INTELLIROUTE: A PROTOTYPE MOBILE SYSTEM FOR NON-RECURRENT CONGESTION MITIGATION DRIVEN BY CROWD-SOURCED DATA

Final Report
April. 2016

Principal Investigator
Gang-Len Chang, Ph.D
Professor
University of Maryland at College Park

Co-Principal Investigator
Yue Liu, Ph.D
Associate Professor
University of Wisconsin at Milwaukee

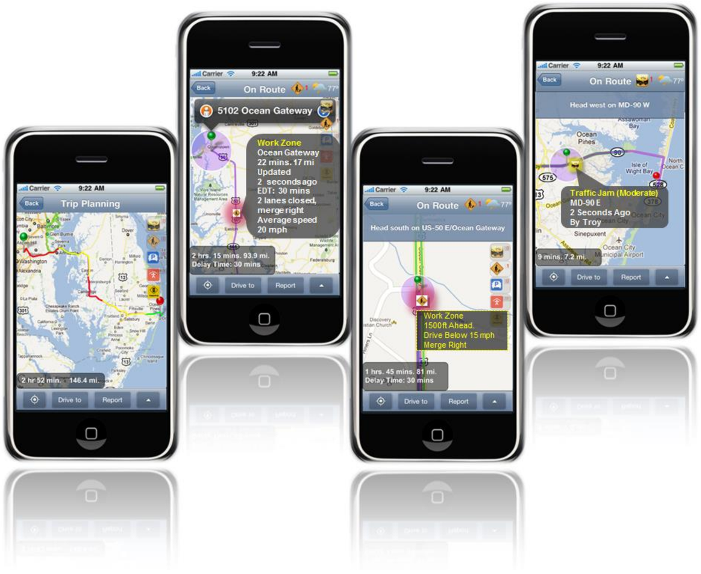


TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This research was conducted under the National Transportation Center at University of Maryland College Park. The authors would like to thank the USDOT, Maryland State Highway Administration, and Wisconsin Department of Transportation for their financial support and technical assistance.

EXECUTIVE SUMMARY

This research develops a smartphone-based prototype system that supplements the 511 to improve its real-time traffic detouring service to state highway users under work zone or non-recurrent traffic conditions. The system makes the best use of the “xml” data feeds and retrieves the useful information using a web service integrated with the SQL database system. To make sure the alternative route guidance is warranted, this study has further developed a multi-criteria decision model, which is integrated with the server side system application. In the Android platform, a smartphone App was developed to provide travelers with real-time routing/re-routing options, traffic and incident information retrieved from the web server. The proposed prototype system has been applied with an actual freeway corridor (the IH-94 corridor between the city of Madison where IH-94 connects with IH-39/90 and the city of Milwaukee where it connects to IH-43). Tests with various scenarios have demonstrated significant overall benefits with system application and effective information provision to travelers in real time. The trajectories of the vehicle carrying the proposed the smartphone system can be automatically recorded into the server side database, which offers the potential for crowd-source traffic dynamics data collection and mining with sufficient number system users.

CHAPTER 1: INTRODUCTION

1.1 Research Background

Highway infrastructure and related transportation elements comprise a crucial lifeline for the United States, as the operations of such systems affect our nation's economic vitality and citizens' way of life. Better information can influence transportation decisions and route choices of travelers in some very clear ways and result in more efficient utilization of highway infrastructure capacity. For example, with better ways to learn how work zones and other incidents affect travel times, commuters can start for work and return home at times more responsive to actual traffic patterns than a fixed schedule. As the effects of congestion become clearer to people, they are more able to consider alternate paths to their destination.

Reduced capacity caused by work zones coupled with overwhelming traffic demand has resulted in significant traffic congestion in the highway system. In such conditions, if proper traffic re-routing strategies could be implemented in time, motorists could circumvent the congested segments by detouring through alternative routes, resulting in significant economic savings by reducing delays, fuel consumption, and vehicle emissions, and by increasing traffic safety

Traveler information has exploded over the past decade with the development and use of advanced technologies able to detect, analyze, and disseminate traffic conditions. The 511, a decision support system that enables the traveling public to make informed decisions and manage their trip details, automates the integration of law enforcement incident data, freeway traffic conditions, roadway construction and detour information, and multimodal bus, rail, and airport conditions. It is capable of providing accurate information to the traveling public, and allows transportation system providers and emergency responders to manage the transportation infrastructure in real time.

The 511 regular/mobile websites and phone service, as the major sources for travelers to access most of the popular 511 services, have not been fully utilized by most of state highway travelers to get real-time detour information when they are experiencing work zone constructions on their way. A very recent study by University of Wisconsin-Milwaukee on travelers' opinions of the I-94 East-West corridor repaving project indicates that although 84% of the 1,556 surveyed drivers have a computer available to get traffic information, most of them are still getting their traffic information from a local radio station (73%) and local television station (66%). Other forms of technology, like 511, are far behind (see Figure 1). Most motorists don't have the numbers on their cell phone's speed dial. As one of the survey respondents noted, "I don't have the time to search on-line or call the number to check traffic congestion ahead in my trip."

Motorists will help manage the traffic demand if they are given accurate, reliable information in a timely manner, and their cooperation is critical for work zone traffic management. Failure to provide real-time and en-route assistance in a timely manner may cause a significant number of travelers to be trapped in traffic congestion. For example, the lack of real-time information stranded 2,000 vehicles on I-39 in February of 2008 when a segment of the interstate highway was closed in Northwest Wisconsin.

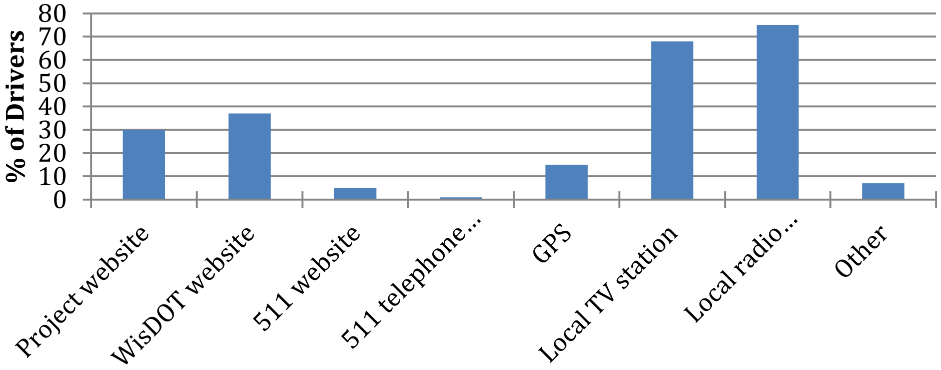


Figure 1.1 Resources for travelers to obtain traffic information during I94 East-West corridor construction

With hundreds of millions of automobiles and over a billion mobile phone-equipped people in the world, cars and humans may turn out to be the carriers of the world's largest and most dynamic sensor networks in the coming years. Such mobile sensor networks have the potential to sense large expanses of the world at much finer fidelity and scale than possibly by static deployments (roadside sensors, cameras, and message signs).

According to recent survey data, 85% of Americans own an Internet-enabled handheld device, and most adults (as high as 93%), ages 18 to 64 years, commonly access the Internet on mobile devices. These survey data indicate that mobile-Internet access is becoming more important in people's daily lives compared to the traditional media (e.g. TV, radio stations, and PC-based web application). The recent survey of I-94 travelers also shows that drivers are showing new trends with more drivers owning smart cell phones than regular cell phones, which can create opportunities of creating smartphone-based tools to supplement and improve the real-time traffic detouring service of 511 system during work zone construction.

Smartphones usually have a full-functional operating system (Android, Apple iOS, Windows Mobile, etc.), and can be thought as a portable computer with regular phone capabilities. With a functional operating system and programming platforms, developers are able to create Smartphone applications for different purposes including entertainment, work, travels, etc. The use of mobile phone applications for a variety of purposes has helped turn Smartphone's into a vital piece of technology at home and in the workplace. The design of a Smartphone application can simplify and complete the same tasks as a typical computer. Most Smartphone applications offer simple interfaces to make it easy for consumers to use and get their information quickly and conveniently.

Detailed and up-to-the-minute information is changing when, where, and how we travel. At the same time, traffic management are experiencing revolution and becoming more data driven. Data analysis on newly fused datasets is enhancing our understanding of how to improve transportation services. The surge of the number of smartphone users brings both opportunities and challenges for public agencies to use mobile sensing for next-gen work zone planning and traffic management. For example, mobile sensing data can be easily collected and mined to analyze traveler diversion behaviors and origin-destination changes due to a major highway construction project. Compared with the traditional survey method of sending questionnaires, it collects much more number of samples at much lower lost and with much better accuracy. According to statistics by USDOT, a small population of users (2,200) can produce high fidelity information in a major metropolitan area. Two-thirds of the smartphone users trust their privacy is protected for traffic management purposes, and high proportion of them (88%) indicates repeated use in the future.

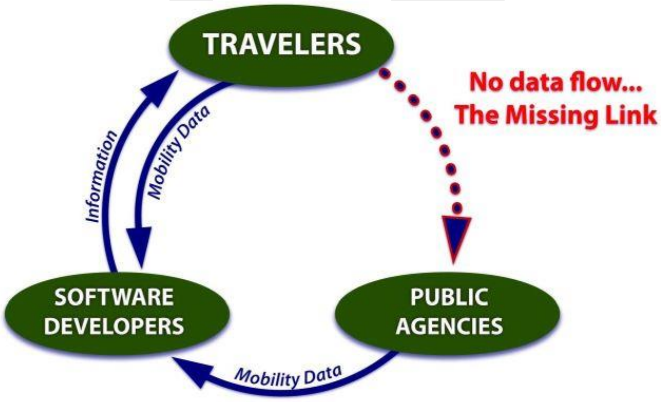


Figure 1.2 Missing feedback loop from travelers to public agencies (Source: USDOT)

In this light, software vendors have started to develop mobile applications and use mobility data from travelers to provide traffic information, which has comprised key components of the transportation mobility data ecosystem illustrated in Figure 2. Notable examples include Waze, Roadify, Beat the Traffic, and etc. However, in such a system, the public agencies have not played an important role as they are supposed to due to the missing data flow from travelers to agencies. How to create the feedback loop to apply crowd-sourced data in next-gen traffic management has become a major challenge to many public agencies.

1.2 Research Objectives

To address the above critical needs and deficiencies, this project develops a smartphone-based App and server system that supplements the 511 to improve its real-time traffic detour service to state highway users under congestion due to work zone construction or other non-recurrent events. The development of the proposed smartphone application shall not require any hardware investment in the current 511 system; the maintenance and further improvements of the Smartphone application could be done through software upgrades, which would make the application reusable for future developments. Finally, the application would provide a critical piece of the long-term push toward enabling the advanced data communication capabilities of the 511 system.

1.3 Report Organization

Based on the research objectives, this study has organized all primary system development work and results into six subsequent chapters. A brief description of the information contained in each chapter is presented next.

Chapter 2 performs a comprehensive review of available literature associated with mobile traveler information provision. *Chapter 3* mainly presents the description and processing of the system data feeds. *Chapter 4* develops the alternative route decision model based on a well-calibrated corridor simulation network and a comprehensive set of experimental scenarios according to the key factors that may affect the system's decision whether or not to implement alternative recommendation. *Chapter 5* reports the development process of the system and the App. *Chapter 6* presents the field test results of the system and benefit analysis results. *Chapter 7* summarizes the primary research findings and their potential applications to improving work zone operational efficiency. Recommendations for future research are also made.

CHAPTER 2: LITERATURE REVIEW

2.1 Smartphone-based traveler information systems

Campolo *et al.* (2012) designed a smartphone-based platform to exploit low-cost dedicated hardware that interacts with sensors on board and in the vehicle surroundings. In their study, a prototype is also developed to assess the technological feasibility of their conceived platform. Similarly, Briante *et al*. (2014) proposed SmartCar (Smartphone-based floating car data collection) platform as an intelligent use of smartphones to collect “augmented” FCD from in-vehicle telematics and external sensors (e.g., pollution detectors for urban sensing), and (ii) the adoption of an offloading strategy that leverages Wi-Fi hotspots to alleviate the burden on the cellular network due to the massive generation of “augmented” data.

Abdulazim *et al.* (2013) presented a data collection framework and its prototype application for personal activity–travel surveys through the use of smartphone sensors. The core components of the framework run on smartphones backed by cloud-based (online) services for data storage, information dissemination, and decision support. The framework employs machine-learning techniques to infer automatically activity types and travel modes with minimum interruption for the respondents. In 2014 TRB meeting, Wang *et al.* discussed the potential of mobile phone positioning data as an alternative data source for travel behavior studies. The study is particularly interested in its potential for analyses in travel behavior dynamics. The authors discussed the relative advantages of mobile positioning over travel diary/GPS tracking as an alternative data collection technique. Eventually, they concluded that mobile phone positioning data is a promising data source for travel behavior study.

Aleyadeh *et al.* (2012) presented a framework developed for both Android and iOS platforms, to enable fine grained sensing and monitoring of road conditions in a ubiquitous fashion. The study employed crowd sensing approaches that are fine-tuned with back-end data analytics, to empower both active and passive sensing schemes. The application, named Smartphone-based Road Monitoring (SRoM), has been tested in real-life scenarios and its accuracy and fidelity have been validated in physical implementations. Current implementations are being communicated with the Ministry of Transportation in Canada for testing on their fleets. Nawaz *et al.* (2013) developed a smartphone based sensing system, *ParkSense*, which detects if a driver has vacated a parking spot.

Manolopoulos *et al.* (2012) has successfully implemented a Smartphone-based Traffic Information System, which functions in estimating traffic conditions solely based on data from smartphones, collecting data securely to protect the TIS as well as achieving privacy preserving manner. Chuang *et al.* (2013) proposed an I-Traffic system that utilizes crowd sourced data from smartphones for the traffic flow mining by shockwave techniques.

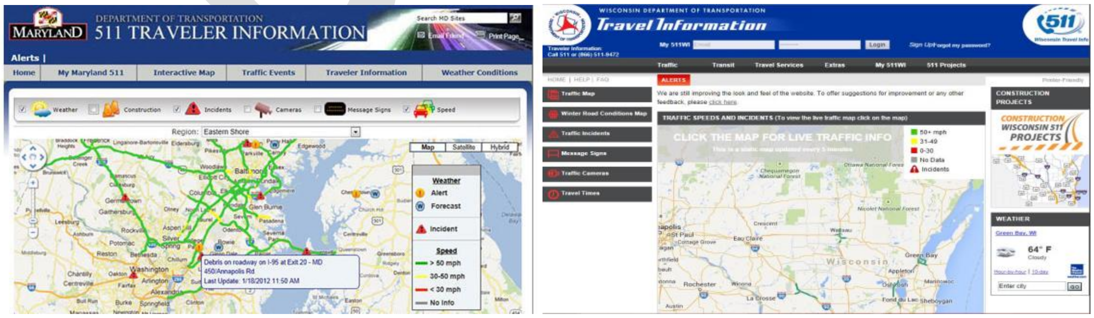
2.2 The 511 systems

On March 8, 1999, the U.S. Department of Transportation (USDOT) petitioned the Federal Communications Commission (FCC) to designate a nationwide three-digit telephone number for travel information. On July 21, 2000 the Federal Communications Commission (FCC) designated 511 as the single travel information telephone number to be made available to states & local jurisdictions across the country. The FCC ruling leaves nearly all implementation issues & schedules to state & local agencies & telecommunications carriers.

Mindful of both the opportunity & challenge 511 presents, the American Association of State Highway & Transportation Officials (AASHTO), in conjunction with many other organizations including the American Public Transportation Association (APTA) and the Intelligent Transportation Society of America (ITS America), with support from the U.S. Department of Transportation, established the 511 Deployment Coalition. The goal of the 511 Deployment Coalition is "the timely establishment of a national 511 traveler information service that is sustainable and provides value to users." After considering a full range of consumer, business, technical and policy issues associated with 511, the Coalition established a national vision for 511 which states that:

“511 will be a customer-driven, multi-modal travel information service, available across the United States, accessed via telephones, and other personal communications devices, realized through locally deployed interoperable systems, enabling a safer, more reliable and efficient transportation system.”

511 has been initialized to offer a wide variety of traffic and transit information for traffic participants. The users are expected to collect or query information link with traffic map, winter road condition map, traffic incidents, road closures, transit map and etc. It aims to build a decision support system that enables the traveling public to make informed decisions and manage their trip details, automates the integration of law enforcement incident data, freeway traffic and roadway conditions, and multimodal bus, rail, and airport conditions. Moreover, it is capable of providing accurate and real time information to the traveling public, and allows transportation system providers and emergency responders to manage the transportation infrastructure in real time.



Debris on roadway on I-95 at Exit 20 - MD 450/Annapolis Rd
Last Update: 1/18/2012 11:50 AM

We are still improving the look and feel of the website. To offer suggestions for improvement or any other feedback, please click here.

Figure 2.1 Maryland and Wisconsin 511

2.3 Crowd-Source data

As its conception explained by Misra (2014), crowdsourcing is an example in which an organizer or an organization is able to use the network of collaborators to solve a problem that would otherwise be cost- or labor-intensive, or in which within a defined organization the expertise is unavailable or insufficient.

So far, crowdsourcing is particularly available and successfully applied into transportation domain because it voluntarily functions in bringing together a large group of people on the same platform to address common issues that affect its members. Crowdsourcing is widely founded to work successfully for local purposes through localized knowledge and acquired experiences because people in a region tend to identify themselves with the region where they live, work, and socialize, and are generally more interested in the systems that affect them.

The recent study of using crowdsourcing in transportation reveals that the predominant purposes are either for the collection of data or feedback from the transportation system's users. For example, one popular use of crowdsourcing is to collect route choice data from bicyclists using the GPS functionality of their cell phones; such data are not readily available through standard data collection procedures, and designing a separate survey for a small population of users is often not cost effective for regional planning agencies. Crowdsourcing in this case helps the geographically dispersed and diverse population of cyclists work together on a common interest without financially burdening the planning agencies. Similarly, crowdsourcing can also help in collecting feedback from a socio-demographically diverse range of users of any transit system, which can be immensely useful for improving transit service quality and standards. Three main systems related with crowdsourcing are widely promoted in transportation area, namely, User Feedback–Based Crowdsourcing Systems, Crowdsourcing Systems for Data Collection and Standalone Crowd-sourced Data Quality Auditor System. All of them have sequentially built the connections in transport participants, crowdsourcing platform developers and public agencies. However, it is noticeably founded a missing link between public agencies and participants at most of cases.

CHAPTER 3: SYSTEM DATA FEEDS

The proposed system framework

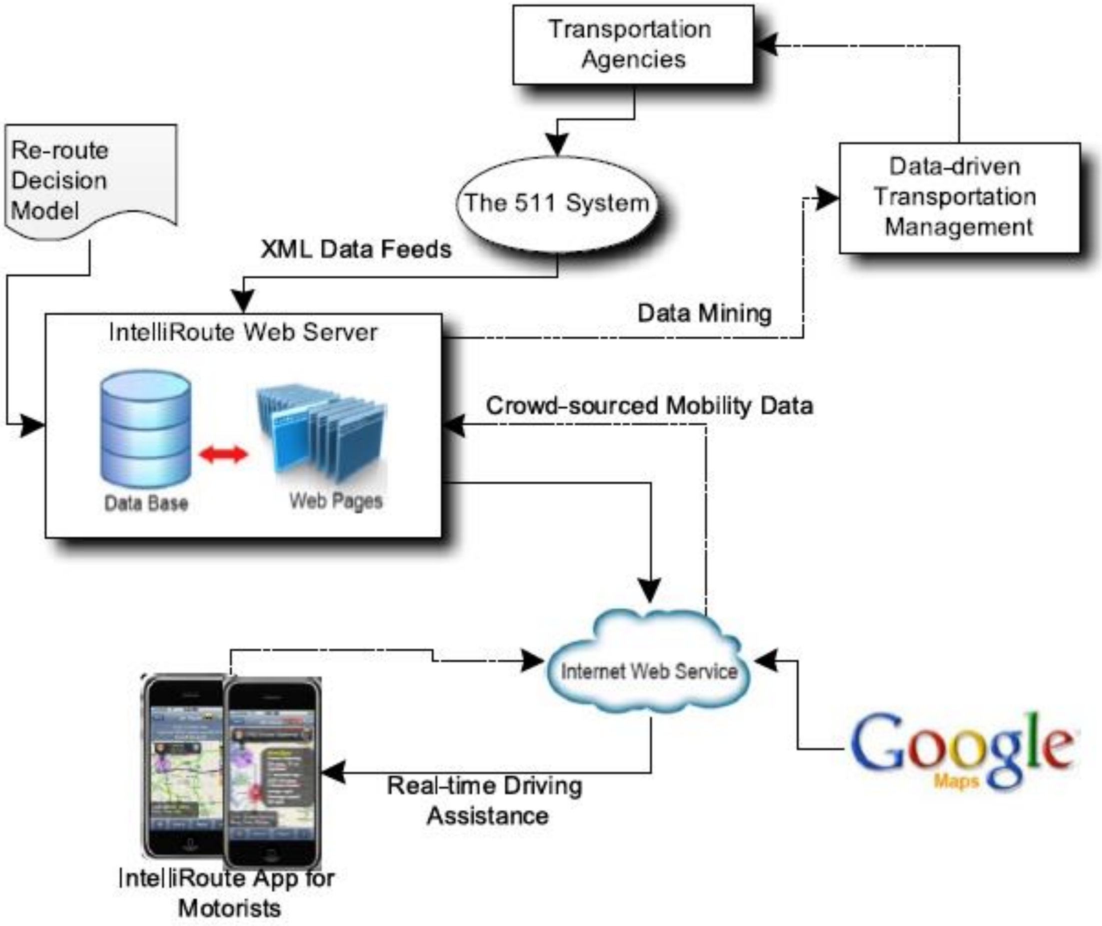


Figure 3.1 The proposed system framework

The project builds a linkage between our proposed smart-phone based prototype system and the Wisconsin 511 system via real-time streaming live traffic information from the 511.xml and 511CS.xml data feeds files. This research seeks to develop a smartphone-based prototype system that supplements the Wisconsin 511 system to improve its real-time traffic routing service to state highway users under non-recurrent traffic congestion. Figure 4 shows the proposed system operational framework that uses mobile-internet to manage the communication between a smartphone app and a web server hosting the database and traffic routing models.

The App features a map-based interface configuration for motorists to visualize the information of interest. It is also designed so that all interactions can be achieved using just one hand when used in portrait mode (although it can also be used in landscape mode, in which case both hands are used). Most importantly, in addition to displaying the information on the map, the app will "read" the user-specific information to motorists with a synthesized voice to minimize motorist distraction while driving.

This chapter will detail the explanation of parsing and extracting data from the 511.xml and 511CS.xml data files.

3.1 Parsing the "511.xml" file

511.xml consists of four sub-parts, namely, "linkeData", "fullEventUpdate", "routeData", and "dMSDeviceStatus", respectively. Each part functions in storing corresponding information.

3.1.1 "linkData" module

The "linkdata" module includes the following data records:

- network-id: Values indicate organization that link IDs belong to.
  - STOC-MILWAUKEE
  - STOC-MADISON
  - STN (State Trunk Highway)
- organization-id: set to WisDOT
- organization-name: set to WisDOT
- center-id: Milwaukee or Madison, whichever center contributed the data
- link-id: an unique numerical designation for the Link within the network.
- link-data-stored:
  - 1 – link volume
  - 2 – current occupancy
  - 3 – current average speed
  - 4 – current delay time
  - 5 – roadway status
  - 6 – daily peak volume and hour of peak volume
- detection-method: source of information
  - 22 – video processing monitoring station
  - other values may be used by other agencies, however, WisDOT appears to utilize only this type.
- link-data-type:
  - 1 – actual
  - 2 – reconstructed
  - 3 – historical
  - 4 – predicted
  - 5 – smoothed
  - 6 – averaged

3.1.2 "routeData" module

In 511.xml, the "routedata" module includes the following data records:

- route-id: identifier up to 32 alphanumeric characters
- route-status: Traffic condition:
    - TravelTimeValid = 0x00000001
    - OverMaxSpeed = 0x00000002
    - UnderMinSpeed = 0x00000004
    - NoLinkData = 0x00000008
    - NotEnoughData = 0x00000010
    - FaxTravelTime = 0x00010000
    - SignTravelTime = 0x00020000
- operational-link-count: number of links on its corresponding route considered to be operational
- total-distance: total length of route in meters
- display-travel-time: travel time to display on or web in seconds. -1 means unavailable, -2 means invalid.
- calculated-travel-time: actual calculated travel time in seconds. -1 means unavailable, -2 means invalid.
- minimum-travel-time: minimum travel time through route using each link's speed limit in seconds. -1 means unavailable, -2 means invalid.
- nominal-travel-time: travel time calculated using each link's travel time in seconds. -1 means unavailable, -2 means invalid.
- maximum-travel-time: maximum realistic travel time through the route in seconds. -1 means unavailable, -2 means invalid.
- delay: difference between the calculated (actual) travel time and the nominal travel time when traffic is flowing less than the speed limit (seconds)
- organization-information
    - organization-id; will be set to WisDOT
    - organization-name; will be set to WisDOT
    - center-id; Milwaukee or Madison, whichever center contributed the data
- last-event-update
    - date; Date given in YYYYMMDD
    - time; Time given in HHMMSSssss where ssss is decimal seconds up to 4 places.
    - offset; valid time offset using 24 hour notation. Given in HHMM format.

3.1.3 "fullEventUpdate" module

This module includes the following data records:

Message-header

- message-type-id: identifies the type of event being sent
    - BEU – Basic event update
    - FEU – Full event update.

- Message-type-version: identifies the version of the message structure being used, e.g. Version 1 of the Event Report Message.
- Message-number: a unique identifier of a specific message instance.

Organization-sending

- organization-id: will be set to WisDOT
- organization-name: will be set to WisDOT
- center-id: Milwaukee or Madison, whichever center contributed the data

Contact Details

- contact-id: string type data with maximum of 32 characters.
- person-name: string type data with maximum of 32 characters.

Message-time-stamp

- date: date given in YYYYMMDD
- time: time given in HHMMSSssss where ssss is decimal seconds up to 4 places.
- offset: valid time offset using 24 hour notation. Given in HHMM format.

Event-reference

- event-id: Incident ID in database or LCS, will be a string with 32 characters maximum.
- event-update: the number of times the log has been modified for a specific roadway event.

Event-indicators

- Event-indicator
    - Status: Populated from STOC's IMS comment fields. For LCS: corresponds to Facility Comment

Event-headline

- Headline
    - Accidents-and-incidents: The ITIS codes used to classify and categorize types of events:

        513 = “accident"
        514 = “serious accident"
        515 = “injury accident"
        516 = “minor accident"
        517 = “multi vehicle accident"
        518 = “numerous accidents"
        519 = “accident involving a bicycle"
        520 = “accident involving a bus"
        521 = “accident involving a motorcycle"
        522 = “accident involving a pedestrian"
        523 = “accident involving a train"
        524 = “accident involving a truck"
        562 = “accident involving a semi trailer"

525 = “accident involving hazardous materials"
526 = “earlier accident"
527 = “medical emergency"
528 = “secondary accident"
529 = “rescue and recovery work REMOVED"
530 = “accident investigation work"
531 = “incident"
532 = “stalled vehicle"
533 = “abandoned vehicle"
534 = “disabled vehicle"
535 = “disabled truck"
536 = “disabled semi trailer"
537 = “disabled bus"
538 = “disabled train"
539 = “vehicle spun out"
540 = “vehicle on fire"
541 = “vehicle in water"
542 = “vehicles slowing to look at accident"
543 = “jackknifed semi trailer"
544 = “jackknifed trailer home"
545 = “jackknifed trailer"
546 = “spillage occurring from moving vehicle"
547 = “acid spill"
548 = “chemical spill"
549 = “fuel spill"
550 = “hazardous materials spill"
551 = “oil spill"
552 = “spilled load"
553 = “toxic spill"
554 = “overturned vehicle"
555 = “overturned truck"
556 = “overturned semi trailer"
557 = “overturned bus"
558 = “derailed train"
559 = “stuck vehicle"
560 = “truck stuck under bridge"
561 = “bus stuck under bridge"
638 = “accident cleared"
639 = “incident cleared"
640 = “unspecified severity"
641 = “on shoulder or median"
642 = “travel lanes or service ramp blocked"
643 = “travel lanes or system ramp blocked"
644 = “full roadway closure in one direction"
645 = “full roadway closure in both directions"

Event-element-details

- event-element-detail
- element-descriptions
- element-description
    - phrase: will be free text
- element-lanes
    - lanes-total-effected: total number of lanes affected.
- element-locations
- element-location
- location-on-link
    - ink-ownership: name of designated owner of link. Any combination of up to 256 ASCII characters
- link-designator: set to STDNAME for main roadway.
- Link-id: a unique numerical designation for the Link within the network.
- Link-direction: string value

Primary-location

- Link-name; set to HWYLST
- Point-name; name of a point along a roadway, a string of up to 256 characters.

Geo-location

- Latitude; latitude in integer micro degrees
- Longitude; longitude in integer micro degrees

Landmark-location

- Landmark-type; string, locally defined
- Landmark-name; landmark name from STN
- Landmark-point-name; Set to STN "RDWY_LINK_ID:LINK_OFFSET", where LINK_OFFSET is in thousandths of a mile (variable length)

Upward-area-reference

- Area-id: FIPS county code
- Area-name: county name

Update-time

- Date; Date given in YYYYMMDD
- Time; Time given in HHMMSSssss where ssss is decimal seconds up to 4 places.
- Offset; valid time offset using 24 hour notation. Given in HHMM format.

Valid-period

- Estimated-duration; estimated remaining duration

3.1.4 "dMSDeviceStatus" module

The "dMSDeviceStatus" module consists of the following data records:

- Organization-information
- Operator-id: unique operator ID consisting of up to 32 characters.
- Device-id: unique device id consisting of up to 32 characters.
- Dms-device-status: device operational status.
    - on (1)
    - off (2)
    - in service (3)
    - out of service (4)
    - unavailable (5)
    - unknown (6)
- dms-current-message: string data type.
- Message-source-code: string data type.

Organization-Information

- organization-id: will be set to WisDOT
- organization-name: will be set to WisDOT
- center-id: Milwaukee or Madison, whichever center contributed the data last-comm-time
- Date: Date given in YYYYMMDD
- Time: Time given in HHMMSSssss where ssss is decimal seconds up to 4 places.
- Offset: valid time offset using 24 hour notation. Given in HHMM format.

3.2 Parsing the "511LCS.xml"

511LCS.xml provides the associated information of the lane closures events. The information is classified into four types: general closure information, location information, scheduling information, and comments and detour information. Furthermore, the generation information contains organization, project description, event headline and etc. The location information contains primary location, secondary location information and etc. Scheduling information contains all the time information related with lane closures. Comments and detour information includes some additional information. In order to track all the lane closure events and locate them in our proposed system and App, this system designs the following parsing algorithms to extract all relevant information:

Algorithm I: tracking the time and location information

Location:

Location is represented by following attributes in 511LCS.XML

```
<primary-location> The begin landmark of the closure
<geo-location>
<latitude>   </latitude>
<longitude>  </longitude>
<secondary-location>  The end landmark of the closure
<geo-location>
<latitude>   </latitude>
<longitude>  </longitude>
```

In 511LCS.XML, it uses both detailed coordination data and description to show the location information. See the following for an example:

```
<primary-location>
<geo-location>
<latitude>44012731</latitude>
<longitude>-88582343</longitude>
</geo-location>
<link-name>US 41 NB</link-name>
<point-name>ON RAMP FROM 9TH AVE</point-name>
```

Here is a closure event caused by construction and it will begin at a point with lat 44.012731 and long -88.582343, where is ON RAMP FROM 9TH AVE in US 41NB. All the link name and point-name are listed in the LCS-landmarks-2010-8-6 database. Noticeably, if we combine the link-name and point-name and search the combination in the above mentioned database, detailed coordinates will be returned, which is exactly Lat 44.012731 and Long -88.582343. All the combined link-name and point-name are connected with unique coordinates, which means just track the Latitude and Longitude in 511LCS.XML and then we can accurately locate a closure event.

Time:

Our proposed system reports the start time and end time of the lane closure to travelers. A timer set is designed in this study to install as a time reminder that activate in the beginning and end of the event.

In 511LCS.XML, time is reacted by the following attributes:

```
<start-time>
<date>   </date>
<time>   </time>
<expected-end-time>
<date>   </date>
<time>   </time>
```

For example:

```
<start-time>
<date>20120323</date>
<time>070000</time>
<expected-end-time>
```

```
<date>20130614</date>
<time>050000</time>
```

The above closure event will start at 2012/3/23 07:00 and will end at 2013/6/14 05:00 (It is a construction event).

Algorithm II: additional description and direction information
The Algorithm II will parse the additional description of the lane closure and the direction of the lane closure from XML and add it into the developed system. For example,

Report the closure reason:

```
<event-headline>
<headline>
<roadwork>construction</roadwork>
```

Add direction into program:

```
<link-direction>n</link-direction>
```

N means “North”.

All the closure events are coded into a uniform format in 511LCS.XML. Consequently, we could use the above algorithms to extract the closure into system and the App.

CHAPTER 4: ALTERNATIVE ROUTE DECISION MODEL

After parsing the 511 data feeds files, the developed system needs to analyze the impact of the events and decide whether or not to issue alternative route recommendation to travelers via the App either prior to the trip or when they are on their route. This study has performed extensive analyses of the past five-year major incident data in the stretch of interstate highway 94 (Madison - Milwaukee) using the Wisconsin Lane Closure System and the InterCAD Traffic Incident Data Exchange System to obtain a comprehensive incident scenario dataset. Alternative route will be recommended for those real-world incident/work zone scenarios in a well-calibrated simulated environment with varying traffic demand levels, driving behavior patterns, geometric configurations, and traffic control parameters.

Alternative route decisions will be evaluated and ranked for each experimental scenario by the developed decision model. A reliable and convenient statistical model will be developed to allow the system to make the best alternative route recommendations to travelers prior to their trip or when they are on the way.

