

Winter 2008

# Project54 vehicle telematics for remote diagnostics, fleet management and traffic monitoring

Ian Cassias

*University of New Hampshire, Durham*

Follow this and additional works at: <https://scholars.unh.edu/thesis>

---

## Recommended Citation

Cassias, Ian, "Project54 vehicle telematics for remote diagnostics, fleet management and traffic monitoring" (2008). *Master's Theses and Capstones*. 412.

<https://scholars.unh.edu/thesis/412>

This Thesis is brought to you for free and open access by the Student Scholarship at University of New Hampshire Scholars' Repository. It has been accepted for inclusion in Master's Theses and Capstones by an authorized administrator of University of New Hampshire Scholars' Repository. For more information, please contact [nicole.hentz@unh.edu](mailto:nicole.hentz@unh.edu).

**PROJECT54 VEHICLE TELEMATICS FOR  
REMOTE DIAGNOSTICS, FLEET MANAGEMENT  
AND TRAFFIC MONITORING**

BY

IAN CASSIAS

BS, University of New Hampshire, 2005

THESIS

Submitted to the University of New Hampshire

in Partial Fulfillment of

the Requirements for the Degree of

Master of Science

in

Electrical Engineering

December, 2008

UMI Number: 1463215

### INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

**UMI**<sup>®</sup>

---

UMI Microform 1463215

Copyright 2009 by ProQuest LLC.

All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

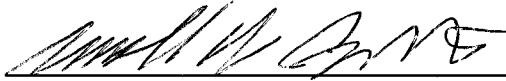
ProQuest LLC  
789 E. Eisenhower Parkway  
PO Box 1346  
Ann Arbor, MI 48106-1346

This thesis has been examined and approved.



---

Thesis Director, Andrew L. Kun  
Associate Professor of Electrical and  
Computer Engineering



---

William H. Lenharth  
Director, Research Computing Center and  
Research Associate Professor of Electrical and  
Computer Engineering



---

W. Thomas Miller, III  
Professor of Electrical and Computer  
Engineering

11/17/38

---

Date

## ACKNOWLEDGEMENTS

First and foremost I thank Dr. Andrew L. Kun for being my advisor and for his patience, guidance, time and support during the development of this thesis.

I thank Dr. William Lenharth for his constant encouragement and for reviewing early drafts, providing helpful suggestions and serving on my thesis committee.

Thank you Dr. W. Thomas Miller, III for serving on my thesis committee and providing valuable feedback.

Thank you to everyone involved in the Consolidated Advanced Technologies Lab program for providing me the opportunity to perform research for Project54 and to pursue a graduate degree at the University of New Hampshire.

Matthew Lape helped to collect radar data for this thesis. I thank Matt for his time, patience and suggestions.

Thank you to all my colleagues at Project54 for your ideas, assistance and encouragement. It has been a pleasure working with you.

Finally, thank you to all my family and friends. I would not be where I am today without your unwavering love and support.

# TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....	iii
TABLE OF CONTENTS.....	iv
LIST OF TABLES .....	viii
LIST OF FIGURES .....	x
LIST OF EQUATIONS .....	xiii
LIST OF ACRONYMS.....	xiv
ABSTRACT.....	xvi
<b>CHAPTER</b>	<b>PAGE</b>
<b>1. INTRODUCTION.....</b>	<b>1</b>
1.1 Problem Description.....	1
1.2 Thesis Objective.....	3
1.3 Approach.....	4
1.3.1 Vehicle Diagnostics Approach.....	4
1.3.2 Fleet Management Approach.....	5
1.3.3 Traffic Monitoring Approach.....	5
1.4 Thesis Organization .....	6
<b>2. BACKGROUND.....</b>	<b>7</b>
2.1 Telematics Services.....	7
2.1.1 Remote Diagnostics.....	7
2.1.2 Fleet Management.....	10
2.1.3 Information Access.....	13

2.1.4 Context Awareness and Traffic Monitoring .....	15
2.2 Architecture .....	18
2.2.1 Hardware .....	18
2.2.2 Middleware .....	20
2.2.3 Software .....	21
2.3 Implementation Issues .....	22
2.3.1 Privacy and Data Security .....	23
2.3.2 Human Factors Design .....	25
<b>3. PROJECT54 VEHICLE DIAGNOSTICS .....</b>	<b>28</b>
3.1 Diagnostic Hardware .....	28
3.1.1 WTPMS Hardware Description .....	29
3.1.2 OBD-II Scan Tool Description .....	30
3.1.3 Diagnostic Hardware Integration .....	31
3.2 Diagnostic Software .....	32
3.2.1 Tire Pressure Application .....	33
3.2.2 OBD-II Application .....	36
3.2.3 Vehicle Diagnostics Application .....	44
3.2.4 Diagnostic Logger Setup Application .....	50
<b>4. FLEET MANAGEMENT .....</b>	<b>53</b>
4.1 Maintenance Reports Application .....	53
4.2 Fleet Management Log Files .....	55
4.2.1 Maintenance and Mileage Reports Format .....	55
4.2.2 P54 and OBD Status Report Format .....	56
4.2.3 Vehicle Diagnostics Log Files .....	58
4.3 Communication Infrastructure .....	59

