at the same time on the server. Hence, we just compare the two remaining methods. Figure 4.8 shows the load average values for the Interval-Execution and the Check-Change-Correct methods. As illustrated, the load average values for the Check-Change-Correct method are significantly less than for the Interval-Execution. As seen in the server overload line, almost all values for the Interval-Execution method are above the server overload capacity. This means that the incoming calculations must wait in the queue. For example in the test time 32, the value is 917 with the highest server capacity of 200. The remaining requests must wait in the queue until the server gets rid of the current calculations. This test was done just for 10 users but in reality a lot more than 10 users must be served by the system. The load average value for the Check-Change-Correct method at the same point is 32. For this case, the server has 168 free capacities. Hence, the Check-Change-Correct method is much less demanding in terms of the needed calculation capacity and, consequently, this method is more reliable for a real-time service. Our complementary test shows that the Check-Change-Correct method is about 50-100 times more efficient than the Interval-Execution method depending on the traffic to be updated. Therefore within this research, the Check-Change-Correct method was selected as the correction method to integrate with the other components of the NCS to generate the service system.

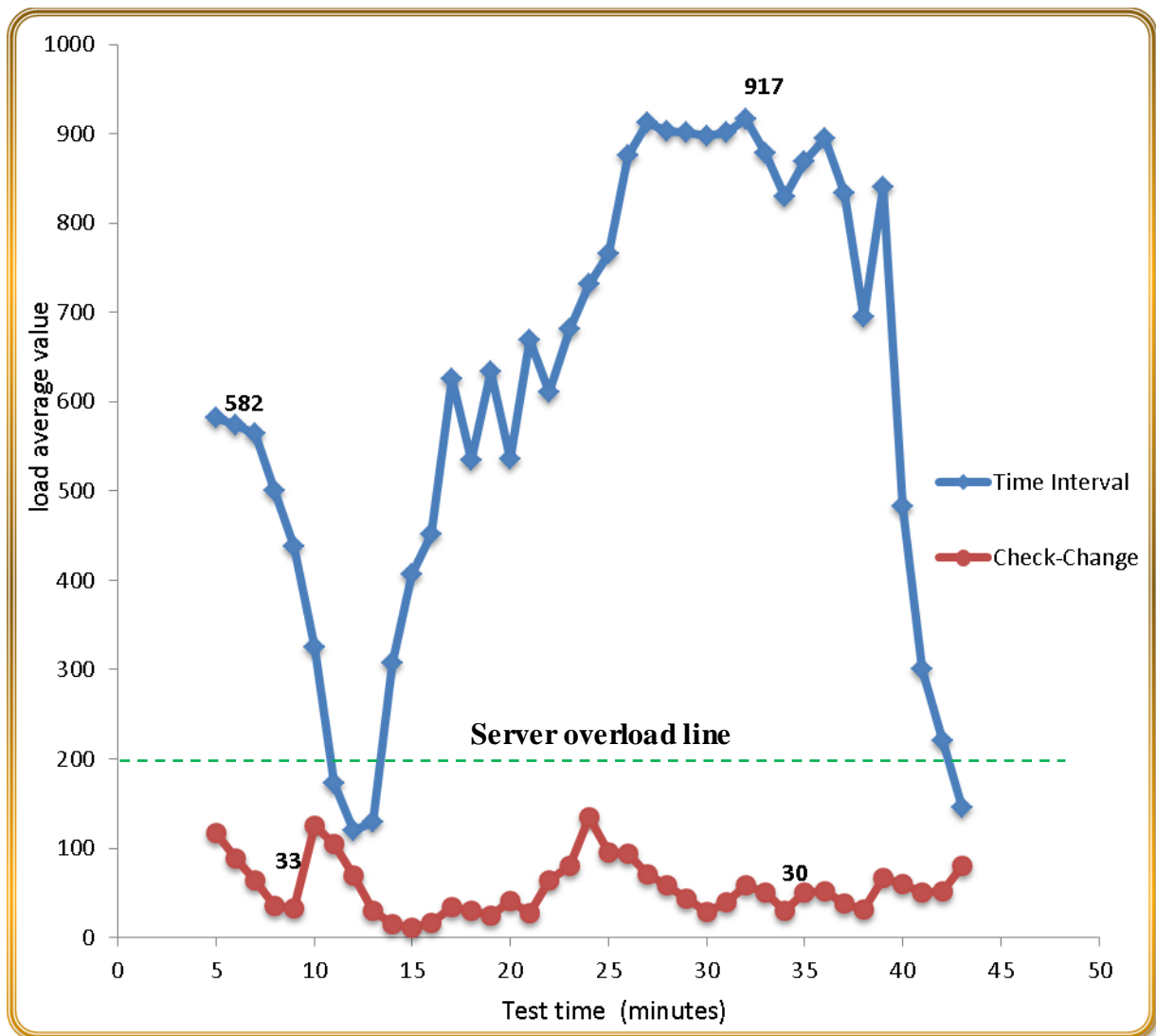


Figure 4.8: The load average values graph for the correction methods

#### 4.5 Integrating all components together to serve the real time routing service over the World Wide Web

Finally, to serve the designed web service over the web, the server offered by University of Osnabrück was the hardware for this thesis project. The IP of the server was registered, three domains for real time service were offered to users, and the controlling system for the administrator and for the vectorial search engine were registered. The client-side files