4.1 Data collection and extraction

4.1.1 Description of Data Collection Network

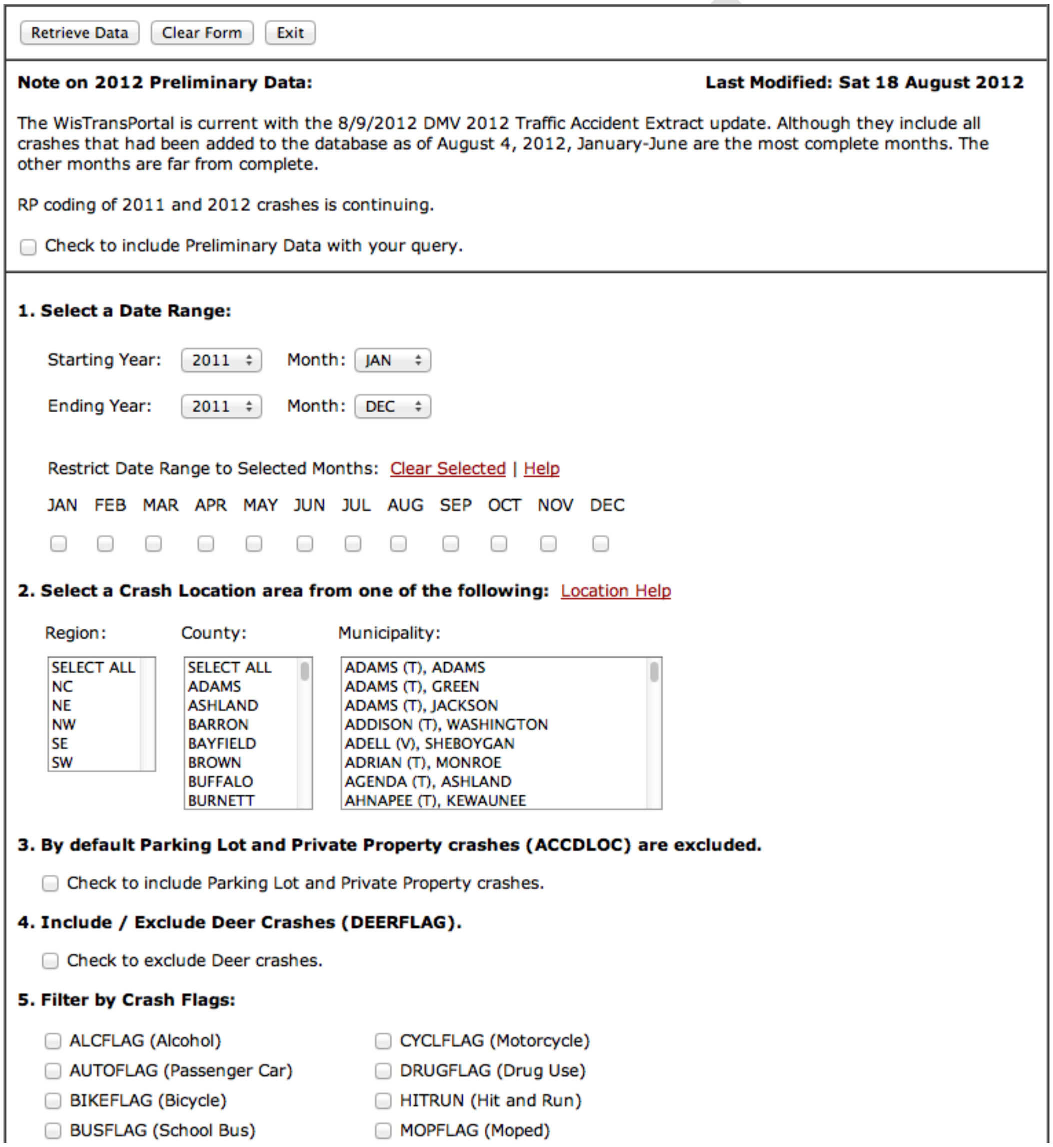
The area of study for alternative route decision model development consists of the IH-94 corridor between the city of Madison where IH-94 connects with IH-39/90 and the city of Milwaukee where it connects to IH-43. The segment covers approximately seventy miles of mostly rural highway from IH-39/90 until reaching Milwaukee County, at which point it continues on as an urban highway.

4.1.2 Data Sources

All data collected for the initial dataset came from the Wisconsin TOPS Laboratory operated by the University of Wisconsin–Madison. There are multiple databases containing crash and incident information maintained by the TOPS Lab. This study chose two, the MV4000 Crash Data database and the InterCAD to complete the preliminary data set. While it would have been preferable to query and use only one database, neither of these databases was complete, and therefore needed to supplement each other. It is for this reason that the dataset is comprised of only two years of data rather than the originally intended five years. While the MV4000 database now covers over eighteen years of incidents, the InterCAD database contains only two years and limits the scope of the data set accordingly.

4.1.2.1 Source 1: MV4000

The MV4000 Crash Data Retrieval Facility is a database maintained by the TOPS Laboratory with crash data from all reportable crashes in Wisconsin with data available from 1994 to the present year. The MV4000 data set contains an abundance of information, providing the majority of information used to build the preliminary data set. The MV4000 database uses standardized data fields to describe each incident. A sample of what the retriever tool looks like is shown in Figure 4.1. Data were retrieved for the years 2010 and 2011 to match the time period that was available from other sources.



Note on 2012 Preliminary Data: **Last Modified: Sat 18 August 2012**

The WisTransPortal is current with the 8/9/2012 DMV 2012 Traffic Accident Extract update. Although they include all crashes that had been added to the database as of August 4, 2012, January-June are the most complete months. The other months are far from complete.

RP coding of 2011 and 2012 crashes is continuing.

□ Check to include Preliminary Data with your query.

1. Select a Date Range:

Starting Year: 2011 Month: JAN

Ending Year: 2011 Month: DEC

Restrict Date Range to Selected Months: Clear Selected | Help

JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
□	□	□	□	□	□	□	□	□	□	□	□

2. Select a Crash Location area from one of the following: Location Help

Region:	County:	Municipality:
SELECT ALL	SELECT ALL	ADAMS (T), ADAMS
NC	ADAMS	ADAMS (T), GREEN
NE	ASHLAND	ADAMS (T), JACKSON
NW	BARRON	ADDISON (T), WASHINGTON
SE	BAYFIELD	ADELL (V), SHEBOYGAN
SW	BROWN	ADRIAN (T), MONROE
	BUFFALO	AGENDA (T), ASHLAND
	BURNETT	AHNAPEE (T), KEWAUNEE

3. By default Parking Lot and Private Property crashes (ACCDLOC) are excluded.

□ Check to include Parking Lot and Private Property crashes.

4. Include / Exclude Deer Crashes (DEERFLAG).

□ Check to exclude Deer crashes.

5. Filter by Crash Flags:

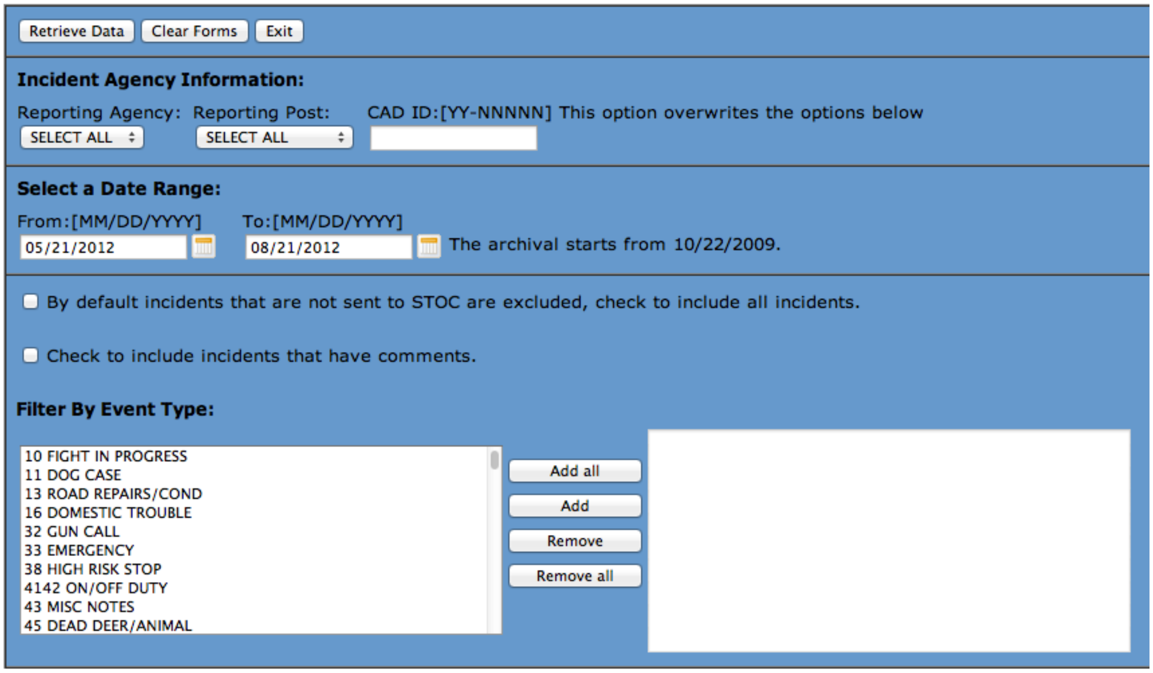
□ ALCFLAG (Alcohol)
□ AUTOFLAG (Passenger Car)
□ BIKEFLAG (Bicycle)
□ BUSFLAG (School Bus)
□ CYCLFLAG (Motorcycle)
□ DRUGFLAG (Drug Use)
□ HITRUN (Hit and Run)
□ MOPFLAG (Moped)

Figure 4.4 Retriever Tool

The retrieval facility provides the user with information in a web-based presentation of the data and allows the user to download the information in a comma separated values (.csv) format.

4.1.2.2 Source II: InterCAD Traffic Incident Data

The second database used for the study was the InterCAD Traffic Incident Data database (InterCAD). This database, while it contains much less data than the MV4000 database, contains the detection and end time for each incident, which is absolutely necessary for a complete database. In rare cases the InterCAD database was able to act as a supplement to MV4000 due to missing or insufficient data. While InterCAD does contain a free text field, these data are not standardized in any way and cannot be compared consistently with other data points. Figure 4.2 shows the user interface for the InterCAD Data Retrieval Facility.



Incident Agency Information:

Reporting Agency: SELECT ALL

Reporting Post: SELECT ALL

CAD ID:[YY-NNNNN] This option overwrites the options below

Select a Date Range:

From:[MM/DD/YYYY] 05/21/2012

To:[MM/DD/YYYY] 08/21/2012

The archival starts from 10/22/2009.

☐ By default incidents that are not sent to STOC are excluded, check to include all incidents.

☐ Check to include incidents that have comments.

Filter By Event Type:

10 FIGHT IN PROGRESS
11 DOG CASE
13 ROAD REPAIRS/COND
16 DOMESTIC TROUBLE
32 GUN CALL
33 EMERGENCY
38 HIGH RISK STOP
4142 ON/OFF DUTY
43 MISC NOTES
45 DEAD DEER/ANIMAL

Figure 4.5 InterCAD Data Retrieval Facility

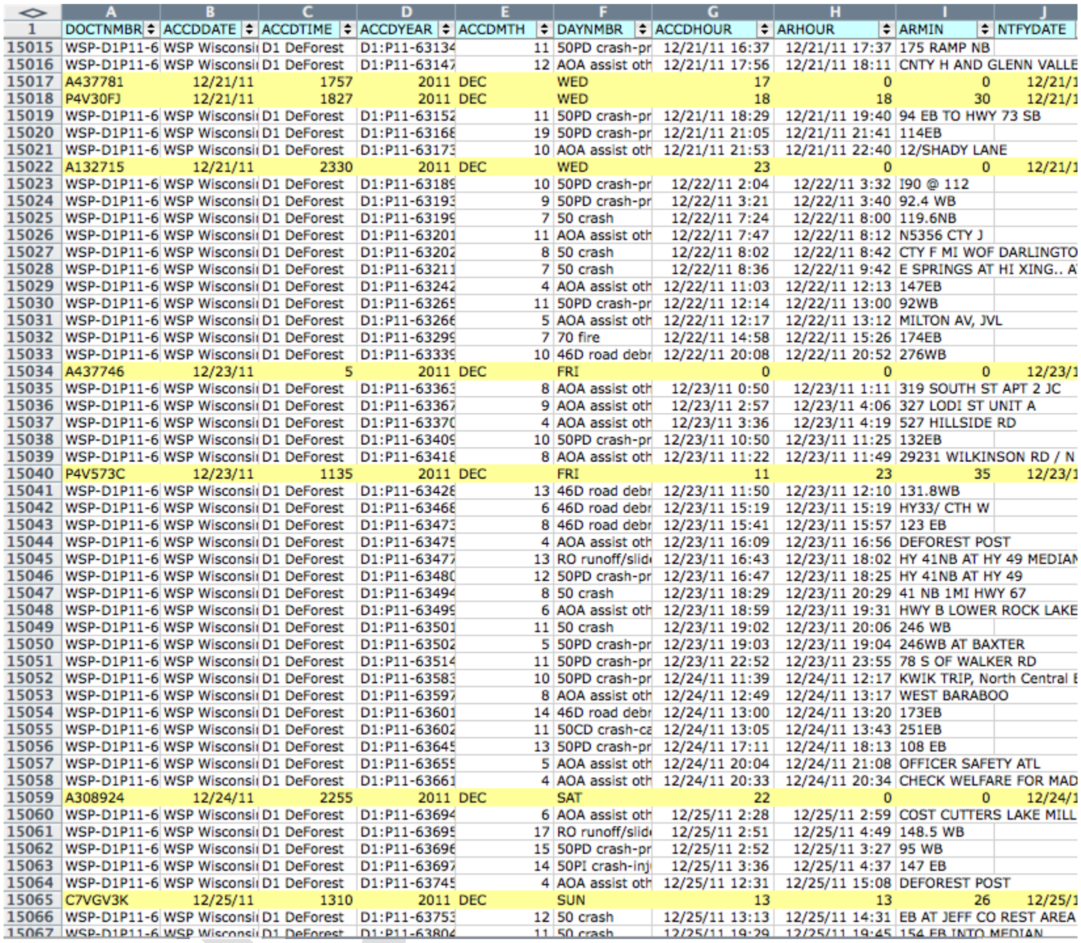
InterCAD, like MV4000, provides users with both a web-based interface as well as an option to download the data in a comma-separated values format.

4.1.3 Data Compilation

4.1.3.1 Database Merging

As stated previously, two databases were used as sources for the decision model development. The goal of the preliminary data collection was to produce a single data set from which to perform the analysis, so it was necessary to combine the two databases. There was no automated way to perform this task. The dataset was constructed by manually matching incidents between MV4000 and interCAD.

Figure 4.3 shows a screenshot of the databases after being combined by the data team. An algorithm written by the data team encoded in a column the date and time of each incident regardless of the database origin. One database was highlighted, and then they were sorted by date and time. By highlighting one database and sorting by date and time, the process of matching data points that described a common incident was simplified. This process was very labor-intensive, as each match must be evaluated on as many factors as possible to ensure that a false match is not made. At many times there were multiple crashes in an area in a fairly short time period. Identifying information such as whether the age of the driver is mentioned in both databases helps to make a positive match. The author was careful to reject a match when in doubt to avoid throwing off any of the data in the final data set.



<>	A	B	C	D	E	F	G	H	I	J
1	DOCTNMBR	ACCDDATE	ACCDTIME	ACCDYEAR	ACCDMTH	DAYNMBR	ACCDHOUR	ARHOUR	ARMIN	NTFYDATE
15015	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63134	11	50PD crash-pr	12/21/11 16:37	12/21/11 17:37	175 RAMP NB	
15016	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63147	12	AOA assist oth	12/21/11 17:56	12/21/11 18:11	CNTY H AND GLENN VALLE	
15017	A437781	12/21/11	1757	2011	DEC	WED	17	0	0	12/21/1
15018	P4V30FJ	12/21/11	1827	2011	DEC	WED	18	18	30	12/21/1
15019	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63152	11	50PD crash-pr	12/21/11 18:29	12/21/11 19:40	94 EB TO HWY 73 SB	
15020	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63168	19	50PD crash-pr	12/21/11 21:05	12/21/11 21:41	114EB	
15021	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63173	10	AOA assist oth	12/21/11 21:53	12/21/11 22:40	12/SHADY LANE	
15022	A132715	12/21/11	2330	2011	DEC	WED	23	0	0	12/21/1
15023	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63189	10	50PD crash-pr	12/22/11 2:04	12/22/11 3:32	I90 @ 112	
15024	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63193	9	50PD crash-pr	12/22/11 3:21	12/22/11 3:40	92.4 WB	
15025	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63199	7	50 crash	12/22/11 7:24	12/22/11 8:00	119.6NB	
15026	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63201	11	AOA assist oth	12/22/11 7:47	12/22/11 8:12	N5356 CTY J	
15027	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63202	8	50 crash	12/22/11 8:02	12/22/11 8:42	CTY F MI WOF DARLINGTO	
15028	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63211	7	50 crash	12/22/11 8:36	12/22/11 9:42	E SPRINGS AT HI XING.. A	
15029	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63242	4	AOA assist oth	12/22/11 11:03	12/22/11 12:13	147EB	
15030	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63265	11	50PD crash-pr	12/22/11 12:14	12/22/11 13:00	92WB	
15031	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63266	5	AOA assist oth	12/22/11 12:17	12/22/11 13:12	MILTON AV, JVL	
15032	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63299	7	70 fire	12/22/11 14:58	12/22/11 15:26	174EB	
15033	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63339	10	46D road debr	12/22/11 20:08	12/22/11 20:52	276WB	
15034	A437746	12/23/11	5	2011	DEC	FRI	0	0	0	12/23/1
15035	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63363	8	AOA assist oth	12/23/11 0:50	12/23/11 1:11	319 SOUTH ST APT 2 JC	
15036	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63367	9	AOA assist oth	12/23/11 2:57	12/23/11 4:06	327 LODI ST UNIT A	
15037	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63370	4	AOA assist oth	12/23/11 3:36	12/23/11 4:19	527 HILLSIDE RD	
15038	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63409	10	50PD crash-pr	12/23/11 10:50	12/23/11 11:25	132EB	
15039	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63418	8	AOA assist oth	12/23/11 11:22	12/23/11 11:49	29231 WILKINSON RD / N	
15040	P4V573C	12/23/11	1135	2011	DEC	FRI	11	23	35	12/23/1
15041	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63428	13	46D road debr	12/23/11 11:50	12/23/11 12:10	131.8WB	
15042	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63468	6	46D road debr	12/23/11 15:19	12/23/11 15:19	HY33/ CTH W	
15043	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63473	8	46D road debr	12/23/11 15:41	12/23/11 15:57	123 EB	
15044	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63475	4	AOA assist oth	12/23/11 16:09	12/23/11 16:56	DEFOREST POST	
15045	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63477	13	RO runoff/slid	12/23/11 16:43	12/23/11 18:02	HY 41NB AT HY 49 MEDIAN	
15046	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63480	12	50PD crash-pr	12/23/11 16:47	12/23/11 18:25	HY 41NB AT HY 49	
15047	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63494	8	50 crash	12/23/11 18:29	12/23/11 20:29	41 NB 1MI HWY 67	
15048	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63499	6	AOA assist oth	12/23/11 18:59	12/23/11 19:31	HWY B LOWER ROCK LAKE	
15049	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63501	11	50 crash	12/23/11 19:02	12/23/11 20:06	246 WB	
15050	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63502	5	50PD crash-pr	12/23/11 19:03	12/23/11 19:04	246WB AT BAXTER	
15051	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63514	11	50PD crash-pr	12/23/11 22:52	12/23/11 23:55	78 S OF WALKER RD	
15052	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63583	10	50PD crash-pr	12/24/11 11:39	12/24/11 12:17	KWIK TRIP, North Central E	
15053	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63597	8	AOA assist oth	12/24/11 12:49	12/24/11 13:17	WEST BARABOO	
15054	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63601	14	46D road debr	12/24/11 13:00	12/24/11 13:20	173EB	
15055	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63602	11	50CD crash-ca	12/24/11 13:05	12/24/11 13:43	251EB	
15056	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63645	13	50PD crash-pr	12/24/11 17:11	12/24/11 18:13	108 EB	
15057	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63655	5	AOA assist oth	12/24/11 20:04	12/24/11 21:08	OFFICER SAFETY ATL	
15058	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63661	4	AOA assist oth	12/24/11 20:33	12/24/11 20:34	CHECK WELFARE FOR MAD	
15059	A308924	12/24/11	2255	2011	DEC	SAT	22	0	0	12/24/1
15060	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63694	6	AOA assist oth	12/25/11 2:28	12/25/11 2:59	COST CUTTERS LAKE MILL	
15061	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63695	17	RO runoff/slid	12/25/11 2:51	12/25/11 4:49	148.5 WB	
15062	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63696	15	50PD crash-pr	12/25/11 2:52	12/25/11 3:27	95 WB	
15063	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63697	14	50PI crash-inj	12/25/11 3:36	12/25/11 4:37	147 EB	
15064	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63745	4	AOA assist oth	12/25/11 12:31	12/25/11 15:08	DEFOREST POST	
15065	C7VGV3K	12/25/11	1310	2011	DEC	SUN	13	13	26	12/25/1
15066	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63753	12	50 crash	12/25/11 13:13	12/25/11 14:31	EB AT JEFF CO REST AREA	
15067	WSP-D1P11-6	WSP Wisconsi	D1 DeForest	D1:P11-63804	11	50 crash	12/25/11 19:29	12/25/11 19:45	154 EB INTO MEDIAN	

Figure 4.6 A Screenshot of the Databases Combined into One Spreadsheet

4.1.3.2 Final Data Set

The merging of the database was not the final step in developing the dataset. The final data set consists of a new layout in the most advantageous manner for this study. Data fields that were deemed useful by the author were included in the final database, as well as fields generated by the author. Examples of fields generated include time parameters that were generated from existing fields in the data, some data that required a conversion from text to numeric form in cases where the author found that it would be more useful, and cases where it was necessary to generate a field that depended on multiple other fields. To create a field that tells the user whether or not trucks were involved in a given incident, that field must be dependent on all fields describing vehicle type.

4.2 Division of Freeway Segments

To ensure that the proposed decision model is effective under a wide range of work zone and incident scenarios and roadway geometric and traffic conditions, the research team has divided the interstate highway 94 (Madison - Milwaukee) into eighteen segments. The purpose of the division is to make sure each segment includes the freeway mainline experiencing an incident/work zone construction, on-ramps, off ramps, upstream and downstream of the incident location, and the connecting parallel detour route. All divided segments are described as follows:

Figure 4.4 shows the configuration of the first segment that starts at County Highway N in Dane County and ends at State Highway 73. Figure 4.4 indicates that the eastbound path and the westbound path both use the same highway segments in reverse. Also noted in Figure 4.4 as well as in the subsequent segment figures is the location of traffic control devices, stop signs, and traffic signals.

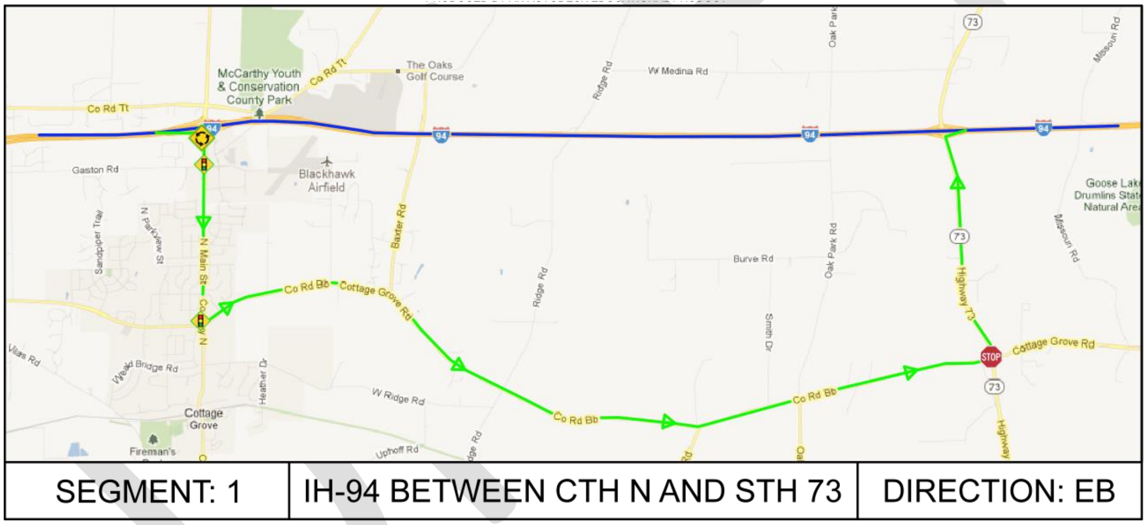


Figure 4.4 Segment 1

Figure 4.5 shows the configuration of the second segment. This segment also uses the same route in both directions. Segment 2 traverses from Dane County to Jefferson County from west to east.

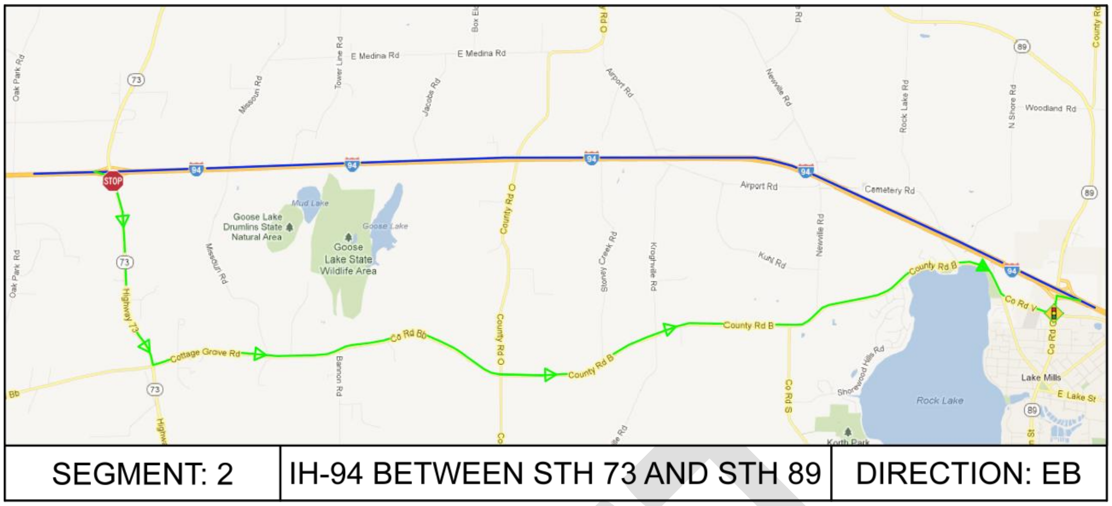


Figure 4.5 Segment 2

The third segment is shown in Figure 4.6. Segment 3, located in Jefferson County, uses the same route in both directions.

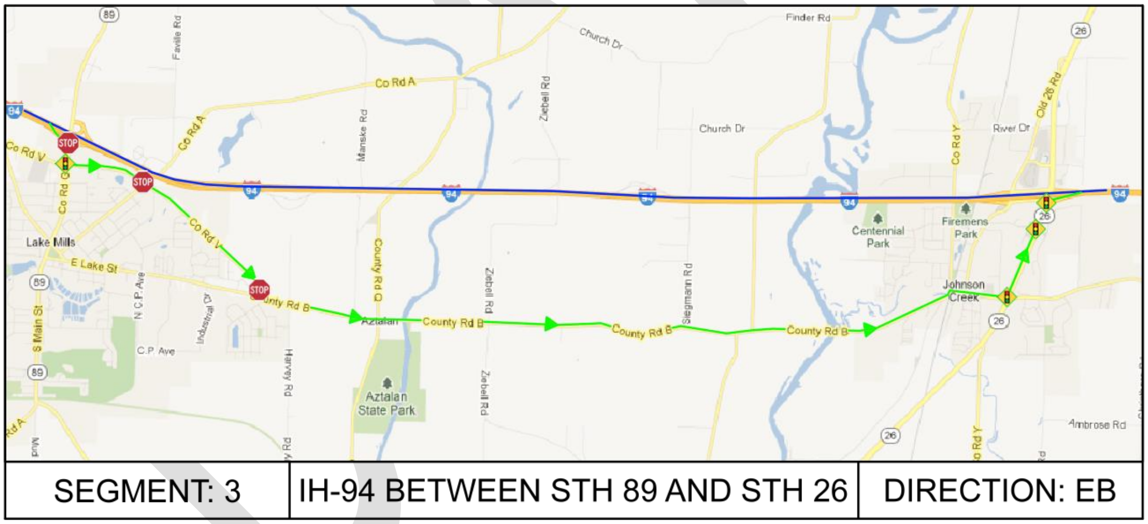


Figure 4.6 Segment 3

Figure 4.7 describes Segment 4. This segment also is located in Jefferson County and uses the same path in both directions.

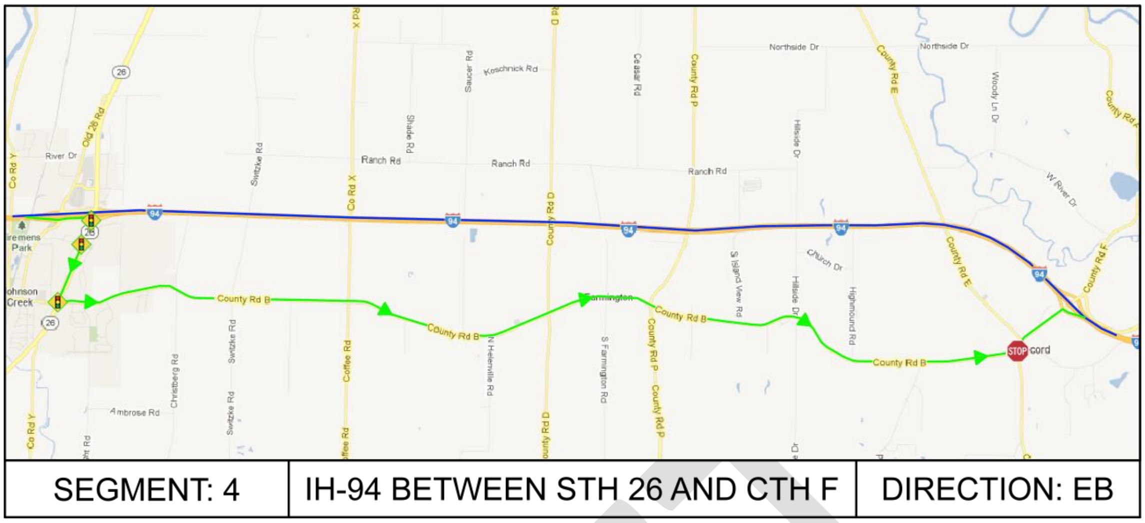


Figure 4.7 Segment 4

Segment 5 is the only segment in which traffic must be diverted to another segment in order to form a full diversion route. The reason that traffic cannot be contained in segment 5 is the lack of an eastbound on-ramp and a westbound off-ramp at the interchange with Willow Glen Road. Because segment 6 has two viable diversion paths, Segment 5 does also, because traffic must be diverted into Segment 6. Segment 5 is also in Jefferson County.

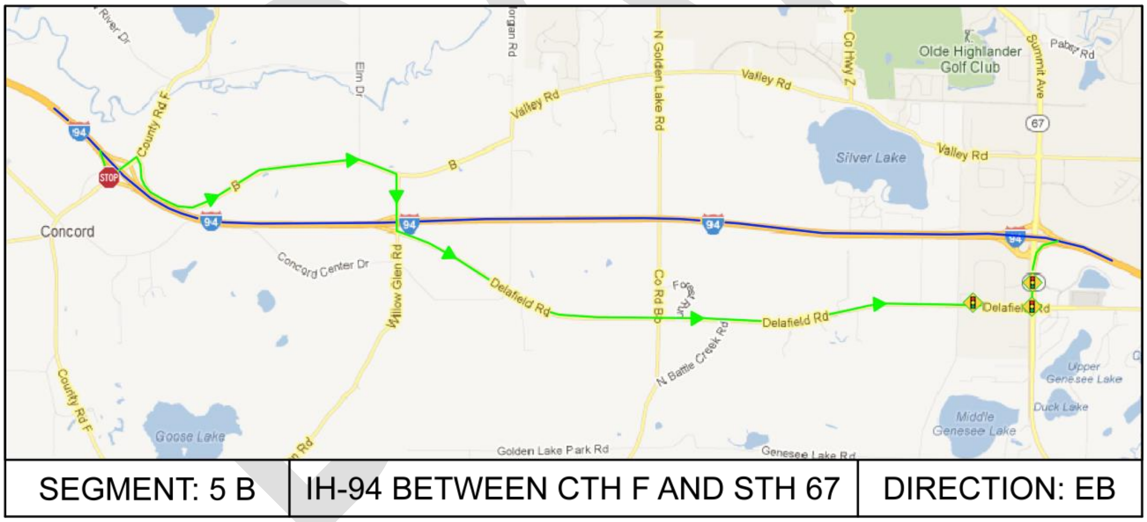


Figure 4.8 Segment 5

Segment 6, shown in Figure 4.9, spans from Jefferson County to Waukesha County from west to east. Segment 6 is used by traffic diverting due to incidents located along segment 5, but is not affected by the nonstandard interchange configuration at Willow Glen Road when incidents occur within Segment 6.

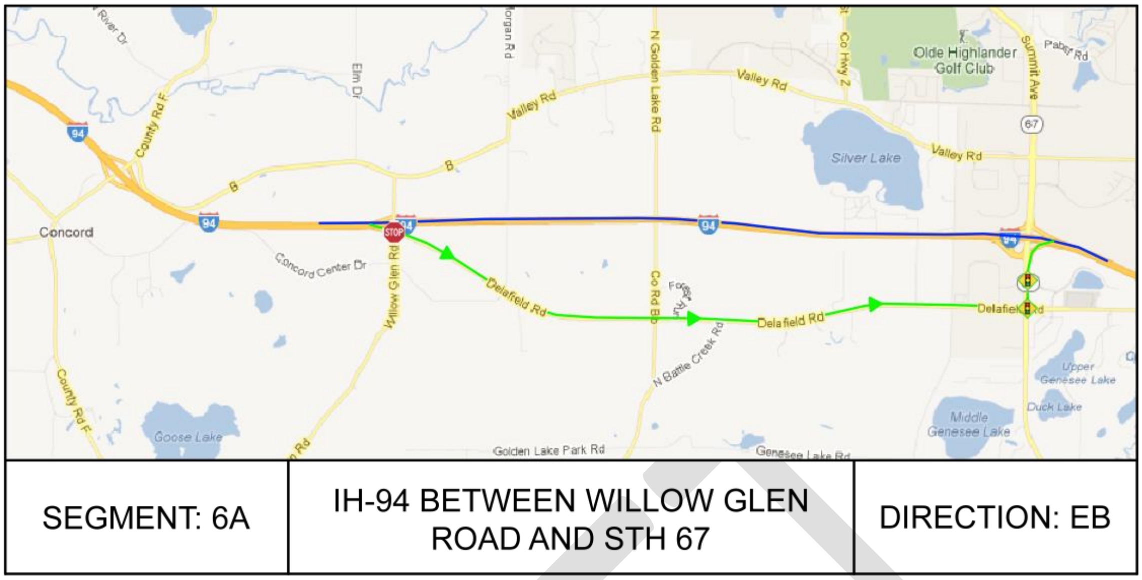


Figure 4.9 Segment 6

Figure 4.10 shows Segment 7 located entirely in Waukesha County. Segment 7 uses the same route for traffic diverting in both directions.