4.4 Data Analysis and Post-Processing .....	61
4.4.1 Diagnostic Data Plotter Application.....	61
4.4.2 Fuel Economy Estimation.....	63
<b>5. TESTING AND DEPLOYMENT .....</b>	<b>66</b>
5.1 Development Testing .....	66
5.2 Deployment Testing .....	68
5.2.1 NHDS Deployment - Phase One .....	69
5.2.2 NHDS Deployment - Phase Two.....	72
5.3 Fleet Management Testing.....	74
<b>6. TRAFFIC MONITORING.....</b>	<b>76</b>
6.1 Traffic Data Collection.....	76
6.2 Traffic Data Analysis and Observations.....	78
6.3 Traffic Congestion Scoring .....	87
<b>7. CONCLUSIONS.....</b>	<b>90</b>
7.1 Vehicle Diagnostics .....	90
7.2 Fleet Management.....	91
7.3 Traffic Monitoring .....	92
<b>8. SUGGESTIONS FOR FUTURE WORK.....</b>	<b>93</b>
8.1 Vehicle Diagnostics Improvements.....	93
8.1.1 Multiple OBD-II Sensor Request Messaging .....	93
8.1.2 Adjustable OBD-II Sensor Sampling Rates.....	94
8.1.3 Additional OBD-II Diagnostic Service Modes.....	94
8.1.4 Inferred Vehicle Parameters .....	95
8.2 Fleet Management Improvements .....	95
8.2.1 Real Time Updates and Additional Data .....	95
8.2.2 Web Based Interface .....	96



8.2.3 <i>Data-Driven Diagnostics and Maintenance</i> .....	96
8.3 Traffic Monitoring .....	97
8.3.1 <i>Expanded Data Collection</i> .....	97
8.3.2 <i>Project54 Traffic Monitoring Application</i> .....	97
<b>BIBLIOGRAPHY</b> .....	<b>99</b>
<b>APPENDIX A</b> .....	<b>107</b>
<b>APPENDIX B</b> .....	<b>110</b>
<b>APPENDIX C</b> .....	<b>112</b>
<b>APPENDIX D</b> .....	<b>115</b>
<b>APPENDIX E</b> .....	<b>119</b>
<b>APPENDIX F</b> .....	<b>121</b>

## LIST OF TABLES

Table 3.1 - Scan Tool Supported OBD-II Protocols.....	31
Table 3.2 - Tire Pressure IDB Connection Settings.....	33
Table 3.3 - Tire Status Message Parameters.....	34
Table 3.4 - OBD-II IDB Connection Settings .....	38
Table 3.5 - O2 Sensor Location Labels .....	40
Table 3.6 - OBD-II Initialization Steps.....	40
Table 3.7 - Standardized and Manufacturer Defined DTC Formats.....	41
Table 3.8 - Messaging Interfaces from OBD-II Application to Clients .....	43
Table 3.9 - Messaging Interfaces from Clients to OBD-II Application .....	43
Table 3.10 - Vehicle Diagnostics GUI Button Functions.....	46
Table 3.11 - OBD Setup Screen GUI Button Functions.....	49
Table 4.1 - Maintenance and Mileage Report Fields.....	56
Table 4.2 - P54 and OBD Status Report Format .....	57
Table 4.3 - Diagnostic Data Plotter Menu Commands .....	63
Table 4.4 - Fuel Consumption Calculation Variables and Coefficients .....	64
Table 5.1 - Phase One Tire Pressure Data Summary.....	69
Table 5.2 - Phase One Project54 Status Data Summary.....	70
Table 5.3 - Phase One OBD-II Data Summary.....	70
Table 5.4 - Phase Two Tire Pressure Data Summary .....	72
Table 5.5 - Phase Two Project54 Status Data Summary .....	72

Table 5.6 - Phase Two OBD-II Data Summary .....	73
Table 6.1 - Description of Traffic Monitoring Datasets .....	78
Table 6.2 - Correlation between Radar Hits and Vehicle Count .....	85
Table 6.3 - Correlation between Average Speed and Vehicle Count .....	86
Table 8.1 - Summary of Supported OBD-II Diagnostic Services .....	94
Table A.1 - Tire Pressure Monitoring System Parts List.....	108
Table B.1 - OBD-II Scan Tool Parts List .....	110
Table C.1 - OBD-II Sensors Supported by Project54.....	112