extension is HTML5 and the server-side is in PHP format. The used RDBMS for this research is PostgreSQL along with PostGIS and routing functions. The web service for all users is accessible at [WWW.persiangulf.co.nr](http://WWW.persiangulf.co.nr). Everybody is entitled to go this website and use it. Another domain name was registered for the administrator to control the system and keep the service real-time. This website is username/password protected and officials like the administrator are able to enter and control the service. Figure 4.9 shows the workflow diagram of the NCS and Figure 4.10 the final real-time routing system interface for regular users.



Client browser

Administrator

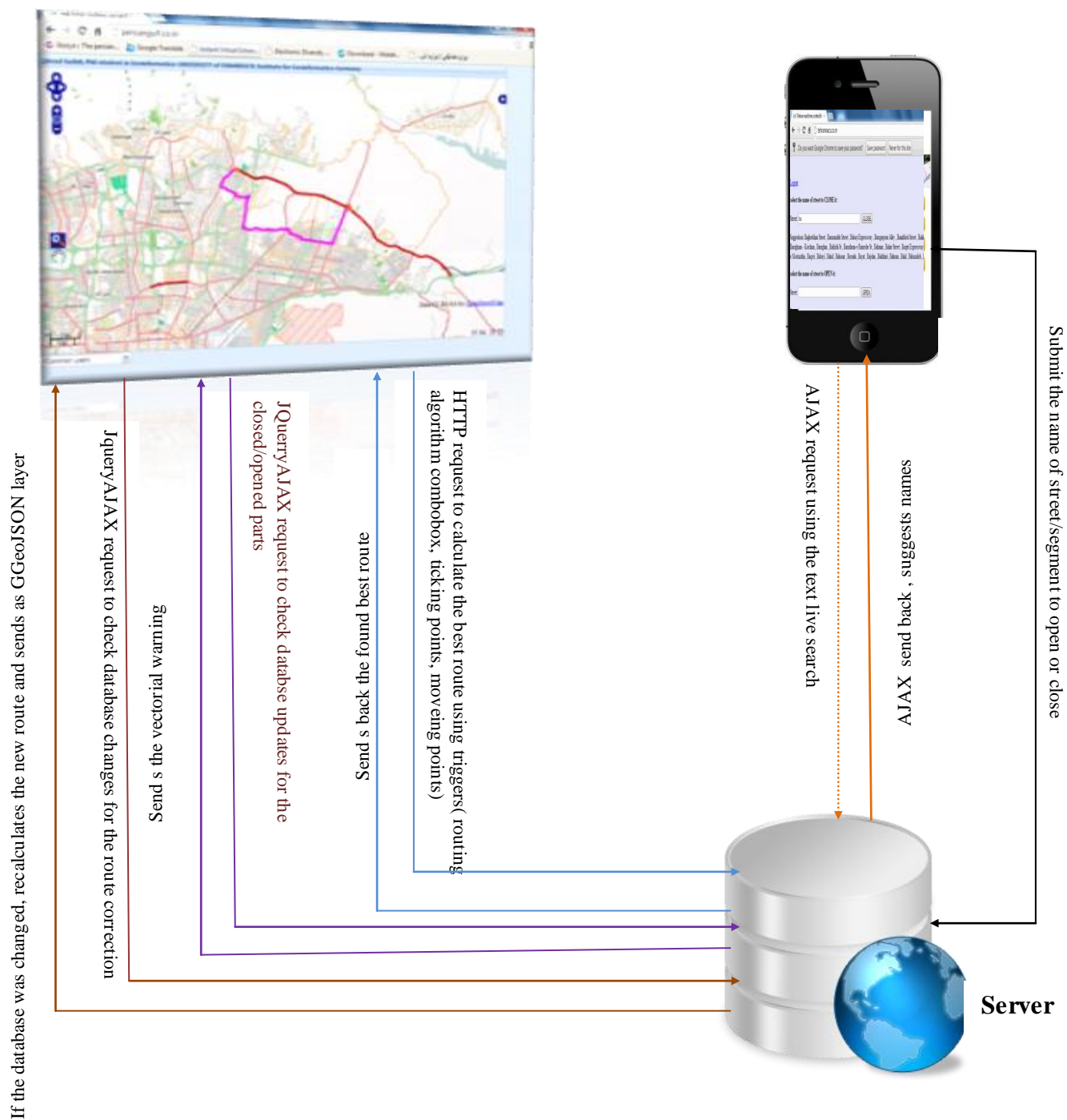


Figure 4.9: Workflow diagram of the NCS to conduct the real-time routing service over the Web