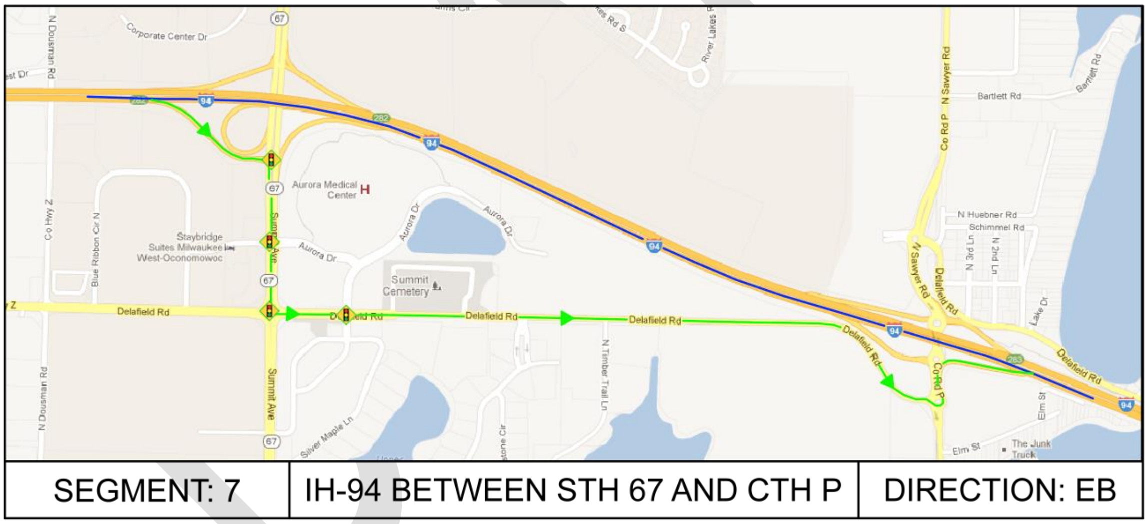
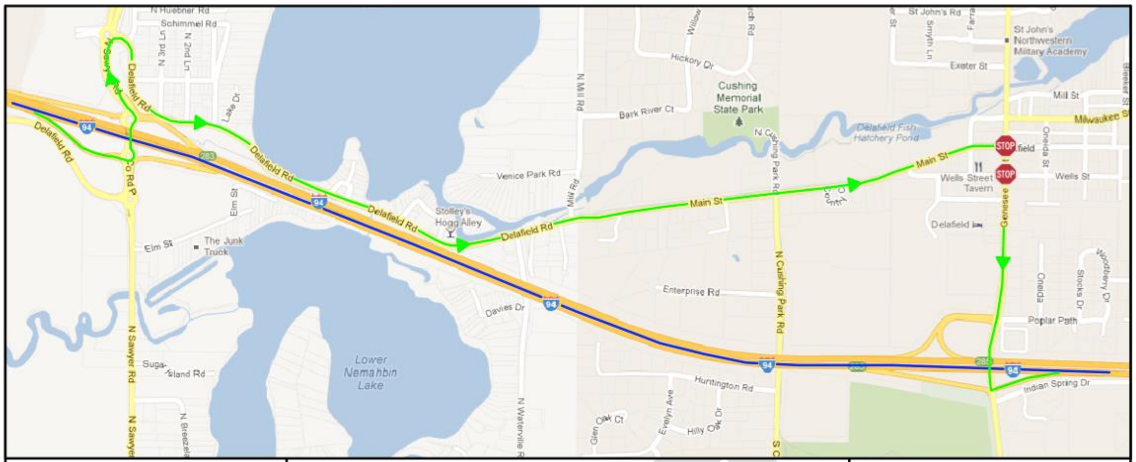


Figure 4.10 Segment 7

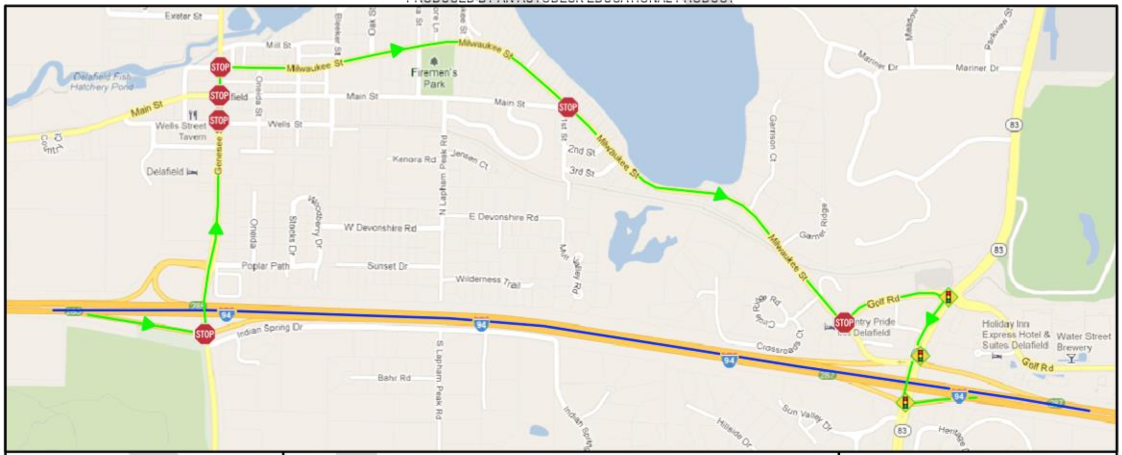
Segment 8 is shown in Figure 4.11. Segment 8 is also located in Waukesha County and uses the same route in both directions.



SEGMENT: 8	IH-94 BETWEEN CTH P AND GENESEE ST	DIRECTION: EB

Figure 4.11 Segment 8

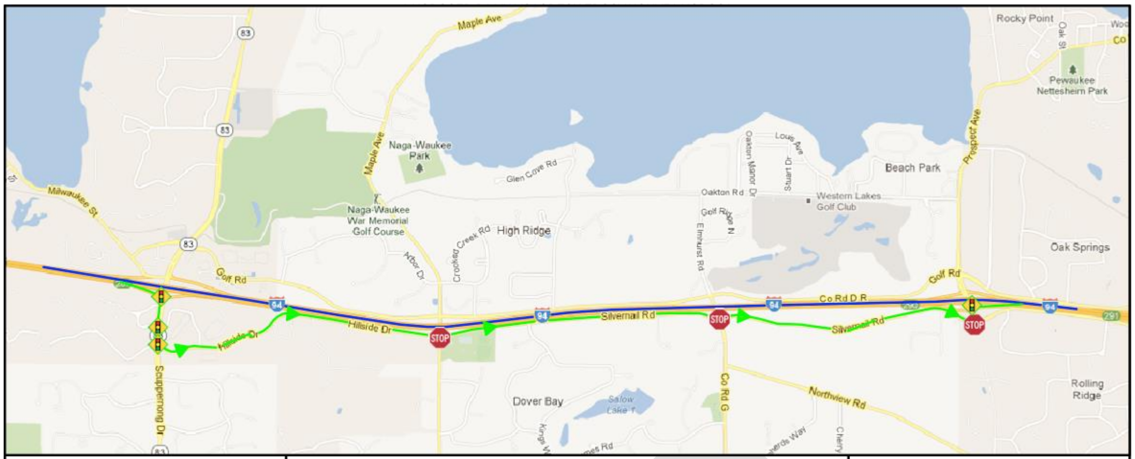
Segment 9 is shown in Figure 4.12. Segment 9 is located in Waukesha County and uses the same routes for diversion traffic in both directions.



SEGMENT: 9	IH-94 BETWEEN GENESEE ST AND STH 83	DIRECTION: EB

Figure 4.12 Segment 9

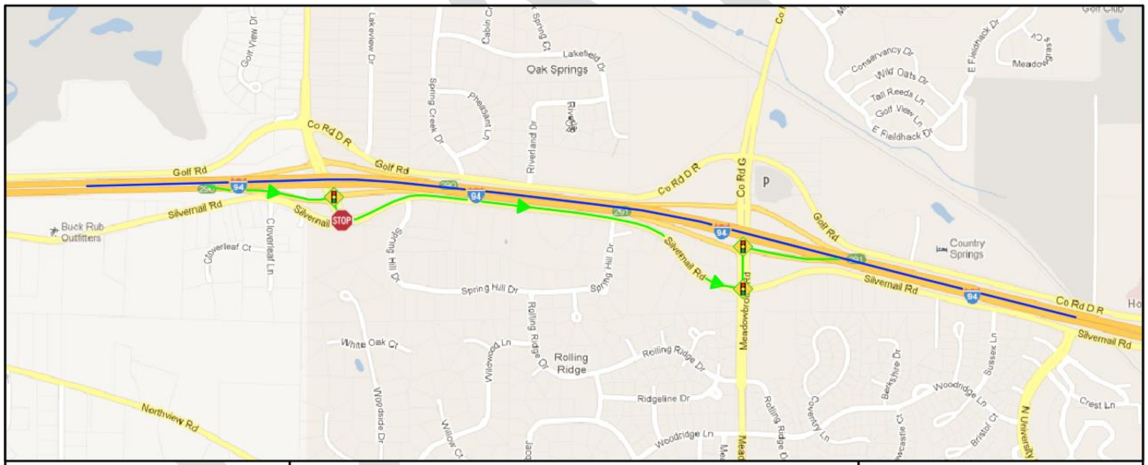
Figure 4.13 shows the configuration of Segment 10. Segment 10 is located in Waukesha County and diversion traffic can travel in either direction using either the road to the north of the freeway segment, Golf Road, or the road to the south of the freeway segment, Silvernail Road. All possible routes were identified in order to find the optimal diversion route for any segment in which multiple routes were available for diversion traffic.



SEGMENT: 10	IH-94 BETWEEN STH 83 ST AND CTH SS	DIRECTION: EB

Figure 4.13 Segment 10

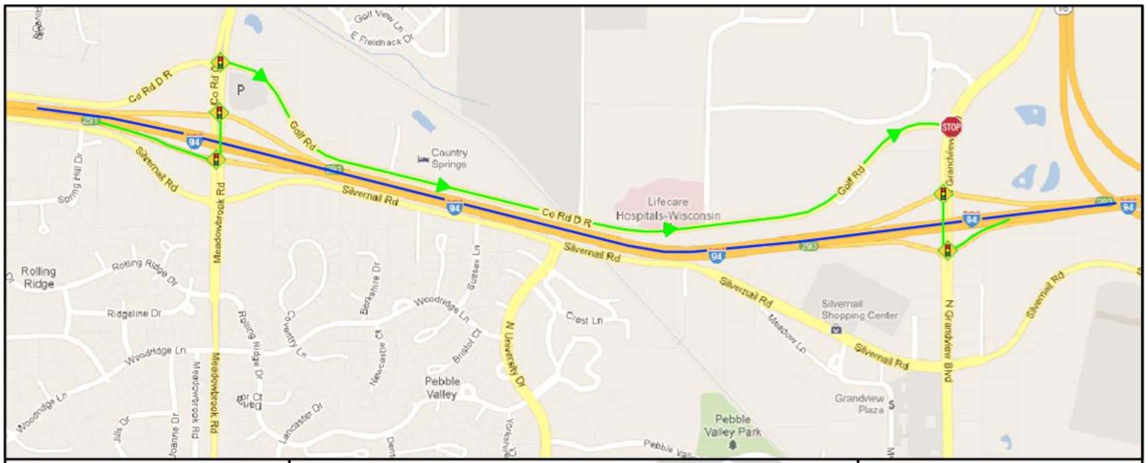
Figure 4.14 shows the eastbound route of diversion traffic for Segment 11. Segment 11 is also located in Waukesha County and can accommodate two different diversion routes, Golf Road to the North, and Silvernail Road to the south.



SEGMENT: 11	IH-94 BETWEEN CTH SS AND CTH G	DIRECTION: EB

Figure 4.14 Segment 11

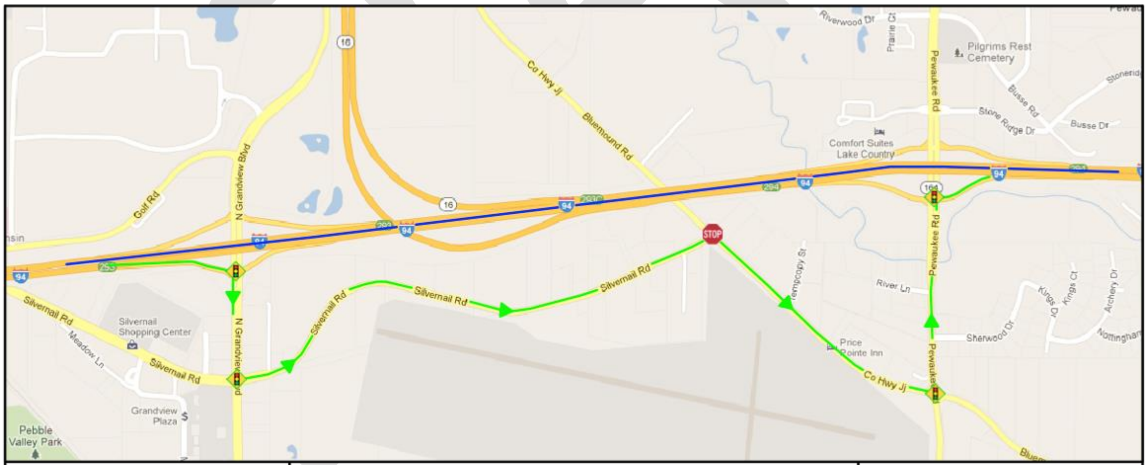
Figure 4.15 shows Segment 12. Again, Segment 12 is located in Waukesha County and allows diversion traffic to travel in two different routes, Golf Road located to the north of the freeway segment, and Silvernail Road located to the south of the freeway segment.



SEGMENT: 12	IH-94 BETWEEN CTH G AND GRANDVIEW BLVD	DIRECTION: EB

Figure 4.15 Segment 12

Segments 13 and 14 are both contained in Figure 4.16, as they are never utilized independent of one another. STH 16 forms an interchange with IH-94 at the dividing line between segments 13 and 14 and is used for a reference point to note incident locations, however STH 16 does not form any part of any diversion route. Segment 13-14 utilizes only one diversion path that accommodated diversion traffic in both directions. Segments 13 and 14 are located in Waukesha County.

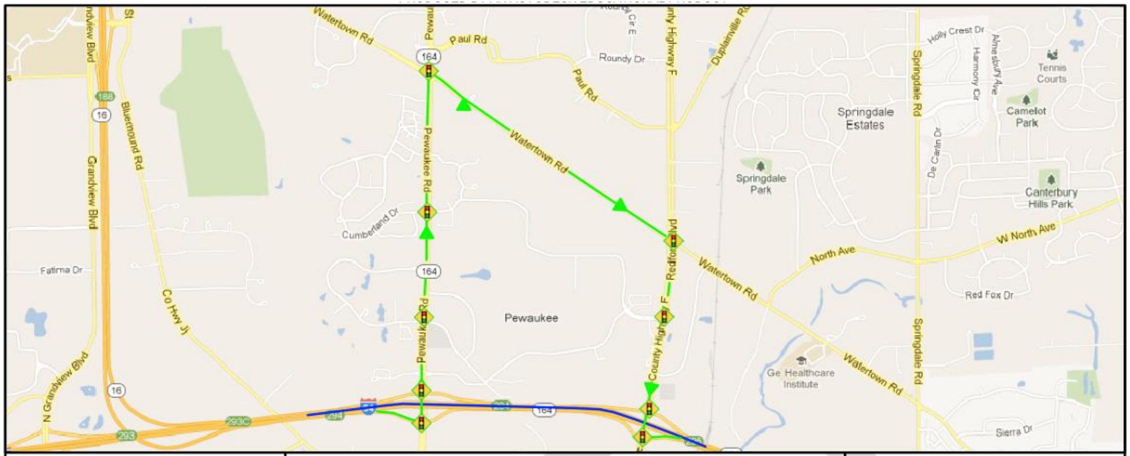


SEGMENT: 13 -14	IH-94 BETWEEN GRANDVIEW BLVD AND WIS 164	DIRECTION: EB

Figure 4.16 Segments 13 and 14

Segment 15 A is shown in Figure 4.17. Segment 15 has two different diversion paths and is represented in a separated figure for each. Figure 4.17 indicated the northern diversion route for Segment 15 that uses Watertown Road. Segment 15 is located in

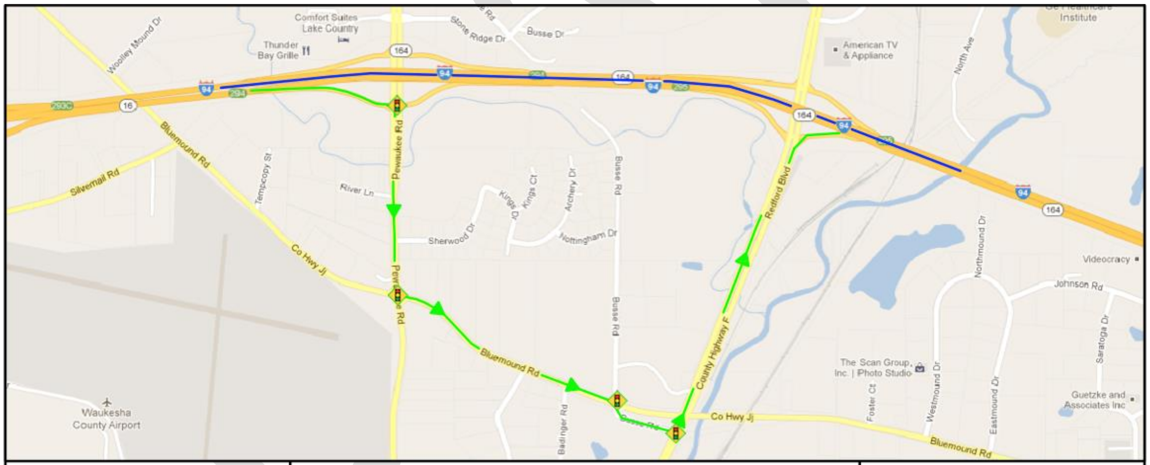
Waukesha County, and both diversion paths can accommodate diversion traffic in both eastbound and westbound directions.



SEGMENT: 15	IH-94 BETWEEN PEWAUKEE RD (WIS 164) AND CTH F (WIS 164)	DIRECTION: EB

Figure 4.17 Segment 15A

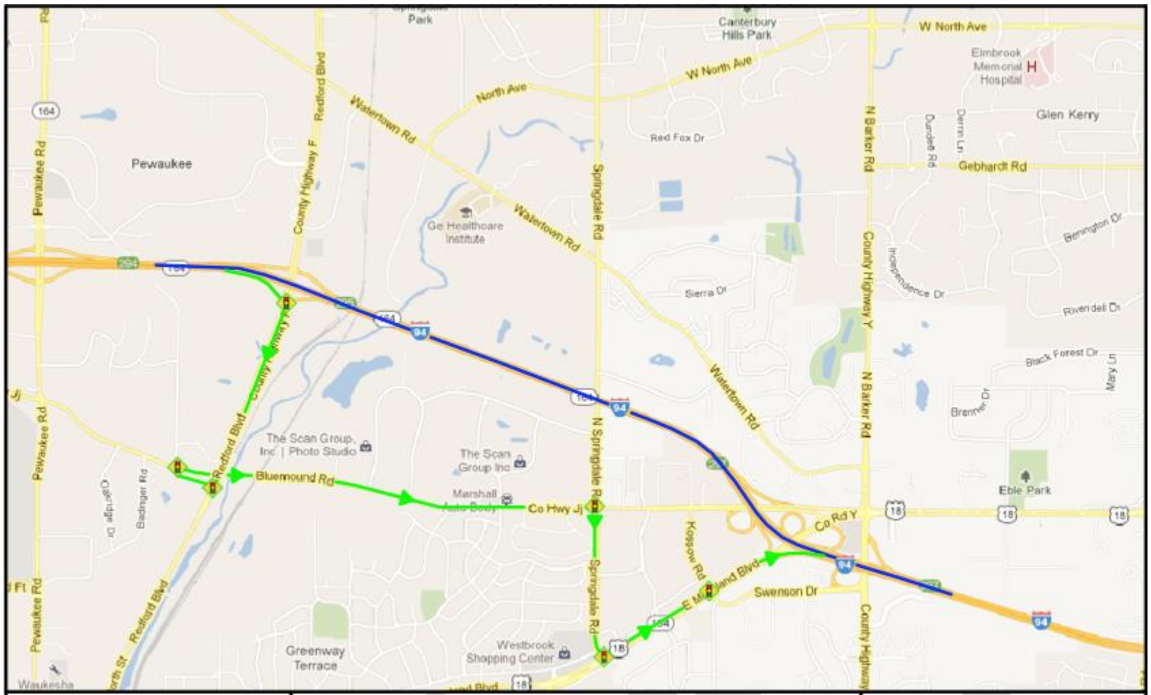
Figure 4.18 shows the southern diversion route for Segment 15 using CTH JJ.



SEGMENT: 15	IH-94 BETWEEN PEWAUKEE RD (WIS 164) AND CTH F (WIS 164)	DIRECTION: EB

Figure 4.18 Segment 15B

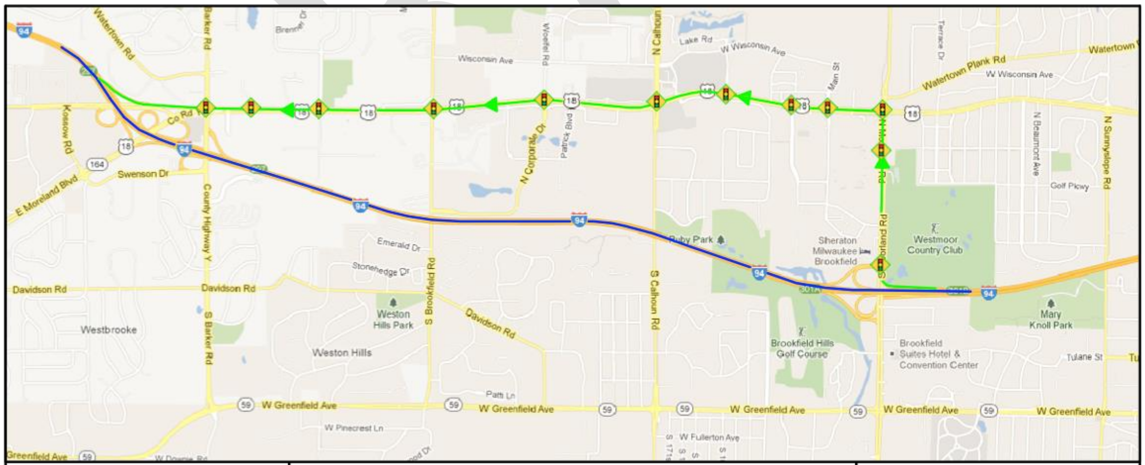
Figure 4.19 shows the southern diversion route for Segment 16. This segment is located in Waukesha County as well. Both diversion routes are capable of accommodating diversion traffic in both directions.



SEGMENT: 16	IH-94 BETWEEN CTH F AND E MORELAND BLVD (US-18)	DIRECTION: EB

Figure 4.19 Segment 16

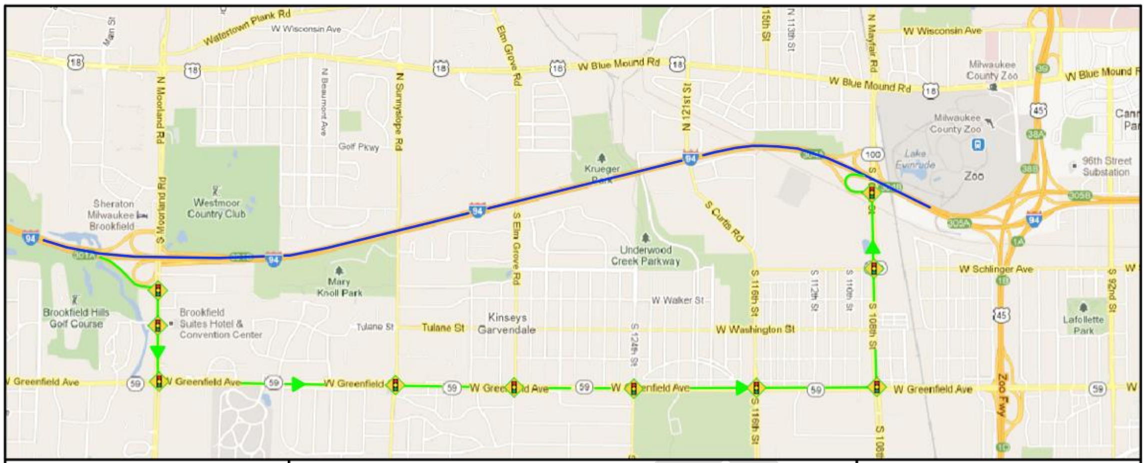
Figure 4.20 indicates the diversion plan for Segment 17. This segment is located in Waukesha County and also has 2 different diversion paths. In Figure 4.20 the northern diversion route is diagramed



SEGMENT: 17	IH-94 BETWEEN E MORELAND BLVD (US - 18) AND S MORELAND RD	DIRECTION: WB

Figure 4.20 Segment 17

Segment 18 is described in Figure 4.21. Segment 18 also has two diversion routes. The southern diversion route is shown in Figure 4.21, using Greenfield Avenue.



SEGMENT: 18	IH-94 BETWEEN S MORELAND RD AND MAYFAIR RD (WIS 100)	DIRECTION: EB

Figure 4.21 Segment 18

4.3 Experimental Design

During an incident/work zone construction, many factors may affect the final decision on whether or not to implement alternative route guidance to travelers, such as traffic volumes on the freeway and the detour route, the incident duration, the number of lanes blocked, and the number of signals on the detour route, etc. To ensure that the proposed decision model is effective under a wide range of incident scenarios and roadway geometric and traffic conditions, freeway segments extracted from Section 4.2 are modeled and calibrated in a microscopic simulation test bed. It will be quite cost-effective to use such an experimental environment to replicate a variety of complex and dynamic traffic patterns as well as the real-world operational characteristics (e.g. turning-bay, delay on ramps, and driving behavior) that may contribute to warranting an alternative route. Conceptual design of the experiment and key contributing factors are shown in Figure 4.22.

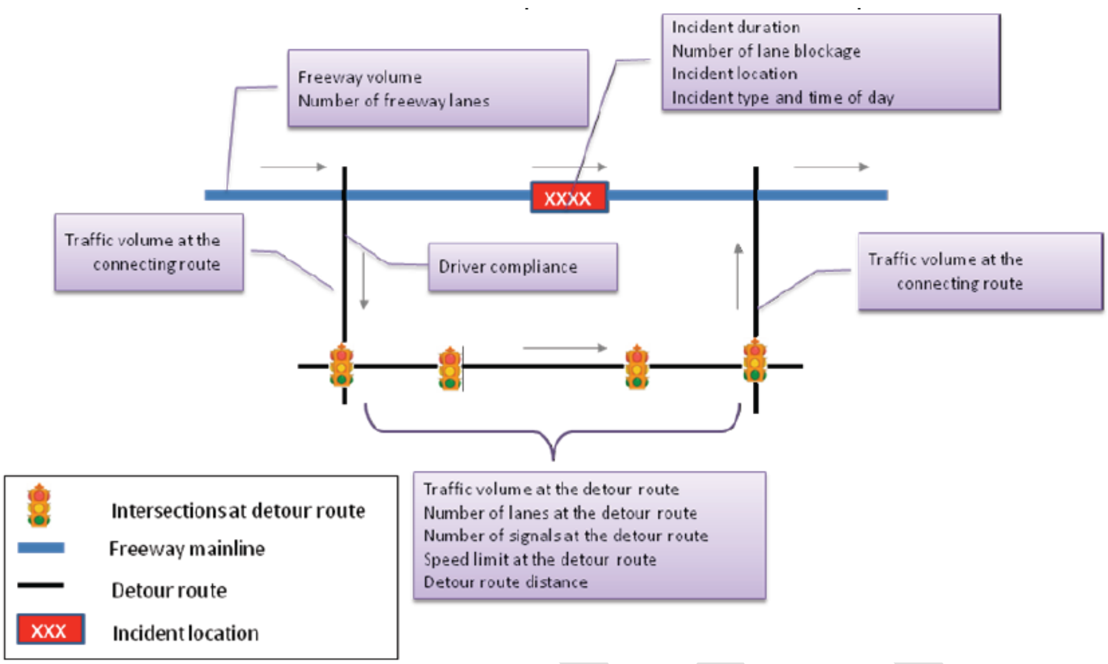

Incident duration
Number of lane blockage
Incident location
Incident type and time of day
Freeway volume
Number of freeway lanes
XXXX
Traffic volume at the connecting route
Driver compliance
Traffic volume at the connecting route
Intersections at detour route
Freeway mainline
Detour route
XXX
Incident location
Traffic volume at the detour route
Number of lanes at the detour route
Number of signals at the detour route
Speed limit at the detour route
Detour route distance

Figure 4.22 Conceptual experimental design and key contributing factors

4.3.1 Simulation Network Construction

To realistically reflect the real-world operational characteristics in the study network (e.g., turning-bay, delay on ramps, and driving behavior), this study has modeled and calibrated each experimental scenario with the widely used micro-simulation package, CORSIM. The networks built with this the graphical interface TRAFED in the TSIS™ software represent the segments.

The simulation network for each segment can be graphically demonstrated given the proper dimension as TRAFED allows the user to use a bitmap image as a background to a network and to specify the real world width. For example, Figure 4.23 shows an overview of a network that has been created in the TSIS™ software package using TRAFED.

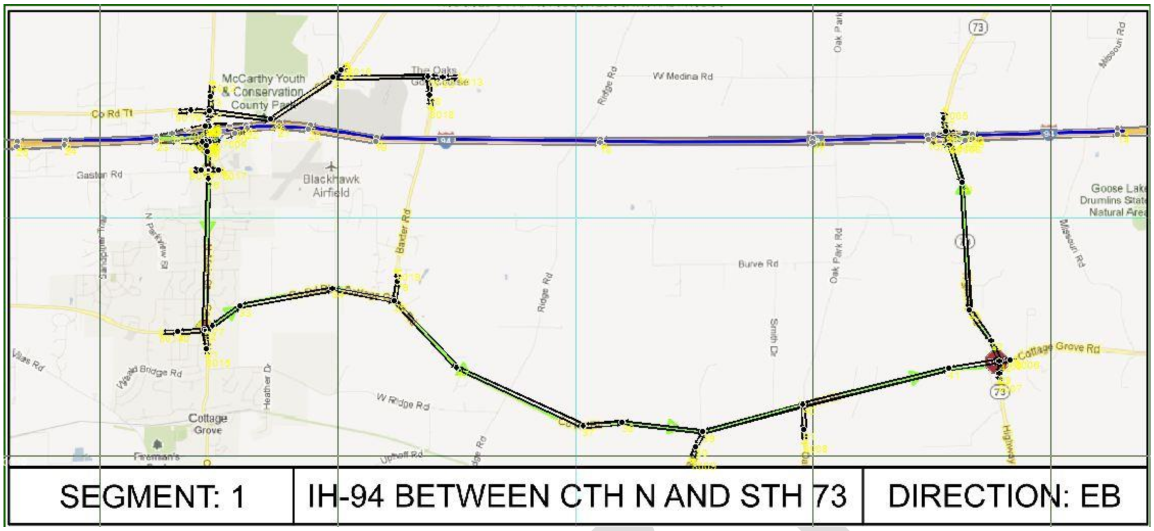


Figure 4.23 An example freeway corridor segment in the simulation test bed

4.3.1.1 Simulation on Interchange

Interchange is a special geometry that needs more efforts to deal with in constructing simulation network. Figure 4.24 is a close-in view of an interchange created in TRAFED that is part of Segments 17 and 18. While the radii are displayed in TRAFVU, they are not considered in the simulation model. The length of the segment is, however, considered. For example, if a segment's end points are 500 feet apart, but the user specifies that the length of segment is 785 feet (if those two points were opposite each other in a semi-circle) the simulation will treat that segment as if it were 785 feet and if the user chooses to display it as a half circle. Unless specified, TSIS™ does not necessarily treat a vehicle leaving a segment to enter another at an angle as a turning vehicle, so the lack of consideration in a curved segment does not matter.

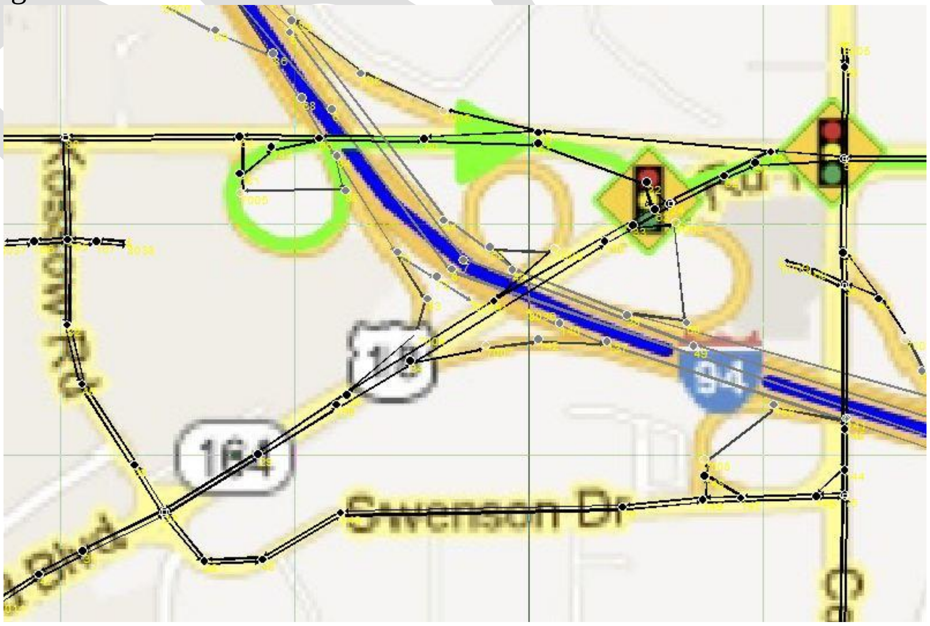


Figure 4.24 A Close-in View of An Interchange

4.3.1.2 Simulation on Intersections

Intersection geometries are important factors in the performance of a high volume traffic network. Figure 4.25 shows a typical intersection layout found in an urban segment as laid out in TSIS™ to represent real world conditions. TRAFVU was not an important tool in ascertaining the performance of the networks. Numerical output parameters were used instead of any graphically observed measures in determining network performance. While TRAFVU was not necessary for any data collection, it was very important when verifying that the network had been laid out correctly. In the TRAFED view, a segment or an intersection would have to be examined in a dialogue box individually to verify that it had been specified correctly. TRAFVU allows the user to examine the entire network by panning it around with parameters such as the number of lanes and the correctly specified number of turning bays easily verified without having to enter into a dialogue box for each component.