## LIST OF FIGURES

Figure 3.1 - Tire Pressure Client Feedback Request Messages .....	34
Figure 3.2 - Tire Status Message Format.....	34
Figure 3.3 - Project54 Component Status Message Format .....	35
Figure 3.4 - Tire Pressure Application Flowchart .....	35
Figure 3.5 - OBD-II IDB Thread Flowchart.....	36
Figure 3.6 - OBD-II Client Feedback Request Messages.....	37
Figure 3.7 - OBD-II Message Thread Flowchart.....	37
Figure 3.8 - OBD-II IDB Thread Initialization Process .....	38
Figure 3.9 - Available Vehicle Sensor Status Message Format.....	39
Figure 3.10 - O2 Location Definition Status Message Format.....	39
Figure 3.11 - OBD-II Client Reset Request Message.....	40
Figure 3.12 - OBD-II DTC Status Message Format .....	41
Figure 3.13 - OBD-II Client DTC Clear Request Message .....	42
Figure 3.14 - Clear DTC Status Message Format.....	42
Figure 3.15 - OBD-II Client Sensor Request Message.....	42
Figure 3.16 - OBD-II Sensor Update Status Message Format.....	43
Figure 3.17 - Project54 Vehicle Diagnostics Application Screenshot.....	44
Figure 3.18 - Project54 PDA Diagnostics Application Screenshot .....	45
Figure 3.19 - PC OBD Setup Screenshot.....	47
Figure 3.20 - PDA OBD Setup Screenshot.....	48

Figure 3.21 - DTC Clear Request Confirmation Dialog.....	50
Figure 3.22 - Diagnostic Data Logger Application Screenshot.....	51
Figure 4.1 - Project54 Maintenance Reports Application Screenshot.....	54
Figure 4.2 - Fleet Management Log File Directory.....	55
Figure 4.3 - Example Maintenance and Mileage Report.....	56
Figure 4.4 - P54 and OBD Status Report Example 1.....	58
Figure 4.5 - P54 and OBD Status Report Example 2.....	58
Figure 4.6 - Vehicle Diagnostic Log File Example.....	59
Figure 4.7 - Project54 Data Update System Physical Network Architecture.....	60
Figure 4.8 - Diagnostic Data Plotter Application Screenshot.....	62
Figure 5.1 – DTC Detected by the Project54 OBD-II Application.....	68
Figure 5.2 – Vehicle Diagnostics Log for 19 DTC Error Report.....	74
Figure 6.1 - Traffic Data Collection Setup.....	77
Figure 6.2 - Traffic Dataset #1, 35 mph Rural Road.....	79
Figure 6.3 - Traffic Dataset #2, 35 mph Rural Road.....	80
Figure 6.4 - Traffic Dataset #3, 35 mph Rural Road.....	81
Figure 6.5 - Traffic Dataset #4, 55 mph Single Lane Highway.....	82
Figure 6.6 - Traffic Dataset #5, 55 mph Single Lane Highway.....	83
Figure 6.7 - Traffic Dataset #6, 55 mph Multilane Highway.....	84
Figure 6.8 - Congested Multilane Highway Interval Z-Score Results.....	89
Figure A.1 - WTPMS Device Image.....	107
Figure A.2 - WTPMS User Interface.....	107
Figure A.3 - WTPMS Connection Diagram.....	107

Figure A.4 - Tire Locations for Monitor Programming.....	108
Figure A.5 - Installed WTPMS Device.....	109
Figure A.6 - WTPMS Common IDB Interface Settings.....	109
Figure B.1 - OBD-II Scan Tool Device Image.....	110
Figure B.2 - OBD-II User Interface.....	110
Figure B.3 - OBD-II Connection Diagram.....	110
Figure B.4 - Installed OBD-II Scan Tool Device.....	111
Figure B.5 - OBD-II Common IDB Interface Settings.....	111

## LIST OF EQUATIONS

Equation 4.1 - Mass Air Flow Based Fuel Consumption Equation.....	63
Equation 4.2 - Intake Manifold Pressure Based Fuel Consumption Equation .....	64
Equation 6.1 - The Z-Score.....	88

## LIST OF ACRYONYMS

AVL .....	Automatic Vehicle Location
AVLDS .....	Automatic Vehicle Location and Dispatch System
CAN .....	Controller Area Network
COM .....	Component Object Model
DTC.....	Diagnostic Trouble Code
ECU.....	Electronic Control Unit
GUI .....	Graphical User Interface
HMI.....	Human Machine Interface
IDB.....	Intelligent Device Bus
MAF .....	Mass Air Flow
MAP.....	Manifold Absolute Pressure
MFD.....	Multi-Function Display
MIL .....	Malfunction Indicator Lamp
NHDS.....	New Hampshire Department of Safety
O2.....	Oxygen (O2 Sensor)
OBD-II .....	On Board Diagnostics II
P25 .....	Project 25
P54 .....	Project54
PID .....	Parameter Identification
PTT .....	Push-To-Talk



RD&M .....	Remote Diagnostics and Maintenance
RFM .....	Remote Fleet Management
SUI .....	Speech User Interface
TPMS .....	Tire Pressure Monitoring System
WTPMS .....	Wireless Tire Pressure Monitoring System