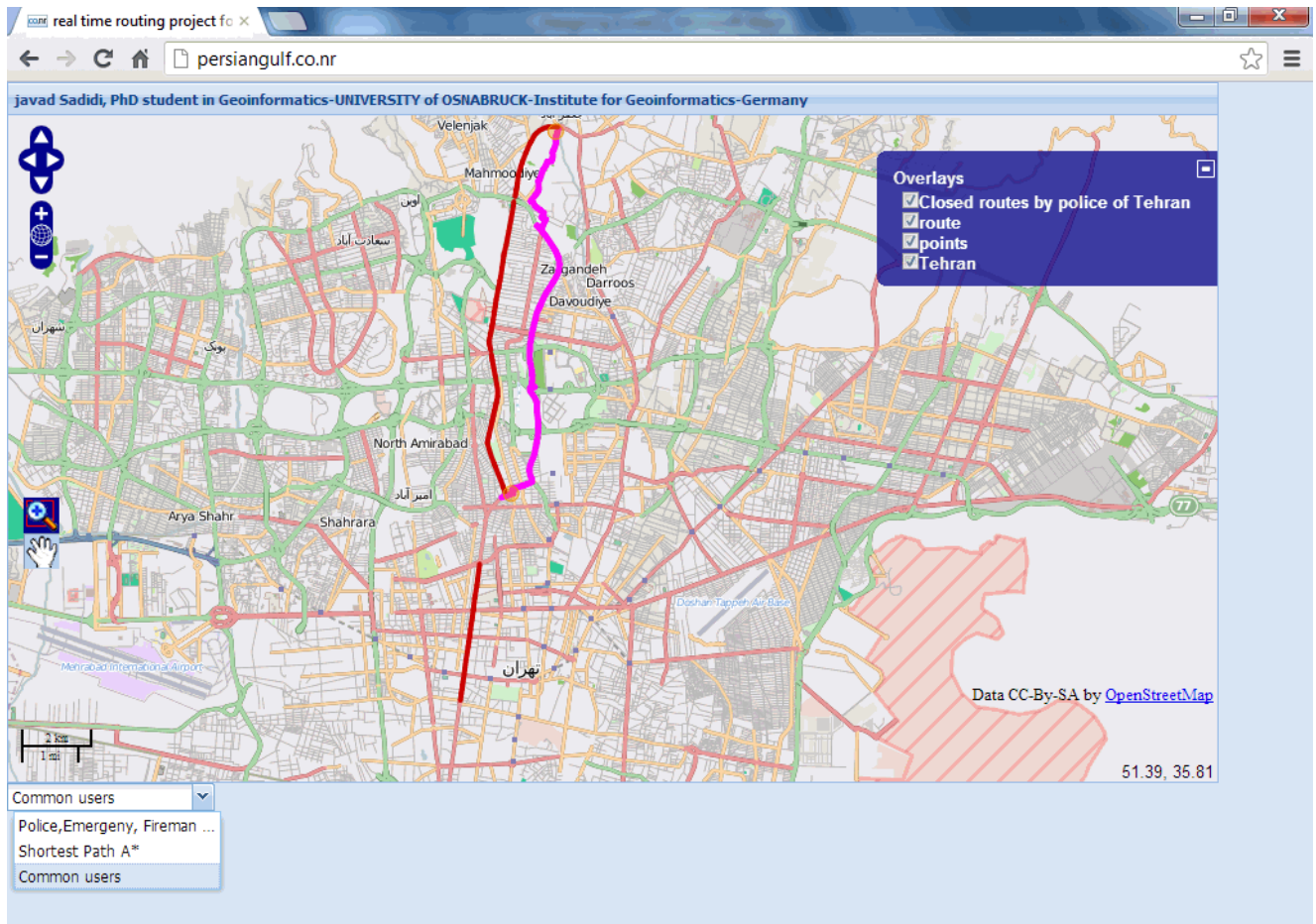


Figure 4.10: The final web-based real-time routing service over the Web

As seen in figure 4.10, the routing service exploits a layer switcher to check or uncheck layers, mouse position in decimal degree, zoom, pan and move facilities. The system offers a combo box in which each type of users can select the proprietary routing algorithms. For example, the officials like police, emergency services and firefighters who have authority to neglect traffic laws can use the special algorithm which considers the shortest path (Dijkstra). In contrast, the common clients like normal people use a heuristic algorithm (ShootingStar).

A vectorial warning is sent to online users whenever the administrator closes or opens a specific part of the network and, concurrently, the correction Check-Change-Correct algorithm, suggested for the first time in this research, corrects the selected route and submits the new route to the clients as a real-time routing service.

# Chapter 5

## Results and Conclusion

In this doctoral thesis, a new approach to offer a controllable and real-time web-based routing service has been presented. The newly designed service (NCS), offers an interactive, dynamic and real-time routing service over the web for classified and regular users along with vectorial warning, a vectorial search engine, a route correction system and a controlling service to keep the calculated route real-time for online users. The research has described a new method to eliminate the disadvantages of the current routing services. This research attempts to integrate the current routing technologies with the newly designed system (NCS) in order to attain a real-time routing service that could be of great benefit to both classified and unclassified users.