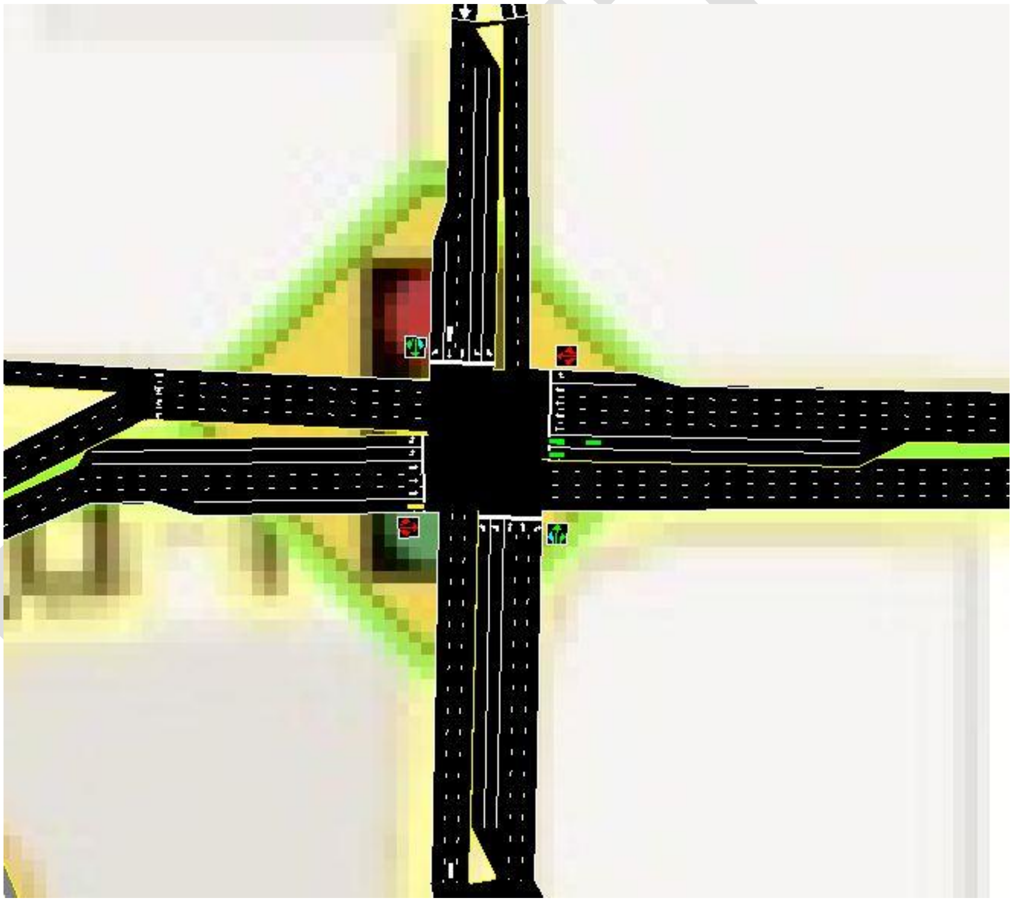
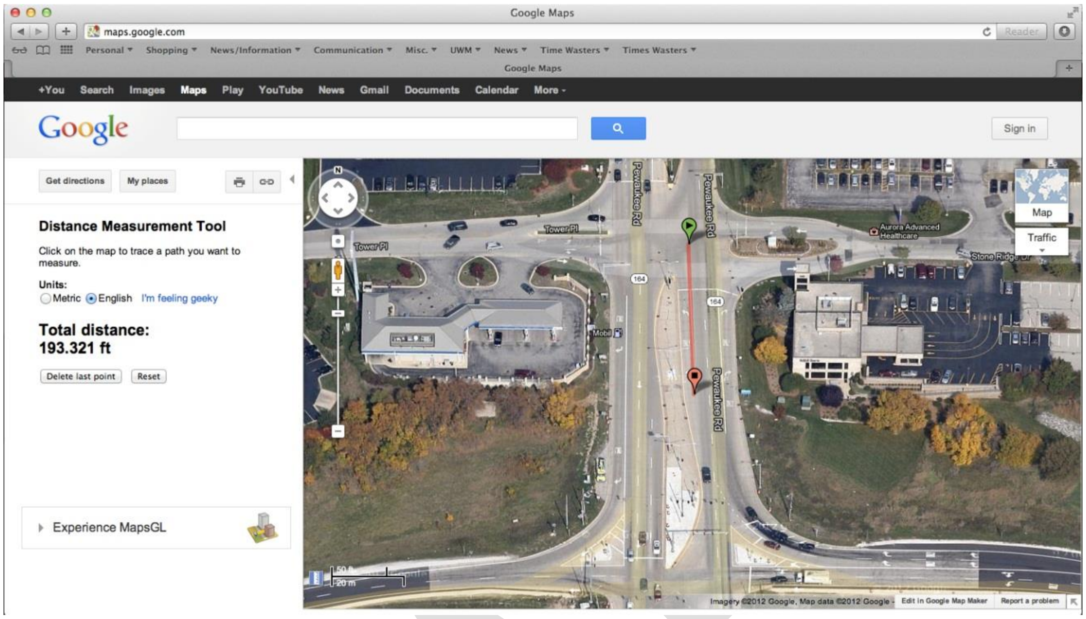


Figure 4.25 A Typical Intersection Layout

4.3.1.3 Technique on Geometric Parameter Estimation

Google Maps was a very important part of the data collection of this study. Without Google Maps, the process of ascertaining the properties described in this section would have become onerous, or the degree of accuracy attained would have been severely diminished.

Using Google Maps, geometric data was collected for each of the segments. In addition to geometric data such as the number of lanes comprising a road segment, using the Distance Measurement Tool facilitates obtaining distances for turn bays, freeway auxiliary lanes, and any other critical dimension.



Distance Measurement Tool

Click on the map to trace a path you want to measure.

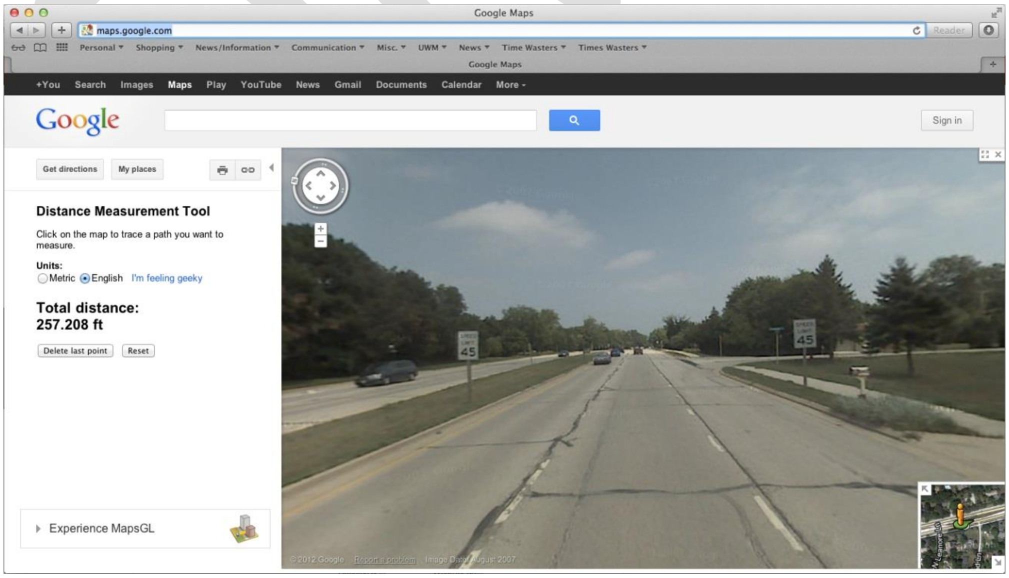
Units:
○Metric ●English I'm feeling geeky

Total distance:
193.321 ft

Delete last point Reset

Figure 4.26 Demonstration of Using Distance Measure Tool

Figure 4.26 demonstrates the use of the Distance Measuring Tool. It is worth noting that while the simulation animation software TRAFVU renders networks in an aesthetically pleasing manner, such as rendering tapers at freeway lane drops, TSIS™ does not recognize partial lanes or assign vehicles to multiple lanes at once. For this reason, features such as turn bays must be measured from the point at which a usable lane width exists, not at the point where the taper begins, as shown in Figure 4.26.



Distance Measurement Tool

Click on the map to trace a path you want to measure.

Units:
○Metric ●English I'm feeling geeky

Total distance:
257.208 ft

Delete last point Reset

Figure 4.27 A Screen Shot of Street View

Some features of the segments such as the speed limit of a local road or freeway segment were ascertained using Google's Street View feature. Figure 4.27 is a screen shot of a road segment in Segment 18. Using Street View, one can see that the speed limit on this segment is 45 miles per hour. One other use of Street View is to corroborate with aerial photos to clarify attributes of a segment. Because not all photos used in Google Maps were taken at the same time, different views can also be useful to ensure that the newest data are used.

In summary, Google Maps is great tool in this study to estimate important geometric parameters such as the distances for turn bays, freeway auxiliary lanes, speed limits of a certain corridor and many other critical geometric attributes. Other parameters such as turn volumes, entrance node volumes and exit percentages on the freeways were necessary only to test that the network performed without any errors, as those parameters would be later specified in the running of the experiment, and many different combinations of values would be used.

4.3.2 Category of Key Variables

With a well-established simulation network for each segment, it is necessary to define the category of key factors that may potentially affect alternative route recommendation. This study organizes all the potential factors associated with each experimental scenario into the following groups:

- **Freeway-related factors**: flow rate on the freeway mainline and the number of lanes on the freeway mainline;
- **Work zone-related factors**: work zone duration and the number of lanes blocked;
- **Alternative route-related factors**: flow rate on the road connecting from the freeway to the alternative route, flow rate on the parallel route, flow rate on the road connecting from the alternative route back to the freeway, and the number of lanes and signals on the alternative route; and
- **Traveler-related factors**: level of traveler compliance rates to the alternative route recommendation.

4.3.3 Range of Variables Values

The range of values of some key factors that will be used in the model development is summarized in Table 4.1; note that these variables and corresponding ranges are original and they may be re-categorized for model construction if needed.

Table 4.1 Key Variables and Range of Values for the Experimental Design

VARIABLES	DESCIRPTION	RANGE OF VALUES
FR_VOL	Freeway mainline volume rate (in vphpl)	250, 750, 1250, 1750, 2200
FR_LN	Number of lanes on the freeway mainline	2, 3, 4

INC_DUR	Incident duration (in mins)	15, 30, 45,60, 75, 90,105, 120
LN_BLK	Number of lanes blocked	1, 2, 3, 4
LC_VOL1	Flow rate on the road connecting from freeway to detour route (in vphpl)	200, 300, 400, 500, 600, 700, 800
LC_VOL2	Flow rate on the detour route (in vphpl)	200, 300, 400, 500, 600, 700, 800
LC_VOL3	Flow rate on the road connecting from detour route to freeway (in vphpl)	200, 300, 400, 500, 600, 700, 800
LC_LN	Number of lanes on the detour route	1, 2, 3
NUM_SIGNAL	Number of signals on the detour route per mile	2, 3, 4, 5, 6, 7

4.3.4 Scenarios Generating

Considering the wide range of values taken by each contributing factor, the total number of experimental scenarios that can be generated from all possible combinations of key factors will be extremely large. For example, assuming each factor takes five possible values, one can generate a total of 5^{13} = 1,220,703,125 scenarios. It will be impossible to evaluate all those scenarios and further use them to develop decision models. To contend with this problem, the research team has adopted a probability sampling approach to randomly select scenarios from the sample space and assure that all scenarios have equal probabilities of being chosen. Using this procedure, this study has generated an experimental scenario set with a relatively compact size of 500. The generated scenario set will then be divided into two subsets, one subset containing 400 experimental scenarios for detour optimization model and decision model development and another subset containing 100 experimental scenarios for model validation.

4.4 Model Development and Validation

This section will develop and calibrate alterative route decision models. Before the development of the decision models, an optimization model developed by Liu et al. (2011) that can generate optimal detour rate will be presented in Section 4.4.1. The generated optimal detour rate will be compared with a threshold in the two-choice model to determine whether an alternative route decision should be made or not.

Section 4.4.2 presents the development and calibration results of a two-choice model considering many potential factors that have been presented in previous sections. A preliminary analysis with Classification and Regression Tree (CART) will be embedded in this section to analyze the significance of selected variables and re-group the variables to better develop the proposed two-choice model. Obviously, this model provides the result of "provide alternative route recommendation" or "do not provide alternative route recommendation," which is effective guidance in the process of incident/work zone management.

4.4.1 Diversion Optimization Model

As stated before, it is necessary to know the optimal detour rate for the development of an alternative route decision model. This study employs an integrated diversion control model developed by Liu et al. (2011) that can determine the best diversion control strategy (i.e., diversion rate, signal timing optimization, ramp metering) that yields the maximum use of corridor capacity for each experimental scenario and the optimal detour rate to the local route. The connection of such a model and CORSIM is illustrated in Figure 4.28. The experimental scenarios served as inputs for the proposed model and the outputs (diversion rate, signal timing optimization, ramp metering) of such a model associated with the experimental scenarios served as the inputs for CORSIM.

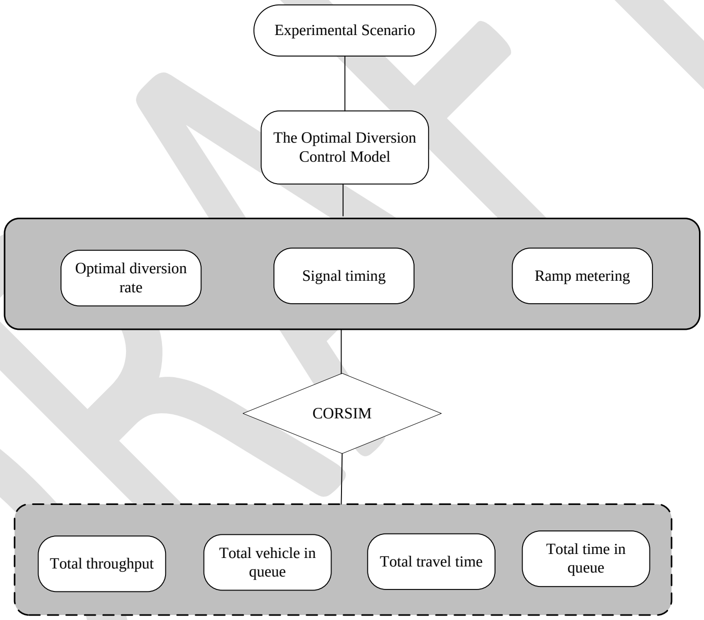
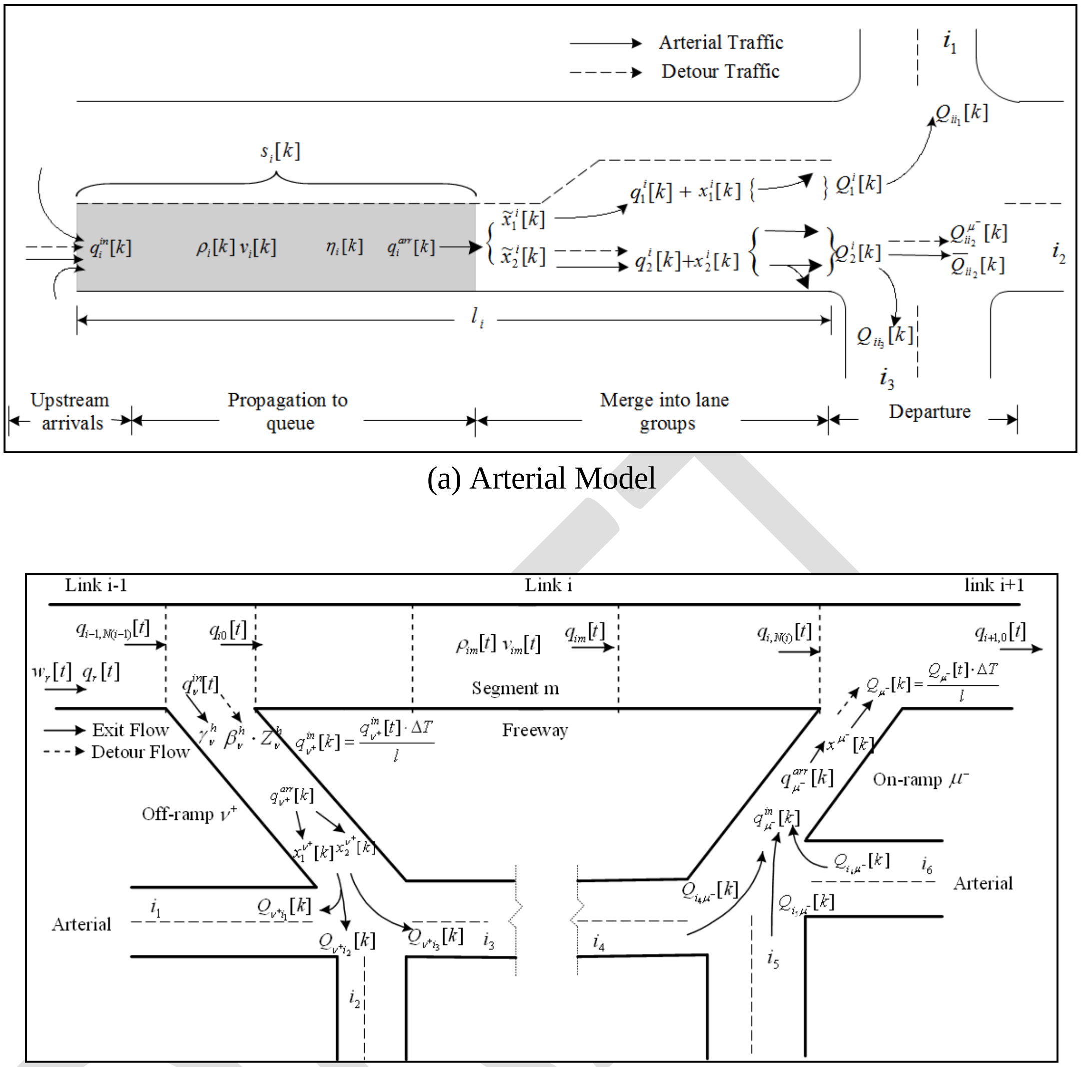


Figure 4.28 Connection of Detour Optimization Model and CORSIM

The integrated diversion control model has effectively integrated a set of macroscopic traffic flow models that can precisely model and predict the traffic evolution along the freeway mainline, arterial link, and on–off ramps (see Figure 4.29).



(a) Arterial Model

(b) Freeway and Ramps

Figure 4.29 Macroscopic Network Flow Modeling in the Integrated Diversion Control Model (Liu et al., 2011)

To facilitate the model presentation, the notations used hereafter are summarized below:

Notation

Δt : Time step for updating arterial status (secs);

T_h : Length of the control time interval h (#. of Δt);

H : The entire control time horizon;

k : Time step index of arterial system corresponds to time $t = k\Delta t$;

S_N : Set of arterial intersections;

$n, n \in S_N$: Index of arterial intersections;

S^U : Set of arterial links;

S^{OUT}: Set of outgoing arterial boundary links;

$i, i \in S^{U}$: Index of links,

S_r : Set of traffic demand entries;

P_n : Set of signal phases at intersection n;

$p,\ p \in P_n$: Index of signal phase at the intersection n;

$\Gamma(i), \Gamma^{-1}(i)$: Set of upstream and downstream links of link i;

l_i : Length of link i (ft);

n_i : Num. of lanes in link i;

N_i : Storage capacity of link i (vehs);

Q_i : Discharge capacity of link i (veh/h);

$\rho^{\min}, v_i^{free}$: Minimum density (veh/mile/lane) and the free flow speed at link i (mph);

$\rho^{jam}, v^{\min}$: Jam density (veh/mile/lane) and the minimum speed (mph);

α, β : Constant model parameters;

S_i^{M} : Set of lane groups at link i;

$m, m \in S_i^{M}$: Index of lane groups at link i;

$\delta_m^{ij},\ j \in \Gamma^{-1}(i)$: A binary value indicating whether the movement from link i to j uses lane group m;

Q_m^{i} : Discharge capacity of lane group m at link i (veh/h);

$d_r[k], r \in S_r$: Demand flow rate at entry r at step k (veh/h);

$q_r[k], r \in S_r$: Flow rate enter the link from entry r at step k (veh/h);

$w_r[k], r \in S_r$: Queue waiting on the entry r at step k (vehs);

$q_i^{in}[k]$: Upstream inflows of link i at step k (vehs);

$\gamma_{ij}[k],\ j \in \Gamma^{-1}(i)$: Relative turning proportion of movement from link *i* to *j*;

$N_i[k]$: Num. of vehicles at link *i* for at step *k* (vehs);

$v_i[k]$: Mean approaching speed of vehicles from upstream to the end of queue at link *i* at step *k* (mph);

$\rho_i[k]$: Density of the segment from upstream to the end of queue at link *i* at step *k* (veh/mile/lane);

$q_i^{arr}[k]$: Flows arriving at end of queue of link *i* at step *k* (vehs);

$s_i[k]$: Available space of link *i* at step *k* (vehs);

$x_i[k]$: Total num. of vehicles in queue at link *i* at step *k* (vehs);

$q_m^{i}[k]$: Flows join the queue of lane group *m* of link *i* at step *k* (vehs);

$x_m^{i}[k]$: Queue length of lane group *m* of link *i* at step *k* (vehs);

$\lambda_m^{ij}[k],\ j \in \Gamma^{-1}(i)$: Percentage of movement from link *i* to *j* in lane group *m*;

$Q_m^{i}[k]$: Flows depart from lane group *m* of link *i* at step *k* (vehs);

$Q_{ij}^{pot}[k]$: Flows potentially depart from link *i* to *j* i at step *k* (vehs);

$Q_{ij}[k]$: Flows actually depart from link *i* to *j* i at step *k* (vehs);

$g_n^p[k]$: Binary value indicating whether signal phase *p* of intersection *n* is set to green at step *k*.

μ^+,ν^+ : Index of the incident upstream on-ramp and off-ramp, respectively

μ^-,ν^- : Index of the incident downstream on-ramp and off-ramp, respectively

$\bar{\gamma}_{ij}[k],\ j\in\Gamma^{-1}(i)$: Relative turning proportion of normal arterial traffic from link i to j

$\gamma_{ij}^{\mu^-},\ j\in\Gamma^{-1}(i)$: A binary value indicating whether detour traffic at link i heading to downstream on-ramp μ^- will use downstream link j or not

$\overline{N}_i[k]$: Num. of vehicles from normal arterial traffic at link i at step k

$N_i^{\mu^-}[k]$: Num. of detour vehicles heading to downstream on-ramp μ^- at link i at step k

$\eta_i[k]$: Fraction of normal arterial traffic in total traffic at link i at step k

$\bar{\lambda}_m^{ij}[k],\ j\in\Gamma^{-1}(i)$: Percentage of normal arterial traffic in lane group m going from link i to j

$\overline{Q}_{ij}[k]$: Normal arterial traffic flows actually depart from link i to link j at step k

$Q_{ij}^{\mu^-}[k]$: Detour traffic flows heading to downstream on-ramp μ^- actually depart from link i to link j at step k

$\{C^h, h\in H\}$: Common cycle length for all intersections in the control interval h

$\{\Delta_n^h, \forall n\in S_N, h\in H\}$: Offset of intersection n for each control interval h

$\{G_{np}^h, \forall n\in S_N, p\in P_n, h\in H\}$: Green time for phase p of intersection n for each control interval h

$\{R_{\mu^+}^h, h\in H\}$: Metering rate at the incident upstream on-ramp μ^+ for each control interval h

$\{Z_{\nu^+}^h, h\in H\}$: Diversion rate at the incident upstream off-ramp ν^+ for each control interval h

The integrated control model aims to maximize the use of the corridor capacity to minimize congestion on the freeway mainline due to an incident with the following control objective:

$$\max \qquad \sum_{t=1}^{H} q_{i+1,0}[t]\cdot\Delta T + \sum_{k=1}^{H}\sum_{i\in S^{OUT}} q_i^{in}[k]$$

$$s.t.\quad s:[C^T, \Delta_n^T, G_{np}^T, Z_{\nu^+}^T]\in S \qquad (4\text{-}1)$$

where $q_{i+1,0}[t]$ is the flow rate entering the freeway link (i+1) downstream of the on-ramp μ^-; S^{OUT} is the set of outgoing links in the arterial network (see Figure 4.29); S denotes the feasible solution set defined by the following network flow and operational constraints:

1) Arterial Demand Entries

$$IN_r[k] = \min\left[D_r[k] + \frac{w_r[k]}{\Delta t}, Q_i, \frac{s_i[k]}{\Delta t}\right] \tag{4-2}$$

$$w_r[k+1] = w_r[k] + \Delta t[D_r[k] - IN_r[k]] \tag{4-3}$$

2) Arterial Upstream Arrivals

$$q_i^{in}[k] = \sum\nolimits_{j \in \Gamma(i)} \overline{Q}_{ji}[k] + \sum\nolimits_{j \in \Gamma(i)} Q_{ji}^{\mu^-}[k] \tag{4-4}$$

3) Arterial Joining Queue End

$$q_i^{arr}[k] = \min\left\{\rho_i[k] \cdot v_i[k] \cdot n_i \cdot \Delta t, \overline{N}_i[k] + N_i^{\mu^-}[k] - x_i[k]\right\} \tag{4-5}$$

4) Arterial Merging Into Lane Groups

$$q_m^i[k] = \min\{\max\{\ N_m^i - x_m^i[k], 0\}, \max\{q_m^{i,pot}[k] \cdot [1 - \sum_{m' \in S_i^M} \omega_{m'm}^i[k]], 0\}\} \tag{4-6}$$

5) Arterial Departing Process

$$Q_{ij}[k] = \min\left\{Q_{ij}^{pot}[k], \frac{Q_{ij}^{pot}[k]}{\sum_{i \in \Gamma(j)} Q_{ij}^{pot}[k]} \cdot s_j[k]\right\} \tag{4-7}$$

$$Q_{ij}^{pot}[k] = \sum_{m \in S_i^M} \min\left\{q_m^i[k] + x_m^i[k], Q_m^i \cdot g_n^p[k]\right\} \cdot \lambda_m^{ij}[k] \tag{4-8}$$

6) Arterial Flow Conservation

$$x_i[k+1] = \sum_{m \in S_i^M} \left(x_m^i[k+1] + \tilde{x}_m^i[k+1]\right) \tag{4-9}$$

$$\overline{N}_i[k+1] = \overline{N}_i[k] + \sum_{j \in \Gamma(i)} \overline{Q}_{ji}[k] - \sum_{j \in \Gamma^{-1}(i)} \overline{Q}_{ij}[k] \tag{4-10}$$

$$N_i^{\mu^-}[k+1] = N_i^{\mu^-}[k] + \sum_{j \in \Gamma(i)} Q_{ji}^{\mu^-}[k] - \sum_{j \in \Gamma^{-1}(i)} Q_{ij}^{\mu^-}[k] \tag{4-11}$$

$$\eta_i[k+1] = \frac{\overline{N}_i[k+1]}{\overline{N}_i[k+1] + N_i^{\mu^-}[k+1]} \tag{4-12}$$

$$s_i[k+1] = N_i - \overline{N}_i[k+1] - N_i^{\mu^-}[k+1] \tag{4-13}$$

7) Freeway Mainline Dynamics

$$\rho_{im}[t+1] = \rho_{im}[t] + \frac{\Delta T}{l_{im} n_{im}} (q_{i,m-1}[t] - q_{im}[t])$$ (4-

14)

$$q_{im}[t] = \rho_{im}[t] \cdot v_{im}[t] \cdot n_{im}$$ (4-

15)

$$V_{im}(\rho_{im}[t]) = v_f^i \exp\left[-\frac{1}{\alpha_f} \left(\frac{\rho_{im}[t]}{\rho_{cr}^i} \right)^{\alpha_f} \right]$$ (4-

16)

$$\mathrm{v}_{\mathrm{im}}[\mathrm{t}+1] = \mathrm{v}_{\mathrm{im}}[\mathrm{t}] + \frac{\mathrm{DT}}{\mathrm{t}}[\mathrm{V}(\mathrm{r}_{\mathrm{im}}[\mathrm{t}]) - \mathrm{v}_{\mathrm{im}}[\mathrm{t}]] + \frac{\mathrm{DT}}{\mathrm{l}_{\mathrm{im}}} \mathrm{v}_{\mathrm{im}}[\mathrm{t}][\mathrm{v}_{\mathrm{i,m-1}}[\mathrm{t}] - \mathrm{v}_{\mathrm{im}}[\mathrm{t}]] - \frac{\mathrm{h} \cdot \mathrm{DT}[\mathrm{r}_{\mathrm{i,m+1}}[\mathrm{t}] - \mathrm{r}_{\mathrm{im}}[\mathrm{t}]]}{\mathrm{t} \cdot \mathrm{l}_{\mathrm{im}}[\mathrm{r}_{\mathrm{im}}[\mathrm{t}] + \mathrm{k}]}$$ (4-

17)

8) On-off Ramps

$$\mathrm{Q}_{\bar{\mathrm{m}}}[\mathrm{t}] = \min\{\frac{\mathrm{x}^{\bar{\mathrm{m}}}[\mathrm{l} \cdot \mathrm{t}] + \sum_{\mathrm{k=lt}}^{\mathrm{l(t+1)-1}} \mathrm{q}_{\bar{\mathrm{m}}}^{\mathrm{arr}}[\mathrm{k}]}{\mathrm{DT}}, \mathrm{Q}_{\bar{\mathrm{m}}} \cdot \mathrm{R}_{\bar{\mathrm{m}}}, \mathrm{Q}_{\bar{\mathrm{m}}} \cdot \min[1, \frac{\mathrm{r}^{\mathrm{jam}} - \mathrm{r}_{\mathrm{i+1,0}}[\mathrm{t}]}{\mathrm{r}^{\mathrm{jam}} - \mathrm{r}_{\mathrm{i}}^{\mathrm{crit}}}]\} \qquad (4\text{-}18)$$

$$\mathrm{q}_{\mathrm{n^+}}^{\mathrm{in}}[\mathrm{t}] = \min\{\mathrm{r}_{\mathrm{i-1,N(i-1)}}[\mathrm{t}] \cdot \mathrm{v}_{\mathrm{i-1,N(i-1)}}[\mathrm{t}] \cdot \mathrm{n}_{\mathrm{i-1,N(i-1)}} \cdot (\mathrm{g}_{\mathrm{n^+}}^{\mathrm{T}} + \mathrm{b}_{\mathrm{n^+}}^{\mathrm{T}} \cdot \mathrm{Z}_{\mathrm{n^+}}^{\mathrm{T}}), \mathrm{Q}_{\mathrm{n^+}}, \frac{\mathrm{s}_{\mathrm{n^+}}[\mathrm{l} \cdot \mathrm{t}] + \sum_{\mathrm{k=lt}}^{\mathrm{l(t+1)-1}} \sum_{\mathrm{j \in G^{-1}(n^+)}} \mathrm{Q}_{\mathrm{n^+j}}[\mathrm{k}]}{\mathrm{DT}}\} \qquad (4\text{-}19)$$

9) Operational Constraints for Control Parameters

$$C^{\min} \le C^T \le C^{\max}, \forall T \in H \qquad (4\text{-}20)$$

$$G_{np}^{\min} \le G_{np}^T < C^T \quad , \forall n \in S_N, p \in P_n, T \in H \qquad (4\text{-}21)$$

$$\sum_{p \in P_n} G_{np}^T + \sum_{p \in P_n} I_{np} = C^T, \forall n \in S_N, p \in P_n, T \in H \qquad (4\text{-}22)$$

$$0 \le \Delta_n^T < C^T, \forall n \in S_N, T \in H \qquad (4\text{-}23)$$

$$\beta_{v^+}^T \cdot Z_{v^+}^T + \gamma_{v^+}^T \le Z^{\max}, T \in H \qquad (4\text{-}24)$$

The arterial dynamics in the diversion optimization model consists of six modules: demand entries, upstream arrivals, joining the end of queue, merging into lane groups, departing process, and flow conservation (see Figure 4.29a). Eq. (4-2) updates the flow entering arterial link i from demand entry r at time step k. Eq. (4-3) calculates the queue waiting at the demand entry during each time step. The arrival flowing to link i at time step k can be formulated as the sum of actual departure flows from all upstream links, including both normal arterial traffic and detour traffic, given by Eq. (4-4). Eq. (4-5) models the evolution of upstream inflows to the end of the queue with the average approaching speed. Eq. (4-6) gives the number of vehicles that can actually merge into their destination lane group m at time step k, considering the potential queue blockage effects from other lane groups (e.g., a fully occupied through lane group may completely block the left-turn traffic). Eqs. (4-7) and (4-8) give the actual departing flows from link i to link j at time step k. The arrival and departure flows at link i should be subject to the flow conservation law, given by Eqs. (4-9) - (4-13).

Eqs. (4-14)-(4-17) capture the network flow dynamics on the freeway mainline (see Figure 4.29b). The key concept is to divide the freeway link into homogeneous segments and update the flow, density, and speed within each segment at every time interval (Messmer and Papageorgiou, 1995). As on-ramps and off-ramps function to exchange diversion flows between the freeway and arterial systems, Eqs. (4-18)-(4-19) are employed to model their interactions.

The integrated diversion control model aims to optimize the diversion rates and retime the signals along the detour route to accommodate the detour traffic. Eqs. (4-20)-(4-24) is the restriction for the control decision variables, including the cycle length (C^T), the offsets ($\Delta_{\mathrm{n}}^{\mathrm{T}}$), the green splits ($G_{np}^T$), diversion rates ($Z_{V^+}^T$).