# **ABSTRACT**

## **PROJECT54 VEHICLE TELEMATICS FOR REMOTE DIAGNOSTICS, FLEET MANAGEMENT AND TRAFFIC MONITORING**

by

Ian Cassias

University of New Hampshire, September, 2008

The Project54 system was developed to introduce advanced technologies into the operations of the New Hampshire Department of Safety and other law enforcement agencies. The application of computing, sensing and telecommunication technologies within the Project54 system enables advanced telematics services that can provide benefits to vehicle operators, fleet managers and the public. This thesis describes the implementation of remote diagnostics and fleet management services for the Project54 system and investigates the use of radar equipped police vehicles as traffic probes.

Aftermarket diagnostic hardware has been integrated in the Project54 system and software applications have been developed to control the hardware and record diagnostic information. An electronic data entry form has been created for tracking vehicle operating expenses and a vehicle status reporting system is described. Additionally, a traffic congestion scoring method using information from traffic radar units is presented.

# CHAPTER 1

## INTRODUCTION

The Project54 (P54) system was developed to introduce advanced technologies into the operations of the New Hampshire Department of Safety (NHDS) and other law enforcement agencies. The system features computer based control of networked vehicle electronics via speech, graphical and hardware user interfaces. Additionally, the P54 system integrates police cruisers into statewide data networks using wireless communications. The application of computing, sensing and telecommunication technologies in the automotive environment enables advanced telematics services such as remote diagnostics, fleet management and traffic monitoring. This thesis describes the implementation of remote diagnostics and fleet management services within the P54 system and investigates the use of radar equipped police vehicles as traffic probes.

### **1.1 Problem Description**

Currently there is information accessible in P54 equipped vehicles that is not being utilized in the operations of the NHDS. The proliferation of electronic devices in automobiles has created a wealth of sensor data about the interior and exterior vehicle context. The storage and processing capabilities of in-vehicle computers can be leveraged to provide interfaces for vehicle occupants to view and interact with this data. Combining vehicle sensing and computing resources with wireless communications enables

telematics services such as remote diagnostics, fleet management and traffic monitoring that can provide benefits to drivers, fleet managers and the public.

There are three problems which will be addressed by this thesis. The first problem is that NHDS personnel cannot easily assess the status and operating condition of their vehicles and the installed aftermarket equipment and may be unaware when conditions requiring maintenance exist. Unknowingly operating a malfunctioning vehicle can exacerbate the problem, leading to more expensive repairs and longer vehicle downtime. Additionally, operating a vehicle in need of maintenance may pose a safety concern for the driver and other motorists.

The second problem is that NHDS fleet managers cannot easily assess the operational status of their vehicle fleet. The maintenance and operating expense information for these vehicles is collected infrequently. The current system for collecting operating expenses uses hardcopy paper forms which must be manually transcribed into an electronic database. This introduces redundant, time consuming effort and the collected information is only updated on a monthly basis. Infrequent information updates can affect purchasing and scheduling decisions and may impact the readiness of the vehicle fleet.

The third and final problem addressed by this thesis is related to traffic congestion. Traffic congestion can increase the response time of NHDS first responders and other time-critical emergency services. Currently, the NHDS does not have a system to monitor or predict traffic congestion. As a result they cannot route their vehicles to avoid congestion. Traffic congestion causes increased wear and tear on vehicles. Additionally, congestion results in decreased productivity and increased pollution for all

motorists. Traffic congestion is a constantly growing problem caused by the ever-increasing number of vehicles on the road. This situation will likely worsen due to the limited addition of new capacity to the road network and the deterioration of existing infrastructure.

## **1.2 Thesis Objective**

The overarching objective of this thesis is to provide telematics services supporting the operations of the NHDS. There are three goals to accomplish this objective. The first goal is to provide a diagnostic service for P54 equipped vehicle operators to assess the status and operating condition of their vehicles. This service should provide early warning of needed repairs. By providing early notification, this service should help reduce vehicle downtime and maintenance expenses.

The second goal of this thesis is to provide a service for fleet managers to assess the status of the NHDS vehicle fleet and track vehicle maintenance and operating expenses. This service should provide frequent updates of vehicle status information and this information should be easily accessible. The information provided by this service should aid fleet managers in making purchasing and scheduling decisions.

The third goal of this thesis is to explore how data gathered from traffic radar units in P54 equipped vehicles could be used to monitor roadway traffic and provide a measure of traffic congestion. There are several existing technologies that can be used to evaluate traffic congestion, but a ubiquitous solution has yet to emerge. One way is to use vehicle mounted sensors such as GPS or video and to combine the results from multiple vehicles to produce a picture of traffic conditions for a given area. This thesis will

investigate a similar method with the unique application of in-vehicle traffic radar units to provide a measure of traffic congestion.