This new conceptual and practical model is a step forward in controlling the spatial web services every second and, subsequently, offers a realistic, efficient and lightweight real-time, web-based service. The research product is applicable in large and small cities to decrease the travelling time and to keep routing real time despite sudden accidents and traffic jams. In cities like Tehran it can potentially reduce pollution (noise, car exhaust, etc.), and travelling costs. It enables emergency vehicles to get faster to accident locations and save lives in this process. Additionally, the potential applications of this research are not limited to street networks. NCS can be used in a large variety of applications related to network systems such as water, gas pipes and other network systems.

This chapter will now recapitulate the results of the thesis and their significance. A conclusion with prospects for further research will also be presented.

## **5.1 Summary of the results and discussion**

The following texts summarize the strategies and basic approaches applied during the research with respect to the objectives:

### **5.1.1 Conceptual model of the NCS**

To design the conceptual model, a study was conducted on current routing services and the disadvantages of these systems were identified. Also, a number of interviews of traffic experts in Tehran were conducted to realize the needed components for the conceptual model. A study on open source software and technologies followed this part and the conceptual model of NCS was designed. Finally, the architecture of NCS was transformed from paper design to computer codes using web programming languages.

### **5.1.2 Description of the NCS**

The NCS is divided into the administrator and the user side. This aspect is mirrored by the programming of server and client components. Although, exchange formats like XML and GeoJSON act as brokers to make connections between the server and client-sides, there is no need for advanced understanding of software and programming to use or administer the service.

As mentioned before, the difference between the NCS and the current services like Google Maps routing service is that it can be offered and managed in real time by the respective authorities and that it provides different routing methods for classified users.

The designed NCS consists of the following components:

1. A secure administrator page acts as entry page to update the database system: PHP plays a key role to program this component. Include and associative array functionalities of PHP along with other functions were used to program the secure admin page.

2. A text live search engine to search the target street/segment: This module transplants the administrator page to the RDBMS. It is used to close or open a target street or a subset by submitting action on the browser. JavaScript, AJAX and PHP functions are used to program this part of the research.
3. A vectorial live search engine to search and locate a vector object: This module of the research is used to search and locate a street and its subsets on Google Maps on the browser. This component is very useful to find and locate small street subsets to consider its neighbor environment on the browser linked to Google Maps or other map servers. It helps the administrator to have a synoptic view of the target part of the network and neighboring subsets and streets. JavaScript, PHP, GeoJSON, GeoExt, SQL and PostGIS were used to program and run this component.
4. Live drawing and removing for the closed streets/segments: This module is designed and programmed to send vectorial warnings for online users. By this, the clients are warned about the closed parts of the network.
5. Live correction algorithm. This component of the research was the most demanding and time consuming part of the research. In this research three new correction algorithms were programmed and evaluated. Then the best algorithm was selected to be embedded in the NCS project. This facility corrects the selected route by online users once a street or a subset of a street is closed or reopened.
6. By integrating the above mentioned components with a simple routing service from FOSG2010 Pgrouting project that includes a module to offer proprietary routing algorithms for classified users, an efficient and user friendly real-time routing service for use as a regular client or for controlling it as an administrator were designed and programmed. The complex aspects of the research are not visible in the intuitive user interface for both, clients and administrators. Using the designed service is so simple that an administrator or a user with only simple reading capacity can use and even control the network.

### 5.1.3 Advantages of the NCS

This real-time routing service offers number of advantages compared to the current systems including:

- The administrator like a policeman can control and update the real-time system when he/she is outside the office in field missions. He/she opens a mobile device, enters the security protected controlling site and closes or opens a specific part of the street without having to inform or access a control center. Once the administrator does any update, the server-side sends a vectorial warning and corrects automatically the selected best route for online users. Using the designed controlling system, the mobile administrator can close a specific part of a street or a segment as soon as an accident occurs or another obstacle prevents users to pass a street. It is important for local administrators because the companies like Google do not let the local administrators access their database.
- The NCS exploits a vectorial search engine to find and locate a special subset of the target street along with its peripheral environment on Google Maps/Earth/Terrain. It gives a possibility to the administrator to close or open a small subset of a street rather than the whole length.
- The designed service also reduces the time of a journey. After any closing or opening of street or segment, the system re-calculates and corrects the past route to the new updated one. This is very important for emergency ambulances, fireman and police as well as normal users for saving their time.
- Finally, integrated system with the simple routing service (provides a possibility to classified users) to select proprietary routing methods and, consequently, a different (faster) route compared to normal users. For example, police, fireman, emergency ambulances and other officials who have the right to neglect traffic laws can use the algorithm that only selects the shortest possible direction regardless of the restrictions. This facility influences the travelling time to reduce it in emergency cases.