A genetic algorithm (GA)-based heuristic integrated with a rolling horizon framework has been used to yield reliable model solutions. Note that the control model has been validated under various traffic conditions and incident scenarios, showing promising properties in freeway corridor incident management. More details about the formulations and solution algorithm of the diversion optimization model can be found in the work by Liu et al. (2011).

4.4.2 A Two-choice Alternative Route Decision Model

Based on each generated experimental scenario in the previous chapter and the optimal detour rate derived from previous sections that yield the maximum total corridor capacity for each scenario, the goal of this section is to determine how to decide whether an alternative route recommendation should be made or not. Before developing the two-choice detour decision model, a preliminary analysis is presented to explore the effects of potential factors on the optimal detour rate in a given scenario.

4.4.2.1 Preliminary Analysis

The preliminary analysis is to explore how factors in each scenario affect the corresponding optimal detour rate. To achieve this goal, a linear regression model is applied in which the independent variables are nine original factors and dependent variables are the optimal detour rate.

Table 4.2 Estimation Results for Linear Regression Model

Variables	Coefficient Estimation	Stand Error	P-value
Intercept	1.765	0.002	0.001
FR_VOL (250, 750, 1250, 1750, 2200)	-2.649	0.239	0.004
FR_LN (2, 3, 4)	6.982	11.300	0.006
INC_DUR (15, 30, 45,60, 75, 90,105, 120)	-3.238	0.963	0.002
LN_BLK (1, 2, 3, 4)	-0.831	1.245	0.003
LC_VOL1 (200, 300, 400, 500, 600, 700, 800)	0.239	16.897	1.230
LC_VOL2 (200, 300, 400, 500, 600, 700, 800)	0.802	2.900	0.003
LC_VOL3 (200, 300, 400, 500, 600, 700, 800)	0.644	20.456	2.098
LC_LN (1, 2, 3)	-6.230	18.908	1.560
NUM_SIGNAL (2, 3, 4, 5, 6, 7)	0.454	1.043	0.002
R Square		0.81	
Adjusted R Square		0.82	
Observation		400	

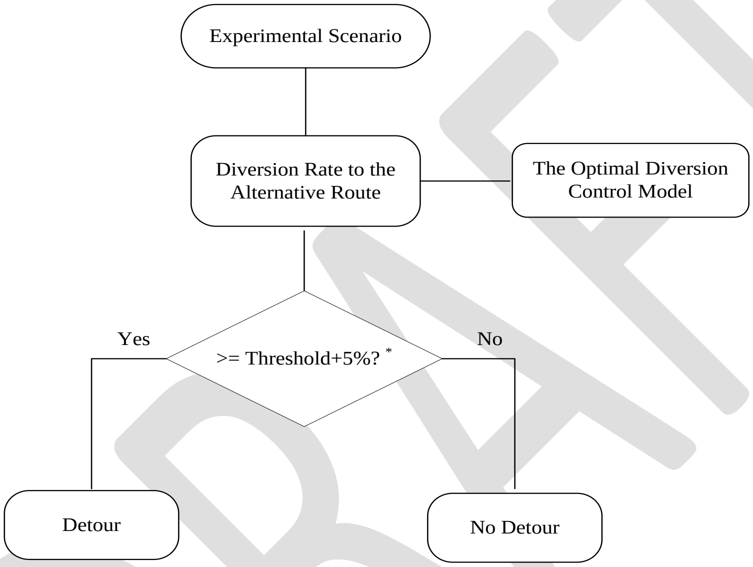
Table 4.2 shows the estimation results for the linear regression model. R square is 81%, which makes this model acceptable. Among nine independent variables, the flow rate on the freeway, the work zone duration, the number of lanes blocked, the flow rate on the alternative route and the number of signals on the alternative route per mile show significance. From the estimated coefficients for each significant variable, the following findings can be reached:

- The increase of flow rate on the freeway has a negative impact on the optimal diversion rate, which means it will get a lower optimal diversion rate when the flow rate on the freeway is higher;
- The work zone duration and the number of lanes blocked show a negative impact on the optimal diversion rate, which implies that travelers are suggested to detour to an alternate route in an early time when the work zone duration is long and too many lanes are blocked on the freeway; and
- The flow rate on the alternative route and the number signals on the alternative route have a positive impact on the optimal diversion rate, which shows that a higher optimal alternative route is derived when the flow rate is higher in the alternative route and there are more signals on the alternative route.

The above analysis can assist SWZDI agencies to figure out how these factors in a given scenario affect the final optimal diversion rate, i.e., what trend (higher or lower) could the optimal detour rate be at a certain incident situation. However, this information cannot help agencies to make a final decision because of the continuity of an optimal diversion rate and the lack of an exact criterion to implement an alternative route decision. In real-time incident/work zone management, agencies prefer to make a decision according to a binary decision variable, i.e., “yes” or “no.” This requirement boosts the selection of a criterion to separate the continuous optimal diversion rate to make a final decision.

4.4.2.2 Concept of Two-choice Alternative Route Decision Model

Based on the aforementioned preliminary analysis, it is necessary to set a minimum threshold value for the diversion rate on the alternate route to convert the decimal diversion rate into a binary decision, which is also the main task in the two-choice detour decision model. Figure 4.30 illustrates the procedure to make the alternative route decision for each experimental scenario, which will be used for the two-choice decision model development. The author assumes that an incident/work zone scenario would warrant a detour operation if its optimal flow distribution state demands more than the summation of this threshold and a normal diversion rate of 5% to divert to the alternative route.


Experimental Scenario
Diversion Rate to the Alternative Route
The Optimal Diversion Control Model
Yes
>= Threshold+5%? *
No
Detour
No Detour

* The threshold is to be decided

Figure 4.30 The Procedure to Determine the Alternative Route Decision

Since the alternative route decision is binary in nature, this study adopts a logistic regression, a commonly used methodology to study a binary dependent variable. The following parts will briefly present the principle of binary logistic regression and detail its development and validation in this study.

The output of a linear regression can be transformed to an appropriate probability using a logit link function as follows:

$$\log it \ \ p = \log \ \ o = \frac{\log p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k \qquad (4\text{-}25)$$

where p is a probability to succeed, and o is the odds representing the ratio of p to 1-p.

Since the odds (o) can be any value in (0, ∞), the log odds (log o) can vary in (-∞, ∞). This value represents what we get from the linear regression on the right hand side of

Eq. (4-25). The inverse of the logit function is the logistic function, thus logit (p) = z can be transformed to:

$$p = \frac{e^z}{1+e^z} \qquad (4\text{-}26)$$

Then, the logistic function maps any value of the right-hand side in Eq. (4-26) to a proportional value in (0, 1). The parameters included in the model (βi) can be estimated with the maximum likelihood method (Allison, 2001). The aforementioned theory implies that a unit additive change in the value of the variable changes the odds by a constant multiplicative amount. More detailed discussion regarding logistic models can be found in many references (Ben-Akiva and Lerman, 1985; Venables and Ripley, 2002; Washington et al., 2003).

4.4.2.3 Model Development

The dependent variables are a series of binary variables indicating whether an alternative route decision should be made or not (1 represents "yes", 0 represents "no"). Note that the minimum threshold has not been set yet. This study will select one from the set (5%, 10%, 15%, 20%, 25%, 30% and 35%) with the principle of providing the greatest performance of the binary logistic regression model. Diversion rates smaller than 5% and greater than 35% are not selected into a threshold set since when the diversion rate is smaller than 5%, the incident/work zone impact is considered trivial, no alternative route recommendation needs to be implemented. When the diversion rate is greater than 35%, obviously, an alternative route plan needs to be implemented in such situations.

Model Development with Original Groups of Variables

This section first applied the original groups of independent variables and their values from Table 4.1 in the previous chapter. Table 4.3 shows the estimation results when the minimum threshold is set as 5%. Among nine independent variables, only incident duration is demonstrated to be significant. Moreover, the predicated model accuracy is only 49.3%, which should be determined to be unacceptable. Therefore, it is necessary to re-group the independent variables to better develop the binary logistic regression model.

Since the overall prediction accuracy is relatively low, it fails to select the optimal minimum threshold. This requires further analysis to get the optimal minimum threshold.

Considering the aforementioned model requirement, the following part will present a preliminary analysis to re-group the independent variables and select the optimal minimum threshold.

Table 4.3 Calibrated Logistic Decision-Model with the Minimum Threshold of 5%

Variables included in the final model	Estimate	Exp(estimate)	Std. Error	z value	p-value
(Intercept)	-2.34500	0.2514	12.54390	-8.54	0.01
FR_VOL (250, 750, 1250, 1750, 2200)	0.45021	0.9738	56.00234	-9.62	1.51
FR_LN (2, 3, 4)	1.78294	3.5678	15.89535	5.08	0.60
INC_DUR (15, 30, 45,60, 75, 90,105, 120)	0.11725	0.7728	0.10723	-2.74	0.04
LN_BLK (1, 2, 3, 4)	-6.72811	1.6958	10.53119	9.02	1.74
LC_VOL1 (200, 300, 400, 500, 600, 700, 800)	0.00036	1.0004	20.00018	6.99	5.05
LC_VOL2 (200, 300, 400, 500, 600, 700, 800)	0.53490	1.8635	58.22140	10.33	7.02
LC_VOL3 (200, 300, 400, 500, 600, 700, 800)	-5.57560	1.8985	23.89450	7.34	2.78
LC_LN (1, 2, 3)	7.50390	4.8565	58.22140	10.33	7.02
NUM_SIGNAL (2, 3, 4, 5, 6, 7)	4.69900	2.9680	13.31660	2.98	0.13
The number of observations used for calibration			400		
Likelihood with constants only			-507.93		
Final value of Likelihood			-1161.605		
Fitted model accuracy			0.520		
Predicted model accuracy			0.493		

Preliminary Analysis for Binary Logistic Regression Model

The goal of this section is to re-categorize the independent variables and select the optimal minimum threshold for the development of binary logistic regression. Classification and Regression Tree (CART) has the ability to organize by variables and identify patterns in the data (Smith and Smith, 2001) that was chosen as a tool for preliminary analysis in this study.

The original independent variables were used as inputs for the building tree. The dependent variable is the same with the binary logistic regression model. Each threshold was used to build a tree. Thus, in total, seven trees were developed for the preliminary analysis. The results show that the significant independent variables are incident duration (INC_DUR), which is categorized into durations under 45 minutes and above 45 minutes, the number of signals on alternative (NUM_SIGNAL), which is categorized into numbers under two and above two, and the volume of the roadway connecting from the freeway to the detour route (LC_VOL1), which is categorized into volume under 600 vphpl and above 600 vphpl. Other variables, such as the number of lanes blocked, the freeway volume for each lane, the number of freeway lanes, the volume on the detour route, and the number of local lanes, were still not significant. This facilitates the combination of the volume of each lane and the number of lanes to model development. Also, this study will try the percentage of capacity drop instead of the number of lanes blocked to analyze its impact on detour decisions.

Table 4.4 summarizes the overall prediction accuracy for each developed tree under different minimum thresholds. From the table, it is obvious that Tree 2 has the highest prediction accuracy of 75.9% in which 10% was set as the minimum threshold. This study will select 10% as the final optimal minimum threshold to develop the binary logistic regression model.

Table 4.4 The Overall Prediction Accuracy of Each Tree

Tree Number	1	2	3	4	5	6	7
Minimum Threshold	5%	10%	15%	20%	25%	30%	35%
Prediction Accuracy	55.1%	75.9%	57.6%	72.4%	65.4%	69.5%	63.8%

Calibration with Re-grouped Variables

With the contribution of preliminary analysis, the final binary logistic regression model used the re-grouped independent variables and a minimum threshold of 10% to calibrate. Table 4.5 summarizes specifications of the model, which demonstrates about 76 percent and 73 percent accuracies for model estimation set and validation set, respectively. The accuracy is determined by whether or not the optimal traffic distribution during the incident management period needs more than twenty percent (additional normal detour volume of five percent) of its total volumes to the local street. In addition, all variables included in the model are significant at a 95 percent confidence level, which also confirms the necessity of re-grouping independent variables.

Table 4.5 Calibrated Logistic Decision-Model

Variables included in the final model	Estimate	Exp (Estimate)	Std. Error	z value	p-value
(Intercept)	-1.38300	0.2508	0.54490	-2.64	0.01
IF(INC_DUR>45) TRUE[1]	0.00725	0.9928	0.00383	-2.34	0.03
IF(NUM_SIGNAL <= 2)TRUE[2]	0.67700	1.9680	0.31220	2.18	0.02
IF(LC_VOL1 < 600)TRUE[3]	0.51490	1.6735	0.22540	2.33	0.01
PER_CAP_DROP	3.42800	1.5958	0.59110	7.02	0.01
LC_VOL2*LC_LN	0.00036	1.0004	0.10018	1.99	0.05
FR_VOL*FR_LN	0.00021	0.9998	0.00304	-4.62	0.04
The number of observations used for calibration			400		
Likelihood with constants only			-317.93		
Final value of Likelihood			-361.605		
Fitted model accuracy			0.765		
Predicted model accuracy			0.733		
The number of observations used for validation			100		

<Note> [1] IF(INC_DUR >45 2)TRUE: 1 if INC_DUR<= 45 ; 0 otherwise

[2] IF (NUM_SIGNAL <= 2) TRUE: 1 if NUM_SIGNAL<= 2; 0 otherwise

[3] IF (LC_VOL1 < 600) TRUE: 1 if LC_VOL1 < 600; 0 otherwise

From the aforementioned findings, it can be concluded that the incident duration alone should not be a sole criterion to decide the need to issue the alternative route recommendation.

Table 4.6 details the re-calibrated logistic model with interaction terms, including INC_DUR: FR_VOL (0.00002/p-value=0.000) and INC_DUR: PER_CAP_DROP (0.05154/p-value=0.000). Although these two interaction terms are not included in the final logistic regression model due to their multicollinearity, the information still can be derived regarding how they interact with each other. It can be observed that both interaction terms are related to incident duration, which confirms its significance again.

Table 4.6 Re-calibrated Logistic Decision Models with Excluded Interaction Terms

Variables included in the final model	Estimate	Exp (estimate)	Std. Error	z value	p-value
(Intercept)	2.29900	9.9642	0.472	4.869	0.000
IF(INC_DUR>45)TRUE	-0.06469	0.9374	0.008	-7.692	0.000
IF(NUM_SIGNAL <= 2)TRUE	0.71610	2.0464	0.316	2.269	0.023
IF(LC_VOL1 < 600)TRUE	0.54460	1.7239	0.227	2.404	0.016
LC_VOL2*LC_LN	0.00043	1.0004	0.000	2.337	0.019
FR_VOL*FR_LN	-0.00047	0.9995	0.000	-5.921	0.000
INC_DUR:FR_VOL	0.00002	1.0000	0.000	4.219	0.000
INC_DUR: PER_CAP_DROP	0.05154	1.0529	0.008	6.766	0.000
The number of observations used for calibration			400		
Likelihood with constants only			-307.93		
Final value of Likelihood			-250.42		
Fitted model accuracy			0.774		
Predicted model accuracy			0.773		

4.4.3 The Final Alternative Route Decision Model

To determine the alternative route decision, one needs to estimate the probability of being a "yes" for a decision regarding a given scenario. Using Eq. (4-27) and the estimated coefficients in Table 4.6, it is possible to estimate u, e^u, and p. If p >= 0.5, the system shall issue the alternative route recommendation to travelers.

$$p = \frac{e^u}{1+e^u} \tag{4-27}$$

where variable u is a measure of the total contribution of all affecting variables used in the model (listed in Table 4.6), and u = -1.383 + 0.00725*IF(INC_DUR>45)TRUE + 0.677*IF(NUM_SIGNAL<=2)TRUE + 0.5149*IF(LC_VOL1<600)TRUE + 3.728*PER_CAP_DROP + 0.00036* LC_VOL2*LC_LN + 0.00021* FR_VOL*FR_LN.

The decision model developed in this chapter will be integrated into the system and used to determine whether or not to issue the alternative route recommendation to travelers.

CHAPTER 5: SYSTEM DEVELOPMENT

This chapter details the development of the smartphone-based prototype system. The system functions to retrieve critical incident/work zone information from the 511.xml and 511LCS.xml files, present information to travelers using an App, make decisions on whether or not to issue alternative recommendation, and re-calculate the best alternative route. To fulfill such functions, there are three main modules in system development: (1) XML-doc retrieving module; (2) SQL database server; and (3) the App.

5.1 XML-doc Retrieving Module

The main goal of this software module is to retrieve information from the 511 web service, parse the XML doc and store it in our local 511 data base. This module is implemented using Visual C# (2010) as a Windows Console application which means that there is no graphical user interface (GUI) for this software module and system administrator needs to run it from command prompt or double click on it in its windows folder. This module is scheduled to run by Windows Task Scheduler every hour. We use VC# because of its ease to communicate with MSSQL database server. The important functions of this module are outlined in the following section.

5.1.1 Functions

All of the settings, definitions and processes are defined in the function main. First the SQL-related variables and objects are declared such as the SQL table name “Route” and the three objects for communicating with the database, specifically, the “daroute” instance from “Route table adapter”, the “dtroute” instance from “Route data table” and the “drroute” instance from “Route row”. These three objects are needed in order to access and modify the database tables in the SQL Server (see the following code snippet).

```
dsrouteTableAdapters.RouteTableAdapter daroute = new
dsrouteTableAdapters.RouteTableAdapter();
dsroute.RouteDataTable dtroute = new dsroute.RouteDataTable();
daroute.Fill(dtroute);
daroute.Update(dtroute);
dsroute.RouteRow drroute = default(dsroute.RouteRow);
```

The universal resource identifier (URI) for the 511 XML document is stored in the variables “remoteUri” and fileName. A Web Client object is instantiated to download the data file from the URI (see the following code snippet).

```
WebClient myWebClient = new WebClisent();
myStringWebResource = remoteUri + fileName;
myWebClient.DownloadFile(myStringWebResource,fileName);
```

As the file is downloaded from the 511 server is compressed in a “gzip” format, decompressing the file is the next step.

```
xmlfilename = Decompress(fileToDecompress);
```

The decompressed file has a XML format so the objects of “XmlDocument” and “XmlNodeList” are defined in order to navigate the XML doc and retrieve the different elements of each node in the XML doc.

```
XmlDocument m_xmld = default(XmlDocument);
XmlNodeList m_nodelist = default(XmlNodeList);
```

In the XML doc the nodes for primary and secondary, latitude and longitude are captured for all incidents and stored in the database, in addition, the starting time (date and time) and the valid period of each incident are stored in the database as well. The nodes for primary and secondary, latitude and longitude are stored in a hierarchical structure as below:

element-locations-> element-location -> location-on-link -> geo-location

Similarly, the nodes for the starting time and the valid period are stored in a hierarchical structure as below:

element-times -> start-time and valid-period -> expected-end-time.

Finally each record is updated in the database table.

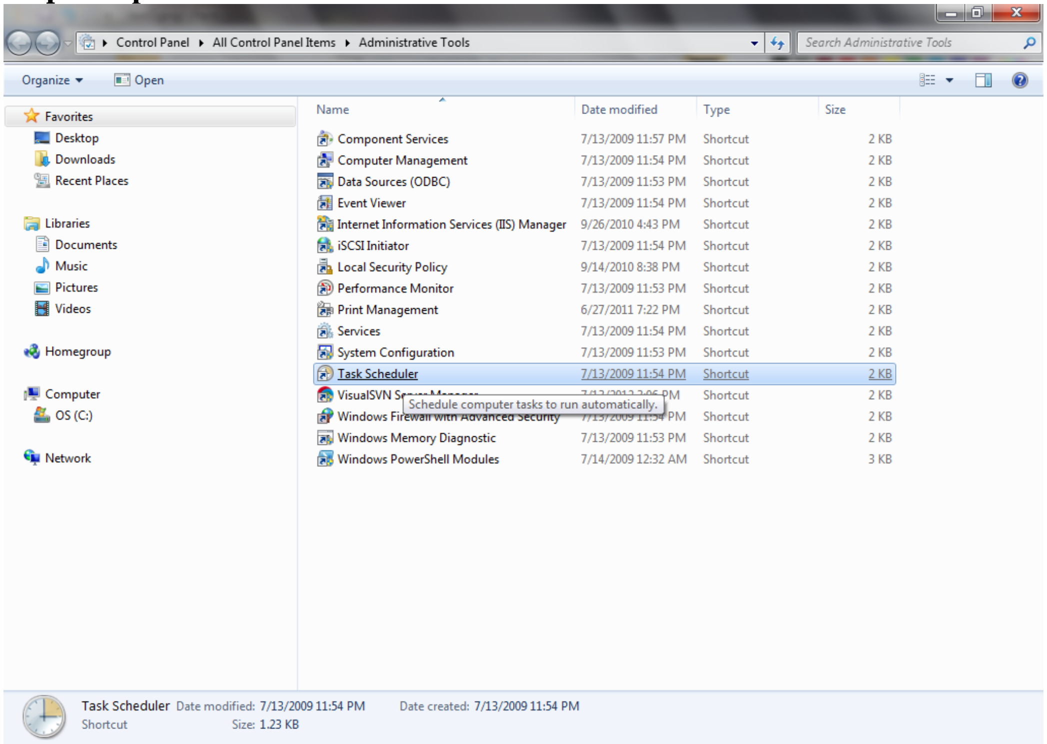
```
dtroute.AddRouteRow(drroute);
daroute.Update(dtroute);
```

The file “read511.exe” is the generated execute file.

5.1.2 Task Scheduler

In order to update the database, the software module “read511.exe” is scheduled as a task to run once every hour. The following steps are needed to setup the task scheduler:

Step 1: Open “Task Scheduler” from Windows Administrative Tools

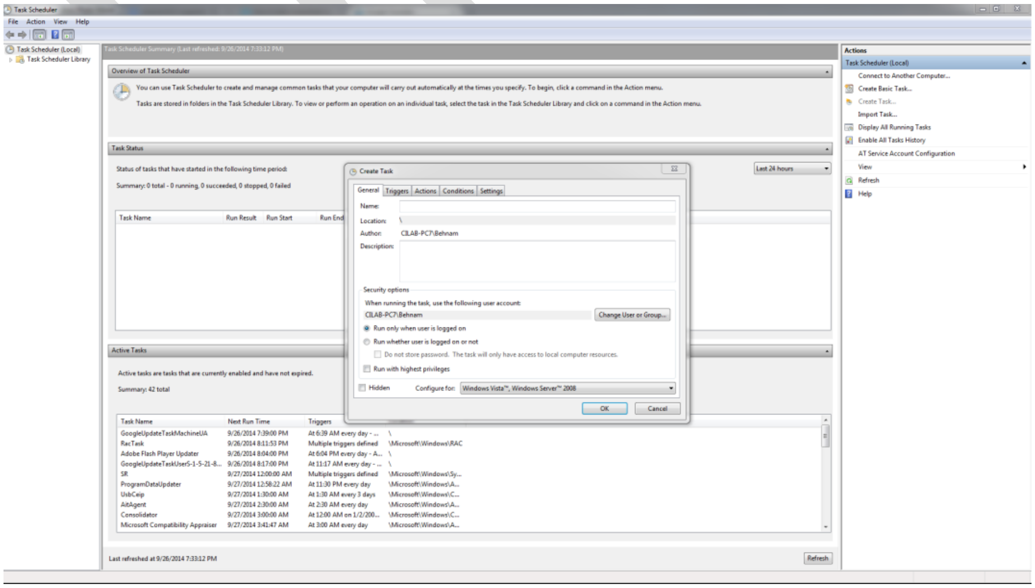


Name	Date modified	Type	Size
Component Services	7/13/2009 11:57 PM	Shortcut	2 KB
Computer Management	7/13/2009 11:54 PM	Shortcut	2 KB
Data Sources (ODBC)	7/13/2009 11:53 PM	Shortcut	2 KB
Event Viewer	7/13/2009 11:54 PM	Shortcut	2 KB
Internet Information Services (IIS) Manager	9/26/2010 4:43 PM	Shortcut	2 KB
iSCSI Initiator	7/13/2009 11:54 PM	Shortcut	2 KB
Local Security Policy	9/14/2010 8:38 PM	Shortcut	2 KB
Performance Monitor	7/13/2009 11:53 PM	Shortcut	2 KB
Print Management	6/27/2011 7:22 PM	Shortcut	2 KB
Services	7/13/2009 11:54 PM	Shortcut	2 KB
System Configuration	7/13/2009 11:53 PM	Shortcut	2 KB
Task Scheduler	7/13/2009 11:54 PM	Shortcut	2 KB
VisualSVN Server Manager	7/13/2012 2:06 PM	Shortcut	2 KB
Windows Firewall with Advanced Security	7/13/2009 11:54 PM	Shortcut	2 KB
Windows Memory Diagnostic	7/13/2009 11:53 PM	Shortcut	2 KB
Windows PowerShell Modules	7/14/2009 12:32 AM	Shortcut	3 KB

Schedule computer tasks to run automatically.

Task Scheduler Date modified: 7/13/2009 11:54 PM Date created: 7/13/2009 11:54 PM
Shortcut Size: 1.23 KB

Step 2: Create a task by clicking on “Create Task …” in Actions list from the right side of Task Scheduler window or from Action menu.



Task Scheduler Summary (Last refreshed: 9/26/2014 7:33:12 PM)

You can use Task Scheduler to create and manage common tasks that your computer will carry out automatically at the times you specify. To begin, click a command in the Action menu.

Tasks are stored in folders in the Task Scheduler Library. To view or perform an operation on an individual task, select the task in the Task Scheduler Library and click on a command in the Action menu.

Status of tasks that have started in the following time period:

Summary: 0 total - 0 running, 0 succeeded, 0 stopped, 0 failed

Task Name	Run Result	Run Start	Run End

Create Task

General | Triggers | Actions | Conditions | Settings

Name:
Location: \
Author: CILAB-PC7\Behnam
Description:

Security options
When running the task, use the following user account:
CILAB-PC7\Behnam [Change User or Group...]
● Run only when user is logged on
○ Run whether user is logged on or not
□ Do not store password. The task will only have access to local computer resources.
□ Run with highest privileges
□ Hidden Configure for: Windows Vista™, Windows Server™ 2008
[OK] [Cancel]

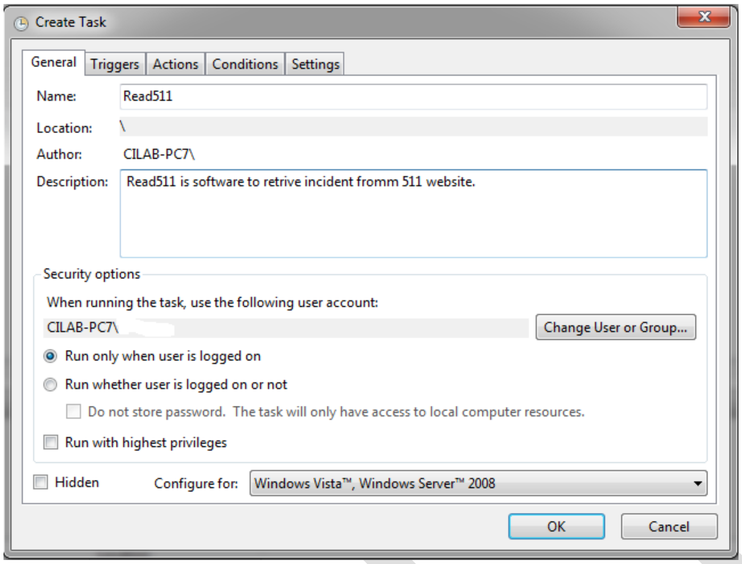
Active tasks are tasks that are currently enabled and have not expired.

Summary: 42 total

Task Name	Next Run Time	Triggers	
GoogleUpdateTaskMachineUA	9/26/2014 7:39:00 PM	At 6:39 AM every day - ...	\
RacTask	9/26/2014 8:11:53 PM	Multiple triggers defined	\Microsoft\Windows\RAC
Adobe Flash Player Updater	9/26/2014 8:04:00 PM	At 6:04 PM every day - A...	\
GoogleUpdateTaskUserS-1-5-21-8...	9/26/2014 8:17:00 PM	At 11:17 AM every day - ...	\
SR	9/27/2014 12:00:00 AM	Multiple triggers defined	\Microsoft\Windows\Sy...
ProgramDataUpdater	9/27/2014 12:58:22 AM	At 11:30 PM every day	\Microsoft\Windows\A...
UsbCeip	9/27/2014 1:30:00 AM	At 1:30 AM every 3 days	\Microsoft\Windows\C...
AitAgent	9/27/2014 2:30:00 AM	At 2:30 AM every day	\Microsoft\Windows\A...
Consolidator	9/27/2014 3:00:00 AM	At 12:00 AM on 1/2/200...	\Microsoft\Windows\C...
Microsoft Compatibility Appraiser	9/27/2014 3:41:47 AM	At 3:00 AM every day	\Microsoft\Windows\A...

Last refreshed at 9/26/2014 7:33:12 PM

Step 3: Fill up the General Information; in this tab Name and Description are defined for the task:



Name: Read511

Location: \

Author: CILAB-PC7\

Description: Read511 is software to retrive incident fromm 511 website.

Security options

When running the task, use the following user account:

CILAB-PC7\

● Run only when user is logged on

○ Run whether user is logged on or not

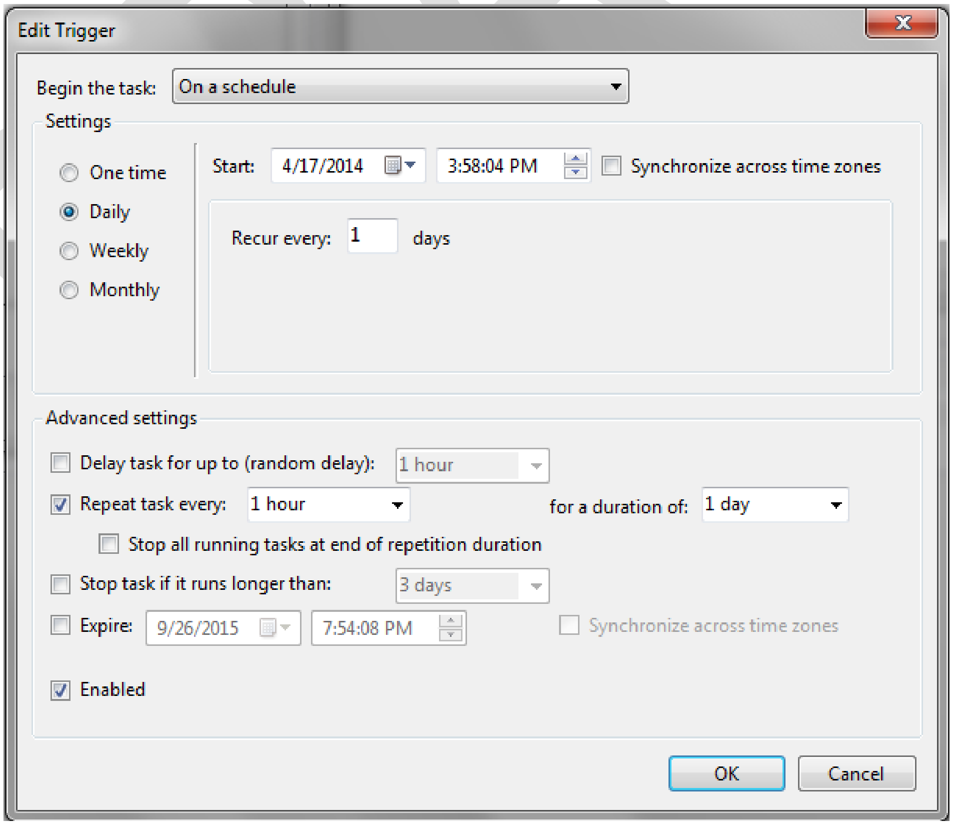
□ Do not store password. The task will only have access to local computer resources.

□ Run with highest privileges

□ Hidden Configure for: Windows Vista™, Windows Server™ 2008

Step 4: Add a new trigger in the trigger tab by clicking on the "new" button. In the New Trigger window make sure the setting is set as below then click on OK:

- Begin the task: On a schedule
- Settings: Daily and Recur every 1 day
- Advanced settings: Repeat task every 1 hour for a duration of 1 day
- Enabled: checked



Begin the task: On a schedule

Settings

○ One time

● Daily

○ Weekly

○ Monthly

Start: 4/17/2014 3:58:04 PM □ Synchronize across time zones

Recur every: 1 days

Advanced settings

□ Delay task for up to (random delay): 1 hour

☑ Repeat task every: 1 hour for a duration of: 1 day

□ Stop all running tasks at end of repetition duration

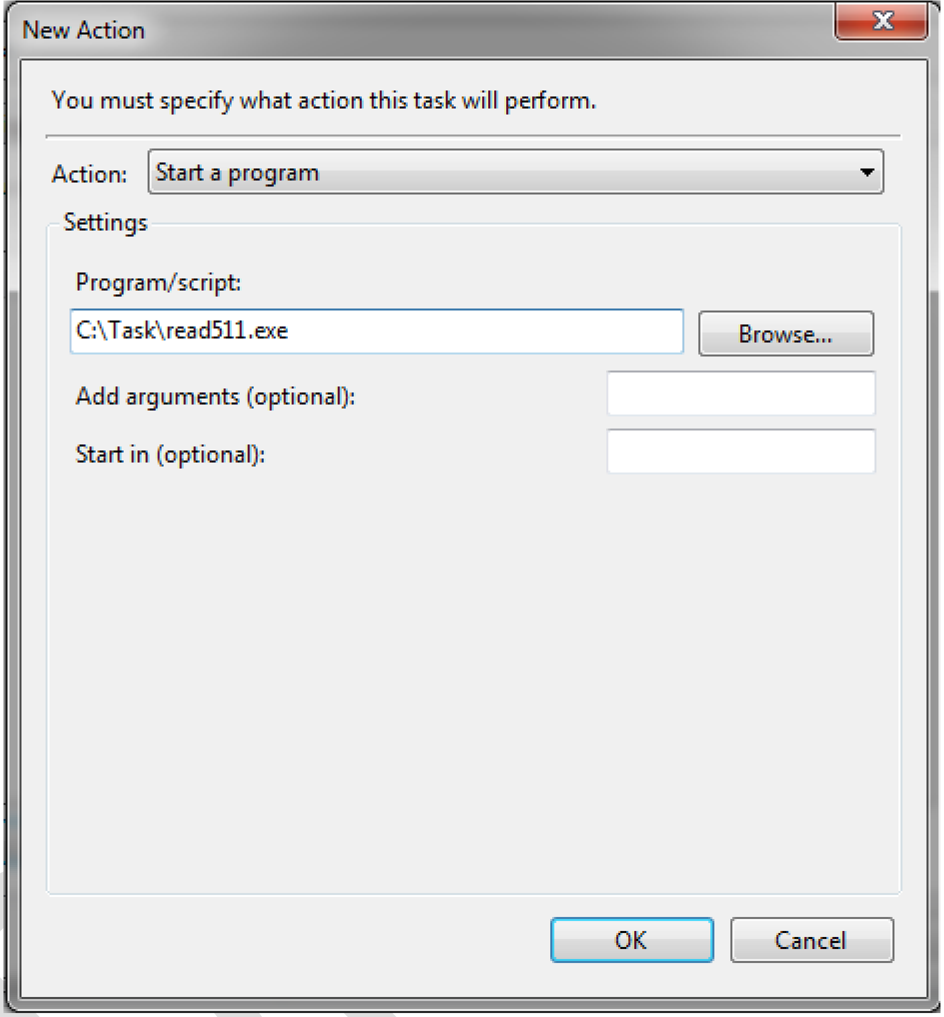
□ Stop task if it runs longer than: 3 days

□ Expire: 9/26/2015 7:54:08 PM □ Synchronize across time zones

☑ Enabled

Step 5: Define the name and the address of the executable file we need to schedule in the New Action tab. Click on "New..." button to add new action.