A traffic monitoring service would allow dispatchers to route NHDS vehicles around congestion and could provide traffic information to the public using a statewide data distribution system. The congestion information could then be used by in-vehicle navigation systems to route traffic around congested areas. This would result in decreased travel times, operating expenses and pollution for all drivers. Police officers and other emergency service personnel are often present or first to arrive when unexpected congestion occurs due to road maintenance or vehicle accidents. First responders could provide early notification of these events to the public.

### **1.3 Approach**

#### ***1.3.1 Vehicle Diagnostics Approach***

The proposed approach to implement a vehicle diagnostics service consists of a series of six steps encompassing the necessary hardware integration and software application development. The first step will be to identify diagnostic hardware offering serial or parallel data connections that can be integrated in the P54 system. Two diagnostic devices supporting RS-232 interfaces are proposed for this purpose: an On-Board Diagnostics II (OBD-II) scan tool and a Wireless Tire Pressure Monitoring System (WTPMS). The second step will be to connect these devices to the P54 Controller Area Network (CAN) using P54 Common Intelligent Device Bus (IDB) Interfaces. The third step will be to develop device-specific software modules to configure, monitor and control the hardware and make the diagnostic data available to other application modules within the P54 system. The fourth step will be to develop an application providing a user

interface for viewing and logging the diagnostic data on a PC and PDA. The fifth step will be to test and deploy this system in P54 equipped NHDS vehicles. The sixth and final proposed step is to collect the diagnostic data for use in the fleet management service.

### ***1.3.2 Fleet Management Approach***

We propose to implement a fleet management service with three components: an in-vehicle data collection system, vehicle-to-server communications and a central monitoring interface for observing the status of fleet vehicles and tracking their operational costs. Four steps are proposed to implement this service. The first step will be to collect pertinent vehicle information in a series of text files. This information will include the data provided by the vehicle diagnostics service and daily records of operating costs for gasoline and other common expenses collected using an electronic data entry form. The second step will be to transmit this information from the NHDS vehicles to a central server. The initial system will employ daily asynchronous updates using a combination of state operated 802.11 wireless hotspots [1] and P25 compliant digital data radios [2]. With future improvements in the coverage and bandwidth of digital radios, real-time updates of all collected data will be possible. The third step will be to present the collected vehicle data to fleet managers using a web interface. The fourth and final step will be to develop post-processing applications that can extract useful parameters such as fuel economy from the collected data.

### ***1.3.3 Traffic Monitoring Approach***

The third and final service proposed for this thesis is to explore the use of radar equipped NHDS vehicles to perform traffic congestion monitoring. The first task will be

to evaluate the feasibility of using radar equipped police vehicles as traffic probes. To do this, a series of tests will be performed to collect radar data for different roads and traffic conditions. Radar data will be collected from a stationary vehicle observing on-coming traffic for both highway and rural road segments. A count of the observed vehicles will be maintained during these tests. The second step will be to analyze the collected data to identify any relationships between the collected radar information and the observed traffic conditions. The third step will be to create a traffic congestion scoring method to assess traffic congestion using only the radar data. The final step will be to develop a P54 application for expanding traffic radar data collection and testing.

#### **1.4 Thesis Organization**

This first chapter has provided an introduction to the work carried out for this thesis including the problems addressed, the objective of the work and the proposed approach. The second chapter will provide a background overview and literature review of the field of vehicle telematics as it pertains to this thesis. Chapter three describes the implementation and functionality of the remote diagnostics service in the P54 system including the OBD-II and WTPMS integration and their related software applications. In chapter four the P54 fleet management implementation and functionality is described. The software and hardware described in this thesis was tested during development and successfully deployed in a trial NHDS vehicle. The testing and deployment effort is described in Chapter five. Chapter six details the traffic radar monitoring tests and the results of the investigation. Chapter seven provides conclusions of the works undertaken during this thesis. Finally, chapter eight provides suggestions for future work to expand and improve upon these efforts.



## **CHAPTER 2**

### **BACKGROUND**

Vehicle telematics is the use of computing, sensing and telecommunication technologies to provide services in an automotive environment. Vehicle telematics service categories include navigation, remote diagnostics, fleet management, safety, information access, context awareness and mobile commerce. Supporting these services requires unique hardware and software architectures. Additionally, issues such as privacy, data security and human factors design must be considered in the implementation of vehicle telematics services. The following sections summarize pertinent work and research in the field of vehicle telematics.