#### **5.1.4 Evaluation between the methods for web based live route correction and introducing a new web based live route correction method**

This part of the research was done to correct the calculated route for online users when a street/segment is closed or reopened by the administrator. To achieve this objective, three correction methods were programmed and evaluated using the load average parameter. Since spatial analysis represents a computer-intensive process for the server, the load average parameter is an appropriate metric to evaluate and compare the methods. Hence, the method with the minimum load average factor is the most efficient one to provide the real-time service for large numbers of simultaneous users. Three correction methods have been introduced including: submit-correct, Interval-Execution and Check-Change-Correct methods. The results show that the submit-correct method consumes the highest amount of computer time because it must correct all routes once the administrator submits an update. This method is not a practical method in the real web era although it works theoretically. To select the best algorithm for route correction, the load average parameters for Interval-Execution and Check-Change-Correct methods were measured using 1, 5 and fifteen minute periods. The Check-Change-Correct method provided the smallest load average values as is illustrated in Figure 5.1. The load average values for Check-Change-Correct method are almost ten times less than Interval-Execution method, and never exceed the server overload line of 200 in this case.

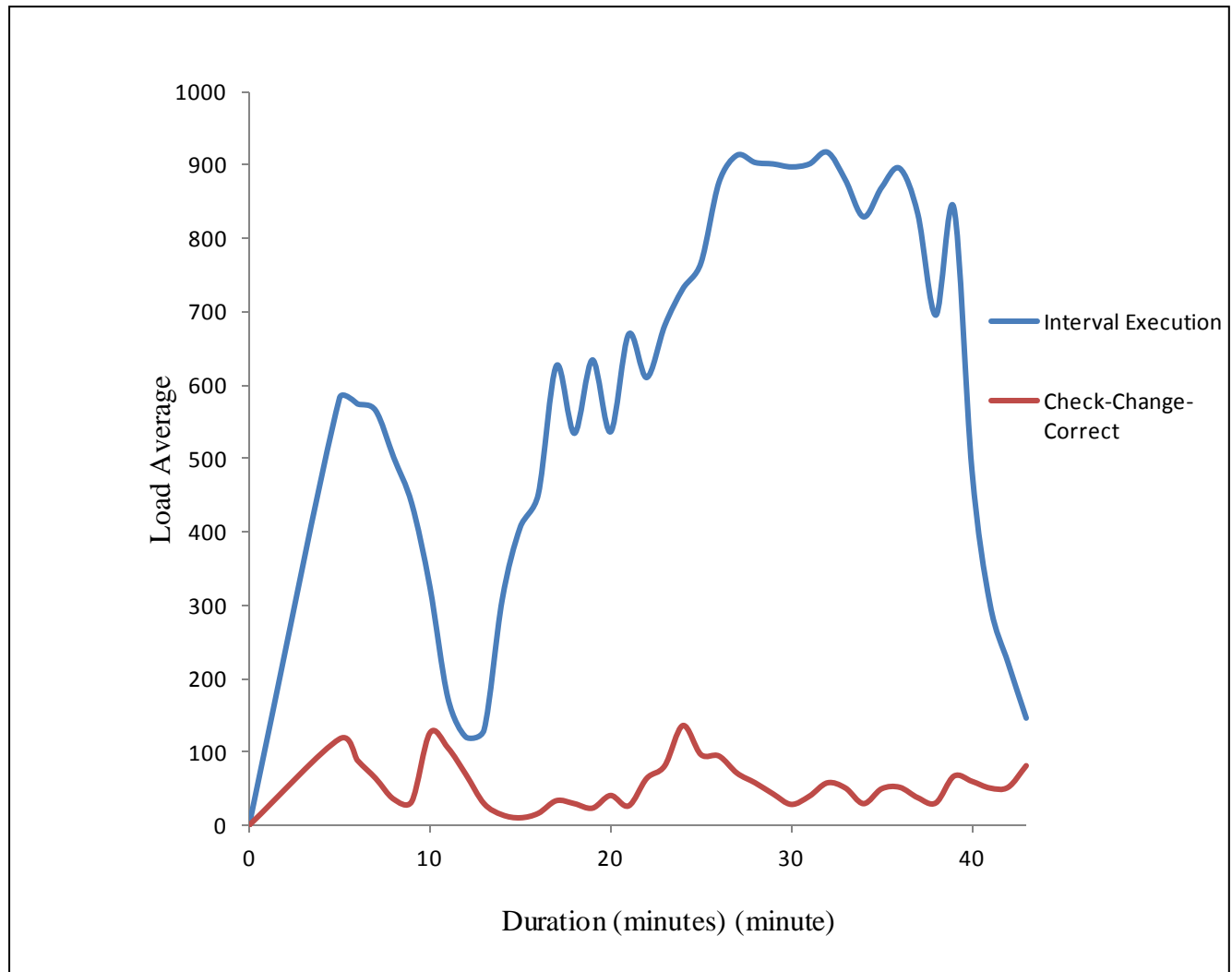


Figure 5.1: The load average values for the one minute period for 10 simultaneous users