In the "action" dropdown menu, select "Start a program" and browse to the "Read511.exe" module location in the Program/script textbox.



New Action

You must specify what action this task will perform.

Action: Start a program

Settings

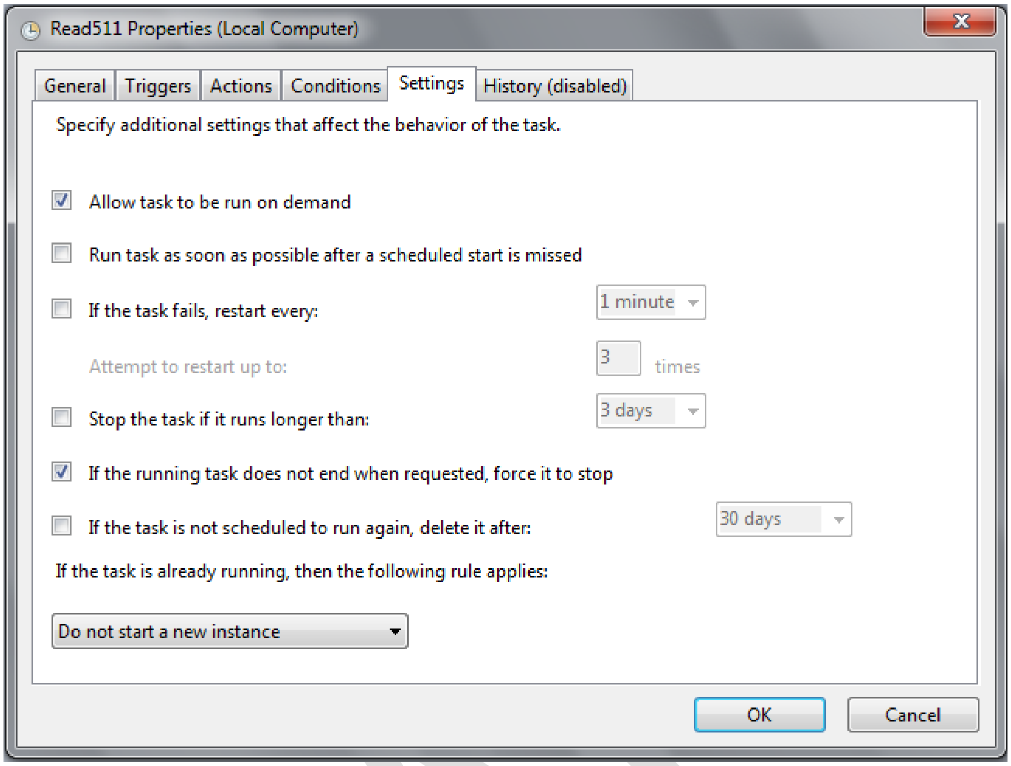
Program/script: C:\Task\read511.exe

Add arguments (optional):

Start in (optional):

Step 6: Check the items as below and make sure the other items left uncheck in the Settings tab.

- Allow task to be run on demand
- If the running task does not end when requested, force it to stop.



Specify additional settings that affect the behavior of the task.

☑ Allow task to be run on demand

□ Run task as soon as possible after a scheduled start is missed

□ If the task fails, restart every: 1 minute

Attempt to restart up to: 3 times

□ Stop the task if it runs longer than: 3 days

☑ If the running task does not end when requested, force it to stop

□ If the task is not scheduled to run again, delete it after: 30 days

If the task is already running, then the following rule applies:

Do not start a new instance

And click on OK to finish creating the task.

5.2 The SQL Database

The information retrieved from the 511 website by the software module "read511.exe" is stored in the SQL Server database. In this database a table named “Route” is defined with a total of nine fields. Each row in the database table corresponds to a unique road event (or incident).

The data types and the related description for each table field are listed below:

Data Types:

1- Routeid
2- Starttimedate
3- Starttimetime
4- Expectedendtimedate
5- Expectedendtimttime
6- Primarylatitude
7- Primarylongitude
8- Secondarylatitude
9- Secondarylongitude

Data Fields:

Field Name:	Routeid
Field type:	uniqueidentifier
Allow Nulls:	No
Default value:	Not define
Description:	Unique id for each row to make find or update easy.
Sample Data	72f97181-b0a6-43b3-96df-01dbd0efd902

Field Name:	Starttimedate
Field type:	Date
Allow Nulls:	Yes
Default value:	Not define
Description:	Starting date for all kind of insident
Sample Data	2014-10-03
Field Name:	Starttimetime
Field type:	Time
Allow Nulls:	Yes
Default value:	Not define
Description:	Starting time for incident
Sample Data	14:00:00

Field Name:	Expectedendtimedate
Field type:	Date
Allow Nulls:	Yes
Default value:	Not define
Description:	Expected end date for incident such as construction
Sample Data	2014-10-03

Field Name:	expectedendtimetime
Field type:	Time
Allow Nulls:	Yes
Default value:	Not define
Description:	Expected end time for incident
Sample Data	15:00:00

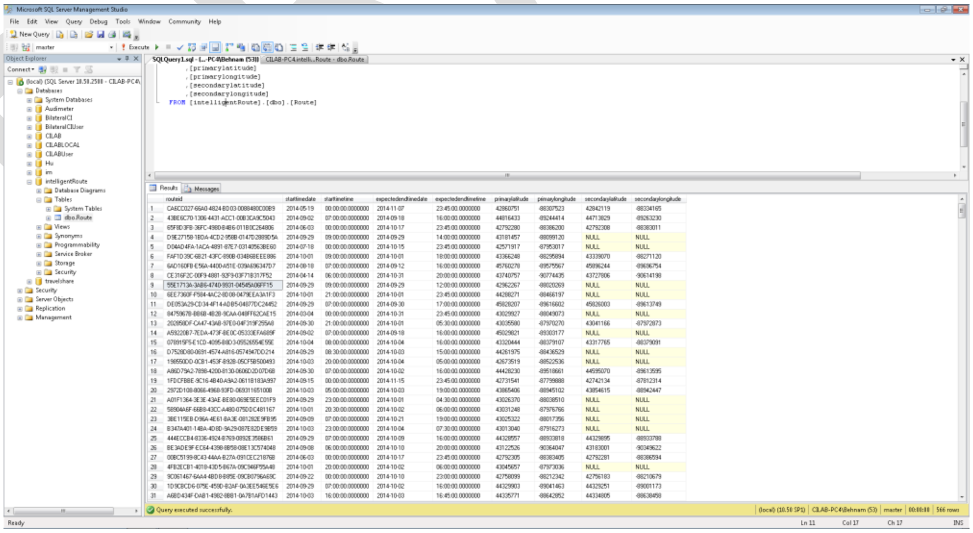
Field Name:	primarylatitude
Field type:	Integer
Allow Nulls:	Yes
Default value:	Not define
Description:	All incidents have a primary location, this filed keep latitude of that location
Sample Data	44500774

Field Name:	primarylongitude
Field type:	Integer

Allow Nulls:	Yes
Default value:	Not define
Description:	Longitude of primary location of incident
Sample Data	-88330622

Field Name:	Secondarylatitude
Field type:	Integer
Allow Nulls:	Yes
Default value:	Not define
Description:	Latitude of Secondary location depend of incident, some of incidents don’t need secondary location in that case value is NULL.
Sample Data	44515132
Field Name:	secondarylongitude
Field type:	Integer
Allow Nulls:	Yes
Default value:	Not define
Description:	Longitude of Secondary location depend of incident, some of incidents don’t need secondary location in that case value is NULL.
Sample Data	-88329956

The database construction and sample are shown in the following figures:

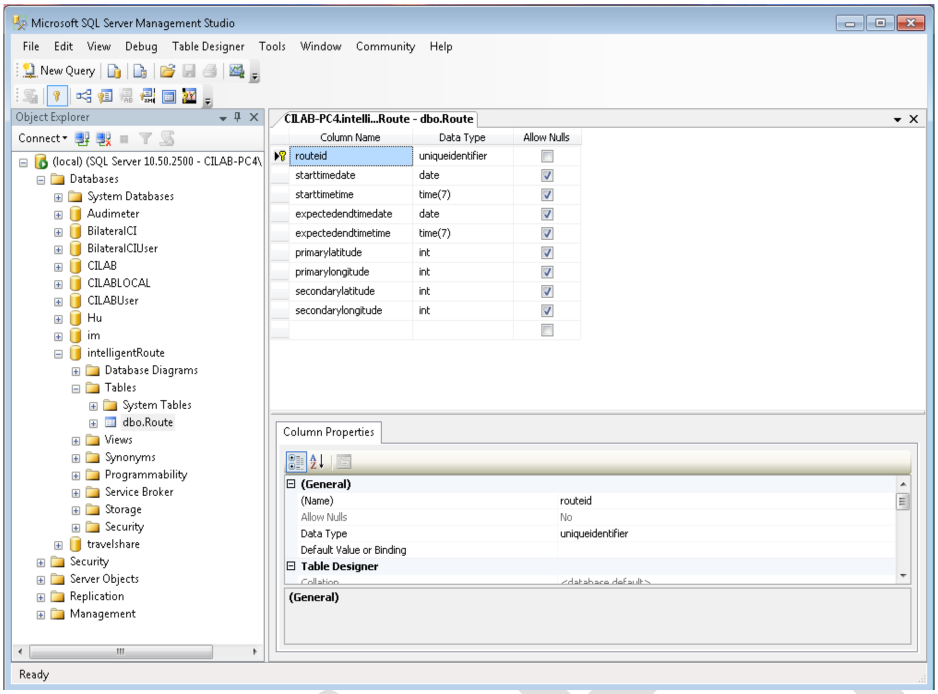


```
      ,[primarylatitude]
      ,[primarylongitude]
      ,[secondarylatitude]
      ,[secondarylongitude]
  FROM [intelligentRoute].[dbo].[Route]
```

	routeid	starttimedate	starttimetime	expectedendtimedate	expectedendtimetime	primarylatitude	primarylongitude	secondarylatitude	secondarylongitude
1	CA6CD027-66A0-4824-BD03-0088480C00B9	2014-05-19	00:00:00.0000000	2014-11-07	23:45:00.0000000	42860751	-88307523	42842119	-88334165
2	43BE6C70-1306-4431-ACC1-00B3CA9C5043	2014-09-02	07:00:00.0000000	2014-09-18	16:00:00.0000000	44816433	-89244414	44713829	-89263230
3	65F8D3FB-36FC-4980-B4B6-011B0C264806	2014-06-03	00:00:00.0000000	2014-10-17	23:45:00.0000000	42792280	-88386200	42792308	-88383011
4	D9E27158-1BDA-4CD2-958B-0147D2889D5A	2014-09-29	09:00:00.0000000	2014-09-29	14:00:00.0000000	43181457	-88099120	NULL	NULL
5	D04AD4FA-1ACA-4891-87E7-03140563BE60	2014-07-18	00:00:00.0000000	2014-10-15	23:45:00.0000000	42571917	-87953017	NULL	NULL
6	FAF1D39C-6B21-43FC-890B-034B6BEEE886	2014-10-01	09:00:00.0000000	2014-10-01	18:00:00.0000000	43366248	-88295894	43339070	-88271120
7	6AD160FB-E56A-4400-A51E-039A696347D7	2014-08-18	07:00:00.0000000	2014-09-12	16:00:00.0000000	45760278	-89575567	45896244	-89696754
8	CE316F2C-00F9-4881-92F9-03F71B317F52	2014-04-14	06:00:00.0000000	2014-10-31	20:00:00.0000000	43740757	-90774435	43727806	-90614198
9	55E1713A-3AB6-4740-9931-04545A06FF15	2014-09-29	09:00:00.0000000	2014-09-29	12:00:00.0000000	42962267	-88020269	NULL	NULL
10	6EE7360F-F584-4AC2-8D08-0479EEA3A1F3	2014-10-01	21:00:00.0000000	2014-10-01	23:45:00.0000000	44288271	-88466197	NULL	NULL
11	DE053A29-CD34-4F14-ADB5-04877DC24452	2014-09-29	07:00:00.0000000	2014-09-30	17:00:00.0000000	45828207	-89616602	45826003	-89613749
12	84759678-B86B-4B28-9CAA-048FF62CAE15	2014-03-04	00:00:00.0000000	2014-10-31	23:45:00.0000000	43029927	-88049073	NULL	NULL
13	202858DF-CA47-43A8-97E0-04F319F255A8	2014-09-30	21:00:00.0000000	2014-10-01	05:30:00.0000000	43035580	-87970270	43041166	-87972873
14	A59220B7-7EDA-473F-8E0C-05333EFA689F	2014-09-02	07:00:00.0000000	2014-09-18	16:00:00.0000000	45029821	-89303177	NULL	NULL
15	078915F5-E1CD-4095-88D3-05526554E55E	2014-10-04	08:00:00.0000000	2014-10-04	16:00:00.0000000	43320444	-88379107	43317765	-88379091
16	D7528D80-0691-4574-A816-0574947DD214	2014-09-29	08:30:00.0000000	2014-10-03	15:00:00.0000000	44261975	-88436529	NULL	NULL
17	198550DD-0CB1-453F-8928-05CF5B500493	2014-10-03	20:00:00.0000000	2014-10-04	05:00:00.0000000	42673519	-88522536	NULL	NULL
18	A86D79A2-7898-4200-8130-0606D2D07D6B	2014-09-30	07:00:00.0000000	2014-10-02	16:00:00.0000000	44428230	-89518661	44595070	-89613595
19	1FDCFB8E-9C16-4B40-A9A2-06118183A997	2014-09-15	00:00:00.0000000	2014-11-15	23:45:00.0000000	42731541	-87799888	42742134	-87812314
20	2972D108-8066-4968-93FD-06931165100B	2014-10-03	05:00:00.0000000	2014-10-03	19:00:00.0000000	43865406	-88945102	43854615	-88942447
21	A01F1364-3E3E-43AE-BE80-069E5EEC01F9	2014-09-29	23:00:00.0000000	2014-10-01	04:30:00.0000000	43026370	-88038510	NULL	NULL
22	58904A6F-66B8-43CC-A480-075DDC481167	2014-10-01	20:30:00.0000000	2014-10-02	06:00:00.0000000	43031248	-87976766	NULL	NULL
23	3BE115EB-D96A-4E61-BA3E-081282E9FB95	2014-09-09	07:00:00.0000000	2014-10-21	19:00:00.0000000	43025322	-88017356	NULL	NULL
24	B347A401-14BA-4D8D-9A29-087E82DE9B59	2014-10-03	23:00:00.0000000	2014-10-04	07:30:00.0000000	43013040	-87916273	NULL	NULL
25	444ECCB4-8336-4924-B769-0892E3586B61	2014-09-29	07:00:00.0000000	2014-10-09	16:00:00.0000000	44328557	-88933818	44329895	-88933788
26	BE3ADE9F-EC64-4398-8B58-08E13C574048	2014-09-08	06:00:00.0000000	2014-10-10	20:00:00.0000000	43122526	-90364047	43183001	-90349622
27	00BC5199-8C43-44AA-B27A-091CEC21876B	2014-06-03	00:00:00.0000000	2014-10-17	23:45:00.0000000	42792305	-88383405	42792281	-88386594
28	4FB2ECB1-4018-43D5-B67A-09C946F55A48	2014-10-01	20:00:00.0000000	2014-10-02	06:00:00.0000000	43045657	-87973036	NULL	NULL
29	9C061467-6AA4-4BD8-B85E-09CB0796A69C	2014-09-22	00:00:00.0000000	2014-10-10	23:00:00.0000000	42758099	-88212342	42756183	-88210679
30	1D9CBCD6-075E-459D-B2AF-0A3EE546E5E6	2014-09-29	07:00:00.0000000	2014-10-02	16:00:00.0000000	44329903	-89041463	44329251	-89001173
31	A6BD434F-DAB1-4982-8BB1-0A7B1AFD1443	2014-10-03	16:00:00.0000000	2014-10-03	16:45:00.0000000	44335771	-88642852	44334805	-88638458

Query executed successfully.

Figure 5.1 Database Construction



Column Name	Data Type	Allow Nulls
routeid	uniqueidentifier	☐
starttimedate	date	☑
starttimetime	time(7)	☑
expectedendtimedate	date	☑
expectedendtimetime	time(7)	☑
primarylatitude	int	☑
primarylongitude	int	☑
secondarylatitude	int	☑
secondarylongitude	int	☑
		☐

(General)

(Name)	routeid
Allow Nulls	No
Data Type	uniqueidentifier
Default Value or Binding	

Table Designer

Collation	<database default>

(General)

Figure 5.2 Data Sample in Table

5.3 The Web Service Module

The protocols and the functionalities related to the web service and the web server are described in this section. The main goal of the web service is to collect geographical information from android devices, and then the incident records corresponding to that location are retrieved from the database and sent back to Android device, as shown in Figure 5.3.

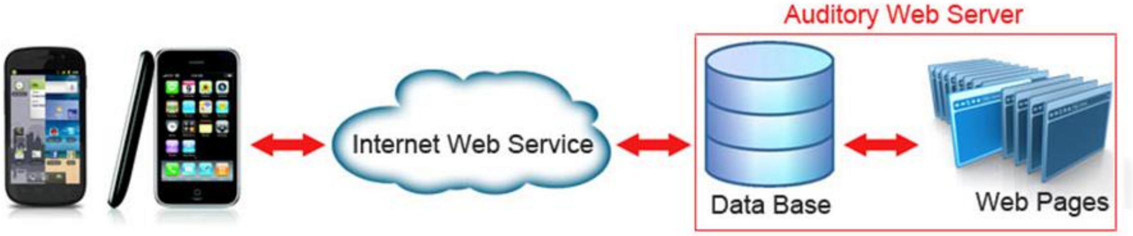


Figure 5.3 The Internet web service

5.3.1 The Internet Web Service

A web service provides communication capability between two electronic devices over the internet. The W3C defines a "web service" as "a software system designed to support interoperable machine-to-machine interaction over a network". It has an interface described in a machine-processable format (specifically Web Services Description Language, known by the acronym WSDL). Other systems interact with the web service in a manner prescribed by its description using Simple Object Access Protocol (SOAP) messages, typically conveyed using Hypertext Transfer Protocol (HTTP) with an Extensible Markup Language (XML) serialization in conjunction with other Web-related standards. The connection protocol is illustrated in Figure 5.4.

Figure 5.4 The Connection Protocol

SOAP is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks. "Large Web services" use XML messages that follow the SOAP standard and have been popular with traditional enterprises. In such systems, there is often a machine-readable description of the operations offered by the service written in the Web Services Description Language (WSDL). The latter is not a requirement of a SOAP endpoint, but it is a prerequisite for automated client-side code generation in many Java and .NET SOAP frameworks (frameworks such as Apache Axis2, Apache CXF, and Spring being notable exceptions). Some industry organizations, such as the WS-I, mandate both SOAP and WSDL in their definition of a Web service.

In this project we used visual studio .Net to create the web service and implement the protocol and structures and used Internet Information Services (IIS) for hosting the service.

After the web service receives the request from android devices that provides the source and destination geo coordinates, it will create a SQL query string to find all road events between these two locations, and basically it checks the square area defined by the source and the destination geo coordinates of the traveler's routes for road events in the database, given as follows:

"primarylatitude < '" & lat1 & "' and primarylatitude > '" & lat2 & "' and
primarylongitude < '" & lng1 & "' and primarylongitude > '" & lng2 & "'",
"[starttimedate] DESC"

After the SQL query string is ready, the retrieval function (dtroute.Select) executes the SQL query statement to obtain the results from database and return the results to android devices. The protocol used for transferring service requests and returned results between Android devices and the web server is GET.

5.3.2 Functions

There are two main functions in the web service, one is for the code running on server to which the client has no access, and the other is for the code of user interface notation, given by:

```
Public Function getinfo(ByVal lat1 As String, ByVal lng1 As String, ByVal lat2
As String, ByVal lng2 As String) As String Implements Iinfofinder.getinfo

<OperationContract()>
   <WebInvoke(Method:="GET", _
   ResponseFormat:=WebMessageFormat.Json, _
   BodyStyle:=WebMessageBodyStyle.Wrapped, _
   UriTemplate:="getinfo/{lat1}/{lng1}/{lat2}/{lng2}")>
```

5.4 The Android App

In Android App a user interface was implanted to obtain from travelers the source and destination addresses that are used to provide travelers with routing options, traffic and incident information retrieved from the web server. In this project we used eclipse IDE and Android SDK to design the Android App. The target Android version is 4.2.2 (API Level 17) and "google-play-services" library is used to access the Google map service 2.0 and its API. A snapshot of the App interface is shown in Figure 5.5.

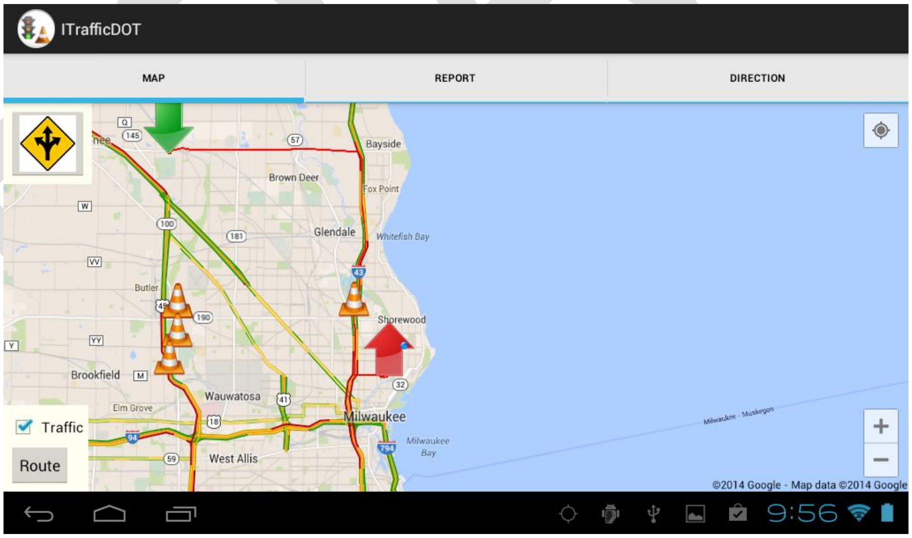


Figure 5.5 Snapshot of the App Interface

This app requires several permissions for security purposes:

```
<uses-permission
        android:name="com.example.googlemaps.permission.MAPS_RECEIVE" />
<uses-permission
        android:name="android.permission.INTERNET" />
<uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission
        android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
<uses-permission
        android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission
        android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission
        android:name="android.permission.ACCESS_NETWORK_STATE" />
```

The first permission gives the Android App access to Google map and its APIs, and the next one gives the App access to Internet in order to obtain the map information and communicate with the web server. The third permission gives the App access to the external memory card so if the phone has an external SD card the App can access the SD card. The forth permission is for the GService, which is part of Globe service and currently it is mandatory if the APP is intended to upload in Google play. The permission “Access_Coarse_Location” allows an app to access approximate location derived from network location sources such as cell towers and Wi-Fi. The permission “Access_Fine_location” allows an App to access precise location from location sources such as GPS, cell towers, and Wi-Fi and finally the permission “Access_Network_State” allows applications to access information about networks.

Also we have three different layouts for main view, Map view and addresses input UI (see Figure 5.6). These layout designs in XML format are located under “res->layout” folder of the Android project.

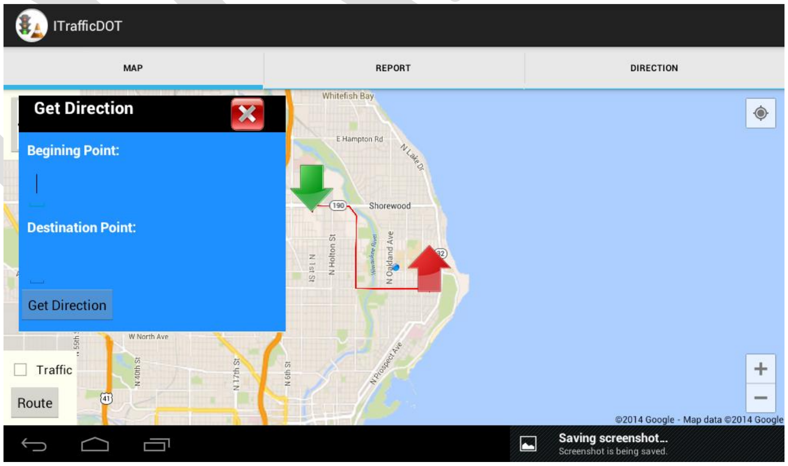


Figure 5.6 Layout of Main View

There are two main classes and these classes are stored under “src” folder of the Android project.

The first class is the “Main.java” class that contains different tabs and the main map view. The second class is the “Map.java” class, and in its onCreate method we define all map settings, setup user interface objects such as “Spinner”, “Check Box” and “Button”, and we also determine the location of device by using “Location Service” (see the following code snippet).

```
    LocationManager locationManager = (LocationManager) getSystemService
(LOCATION_SERVICE);
    Location lastLoc =
locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
    map.animateCamera(CameraUpdateFactory.newLatLngZoom(new LatLng(
    lastLoc.getLatitude(), lastLoc.getLongitude()), 15));
```

Several important functions are listed below:

This function is used to convert physical addresses to Geo coordinates (Latitude and Longitude):

```
    public LatLng AddresstoPoint(String strAddress);
```

This function updates current location when it changes:

```
    public void onLocationChanged(Location location);
```

This function gets the location of the source address and the destination address and returns the direction:

```
    private String getDirectionsUrl(LatLng origin, LatLng dest);
```

This function is used to download the road events/incidents from the web service:

```
    private class DownloadTask extends AsyncTask<String, Void, String>;
```

This function is called by “DownloadTask” to retrieve data:

```
    private String downloadUrl(String strUrl) throws IOException {}
```

5.5 The App showcase

The interface built on the Android platform is shown in the following figures and source codes are presented in the Appendix A.

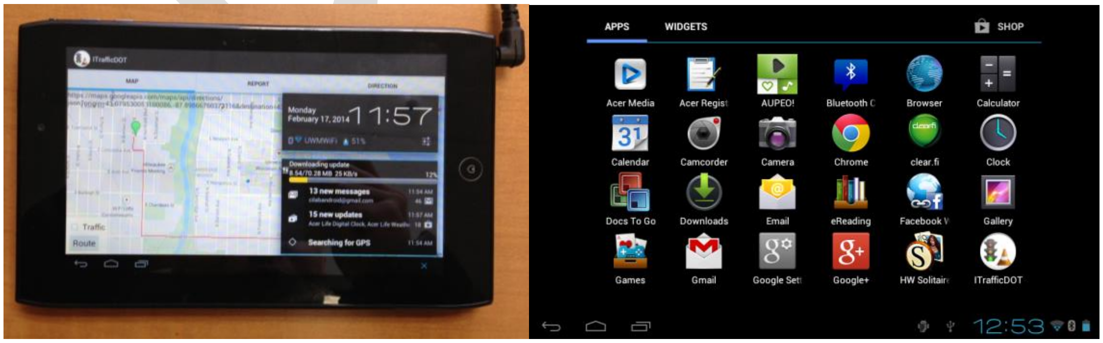


Figure 5.7 The App developed in Android platform

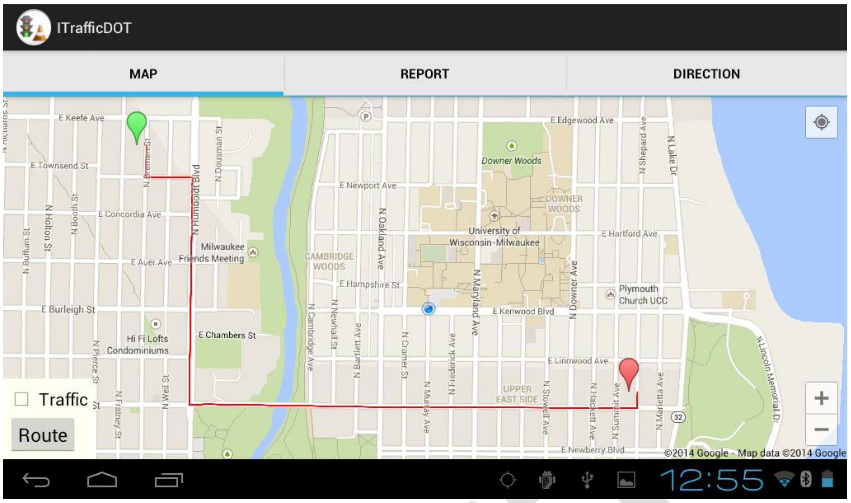


Figure 5.8 The route planner in the developed App

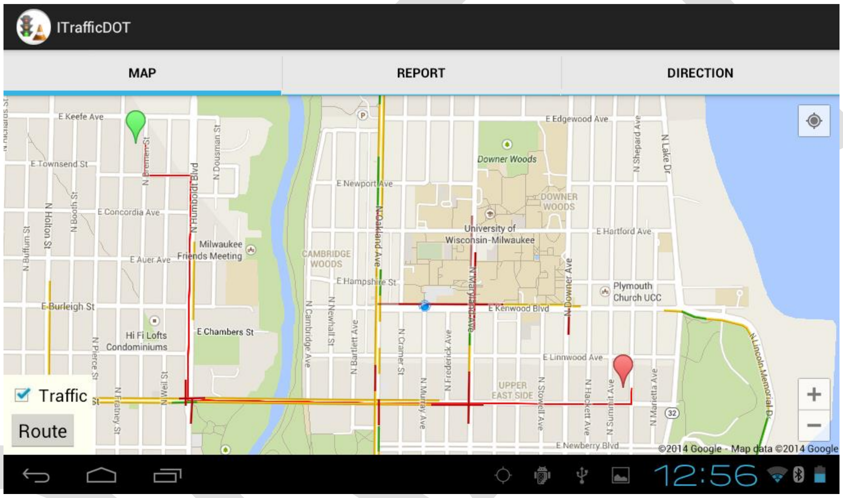


Figure 5.9 The traffic condition visualizer in the developed App

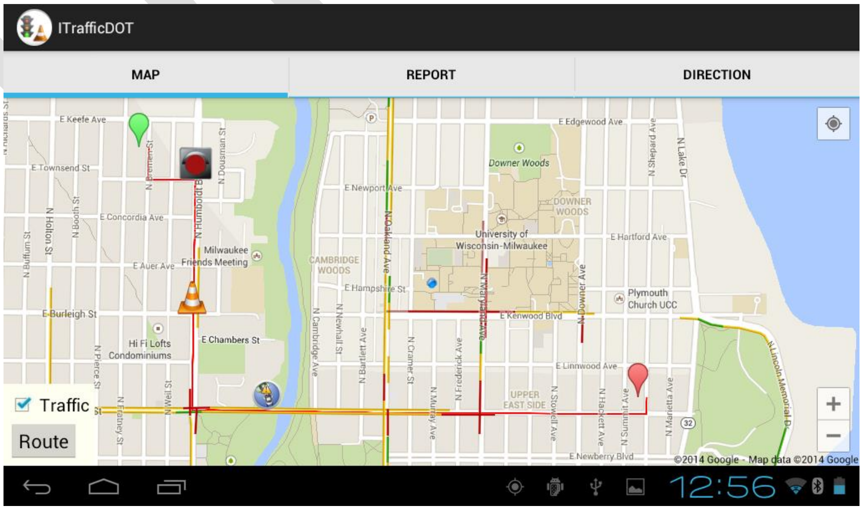


Figure 5.10 Warning messages of incident events customized to the selected route

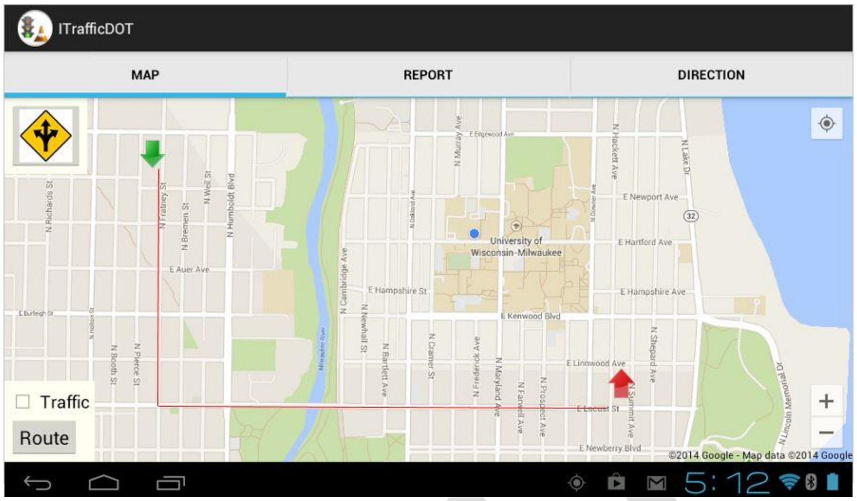

ITrafficDOT
MAP
REPORT
DIRECTION
University of Wisconsin-Milwaukee
□ Traffic
Route
©2014 Google - Map data ©2014 Google
5:12

Figure 5.11 Alternative route guidance

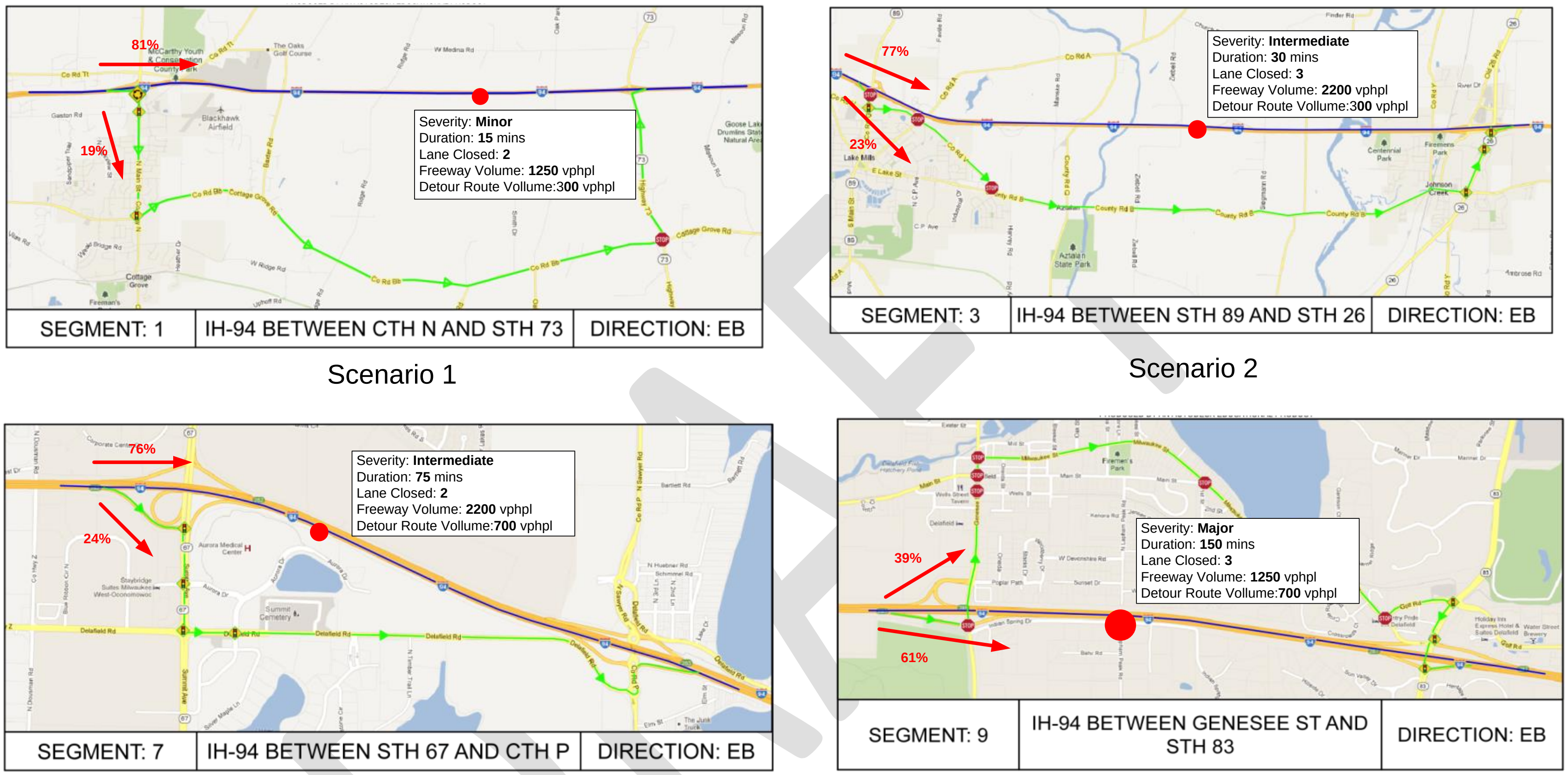
CHAPTER 6: FIELD TESTS AND BENEFIT ANALYSES

This chapter will present the field test results of the developed App with respect to its applicability in real time. In addition, since the primary goal of providing real-time incident and routing information to travelers through the developed App is to mitigate the congestion and the resulting delay due to an unexpected lane closure, from the perspective of responsible agencies, one needs to consider the resulting benefits of the alternative route guidance. This chapter briefly illustrates how to estimate the potential benefits resulting from the alternative route guidance. This benefit analysis can be a way to validate the developed alternative route guidance decision model, since it shows us whether the implemented alternative route guidance is truly beneficial or not, from the overall system perspective.