#### **2.1 Telematics Services**

##### ***2.1.1 Remote Diagnostics***

The transmission of vehicle sensor and diagnostic data enables routine diagnostics and maintenance operations without the need to schedule service. Remote diagnostics systems detect and report fault conditions and alert if unexpected maintenance is required. Additionally, continuous monitoring of vehicle components and operating conditions may lead to advanced prognostics services which further reduce downtime and maintenance costs [3]. A comprehensive overview of remote vehicle diagnostics is given by You *et al.* [4]. According to You *et al.*, current maintenance strategies can be classified as corrective or preventative in nature. Corrective maintenance addresses

faults after they occur and can lead to higher repair costs and reduced vehicle availability. Preventative maintenance involves replacing parts and fluids according to a fixed schedule based on the mileage, elapsed time and operating environment of the vehicle. It is often difficult to accurately determine the correct level of preventative maintenance for a given vehicle due to variation in the operating environment and usage levels. Preventative maintenance may result in increased cost of ownership due to the replacement of parts and fluids before the end of their useful service life [4].

You *et al.* also identify three major components for a Remote Diagnostics and Maintenance (RD&M) system. The first is an in-vehicle component capable of accepting downloads or upgrades and providing trouble codes, sensor values and maintenance information to a RD&M center. The second component consists of real-time diagnostic and maintenance modules in the vehicle and at the remote service center with appropriate human-machine interfaces. The final component is the RD&M service center which performs advanced diagnostics and maintenance routines and interacts with the driver [4].

The OBD-II network [5] is a fundamental enabling technology for remote diagnostics. Commercially available scan tools can connect to the OBD-II network and retrieve sensor and diagnostic data which can be collected by a computer and transmitted to a monitoring center for remote diagnostics. OBD-II consists of an in-vehicle data network of electronic control units (ECU) and vehicle sensors connected to a data bus. OBD-II monitors the powertrain, chassis, body and network of the vehicle and performs diagnostic routines on emissions related systems.

All vehicles manufactured for sale in the US since 1996 have been equipped with OBD-II. The intent of OBD-II is to regulate emissions by detecting faults that could lead

to increased vehicle pollution. When the OBD-II system detects a fault a DTC describing the nature of fault is stored in an ECU and the check engine light is illuminated to alert the driver of the error condition. Originally there were four different standards describing OBD-II data network implementations: SAE J1850, ISO 9141, ISO 14230 and ISO 15765. For all vehicles manufactured from the 2008 model year forward the only protocol used will be ISO 15765-4 which is based on the CAN standard operating at 500 kbps [6].

In addition to the sensor readings and diagnostic data available with OBD-II, wireless sensor networks can be deployed within the vehicle for monitoring components and creating aftermarket diagnostics systems [7,8]. Marcy *et al.* introduce the development of wireless integrated networked sensors for area monitoring and vehicle health management [7]. These distributed microsensors are self configuring and include on-board signal processing to reduce the amount of raw data that must be transmitted over the network resulting in increased battery life.

Polar *et al.* describe an in-vehicle Bluetooth piconet used to collect and transmit vehicle data for remote diagnostics [8]. The system consists of a master node and seven sensor equipped slave nodes. In order to evaluate the performance of the network, tests were performed for bit error rate, packet loss, data integrity and the time required for a lost node to reconnect to the network for different locations in the vehicle with the engine on and off. Their results suggest that the Bluetooth network is well suited for diagnostics in the automotive environment.

Remote diagnostics can lower operating costs and improve safety by detecting low tire pressure and alerting the driver of the situation [4]. Proper tire pressure is

important for the efficient operation and safety of an automobile as it improves fuel economy, increases tread life and reduces stopping distances [9]. For these reasons, the federal government has mandated that Tire Pressure Monitoring Systems (TPMS) be installed in all vehicles after September 1, 2008 [10]. Remotely monitoring the tire pressures for large vehicle fleets such as taxis, rental cars and public safety services would make them easier to maintain and could lower fuel costs and improve safety. The remote diagnostics service implemented for this thesis utilizes a commercial OBD-II scan tool and WTPMS to provide an overview of the vehicle operating condition and offers early notification of needed repairs. The service also provides the status of installed P54 system components.

This work expands upon prior work at P54 by S.Y. Kim. In [11] a prototype remote vehicle diagnostic system using an OBD-II scan tool was implemented and tested in laboratory conditions. This application supported a predefined set of sensors and the scan tool employed supported only one OBD-II protocol. For this thesis, a generic OBD-II scan tool was selected supporting all current OBD-II protocols. Additionally, the OBD-II software application developed supports all OBD-II sensors available in the vehicle as opposed to a fixed list. Interactive client and server side applications have been developed and the P54 component status has been included as recommended by S. Y. Kim.

### ***2.1.2 Fleet Management***

Fleet management services typically utilize vehicle location and diagnostics capabilities for the remote monitoring and management of vehicle assets. Fleet management is intended to improve the logistics, scheduling, reliability and availability

of services supported by vehicle fleets such as carrier trucking, car rental, taxi dispatch and public safety operations. Early research in advanced logistics services was stimulated by the creation of the Single European Market in 1993 [12]. There were three reasons for this: increased demand for transportation services, the need for higher quality services, such as just-in-time delivery and online tracking, and increased competition in the European transport industry [12]. Giannopoulos and Boulougaris identified the lack of infrastructure, the cost of implementation and the absence of standards as three challenges for the early development of advanced logistics services [12].