## 5.2 Conclusion

In conclusion, this thesis has provided a case study of a web based real-time routing service that was designed, programmed, implemented and executed using solely open source technologies. The integrated system obtained from NCS and the simple routing service differentiates between public users and classified users such as emergency fleet like police, fireman and hospital ambulances with real-time capability. The method presented in this research to upgrade a simple routing service to real time is new, innovative and was first implemented at a server of the Institute for Geoinformatics and Remote Sensing of the



University of Osnabrück (Germany). While the experiment undertaken in this thesis is only related to street networks, it still provides directions for practical considerations that are required and can be used for integrating the NCS with other network applications. There are some considerations on this research that can be summarized as the following:

- The key issue to design an efficient and practical real-time routing service on the World Wide Web in real world is programming an algorithm with the smallest load average possible, because the live correction algorithm put a big computational burden on the server. This is a critical point in designing real-time geospatial web services because the execution of each spatial analysis function consumes a significant proportion of the server capacity. In this situation, with an increasing number of simultaneous users, the probability of delay in response time for the queued requested calculations is increased and would be detrimental to the desired real-time policy.
- There is a need to focus on the scale of the used data for a higher resolution, and to present the spatial relationships in large-scale areas, especially for officials like police and emergency services. For example, each house can have a street address or a zip code and when a person needs help from the police or a fireman, the person can send the code to the police by a mobile phone and the police can enter this code in the real-time routing service. This may help to lead the police to the exact address. This process also helps emergency services to locate the exact address in case of an emergency.
- There is a need to use multi-server platforms and cloud computing for real applications. By using this system, each client connects to the nearest server to request the service. For this, there is a need for skillful programming functions to divide the requested routing jobs between the servers and to lead the requests to the server with free calculation capacity.
- The developed system does not aim to create a tool that can replace the current routing systems like Google Maps; instead, a focus was given to add the lightweight real-time functionality with minimum load average values for real-time applications on the web.

### 5.3 Future Work

The research presented in this thesis can be extended in different directions. Some of these are:

1- Currently, this research is limited to design a NCS for offering real-time capability for web based routing services. However, further work could be undertaken to investigate its uses in other fields including water and gas pipelines.

2- To correct the calculated route, this research has suggested the Check-Change-Correct algorithm. This algorithm exploits JQuery-AJAX as a new technology but further research could focus on the development of this method for the following cases:

- Further development of the algorithm on the client-side as well as on the server-side to recognize if the update (closed or opened part) is spatially related to the client route? If the updated part of the network is not located in the client's way, there is no need to request a new route from the server. This can cause a further decrease in the load average values. Subsequently, the server is able to provide a faster service to more simultaneous users.
- Further development of the algorithm on the client-side to receive and keep free capacity of each server in time intervals to decide to which server the routing request must be sent. This process avoids server overload and submits the requests to the appropriate servers which produces even faster response times in real-time services.

# References

- Anderson, G., and Moreno-Sanchez, R. (2003). Building Web-Based Spatial Information Solutions around Open Specifications and Open Source Software. *Transactions in GIS*.7(4), 447–466.
- Butler, H. (2008). The GeoJSON format specification. <http://www.geojson.org/>. Retrieved on July, 16, 2011, from <http://www.geojson.org/geojson-spec.html>.
- Chertok, Ph. (2010). Introduction to A\* (A-Star) Pathfinding in ActionScript 3 (AS3). Retrieved on September, 3, 2013, from <http://www.untoldentertainment.com/blog/introduction-to-a-a-star-pathfinding-in-actionscript-3-as3-2>.
- Dijkstra, E. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*. 1, 269-271.
- Ferrari, M. and Zhou, S. (1988). "An Empirical Investigation of Load Indices for Load Balancing Applications. Proceeding of the 12th Int'l Symposium On Computer Performance Modeling, Measurement, and Evaluation, North Holland Publishers, Amsterdam.
- Fischer, M. (2003). GIS and Network analysis, proceeding of Ersä 2003 congress, Vienna.
- Fu, P., and Sun, J. (2010). *Web GIS: Principles and Applications*. Redlands: ESRI Press.
- Garrett, J. (2005). *Ajax: A New Approach to Web Applications*. Adaptive path. Retrieved on May, 10, 2011, from <http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications>.
- Geisberger, R. (2008). *Contraction Hierarchies: Faster and Simpler Hierarchical Routing in Road Networks*. Master thesis, university of Karlsruhe, Germany.
- Hart, P., Nilsson, N., and Raphael, B. (1968). A formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics* SSC4 4 (2), 100–107.