6.1 Scenario Selection

To illustrate how benefits from alternative route guidance would vary depending on different traffic conditions and incident severities, this study selected four different scenarios that resulted in the decision to implement alternative route guidance based on the developed decision model in Chapter 4. Figure 6.1 illustrates certain attributes of these four scenarios, which are located in segments 1, 3, 7 and 9, independently, along the I94 east-west corridor. Note that the segments presented here are consistent with those in section 4.2. The main flow rate and diversion flow rate derived from the integrated diversion control model are shown for each scenario in this figure. Table 6.1 summarizes the outputs for the four scenarios with the developed detour decision models. The output for all scenarios is “Yes”, which means that they call for alternative route guidance using the App.

Corresponding to each scenario, we will update the system data feeds and the database to evaluate how fast the developed App will communicate with the server to retrieve the information and broadcast to a traveler. Then, benefit analyses for each scenario will be performed.


81%
19%
Severity: Minor
Duration: 15 mins
Lane Closed: 2
Freeway Volume: 1250 vphpl
Detour Route Vollume:300 vphpl
SEGMENT: 1
IH-94 BETWEEN CTH N AND STH 73
DIRECTION: EB
77%
23%
Severity: Intermediate
Duration: 30 mins
Lane Closed: 3
Freeway Volume: 2200 vphpl
Detour Route Vollume:300 vphpl
SEGMENT: 3
IH-94 BETWEEN STH 89 AND STH 26
DIRECTION: EB
76%
24%
Severity: Intermediate
Duration: 75 mins
Lane Closed: 2
Freeway Volume: 2200 vphpl
Detour Route Vollume:700 vphpl
SEGMENT: 7
IH-94 BETWEEN STH 67 AND CTH P
DIRECTION: EB
39%
61%
Severity: Major
Duration: 150 mins
Lane Closed: 3
Freeway Volume: 1250 vphpl
Detour Route Vollume:700 vphpl
SEGMENT: 9
IH-94 BETWEEN GENESEE ST AND STH 83
DIRECTION: EB

Scenario 1

Scenario 2

Scenario 3

Scenario 4

Figure 6.1 Selected Scenarios for Field Tests and Benefit Analyses

Table 6.1 Alternative Route Decision for the Selected Scenarios

Scenario	The Alternative Route Decision Model
1	Yes
2	Yes
3	Yes
4	Yes

6.2 Field Tests

The field test is conducted by four drivers from the research team driving separately along the I-94 from Madison to Milwaukee and we simulated the virtual construction zones and accident scenarios along the route and evaluate how the system and the App would respond to the events and provide real-time routing information to the drivers. The test results show that there are no real-time communication delay between the App and the server when the incident is detected in the system, and the traveler is able to receive the system-warranted alternative route recommendation in real time in all scenarios. The resulting benefits from the alternative routing will also be analyzed in section 6.3.

Scenario I: An event of construction is detected at I94 between CTH N and STH73

When the driver is approaching the upstream interchange, a warning message is going to display on the App along with a recommendation of the alternative route.

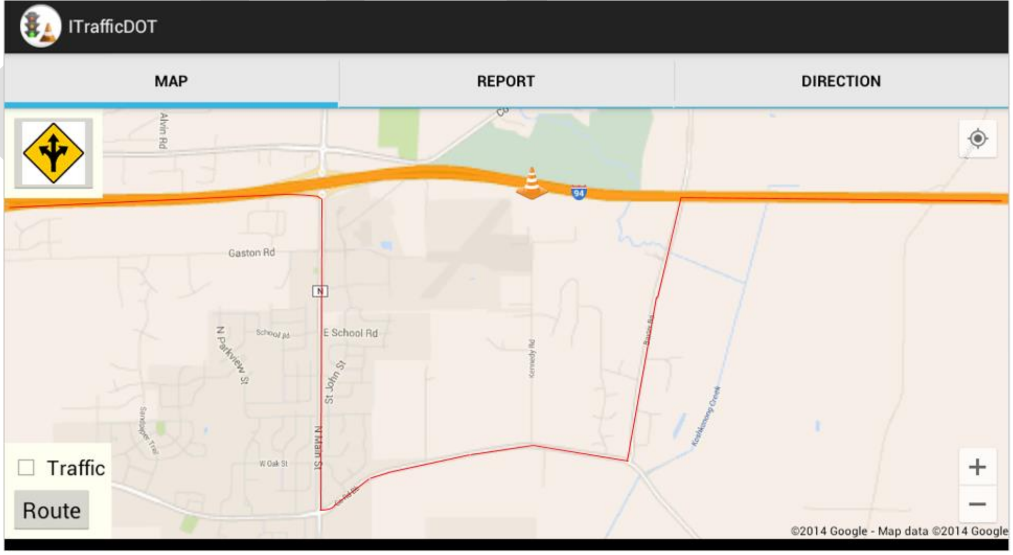


Figure 6.2 Scenario I

Scenario II: An event of accident is detected at I94 between STH 89 and STH 26

When a driver just passes Lake Mills, another accident event pops up in the App. Consequently, a warning message and its corresponding re-routing advice were sent to the driver.

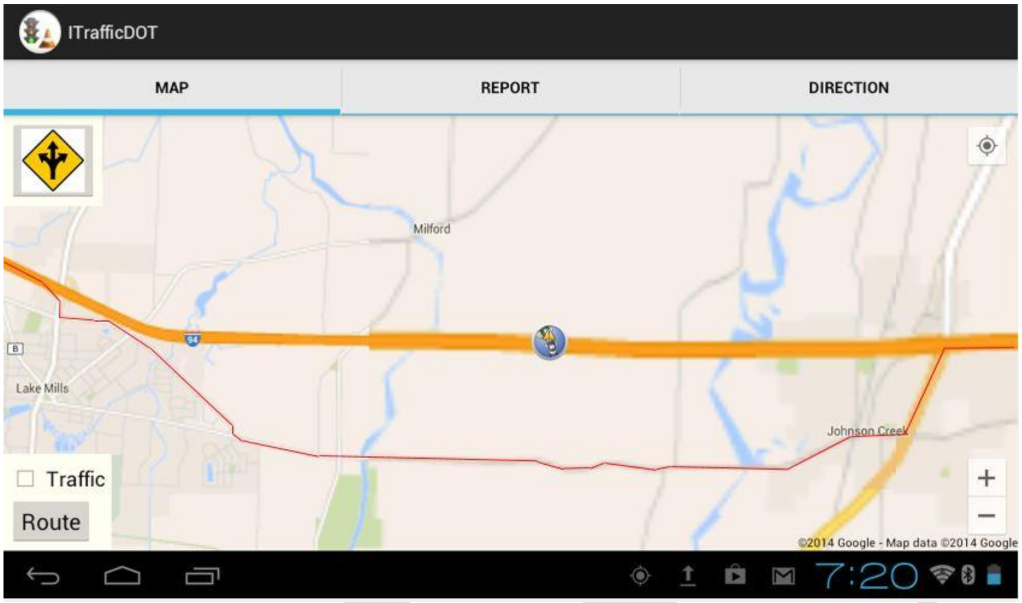


Figure 6.3 Scenario II

Scenario III: An event of construction is detected at I94 between STH67 and CTH P

A new construction work zone is identifying between STH67 and CTH P, and then the App suggests an alternative route to make the driver avoid encountering the congestion.

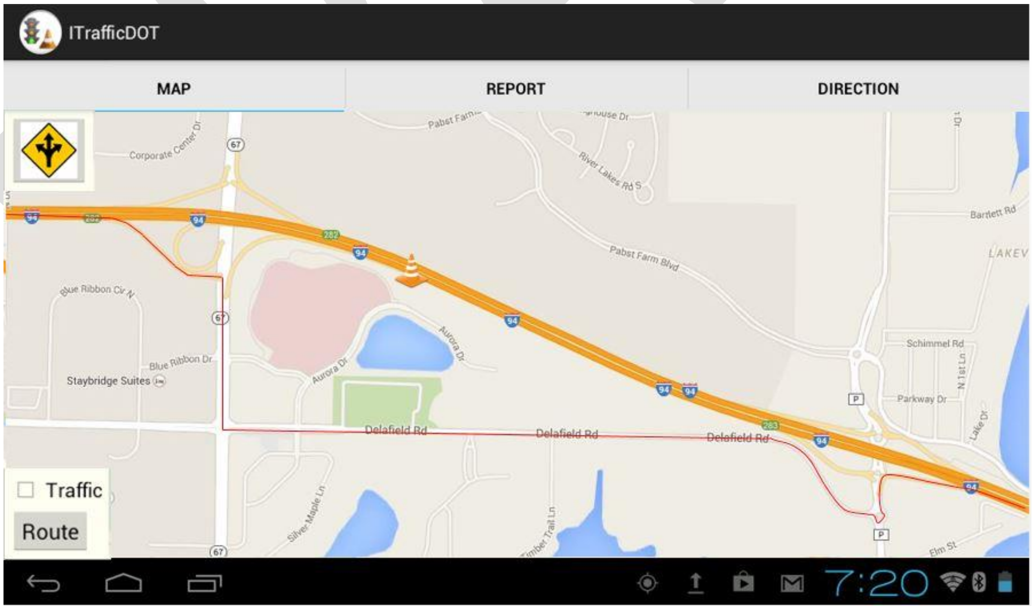


Figure 6.4 Scenario III

Scenario IV: An event of construction is detected at I94 between STH67 and CTH P

A new accident is identifying between STH67 and CTH P, and then the system suggests an alternative route as shown in Figure 6.5.

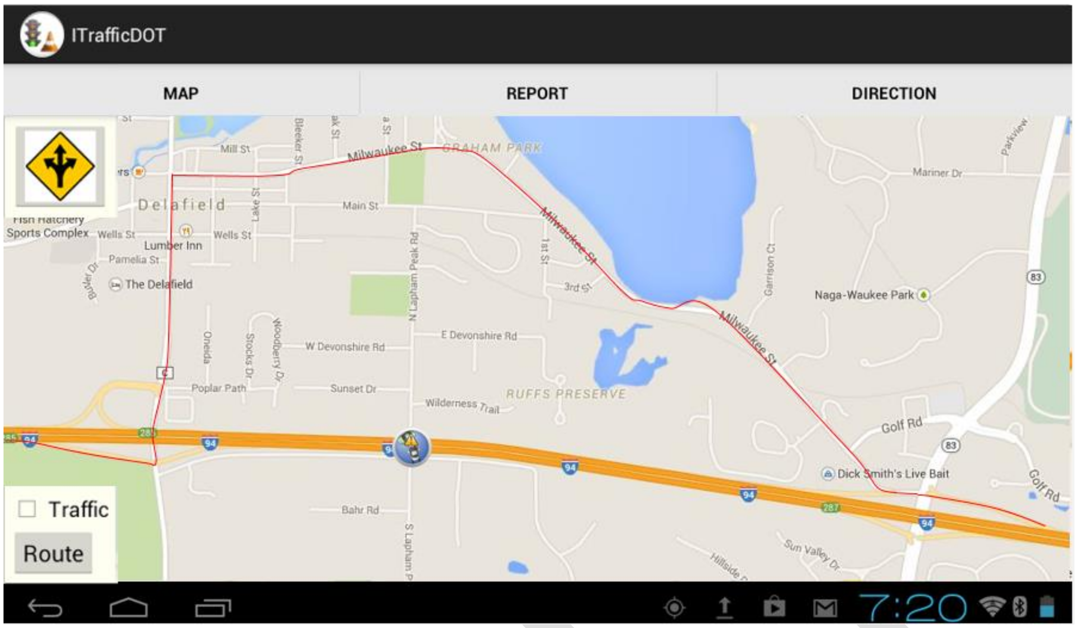


Figure 6.5 Scenario IV

6.3 System Benefit Analysis

This study estimated benefits of selected scenarios with the following procedure:

Step 1: Compute the difference in delay between with and without alternative route guidance

In this research the ***total travel time*** and ***total time in queue*** from the integrated corridor control model output are used to compute the reduced delay due to alternative route guidance.

Step 2: Select other impacts that could also be part of the benefit analysis

Once the delay decreases for any reason, associated by-products also decrease. This study includes reduced fuel consumptions and emissions (i.e., HC, CO, NO, and CO2) in this benefit estimation procedure.

Step 3: Estimate the reduced amount of each by-product based on related references

Assuming that all vehicles are passenger cars, the author estimates the fuel consumption reduction directly from the reduced delays using a conversion factor, 0.156 gallons of gasoline/hour, which is provided by the *Ohio Air Quality Development Authority* (Koerner, 2008). It should be mentioned that the assumption of passenger cars is only made for convenience of presentation and has nothing to do with the presented methodology and the proposed decision model. The inclusion of truck data will change only the estimated parameter values, but not the model structure or the research methodology.

Similarly, the reduced emissions can be estimated based on either the reduced delay or fuel consumption using conversion factors as follows:

- HC: 13.073 grams / hour of delay (provided by MDOT in 2000)
- CO: 146.831 grams / hour of delay (provided by MDOT in 2000)

- NO: 6.261 grams / hour of delay (provided by MDOT in 2000)
- CO2: 19.56 lbs CO2 / gallon of gasoline (Energy Information Administration in 2009)

Step 4: Convert the reduced delay, fuel consumption, and emissions to their monetary values

Similar to Step 3, we use monetary conversion factors to estimate the reduced delay and associated by-products as monetary values. Following are the values, and the sources for the factors.

- Delay: $27.37/ hour (U.S. Census Bureau in 2008)
- Fuel: $2.32/gallon (Energy Information Administration in 2009)
- HC: $6,700/ton (DeCorla-Souza, 1998)
- CO: $6,360/ton (DeCorla-Souza, 1998)
- NO: $12,875/ton (DeCorla-Souza, 1998)
- CO_2: $23 / metric ton (CBO (Congressional Budget Office)'s cost estimate for S. 2191, America's Climate Security Act of 2007)

Table 6.2 further displays the details for selected scenarios and corresponding outputs from the alternative route decision model, while Table 6.3 shows the benefits estimated from the aforementioned procedure.

Table 6.2 Descriptions of Scenarios for Benefit Analysis Illustrations

Categories		Scenario 1	Scenario 2	Scenario 3	Scenario 4
	Freeway : Detour Route Volume Level Incident Severity Lane Closure Status	L:L* Minor Moderate	H:L Intermediate Severe	H:H Intermediate Light	L:H Major Severe
Simulation Model Inputs	Number of Freeway	4	4	4	4
	Number of Lane Closures	2	3	2	3
	Incident Duration (minute)	15	30	75	150
	Freeway Volume (vphpl)	1250	2200	2200	1250
	Local Volume 1 (vphpl)	300	300	500	600
	Local Volume 2 (vphpl)	300	300	700	700
	Local Volume 3 (vphpl)	200	200	200	800
	Number of Signal on Primary Detour Route	2	4	2	5
	Ratio of Lane Closures	0.50	0.75	0.50	0.75
	Percentage Capacity Reduction	0.75	0.87	0.75	0.87
Flow Distribution for Each Route	Main Flow Rate	0.81	0.77	0.76	0.61
	Detour Flow Rate	0.19	0.23	0.24	0.39
Saved Outputs (w/o – w/ Detour)	Total Throughput	11432	12583	12492	15180
	Total vehicles in queue	3873	1035	1317	1252
	Total travel time (veh-hr)	1204.70	1548.04	1738.93	1964.18
	Total queue time (veh-hr)	432.85	407.72	571.75	910.16
	Total delay reduction (veh-hr)	**1637.55**	**1955.76**	**2310.78**	**2874.34**

* L: Light H: Heavy

Table 6.3 Estimated Benefit Based on Saved Delays

Estimated Benefit ($)	Scenario 1	Scenario 2	Scenario 3	Scenario 4
Delay	44,819.77	53,529.24	63,243.33	78,670.76
Fuel	592.66	707.83	836.28	1,040.28
HC	143.43	171.30	202.39	251.76
CO	1,529.22	1,826.38	2,157.82	2,684.19
NO	132.00	157.65	186.26	231.70
CO_2	52.13	62.26	73.56	91.50
Total	**47,269.21**	**56,454.70**	**66,699.65**	**82,970.20**

As shown in Table 6.2, selected scenarios cover four combinations of traffic conditions (heavy and light volumes) on both freeway and alternate routes. A significant reduction in delay and its resulting benefits have been shown in Table 6.3. Notice that considerable savings ($47,269.21) have been demonstrated in the first scenario that just reflects a minor incident case with relatively light volumes on both the freeway and detour route.

The second scenario with a greater diversion flow rate and a higher level of incidents shows a more considerable savings than scenario 1. Both the benefits of scenario 2 and 3 are considerable, which further validates the proposed alternative route decision model.

The last scenario demonstrates more promising benefits of implementing alternative route recommendation than the first three scenarios. The benefits of almost $83,000 are observed in the last scenario characterized by a major incident with a long duration.

CHAPTER 7: CONCLUSIONS

Despite the increasing attention to minimizing work zone/incident-incurred congestion with advanced traveler information systems, effective tools providing alternative route guidance to travelers are very limited.

This research develops a smartphone-based prototype system that supplements the 511 to improve its real-time traffic detouring service to state highway users under work zone or non-recurrent traffic conditions. The system makes the best use of the “xml” data feeds and retrieves the useful information using a web service integrated with the SQL database system. To make sure the alternative route guidance is warranted, this study has further developed a multi-criteria decision model, which is integrated with the server side system application. In the Android platform, a smartphone App was developed to provide travelers with real-time routing/re-routing options, traffic and incident information retrieved from the web server. The proposed prototype system has been applied with an actual freeway corridor (the IH-94 corridor between the city of Madison where IH-94 connects with IH-39/90 and the city of Milwaukee where it connects to IH-43). Tests with various scenarios have demonstrated significant overall benefits with system application and effective information provision to travelers in real time. The trajectories of the vehicle carrying the proposed the smartphone system can be automatically recorded into the server side database, which offers the potential for crowd-source traffic dynamics data collection and mining with sufficient number system users.

REFERENCES

About 511 http://www.ops.fhwa.dot.gov/511/about511/history.htm;

America's Travel Information Number Implementation and Operational Guidelines for 511 Services, 2005, 511 Deployment Coalition;

Aleyadeh, S., Sharief M.A., Oteafy., Hamid, S.A., Hassanein, H.S., 2012. Smartphone-based traffic information system for sustainable cities; ACM SIGMOBILE Mobile Computing and Communications Review Volume 16 Issue 4, October 2012 Pages 30-31;

Briante, O., Campolo, C., Iera, A., Molinaro, A., Paratore, Y. S., Ruggeri, S., 2014. Supporting augmented floating car data through smartphone-based crowd-sensing, Vehicular Communications, In press;

Boriboonsomsin, K., Barth, M.J., Zhu, W.H., VU, A., 2012. Eco-Routing Navigation System Based on Multisource Historical and Real-Time Traffic Information. IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 13, NO. 4, DECEMBER 2012;

Campolo, C., Iera, A.,Molinaro, A., Paratore, Y. S., Ruggeri, G., 2012. Smartcar: An Integrated Smartphone-based Platform to Support Traffic Management Applications, 2012 First International Workshop on Vehicular Traffic Management for Smart Cities (VTM);

Chuang, Y.T., Yi, C.W., Lu, Y.C., Tsai, P.C., 2013. iTraffic: A Smartphone-based Traffic Information System 2013. 2013 42nd International Conference on Parallel Processing;

Emerging Technologies, vol. 18, no. 4, pp. 626–638, January 2010. N. Geroliminis and A. Skabardonis, "Identification and analysis of queue spillovers in city street networks," IEEE Transactions on Intelligent Transportation Systems, vol. pp, no. 99, pp. 1–9, May 2011;

Traffic Diversion and Impact Research for IH-94 East-West Corridor Repaving Project, Department of Transportation, State of Wisconsin.

The New Analytics for Transportation Management: Using "Big" and Crowd-sourced Data, US Department of Transportation, March 2012;

Manolopoulos, V., Tao, S., Rusu, A., Papadimitratos, P., HotMobile 2012 Demo: Smartphone-based Traffic Information System for Sustainable Cities2012. Mobile Computing and Communications Review, Volume 16, Number 4, October 2012;

Misra, A., Gooze, A., Watkins, K., Asad, M., Le, C.A., Dantec., 2014. Crowdsourcing and Its Application to Transportation Data Collection and Management, Transportation Research Record: Journal of the Transportation Research Board, No. 2414, Transportation Research Board of the National Academies, Washington, D.C., 2014, pp. 1–8;

Nawaz, S., Efstratiou,C., Mascolo., C., 2013. ParkSense: A Smartphone Based Sensing System for On-Street Parking, MobiCom'13;

X. Wu, H. X. Liu, and D. Gettman, "Identification of oversaturated intersections using high-resolution identification of oversaturated intersections using high-resolution," Transportation Research Part C;

Yue, Y.Q., Zhang, K.W., Jacobsen, H.A., 2013. Smart Phone Application for Connected Vehicles and Smart Transportation, Middleware Systems Research Group, University of Toronto, CA;

APPENDIX A: SOURCE CODES OF THE APP

A.1 Code-511File Reader

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using System.IO.Compression;
using System.Xml;
using System.Globalization;
using System.Net;


namespace read511
{
    class Program
    {
        static void Main(string[] args)
        {
            dsrouteTableAdapters.RouteTableAdapter daroute = new
dsrouteTableAdapters.RouteTableAdapter();
dsroute.RouteDataTable dtroute = new dsroute.RouteDataTable();
daroute.Fill(dtroute);
for (int i = 0; i <= dtroute.Rows.Count - 1; i++) {
    dtroute.Rows[i].Delete();
}


//dtroute.Clear()
daroute.Update(dtroute);
dsroute.RouteRow drroute = default(dsroute.RouteRow);
string path = Directory.GetCurrentDirectory();
string remoteUri = "http://www.dot.state.wi.us/travel/511/";
string fileName = "Wis511LCS.xml.gzip", myStringWebResource = null;
WebClient myWebClient = new WebClient();
myStringWebResource = remoteUri + fileName;
myWebClient.DownloadFile(myStringWebResource,fileName);

DirectoryInfo directorySelected = new DirectoryInfo(path);
string xmlfilename = "";
```

```
foreach (FileInfo fileToDecompress in directorySelected.GetFiles("*.gzip"))
{
  xmlfilename = Decompress(fileToDecompress);
}
XmlDocument m_xmld = default(XmlDocument);
XmlNodeList m_nodelist = default(XmlNodeList);
//XmlNode m_node ;
m_xmld = new XmlDocument();
m_xmld.Load(xmlfilename);
m_nodelist = m_xmld.GetElementsByTagName("event-element-detail");
//   m_nodelist = m_xmld.ChildNodes.Item(1).SelectNodes("/fullEventUpdate/event-
element-details/event-element-detail")
dynamic locationItemAttr = new string[] {
    "primary-location",
    "secondary-location"
};
// Console.WriteLine(m_nodelist.Count)
int counter = 0;
foreach (XmlNode m_node in m_nodelist)
{
    drroute = dtroute.NewRouteRow();
    drroute.routeid = Guid.NewGuid();

    XmlNode node1 = default(XmlNode);
    XmlNode node2 = default(XmlNode);
    XmlNode node3 = default(XmlNode);
    counter = counter + 1;
    // Console.Write(counter)
    node1 = m_node.SelectNodes("element-locations").Item(0);
    node2 = node1.SelectNodes("element-location").Item(0);
    node3 = node2.SelectNodes("location-on-link").Item(0);
    for (int i = 0; i <= locationItemAttr.Length - 1; i++) {
        XmlNode geoNode = default(XmlNode);
        geoNode = node3.SelectNodes(locationItemAttr[i]).Item(0);
        //  Console.WriteLine(locationItemAttr(i))
        if ((geoNode != null)) {
            XmlNode LocationNode = default(XmlNode);
            XmlNode latitudeNode = default(XmlNode);
            XmlNode longitudeNode = default(XmlNode);
            LocationNode = geoNode.SelectNodes("geo-location").Item(0);
            latitudeNode = LocationNode.SelectNodes("latitude").Item(0);

            if ((latitudeNode != null)) {
                // Console.Write("latitude: ")
                if (i == 0) {
                    drroute.primarylatitude = latitudeNode.InnerText;
                } else {
                    drroute.secondarylatitude = latitudeNode.InnerText;
                }
            }
            longitudeNode = LocationNode.SelectNodes("longitude").Item(0);
            if ((longitudeNode != null)) {
                //  Console.Write("longitude: ")
                if (i == 0) {
                    drroute.primarylongitude =
longitudeNode.InnerText;
                } else {
                    drroute.secondarylongitude =
longitudeNode.InnerText;
                }
            }
            // Console.WriteLine()
        }
        // Console.WriteLine(locationItemAttr(i))
    }
    XmlNode dateNode = default(XmlNode);
    XmlNode timeNode = default(XmlNode);
    node1 = m_node.SelectNodes("element-times").Item(0);
    node2 = node1.SelectNodes("start-time").Item(0);
    dateNode = node2.SelectNodes("date").Item(0);
    if ((dateNode != null)) {
        //   Console.Write(dateNode.InnerText)
        drroute.starttimedate = System.DateTime.ParseExact(dateNode.InnerText,
"yyyyMMdd", CultureInfo.CurrentCulture).Date;

    }

    timeNode = node2.SelectNodes("time").Item(0);
    if ((timeNode != null)) {
        // Console.Write(timeNode.InnerText)
        drroute.starttimetime =
System.DateTime.ParseExact(timeNode.InnerText, "HHmmss",
CultureInfo.CurrentCulture).TimeOfDay;
    }

    node2 = node1.SelectNodes("valid-period").Item(0);
    node3 = node2.SelectNodes("expected-end-time").Item(0);
    dateNode = node3.SelectNodes("date").Item(0);
    if ((dateNode != null)) {
        //Console.Write(dateNode.InnerText)
```

```
                        drroute.expectedendtimedate = 
System.DateTime.ParseExact(dateNode.InnerText, "yyyyMMdd", 
CultureInfo.CurrentCulture).Date;
            }
            timeNode = node3.SelectNodes("time").Item(0);
            if ((timeNode != null)) {
                        //  Console.Write(timeNode.InnerText)
                        drroute.expectedendtimetime = 
System.DateTime.ParseExact(timeNode.InnerText, "HHmmss", 
CultureInfo.CurrentCulture).TimeOfDay;
            }
            dtroute.AddRouteRow(drroute);
            daroute.Update(dtroute);
}
        }

        public static string Decompress(FileInfo fileToDecompress)
        {
            using (FileStream originalFileStream = fileToDecompress.OpenRead())
            {
                string currentFileName = fileToDecompress.FullName;
                string newFileName = currentFileName.Remove(currentFileName.Length - 
fileToDecompress.Extension.Length);

                using (FileStream decompressedFileStream = File.Create(newFileName))
                {
                    using (GZipStream decompressionStream = new 
GZipStream(originalFileStream, CompressionMode.Decompress))
                    {
                        decompressionStream.CopyTo(decompressedFileStream);
                        Console.WriteLine("Decompressed: {0}", fileToDecompress.Name);
                    }
                }
                return newFileName;
            }
        }
    }
}
```

A.2 Route Service

```
Public Class infofinder
    Implements Iinfofinder

    Public Function getinfo(ByVal lat1 As String, ByVal lng1 As String, ByVal lat2 As
String, ByVal lng2 As String) As String Implements Iinfofinder.getinfo
        Dim retstr As String = ""
        Dim i As Integer
        Dim dtroute As New Dsroute.RouteDataTable
        Dim daroute As New DsrouteTableAdapters.RouteTableAdapter
        daroute.Fill(dtroute)

        Dim drroute As Dsroute.RouteRow()
        drroute = dtroute.Select("primarylatitude < '" & lat1 & "' and primarylatitude > '" &
lat2 & "' and primarylongitude < '" & lng1 & "' and primarylongitude > '" & lng2 & "'",
"[starttimedate] DESC")
        Dim drrep As Dsroute.RouteRow
        retstr = "ok"
        Dim j As Integer
        j = 0
        If drroute.Count > 0 Then
            retstr = "**<total**" & drroute.Count & "**total>**"
            For i = 1 To drroute.Count
                If j < 6 Then
                    drrep = drroute(i - 1)
                    If IsNothing(drrep) Then
                        retstr = "error"
                    Else
                        j = j + 1
                        retstr = retstr & "**<**" & i & "**>**" & drrep.primarylatitude & "**" &
drrep.primarylongitude & "*startd*" & drrep.starttimedate.Date.ToString & "*startt*" &
drrep.starttimetime.ToString & "*endd*" & drrep.expectedendtimedate.Date.ToString &
"*endt*" & drrep.expectedendtimetime.ToString
                    End If
                End If
            Next
        End If
        Return (retstr)
    End Function
End Class

 <ServiceContract()>
Public Interface Iinfofinder

    <OperationContract()>
    <WebInvoke(Method:="GET", _
    ResponseFormat:=WebMessageFormat.Json, _
    BodyStyle:=WebMessageBodyStyle.Wrapped, _
    UriTemplate:="getinfo/{lat1}/{lng1}/{lat2}/{lng2}")>
    Function getinfo(ByVal lat1 As String, ByVal lng1 As String, ByVal lat2 As String, ByVal lng2 As String) As String



End Interface
```

A.3 App Development

```
<uses-permission
        android:name="com.example.googlemaps.permission.MAPS_RECEIVE" />

<uses-permission
        android:name="android.permission.INTERNET" />

<uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<uses-permission
        android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />

<uses-permission
        android:name="android.permission.ACCESS_COARSE_LOCATION" />

<uses-permission
        android:name="android.permission.ACCESS_FINE_LOCATION" />

<uses-permission
        android:name="android.permission.ACCESS_NETWORK_STATE" />
```

A.3.1 Direction.java

```
package com.itrafficDot.cilab;

import android.app.Activity;
import android.os.Bundle;
```

```
public class Direction extends Activity {
            public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.direction);
    }

}
```

A.3.2 DirectionsJSONParser.java

```
package com.itrafficDot.cilab;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import com.google.android.gms.maps.model.LatLng;

public class DirectionsJSONParser {

    /** Receives a JSONObject and returns a list of lists containing latitude and longitude
*/
    public List<List<HashMap<String,String>>> parse(JSONObject jObject){

        List<List<HashMap<String, String>>> routes = new
ArrayList<List<HashMap<String,String>>>() ;
        JSONArray jRoutes = null;
        JSONArray jLegs = null;
        JSONArray jSteps = null;

        try {

            jRoutes = jObject.getJSONArray("routes");

            /** Traversing all routes */
            for(int i=0;i<jRoutes.length();i++){
                jLegs = ( (JSONObject)jRoutes.get(i)).getJSONArray("legs");
                List path = new ArrayList<HashMap<String, String>>();