Advances in communications, computing and sensing technologies have led to the development of a number of commercial off the shelf telematics systems that can provide fleet management services. Goel and Gruhn propose a strategy for integrating such systems with existing carrier IT infrastructures to improve functionality and avoid the cost of overhauling existing systems to accommodate telematics [13]. They suggest that the vehicle information provided via a telematics system can supplement the existing IT infrastructure through the use of an intermediary application that is capable of communicating with both systems. They implemented such a system for a real carrier and found that it improved the quality of transportation services by reducing driver and dispatcher misunderstandings and improving the information flow but with the tradeoff of increased communication costs.

The dispatching operations for large vehicle fleet services such as taxis and rental cars can benefit from the adoption of fleet management services. One example of this can be found in the work of Ziqi Liao who proposed an Automatic Vehicle Location and Dispatch System (AVLDS) to overcome inefficiencies in traditional radio-paging

dispatch systems for taxis [14]. The AVLDS system combined GPS, wireless communication and computer aided dispatch technologies to automate taxi dispatch services in a large metropolitan environment. Using AVLDS a customer requests a taxi and an automated job request is sent to the nearest available vehicle. If the taxi accepts the fare, the ETA and taxi number are relayed back to the customer. In a study of these systems, the author found that they improved accuracy and efficiency when compared to traditional dispatch methods but that driver training was required to overcome initial resistance to their adoption.

Fleet management services often combine vehicle location data with performance and status information to improve logistics, maintenance and scheduling. One project at Mississippi State University [15] focused on the implementation of a system for remote vehicle location tracking as well as web based performance monitoring. This project utilized technologies including GSM/GPRS for data communication, GPS for localization and OBD-II diagnostics to continuously monitor the status of a bus fleet. The information collected by the system is presented to the public in real time using a web based interface.

The improved logistics capabilities offered by fleet management services have the potential to increase the availability of a vehicle fleet. In [16], Fass and Miller performed a simulation to predict the impact of an Autonomic Logistics System (ALS) on logistics efficiency and aircraft availability for the Joint Strike Fighter aircraft program. ALS is a proactive maintenance service for vehicles employing prognostics and health management systems. ALS is used to detect and isolate faults and automate the maintenance process before failures occur. Their simulation results indicate increased

availability when ALS is employed with a potential 8% improvement in the mission capable rate of aircraft.

A prototype Remote Fleet Management (RFM) application for police cruisers using the P54 system was proposed by Kim *et al.* [17]. The RFM system consisted of two modules: an Automatic Vehicle Location (AVL) module and a prototype remote diagnostics module for interfacing with the OBD-II network. The AVL module was intended to connect to commercial computer aided dispatch systems and provide them location updates using digital police radio data channels. The bandwidth limitations of the radio data channel required the use of heuristic rules to limit the number of location updates transmitted. One method used was to change the transmission rate depending on vehicle speed and distance traveled.

The P54 fleet management service described in this thesis expands upon the effort described in [17]. The information made available to fleet managers in this system includes the vehicle status and diagnostic data from the remote diagnostics service and a record of daily operating expenses for gas, oil changes and other routine maintenance expenses. Ongoing efforts at P54 will implement advanced in-vehicle navigation services and the location information they provide may supplement the dispatch operations of the NHDS using the fleet management service.

### ***2.1.3 Information Access***

Information Access services provide vehicle occupants access to information sources located outside the vehicle. One of the earliest examples is the now ubiquitous AM and FM radio receiver which allows vehicles to receive information and entertainment content. Such systems have evolved to include satellite radio and HD radio

[18]. Recent efforts at providing information access in automobiles have focused on creating a Transmission-Control Protocol/Internet Protocol (TCP/IP) connection enabling bi-directional communication of all types of data through a standardized interface [19].

Another area of information access involves providing information about vehicles to the public. Maclean and Dailey [20] describe a project called MyBus intended to provide real time information about bus arrival and departure times so that bus riders can make informed decisions about their travel options. A large fleet of busses are equipped with AVL systems which track their positions. This information is used by a predictor to estimate arrival and departure times and the results are made available to the public via a web interface which can be viewed from cell phones.

Access to accurate, complete and up to date information is critically important for supporting public safety operations. A number of efforts at P54 have addressed this need by expanding and improving information access for officers in the field. LeBlanc *et al.* describe a system for the collection and distribution of data using state operated WiFi hotspots [1]. This system is designed to address the issues of slow transfer speed and limited availability currently encountered using traditional analog and digital police radio systems. It enables temporary, high-speed wireless access to large amounts of data which can be stored locally in cruisers for later use. It also allows cars to quickly upload data to a central location for data collection and auditing. Using handheld computers and distributed computing components, officers can perform drivers license records checks and control in-car devices while outside their vehicles using the system described by Krstovski *et al.* [21]. Another information access technology being implemented at P54 involves the use of excess bandwidth available in the public television broadcast



spectrum to provide a high data rate connection to cruisers for delivering images, audio and text to officers in the field [22].