- <http://api.jquery.com/> (1, September, 2013). JQuery.  
<http://api.jquery.com/category/deferred-object/>.
- <http://developer.mapquest.com/> (12, March, 2013). Mapquest developers.  
<http://developer.mapquest.com/web/products/open/directions-service>.
- <http://dictionary.reference.com/> (13, March, 2011). Dictionary.  
<http://dictionary.reference.com/browse/html>.
- <http://httpd.apache.org/> (20, February, 2012). Apache HTTP server project.  
<http://httpd.apache.org/>.
- <http://oxforddictionaries.com> (10, April, 2011). Oxford dictionaries.  
<http://oxforddictionaries.com/definition/Internet>.
- [http://wiki.openstreetmap.org/wiki/Main\\_Page](http://wiki.openstreetmap.org/wiki/Main_Page) (01, March, 2013). OpenTripPlanner.  
<http://wiki.openstreetmap.org/wiki/OpenTripPlanner>.
- [http://wiki.openstreetmap.org/wiki/Main\\_Page](http://wiki.openstreetmap.org/wiki/Main_Page) (15, March, 2013). Traveling salesman. [http://wiki.openstreetmap.org/wiki/Traveling\\_salesman](http://wiki.openstreetmap.org/wiki/Traveling_salesman).
- <http://www.desktoplinux.com/> (14, March, 2011).  
<http://www.desktoplinux.com/articles/AT5836989728.html?regen=yes&display=yes>.
- <https://code.google.com/p/astar-search/> (1, November, 2013). astar-search.  
<https://code.google.com/p/astar-search>.
- <https://developers.google.com/> (31, October, 2013). Google Maps API.  
<https://developers.google.com/maps/documentation/>.
- Lan, Y., Zhipeng, G., Wei, G., Shengyu, S., and Huayi, W. (2012). A geospatial web services composition framework supporting real-time status monitoring. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume I-4, , proceeding of XXII ISPRS Congress, Melbourne, Australia
- Li, S., Saborowski, J., Nieschulze, J., Zeng-yuan, L., Yuan-chang, L. and Er-xue, C. (2007). Web service based spatial forest information system using an open source software approach. Journal of Forestry Research. 18 (2), 85-90.
- Melby, J. (2012), How to Compare Vehicle Navigation Systems. Retrieved in 2012, from <http://www.roadmapgps.com/articles/a0110.shtml>.
- Miller, M., Drazen, O., and Medak, d. (2010). Facing the Challenges Building the Capacity. Proceeding of FIG Congress, Sydney.

- Percivall, G. (2010). The application of open standards to enhance the interoperability. *International Journal of Digital Earth*. 3(1), 14-30.
- Raghavan, V., Santitamont, P., Masumoto, S. and Honda, K. (2002). Implementing Web GIS Applications using Open Source Software. Map Asia conference, Bangkok.
- Roberts, A., Park, G., Melland, A., and Miller, I. (2009). Trialing a web-based spatial information management tool with Land Managers in Victoria, Australia. *Journal of Environmental Management*. 91(2), 523-531.
- Schaub, T., Doyle, A., Daly, M., Gillies, S., and Turner, A. (2008). GeoJSON draft version 6. [http://wiki.geojson.org/Main\\_Page](http://wiki.geojson.org/Main_Page). Retrieved on 2 November 2013, from [http://wiki.geojson.org/GeoJSON\\_draft\\_version\\_6](http://wiki.geojson.org/GeoJSON_draft_version_6).
- Uan, M. (1997). Development of a Global Conceptual Schema for Interoperable Geographic Information. The International Specialist Meeting on Interoperating Geographic Information Systems, sponsored by NSF and OpenGIS, Santa Barbara, CA.
- Venneman, K. (2009). Implementing Web GIS Solutions using Open Source Geospatial Software, proceeding of GIS User Conference, Sunriver, Oregon.
- [www.blog.scoutapp.com](http://www.blog.scoutapp.com) (13, September, 2012). Understanding Linux CPU Load. <http://blog.scoutapp.com/articles/2009/07/31/understanding-load-averages>.
- [www.codeclouds.com](http://www.codeclouds.com) (1, November, 2013). What is ExtJS and why do we care? <http://www.codeclouds.com/community/what-is-extjs-and-why-do-we-care/>.
- [www.geoext.org/](http://www.geoext.org/) (2, September, 2012). JavaScript Toolkit for Rich Web Mapping Applications. <http://www.geoext.org/>.
- [www.jamejamonline.ir/newstext.aspx](http://www.jamejamonline.ir/newstext.aspx) (28, March, 2012). Jame Jam newspaper. <http://www.jamejamonline.ir/newstext.aspx>.
- [www.java-samples.com/](http://www.java-samples.com/showtutorial.php?tutorialid=891) (11, February, 2012). Java samples. <http://www.java-samples.com/showtutorial.php?tutorialid=891>.
- [www.jquery.com/](http://www.jquery.com/) (10, June, 2012). JQuery. <http://api.jquery.com/jquery.get/#jqxhr-object>
- [www.openlayers.org/](http://www.openlayers.org/) (2, 11, 2013). OpenLayers: Free Maps for the Web. <http://openlayers.org/>.