                /** Traversing all legs */
                for(int j=0;j<jLegs.length();j++){
                    jSteps = ( (JSONObject)jLegs.get(j)).getJSONArray("steps");
                    /** Traversing all steps */
                    for(int k=0;k<jSteps.length();k++){
                        String polyline = "";
                        polyline = 
(String)((JSONObject)((JSONObject)jSteps.get(k)).get("polyline")).get("points");
                        List<LatLng> list = decodePoly(polyline);

                        /** Traversing all points */
                        for(int l=0;l<list.size();l++){
                            HashMap<String, String> hm = new HashMap<String, String>();
                            hm.put("lat", Double.toString(((LatLng)list.get(l)).latitude) );
                            hm.put("lng", Double.toString(((LatLng)list.get(l)).longitude) );
                            path.add(hm);
                        }
                    }
                    routes.add(path);
                }
            }

        } catch (JSONException e) {
            e.printStackTrace();
        }catch (Exception e){
        }
        return routes;
    }

    /**
    * Method to decode polyline points
    * Courtesy : jeffreysambells.com/2010/05/27/decoding-polylines-from-google-maps-
direction-api-with-java
    * */
    private List<LatLng> decodePoly(String encoded) {

        List<LatLng> poly = new ArrayList<LatLng>();
        int index = 0, len = encoded.length();
        int lat = 0, lng = 0;

        while (index < len) {
            int b, shift = 0, result = 0;
            do {
                b = encoded.charAt(index++) - 63;
                result |= (b & 0x1f) << shift;
                shift += 5;
            } while (b >= 0x20);
            int dlat = ((result & 1) != 0 ? ~(result >> 1) : (result >> 1));
            lat += dlat;

            shift = 0;
            result = 0;
            do {
                b = encoded.charAt(index++) - 63;
                result |= (b & 0x1f) << shift;
                shift += 5;
            } while (b >= 0x20);
            int dlng = ((result & 1) != 0 ? ~(result >> 1) : (result >> 1));
            lng += dlng;

            LatLng p = new LatLng((((double) lat / 1E5)),
                                 (((double) lng / 1E5)));
            poly.add(p);
        }
        return poly;
    }
}
```

A.3.3Main.java

```
package com.itrafficDot.cilab;

import android.os.Bundle;
import android.app.Activity;
import android.app.TabActivity;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences.Editor;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.View;
import android.view.ViewGroup;
import android.view.View.OnClickListener;
import android.widget.AbsoluteLayout;
import android.widget.Button;
import android.widget.PopupWindow;
import android.widget.TabHost;
import android.widget.TabHost.TabSpec;

public class Main extends TabActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.main);
        @SuppressWarnings("deprecation")
        TabHost tabHost = getTabHost();

        // Tab for Map
        TabSpec mapspec = tabHost.newTabSpec("Map");
        // setting Title and Icon for the Tab
        mapspec.setIndicator("Map",
                getResources().getDrawable(R.drawable.map_tab));
        Intent mapIntent = new Intent(this, Map.class);
        mapspec.setContent(mapIntent);

        // Tab for Direction
        TabSpec directionspec = tabHost.newTabSpec("Direction");

        directionspec.setIndicator("Direction",
                getResources().getDrawable(R.drawable.direction_tab));
        Intent directionIntent = new Intent(this, Direction.class);
        directionspec.setContent(directionIntent);

        // Tab for Report
        TabSpec reportspec = tabHost.newTabSpec("Report");
        reportspec.setIndicator("Report",
                getResources().getDrawable(R.drawable.report_tab));
        Intent reportIntent = new Intent(this, Report.class);
        reportspec.setContent(reportIntent);

        // Adding all TabSpec to TabHost
        tabHost.addTab(mapspec); // Adding map tab

        tabHost.addTab(reportspec); // Adding report tab
        tabHost.addTab(directionspec); // Adding dir tab




    }

}
```

A.3.4 Map_Routing.java

```
package com.itrafficDot.cilab;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.GooglePlayServicesUtil;
import com.google.android.gms.location.LocationListener;
import com.google.android.gms.maps.CameraUpdate;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.GoogleMap.OnMapClickListener;
import com.google.android.gms.maps.GoogleMap.OnMapLongClickListener;
import com.google.android.gms.maps.GoogleMap.OnMarkerDragListener;
import com.google.android.gms.maps.GoogleMapOptions;
import com.google.android.gms.maps.MapFragment;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.maps.model.Polygon;
import com.google.android.gms.maps.model.PolygonOptions;

import android.R.integer;
import android.location.Address;
import android.location.Criteria;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationManager;
import android.os.Bundle;
import android.os.StrictMode;
import android.app.Activity;
import android.app.AlertDialog;
// *** for case 1 Using MapFragment instead of SupportMapFragment
import android.app.FragmentManager;
import android.content.Context;
/// for case 1  ***

// **** for case 2
//import android.support.v4.app.FragmentActivity;
//import android.support.v4.app.FragmentManager;
/// for case 2 ***GoogleMap from MapFragment/SupportMapFragment

import android.util.Log;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemSelectedListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.PopupWindow;
import android.widget.RelativeLayout;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

import java.io.BufferedReader;
import java.io.ByteArrayInputStream;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import com.google.android.gms.maps.model.PolylineOptions;

import android.os.AsyncTask;
import android.graphics.Color;

import android.support.v4.app.FragmentActivity;
//import android.support.v4.app.FragmentManager;
import android.util.Log;
import android.view.Menu;
import android.view.View.OnClickListener;

import java.util.Random;

public class Map extends Activity implements OnMapClickListener,
		OnItemSelectedListener, LocationListener {

	private String SERVICE_URI;
	private String dir_origin = "";
	private String dir_destination = "";
	private String events;
	private ArrayList<LatLng> traficpoint = null;
	private ArrayList<LatLng> accidentpoint = null;
	private ArrayList<LatLng> workpoint = null;
	private MarkerOptions myMarkerOptions = null;

	public PopupWindow pwd;
	public Button btnpopclosedirection;
	public Button btnpopdirection;
	public Button btnpopgetdirection;
	public EditText etorgin;
	public EditText etdist;
	public ArrayList<LatLng> markerPoints;
	public ArrayList<LatLng> markerworkzone;
	public int latorg,lngorg,latdes,lngdes;
	String routePoints = ""; // to store points to be submitted to server

	String a1, b1, c1;

	public GoogleMapOptions compassEnabled(boolean enabled) {
		return null;
	}

	public GoogleMapOptions MyLocationButtonEnabled(boolean enabled) {
		return null;
	}

	private CheckBox mTrafficCheckbox;
	private Button mMyLocationCheckbox;
```

```
    final int RQS_GooglePlayServices = 1;
    private GoogleMap map;
    Location myLocation;
    TextView tvLocInfo;
    boolean markerClicked;
    PolygonOptions polygonOptions;
    Polygon polygon;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.map);
        SERVICE_URI = getResources().getString(R.string.service_url);

        markerPoints = new ArrayList<LatLng>();
        markerworkzone = new ArrayList<LatLng>();

        FragmentManager myFragmentManager = getFragmentManager();
        MapFragment myMapFragment = (MapFragment) myFragmentManager
                .findFragmentById(R.id.map);
        map = myMapFragment.getMap();
        map.getUiSettings().setRotateGesturesEnabled(true);
        map.getUiSettings().setScrollGesturesEnabled(true);
        map.getUiSettings().setTiltGesturesEnabled(true);
        map.getUiSettings().setZoomGesturesEnabled(true);
        map.animateCamera(CameraUpdateFactory.zoomTo(10), 2000, null);

        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder()
                .permitAll().build();
        StrictMode.setThreadPolicy(policy);

        map.setOnMapClickListener(this);
        // map.setOnMapLongClickListener(this);
        // map.setOnMarkerDragListener(this);

        tvLocInfo = (TextView) findViewById(R.id.locinfo);

        Spinner spinner = (Spinner) findViewById(R.id.layers_spinner);
        ArrayAdapter<CharSequence> adapter =
ArrayAdapter.createFromResource(
                this, R.array.layers_array,
                android.R.layout.simple_spinner_item);

    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown
_item);
```

```
                spinner.setAdapter(adapter);
                spinner.setOnItemSelectedListener(this);
                // this can be done because of the 'implements' interface,
                // see
                // http://developer.android.com/guide/topics/ui/controls/spinner.html

                mTrafficCheckbox = (CheckBox) findViewById(R.id.traffic);
                mMyLocationCheckbox = (Button) findViewById(R.id.my_location);
                // setUpMapIfNeeded();
                // Enable MyLocation Button in the Map
                map.setMyLocationEnabled(true);

                try {
                        LocationManager locationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);
                        // Criteria criteria = new Criteria();
                        // // criteria.setAccuracy(Criteria.ACCURACY_FINE);
                        // String provider = locationManager.getBestProvider(criteria,
                        // true);
                        Location lastLoc = locationManager

        .getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
                        map.animateCamera(CameraUpdateFactory.newLatLngZoom(new
LatLng(
                                        lastLoc.getLatitude(), lastLoc.getLongitude()), 15));

                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }


                // Popup Window

                                btnpopdirection = (Button)findViewById(R.id.btndirect);
                                btnpopdirection.setOnClickListener(new OnClickListener()
{
                                     @Override
                                     public void onClick(View vv) {

                                        pwd.showAtLocation((RelativeLayout )
findViewById(R.id.settup_view), Gravity.LEFT | Gravity.TOP, 20, 120);

                                     }
                                  });
```

```
                         LayoutInflater inflater = (LayoutInflater)
this.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
                         View layoutpm =
inflater.inflate(R.layout.popup_getdirection,
(ViewGroup)findViewById(R.id.popup_direction_root));
                         pwd = new PopupWindow(layoutpm,
                      350,
                      310,
                      true);
                         btnpopclosedirection =
(Button)layoutpm.findViewById(R.id.popclose);
                         btnpopclosedirection.setOnClickListener(new
OnClickListener() {
                             @Override
                             public void onClick(View vv) {

                               pwd.dismiss();
                             }
                           });
                         etorgin = (EditText)layoutpm.findViewById(R.id.etbegin);
                         etdist =
(EditText)layoutpm.findViewById(R.id.etdestination);
                         btnpopgetdirection =
(Button)layoutpm.findViewById(R.id.Btngetdirect);
                         btnpopgetdirection.setOnClickListener(new
OnClickListener() {
                             @Override
                             public void onClick(View vv) {

                              // Get Direction


                             //  etorgin.setText("4040 N Wilson Dr, Milwaukee,
WI");
                             //  etdist.setText("2300 E Kenwood Blvd, Milwaukee,
WI");
                               LatLng origin =
AddresstoPoint(etorgin.getText().toString());
                                           LatLng dest =
AddresstoPoint(etdist.getText().toString());


                                           latorg=(int)(origin.latitude *1E6);
                                     lngorg=(int)(origin.longitude *1E6);

                                           latdes=(int)(dest.latitude *1E6);
                                           lngdes=(int)(dest.longitude *1E6);
```

```
                            pwd.dismiss();
                            //Log.e("iTrafic", "Check");
                            //Log.e("iTrafic",
String.valueOf(origin.latitude));
                            //Log.e("iTrafic",
String.valueOf(origin.longitude));
                            // Getting URL to the Google
Directions API
                            String url;
                            try {
                                url = getDirectionsUrl(origin,
dest);
                                //Log.e("iTrafic", url);
                                tvLocInfo.setText(url);

                                // Start downloading json data
from Google Directions API
                                new
DownloadTask().execute(url);


                                    markerPoints.clear();
                                    map.clear();
                                    routePoints = "";

                                // Adding new item to the
ArrayList
                                markerPoints.add(origin);

                                MarkerOptions options =
new MarkerOptions();
                                options.position(origin);

    //options.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.
HUE_BLUE));

    options.icon(BitmapDescriptorFactory.fromResource(R.drawable.in));
                                map.addMarker(options);

                                markerPoints.add(dest);

                                MarkerOptions options2 =
new MarkerOptions();
                                options2.position(dest);
```

```
        //options2.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactor
y.HUE_RED));

        options.icon(BitmapDescriptorFactory.fromResource(R.drawable.out));
                                                                map.addMarker(options2);
                                                        } catch (Exception e) {
                                                                // TODO Auto-generated
catch block
                                                                e.printStackTrace();
                                                        }
                                     }
                                   });

        }

        public LatLng AddresstoPoint(String strAddress)
        {
                Geocoder coder = new Geocoder(getApplicationContext(),Locale.US);
                List<Address> address;

                LatLng p1 = new LatLng(0, 0);
                try {
                   address = coder.getFromLocationName(strAddress,1);
                   if (address == null) {
                        Log.e("iTrafic", "null");
                      return null;
                   }
                   Address location = address.get(0);
                   double lat = location.getLatitude();
                   double lng = location.getLongitude();
                   LatLng position = new LatLng(lat, lng);
                   Log.e("iTrafic", String.valueOf(position.latitude));
                        Log.e("iTrafic", String.valueOf(position.longitude));
                   p1=position;

                }catch (IOException e)
                {
                        Log.e("iTrafic", e.toString());

                }
                return p1;
        }


        @Override
```

```
public void onLocationChanged(Location location) {

    // Getting latitude of the current location
    double latitude = location.getLatitude();

    // Getting longitude of the current location
    double longitude = location.getLongitude();

    // Setting latitude and longitude in the TextView tv_locinfo
    tvLocInfo.setText("Latitude:" + latitude + ", Longitude:" + longitude);

}

public void onStatusChanged(String provider, int status, Bundle extras) {
    // TODO Auto-generated method stub
}

private String getDirectionsUrl(LatLng origin, LatLng dest) {

    // Origin of route
    String str_origin = "origin=" + origin.latitude + ","
            + origin.longitude;
    dir_origin = str_origin;

    // Destination of route
    String str_dest = "destination=" + dest.latitude + "," + dest.longitude;
    dir_destination = str_dest;

    // Sensor enabled
    String sensor = "sensor=false";

    // Building the parameters to the web service
    String parameters = str_origin + "&" + str_dest + "&" + sensor;

    // Output format
    String output = "json";

    // Building the url to the web service
    String url = "https://maps.googleapis.com/maps/api/directions/"
            + output + "?" + parameters;

    return url;
}

/** A method to download json data from url */
private String downloadUrl(String strUrl) throws IOException {
        String data = "";
        InputStream iStream = null;
        HttpURLConnection urlConnection = null;
        try {
            URL url = new URL(strUrl);

            // Creating an http connection to communicate with url
            urlConnection = (HttpURLConnection) url.openConnection();

            // Connecting to url
            urlConnection.connect();

            // Reading data from url
            iStream = urlConnection.getInputStream();

            BufferedReader br = new BufferedReader(new
InputStreamReader(
                    iStream));

            StringBuffer sb = new StringBuffer();

            String line = "";
            while ((line = br.readLine()) != null) {
                sb.append(line);
            }

            data = sb.toString();

            br.close();

        } catch (Exception e) {
            Log.d("Exception while downloading url", e.toString());
        } finally {
            iStream.close();
            urlConnection.disconnect();
        }
        return data;
    }

    // Fetches data from url passed
    private class DownloadTask extends AsyncTask<String, Void, String> {

        // Downloading data in background thread
        @Override
        protected String doInBackground(String... url) {
            // For storing data from web service
            String data = "";

            try {
                // Fetching the data from web service
                data = downloadUrl(url[0]);
            } catch (Exception e) {
                Log.d("Background Task", e.toString());
            }
            return data;
        }

        // Executes in UI thread, after the execution of
        // doInBackground()
        @Override
        protected void onPostExecute(String result) {
            super.onPostExecute(result);

            // Invokes the thread for parsing the JSON data
            new ParserTask().execute(result);
        }
    }

    /** A class to parse the Google Places in JSON format */
    private class ParserTask extends
            AsyncTask<String, Integer, List<List<HashMap<String,
String>>>> {

        // Parsing the data in non-ui thread
        // @Override
        protected List<List<HashMap<String, String>>> doInBackground(
                String... jsonData) {

            JSONObject jObject;
            List<List<HashMap<String, String>>> routes = null;

            try {
                jObject = new JSONObject(jsonData[0]);
                DirectionsJSONParser parser = new
DirectionsJSONParser();

                // Starts parsing data
                routes = parser.parse(jObject);
            } catch (Exception e) {
                e.printStackTrace();
            }
            return routes;
        }

        // Executes in UI thread, after the parsing process
        // @Override
        protected void onPostExecute(List<List<HashMap<String, String>>>
result) {
            ArrayList<LatLng> points = null;
            PolylineOptions lineOptions = null;
            String routePoints_tmp = "";

            // Traversing through all the routes
            for (int i = 0; i < result.size(); i++) {
                points = new ArrayList<LatLng>();
                lineOptions = new PolylineOptions();

                // !!experimental code to add points of interests
                traficpoint = new ArrayList<LatLng>();
                accidentpoint = new ArrayList<LatLng>();
                workpoint = new ArrayList<LatLng>();

                // Fetching i-th route
                List<HashMap<String, String>> path = result.get(i);

                // Fetching all the points in i-th route
                for (int j = 0; j < path.size(); j++) {
                    HashMap<String, String> point = path.get(j);

                    double lat = Double.parseDouble(point.get("lat"));
                    double lng = Double.parseDouble(point.get("lng"));
                    LatLng position = new LatLng(lat, lng);

                    points.add(position);
                    routePoints_tmp = routePoints_tmp +
String.valueOf(lat)
                            + "," + String.valueOf(lng) + ";";
                }

                // Adding all the points in the route to LineOptions
                lineOptions.addAll(points);
                lineOptions.width(2);
                lineOptions.color(Color.RED);

                // !!experimental code to add points of interests by
randomly selected indices
```

```
                traficpoint.add(points.get(new
Random().nextInt(points.size())));
                accidentpoint.add(points.get(new
Random().nextInt(points.size())));
                workpoint.add(points.get(new
Random().nextInt(points.size())));


            }

            // removing the last separator "|"
            routePoints = routePoints_tmp.substring(0,
                    routePoints_tmp.length() - 1);

            // Drawing polyline in the Google Map for the i-th route
            map.addPolyline(lineOptions);

        }
    }


    public void onMyLocationToggled(View view) {
        updateMyLocation();
    }

    private void updateMyLocation() {
        if (!checkReady()) {
            return;
        }
        if (routePoints != "") {
            String lat1,lat2,lng1,lng2;

            if (latorg>latdes)
            {
                lat1=String.valueOf(latorg);
                lat2=String.valueOf(latdes);
            }else
            {
                lat1=String.valueOf(latdes);
                lat2=String.valueOf(latorg);
            }
            if (lngorg>lngdes)
            {
                lng1=String.valueOf(lngorg);
                lng2=String.valueOf(lngdes);
            }else
            {
                lng1=String.valueOf(lngdes);
                lng2=String.valueOf(lngorg);
            }
            //new Getinfodata().execute("45884895","-
87954185","42892854","-87969464");
            new Getinfodata().execute(lat1,lng1,lat2,lng2);




            //new submitfile().execute(routePoints);
        }
    }

    class submitfile extends AsyncTask<String, Void, String> {

        @Override
        protected String doInBackground(String... fn) {
            String res = "0";
            try {
                res = uploadstring(fn[0], "idcode"); // upload route points

            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                Log.e("sending", e.getMessage());
            }
            return res;
        }

        protected void onPostExecute(String feed) {
            // TODO: check this.exception
            // TODO: do something with the feed
            events = feed;
            tvLocInfo.setText(events);

            for (int j= 0; j< traficpoint.size(); j++){
                myMarkerOptions = new MarkerOptions();
                myMarkerOptions.position(traficpoint.get(j));

    myMarkerOptions.icon(BitmapDescriptorFactory.fromResource(R.drawable.traffi
cs));
//              myMarkerOptions.title("?");
                myMarkerOptions.snippet("Traffic");
                map.addMarker(myMarkerOptions);
            }

            for (int j= 0; j< accidentpoint.size(); j++){
                myMarkerOptions = new MarkerOptions();
                myMarkerOptions.position(accidentpoint.get(j));

    myMarkerOptions.icon(BitmapDescriptorFactory.fromResource(R.drawable.acci
dent));
//              myMarkerOptions.title("*");
                myMarkerOptions.snippet("accident");
                map.addMarker(myMarkerOptions);
            }

            for (int j= 0; j< accidentpoint.size(); j++){
                myMarkerOptions = new MarkerOptions();
                myMarkerOptions.position(workpoint.get(j));

    myMarkerOptions.icon(BitmapDescriptorFactory.fromResource(R.drawable.wor
kzone));
//              myMarkerOptions.title("*");
                myMarkerOptions.snippet("work zone");
                map.addMarker(myMarkerOptions);
            }

        }

    }
    class Getinfodata extends AsyncTask<String, Void, String> {

        @Override
        protected String doInBackground(String... fn) {
            String res = "0";
            try {
                res = getinfosub(fn[0],fn[1],fn[2],fn[3]); // upload route
points

            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                Log.e("sending", e.getMessage());
            }
            return res;
        }

        protected void onPostExecute(String feed) {
            // TODO: check this.exception
            // TODO: do something with the feed
            events = feed;
            tvLocInfo.setText(events);
        //  Log.e("sending",events);

        //  Log.e("workzonesize",String.valueOf(markerworkzone.size()));
        for (int j= 0; j< markerworkzone.size(); j++){
                myMarkerOptions = new MarkerOptions();
                myMarkerOptions.position(markerworkzone.get(j));

    myMarkerOptions.icon(BitmapDescriptorFactory.fromResource(R.drawable.wor
kzone));
//              myMarkerOptions.title("*");
                myMarkerOptions.snippet("work zone");
                map.addMarker(myMarkerOptions);
            }

        }

    }
    public String uploadstring(String data, String uid) throws IOException {
        URL url = null;
        HttpURLConnection conn = null;
        DataOutputStream dos = null;
        DataInputStream inStream = null;
        String lineEnd = "\r\n";
        String twoHyphens = "--";
        String boundary = "*****";

        int bytesRead, bytesAvailable, bufferSize;
        byte[] buffer;
        int maxBufferSize = Integer.MAX_VALUE;

        url = new URL(SERVICE_URI + "/gettxtfile/" + uid);
        Log.e("Debug url", SERVICE_URI + "/gettxtfile/" + uid);
        try {
            // ------------------ CLIENT REQUEST
            // FileInputStream fileInputStream = new FileInputStream(new
            // File(existingFileName) );

            // Open a HTTP connection to the URL
            conn = (HttpURLConnection) url.openConnection();
            // Allow Inputs
            conn.setDoInput(true);
            // Allow Outputs
            conn.setDoOutput(true);
            // Don't use a cached copy.
            conn.setUseCaches(false);
            // Use a post method.
            conn.setRequestMethod("POST");
            conn.setRequestProperty("Connection", "Keep-Alive");
            conn.setRequestProperty("Content-Type", "application/stream");
            dos = new DataOutputStream(conn.getOutputStream());
            InputStream streamin = new ByteArrayInputStream(
                    data.getBytes("UTF-8"));
            // create a buffer of maximum size
            bytesAvailable = streamin.available();
            bufferSize = Math.min(bytesAvailable, maxBufferSize);
            buffer = new byte[bufferSize];
            // read file and write it into form...
            bytesRead = streamin.read(buffer, 0, bufferSize);
            Log.e("Debug stat", "before start");
            while (bytesRead > 0) {
                dos.write(buffer, 0, bufferSize);
                bytesAvailable = streamin.available();
                bufferSize = Math.min(bytesAvailable, maxBufferSize);
                bytesRead = streamin.read(buffer, 0, bufferSize);

            }

            // send multipart form data necesssary after file data...
//          dos.writeBytes(lineEnd);

            // close streams
            Log.e("Debug s1", twoHyphens + boundary + twoHyphens +
lineEnd);
            streamin.close();
            dos.flush();
            dos.close();
        } catch (MalformedURLException ex) {
            Log.e("Debug s2", "error: " + ex.getMessage(), ex);
        } catch (IOException ioe) {
            Log.e("Debug s3", "error: " + ioe.getMessage(), ioe);

        }
        String str = "test";
        String strm = "";
        // ------------------ read the SERVER RESPONSE
        try {
            inStream = new DataInputStream(conn.getInputStream());

            while ((str = inStream.readLine()) != null) {
                Log.e("Debug s4", "Server Response " + str);
                strm = strm + str;

            }
            inStream.close();

        } catch (IOException ioex) {
            Log.e("Debug s5", "error: " + ioex.getMessage(), ioex);
        }
        Log.e("Data", strm);

        conn.disconnect();
        return strm;
    }

    public String getinfosub(String lat1, String lng1,String lat2,String lng2) throws
IOException {
        String resdata="";
        try {
          HttpGet request = new HttpGet(SERVICE_URI + "/getinfo/" + lat1 +
"/" + lng1 + "/" + lat2 + "/" + lng2);
          Log.e("bb",SERVICE_URI + "/getinfo/" + lat1 + "/" + lng1 + "/" + lat2
+ "/" + lng2);
       request.setHeader("Accept", "application/getinfo");
       request.setHeader("Content-type", "application/getinfo");


       DefaultHttpClient httpClient = new DefaultHttpClient();
       HttpResponse response = httpClient.execute(request);

       HttpEntity responseEntity = response.getEntity();

       // Read response data into buffer
       char[] buffer = new char[(int)responseEntity.getContentLength()];
       InputStream stream = responseEntity.getContent();
       InputStreamReader reader = new InputStreamReader(stream);
       reader.read(buffer);
       stream.close();

       JSONArray plates = new JSONArray("[" + new String(buffer) + "]");
      // Log.e("check point","Json array");

       for (int i = 0; i < plates.length(); ++i) {
          JSONObject jsonUser = plates.getJSONObject(i);
          resdata= jsonUser.getString("getinfoResult");
       }
      //  Toast.makeText(getApplicationContext(), resdata,
Toast.LENGTH_LONG).show();

      //   Toast.makeText(getApplicationContext(), resdata,
Toast.LENGTH_LONG).show();
     //   Log.e("check point","retrievedata");
        retrievedata(resdata);
     //     Log.e("check point","after retrievedata");
            } catch (Exception e) {
           e.printStackTrace();
           Toast.makeText(getApplicationContext(),  e.getMessage(),
Toast.LENGTH_LONG).show();
          //   resdata=resdata + e.getMessage();
          }
        return "OK";
    }
    private void retrievedata(String datain)
    {
        int pos1,pos2,pos3;
        try{


        //Log.e("check point",datain);
        pos1=datain.indexOf("**total>****");
        if (pos1==-1)
        {
            markerworkzone.clear();
        }else
        {
        String total;
        //Log.e("check point",String.valueOf(pos1));
        total=datain.substring(10,pos1);
        //Log.e("Total",total);
        int itotal=Integer.valueOf(total);
        // ***********************************************************
        // ***********************************************************
        // *******************This is just for Get Methods***********
        // ***********************************************************
        // ***********************************************************
        if (itotal>6)
        {
            itotal=6;
        }
        // ***********************************************************
        // ***********************************************************
```

```
                        // ********************************************************
                        // ********************************************************
                        // ********************************************************
                        int i;

                        String startstring;
                        double[] lati = new double[itotal];
                        double[] lngi = new double[itotal];

                for (i=0;i<itotal ;i++)
                {
                        startstring="**<**"+String.valueOf(i+1)+"**>**";
                //      Log.e("Total",startstring);
                        pos2=datain.indexOf(startstring);
                        datain=datain.substring(pos2,datain.length());
                        pos2=0;
                        pos2=pos2+startstring.length();
                        lati[i]=Double.valueOf(datain.substring(pos2,pos2+8))/1e6;
                        pos3=datain.indexOf("*startd*");
                        //Log.e("Total",datain.substring(pos2+10,pos3));
                        lngi[i]=Double.valueOf(datain.substring(pos2+10,pos3))/1e6;

                }
                markerworkzone.clear();

                for (int j= 0; j< itotal; j++){

                                    LatLng newposition = new LatLng(lati[j], lngi[j]);
                                    markerworkzone.add(newposition);
                        }
            //          Log.e("Total",total);
                        }} catch (Exception e) {
                    e.printStackTrace();

                }


                        //Toast.makeText(getApplicationContext(), "Type:" + type +
";\nLocation:" +location + ";\nStarttime:" + starttime, Toast.LENGTH_LONG).show();
            //nlat=(int)(Float.parseFloat(lat)*1E6);
            //nlng=(int)(Float.parseFloat(lng)*1E6);
            };
            /*
             * private void setUpMapIfNeeded() {
             *
             * if (map == null) { FragmentManager myFragmentManager =
     * getFragmentManager(); MapFragment myMapFragment = (MapFragment)
     * myFragmentManager .findFragmentById(R.id.map); map =
     * myMapFragment.getMap(); } }
     */

    private boolean checkReady() {
        if (map == null) {
            Toast.makeText(this, R.string.map_not_ready,
Toast.LENGTH_SHORT)
                    .show();
            return false;
        }
        return true;
    }

    // Called when the Traffic checkbox is clicked.
    public void onTrafficToggled(View view) {
        updateTraffic();
    }

    private void updateTraffic() {
        if (!checkReady()) {
            return;
        }
        map.setTrafficEnabled(mTrafficCheckbox.isChecked());
    }

    // @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position,
            long id) {

        setLayer((String) parent.getItemAtPosition(position));
    }

    private void setLayer(String layerName) {
        if (!checkReady()) {
            LocationManager locationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);
            Location lastLoc = locationManager

    .getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

            Criteria criteria = new Criteria();
            String provider = locationManager.getBestProvider(criteria, true);

            double latitute = lastLoc.getLatitude();
```

```
                double longitute = lastLoc.getLongitude();
                double nyspeed = lastLoc.getSpeed();

                a1 = String.valueOf(latitute);
                b1 = String.valueOf(longitute);

                LatLng latlng = new LatLng(latitute, longitute);
                c1 = String.valueOf(latlng);
                tvLocInfo.setText(a1);

                return;
        }
        if (layerName.equals(getString(R.string.normal))) {
                map.setMapType(GoogleMap.MAP_TYPE_NORMAL);
        } else if (layerName.equals(getString(R.string.hybrid))) {
                map.setMapType(GoogleMap.MAP_TYPE_HYBRID);
        } else if (layerName.equals(getString(R.string.satellite))) {

                map.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
        } else if (layerName.equals(getString(R.string.terrain))) {
                map.setMapType(GoogleMap.MAP_TYPE_TERRAIN);
        } else {
                Log.i("LDA", "Error setting layer with name " + layerName);
        }
}

@Override
public void onNothingSelected(AdapterView<?> parent) {
        // Do nothing.
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {

        return super.onOptionsItemSelected(item);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
}

@Override
public void onMapClick(LatLng point) {
        tvLocInfo.setText(point.toString());

        // Already two locations
        if (markerPoints.size() == 2) {
            markerPoints.clear();
            map.clear();
            routePoints = "";
        }

        // Adding new item to the ArrayList
        markerPoints.add(point);

        // Creating MarkerOptions
        MarkerOptions options = new MarkerOptions();

        // Setting the position of the marker
        options.position(point);

        /**
         * For the start location, the color of marker is GREEN and for the end
         * location, the color of marker is RED.
         */
        if (markerPoints.size() == 1) {
        //
    options.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.H
UE_BLUE));

    options.icon(BitmapDescriptorFactory.fromResource(R.drawable.in));
            latorg=(int)(point.latitude*1E6);
            lngorg=(int)(point.longitude*1E6);

        } else if (markerPoints.size() == 2) {

    //options.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.
HUE_RED));

    options.icon(BitmapDescriptorFactory.fromResource(R.drawable.out));
            latdes=(int)(point.latitude*1E6);
            lngdes=(int)(point.longitude*1E6);
        }

        // Add new marker to the Google Map Android API V2
        map.addMarker(options);

        // Checks, whether start and end locations are captured
        if (markerPoints.size() == 2) {
            LatLng origin = markerPoints.get(0);
            LatLng dest = markerPoints.get(1);

            // Getting URL to the Google Directions API
            String url;
            try {
                url = getDirectionsUrl(origin, dest);
                tvLocInfo.setText(url);

            // Start downloading json data from Google Directions API
                new DownloadTask().execute(url);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
}
```

A.3.5 Report.java

```
package com.itrafficDot.cilab;

import android.app.Activity;
import android.os.Bundle;
import android.os.StrictMode;
import android.widget.Button;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONObject;
import org.xmlpull.v1.XmlSerializer;

import android.content.Intent;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;

import android.os.Environment;
import android.util.Log;
import android.util.Xml;
import android.view.View;
import android.view.Window;
import android.view.View.OnClickListener;

import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class Report extends Activity {
        private String SERVICE_URI;

        private Button submit;
        private int setingresult = 0;
        private EditText lb, lat, longi;
        private TextView terorr;

        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.report);
                // Window window = getWindow();
                // setRequestedOrientation(1);
                // window.setBackgroundDrawable(new ColorDrawable(Color.rgb(255,
255,
                // 255)));
                SERVICE_URI = getResources().getString(R.string.service_url);

                lat = (EditText) findViewById(R.id.editLatitute);
                longi = (EditText) findViewById(R.id.editLongitude);

                // final Button coordinates = (Button) findViewById(R.id.BtnSubmit);
                // coordinates.setOnClickListener(SubmitButtonListener);


        }

        public void SubmitCoordinates(View v) {

                String a, b, c;
                a = lat.getText().toString();
                b = longi.getText().toString();
        c = a + "," + b;

        String locdata = "";

        try {

            HttpGet request = new HttpGet(SERVICE_URI + "/sendLongLat/"
+ c);
            request.setHeader("Accept", "application/sendLongLat");
            request.setHeader("Content-type", "application/sendLongLat");

            DefaultHttpClient httpClient = new DefaultHttpClient();
            HttpResponse response = httpClient.execute(request);

            HttpEntity responseEntity = response.getEntity();

            // Read response data into buffer
            char[] buffer = new char[(int) responseEntity.getContentLength()];
            InputStream stream = responseEntity.getContent();
            InputStreamReader reader = new InputStreamReader(stream);
            reader.read(buffer);
            stream.close();

            JSONArray plates = new JSONArray("[" + new String(buffer) +
"]");

            for (int i = 0; i < plates.length(); ++i) {
                JSONObject jsonUser = plates.getJSONObject(i);
                locdata = jsonUser.getString("sendLongLatResult");
            }

        } catch (Exception e) {
            e.printStackTrace();
            Toast.makeText(getApplicationContext(),
                    "Error occurred at:" + SERVICE_URI,
Toast.LENGTH_LONG).show();
        }
        lat.setText("");
        longi.setText("");

        Toast.makeText(getApplicationContext(), locdata,
Toast.LENGTH_LONG)
                .show();
    }

}
```