- [www.opentripplanner.com](http://www.opentripplanner.com/vendors/) (2, November, 2013). OpenTripPlanner.  
<http://www.opentripplanner.com/vendors/>.
- [www.pgrouing.org](http://workshop.pgrouing.org/chapters/shortest_path.html#shortest-path-a) (1, November, 2013). pgRouting Algorithms.  
[http://workshop.pgrouing.org/chapters/shortest\\_path.html#shortest-path-a](http://workshop.pgrouing.org/chapters/shortest_path.html#shortest-path-a).
- [www.pgrouing.org](http://www.pgrouing.org/docs) (06, July, 2011). Shortest path search (FOSS4G workshop 2010). <http://www.pgrouing.org/docs>.
- [www.php.net](http://www.php.net/manual/en/intro.array.php) (22, February, 2011). PHP Hypertext Preprocessor.  
<http://www.php.net/manual/en/intro.array.php>.
- [www.postgresql.org/](http://www.postgresql.org/) (14, December, 2011). PostgreSQL extensions (PostPic).  
<http://www.postgresql.org/>.
- [www.postgresql.org/](http://www.postgresql.org/download/products/6-postgresql-extensions/) (14, December, 2011). Software Catalogue - PostgreSQL extensions (PostGIS). [www.postgresql.org/download/products/6-postgresql-extensions/](http://www.postgresql.org/download/products/6-postgresql-extensions/).
- [www.princeton.edu](http://www.princeton.edu/~achaney/tmve/wiki100k/docs/Heuristic.html) (3, November, 2013). Heuristic.  
<http://www.princeton.edu/~achaney/tmve/wiki100k/docs/Heuristic.html>.
- [www.redblobgames.com/](http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html) (2, September, 2013). Introduction to A\*.  
<http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html>.
- [www.sce.carleton.ca](http://www.sce.carleton.ca/faculty/chinneck/po.html) (2011). Carleton Univeristy. <http://www.sce.carleton.ca/faculty/chinneck/po.html>
- [www.stanford.edu/](http://theory.stanford.edu/~amitp/GameProgramming/MovingObstacles.html) (1, November, 2013). Dealing with moving obstacles.  
<http://theory.stanford.edu/~amitp/GameProgramming/MovingObstacles.html>.
- [www.tomtom.com](http://www.tomtom.com/lib/doc/licensing/RTTHT.US.pdf) (2, November, 2013). Real-time & historical traffic.  
<http://www.tomtom.com/lib/doc/licensing/RTTHT.US.pdf>.
- [www.tomtom.com](http://www.tomtom.com/en_gb/licensing/products/traffic/real-time-traffic/) (25, February, 2013). TomTom.  
[http://www.tomtom.com/en\\_gb/licensing/products/traffic/real-time-traffic/](http://www.tomtom.com/en_gb/licensing/products/traffic/real-time-traffic/).
- [www.w3schools.com](http://www.w3schools.com/php/php_functions.asp) (1, November, 2013). PHP Functions.  
[http://www.w3schools.com/php/php\\_functions.asp](http://www.w3schools.com/php/php_functions.asp).
- [www.w3schools.com](http://www.w3schools.com/ajax/default.asp) (11, May, 2012). AJAX Tutorial.  
<http://www.w3schools.com/ajax/default.asp>.
- [www.w3schools.com](http://www.w3schools.com/xml/xml_what_is.asp) (13, May, 2012). Introduction to XML.  
[http://www.w3schools.com/xml/xml\\_what\\_is.asp](http://www.w3schools.com/xml/xml_what_is.asp).

- [www.w3schools.com](http://www.w3schools.com/php/php_arrays.asp) (14, May, 2012). PHP functions.  
[http://www.w3schools.com/php/php\\_arrays.asp](http://www.w3schools.com/php/php_arrays.asp), 2012.
- [www.w3schools.com](http://www.w3schools.com/js/js_intro.asp) (18, June, 2012). JavaScript Introduction.  
[http://www.w3schools.com/js/js\\_intro.asp](http://www.w3schools.com/js/js_intro.asp).
- [www.w3schools.com](http://www.w3schools.com/js/js_timing.asp) (18, May, 2012). JavaScript Timing Events.  
[http://www.w3schools.com/js/js\\_timing.asp](http://www.w3schools.com/js/js_timing.asp).
- [www.wikipedia.org](http://en.wikipedia.org/wiki/Real-time_web) (1, November, 2013) Real-time web.  
[http://en.wikipedia.org/wiki/Real-time\\_web](http://en.wikipedia.org/wiki/Real-time_web)
- [www.wikipedia.org](http://en.wikipedia.org/wiki/Load_computing) (1, November, 2013). Load (computing).  
[http://en.wikipedia.org/wiki/Load\\_computing](http://en.wikipedia.org/wiki/Load_computing).
- [www.wikipedia.org](http://en.wikipedia.org/wiki/Open_Source_Routing_Machine) (3, November, 2013). Open Source Routing Machine.  
[http://en.wikipedia.org/wiki/Open\\_Source\\_Routing\\_Machine](http://en.wikipedia.org/wiki/Open_Source_Routing_Machine).
- Zhang, C., and Li, W. (2005). The Roles of Web Feature and Web Map Services in Real-time Geospatial Data Sharing for Timecritical Applications. *Cartography and Geographic Information Science*. 32(4), 269-283.