The P54 information access methods described above will provide the necessary communications pathways for supporting the services detailed in this thesis. The services will use a combination of in-vehicle digital data radios and state operated 802.11 wireless hotspots to transmit information. The digital data radios will be used for uploading brief reports including the vehicle mileage and status summary. The wireless hotspots will be used to upload detailed information including logs of measured vehicle sensor data. As the cost and performance of mobile data communications improves it will be possible to transmit all of the information in real-time.

#### ***2.1.4 Context Awareness and Traffic Monitoring***

Context aware vehicle telematics services are based on the collection and distribution of information regarding the internal and external vehicle environment. This information can include vehicle component states, traffic, weather and pollution conditions and information about nearby points of interest. Two examples of context aware vehicle platforms are given in [23,24].

In [23] a sentient car is presented which uses context information to produce adaptive pollution maps. Pollution is monitored using exhaust emissions sensors and the vehicle context is provided via multiple sensors including GPS and a connection to the vehicle ECU for speed, acceleration, temperature and steering wheel position data. A computer interfaces with the sensors and a GSM data modem is combined with an 802.11b network card to provide communications for transmitting pollution information and receiving updated map data. McCall *et al.* [24] describe an integrated vehicle test-bed

designed for studying driver behavior and developing algorithms for intelligent transportation systems. This test-bed provides a complete picture of the interior and exterior vehicle context through a combination of sensors including omnidirectional and rectilinear cameras, radar, microphones, GPS and vehicle state sensors.

Traffic information services for detecting, predicting and avoiding congestion are one of the most common context-based telematics services. Typically these systems employ a fixed network of sensors and communications equipment along a roadway to measure traffic and transmit congestion information. Sensors commonly used for detecting vehicles include inductive loops, video cameras, radars and lasers [25,26]. Fixed sensor networks for monitoring traffic can be expensive to deploy and provide only localized traffic conditions. For these reasons, recent research has investigated using vehicles as mobile traffic probes to detect and report traffic conditions [27-29].

Ishizaka *et al.* conducted a field test in which taxis were equipped with GPS and data logging equipment to record travel times for different road segments [29]. They determined that sampling the vehicle positions at five second intervals provided sufficient accuracy for estimating segment travel times. Using probe vehicles to detect real-time traffic conditions requires a significant communications infrastructure for collecting and processing the data produced by a large number of vehicles.

Chen *et al.* [28] addressed the issue of scaling such systems when a centralized server is used for data collection and path routing. They suggest inserting programmable middleboxes to act as intermediaries between vehicles and the server. The middleboxes collect and process traffic data from vehicles in their vicinity before passing the aggregated data to the central server. This allows a large number of vehicles to participate

without increasing the workload of the server. For their test setup the use of middleboxes reduced the amount of packets the server had to handle by a factor of ten.

Traffic monitoring systems often produce large volumes of multi-modal sensor data which must be analyzed in real time to provide continuous congestion information updates. Grossman *et al.* [30] describe a system which utilized real time data from over 830 traffic sensors combined with data about weather conditions and upcoming events that may impact traffic to detect changes in traffic conditions. A large set of historic traffic data was used to establish numerous baseline models. Changes in traffic conditions were detected by scoring current data against the baseline models. Real time updates were sent when deviations from the baselines were detected.

The system described in [30] is centralized with all traffic information collected and alerts sent from a single site. An alternative to this approach is to employ a distributed system as described in [25]. Here, Utamaphethai and Ghosh propose a network of distributed traffic management centers which would collect congestion information for highway segments and propagate their information to other nodes in the network using a flooding algorithm. The majority of the processing for routing and traffic avoidance would be handled by in-vehicle navigation systems which would receive updated congestion measures for highway segments whenever they are within communications range of a management center.

The traffic congestion monitoring investigated in this thesis utilizes post-processing of collected radar data. The intent is to establish baseline traffic conditions for a given location and to score current radar data collected during a fixed time interval against the baseline data to produce a congestion measure for that interval. Congestion

measures would be collected by officers and periodically transmitted to a central server. Congestion updates could then be provided to the public using datacasting [22] or some other form of wireless communication. Further testing will be needed to determine if radar data can be combined with GPS data to produce traffic updates for various radar configurations such as when the vehicle is mobile and when traffic is both closing and moving away from the observer.

## **2.2 Architecture**

### ***2.2.1 Hardware***

Vehicle telematics services are made possible by in-vehicle data networks, integrated sensors, powerful low-cost computers and ubiquitous wireless connectivity. A comprehensive survey of automotive sensors for powertrain, chassis and body systems is provided by W.J. Fleming in [31]. He also identifies a number of emerging state-of-the-art sensor technologies such as oil quality/deterioration sensing and multi-axis micromachined inertial sensors which may have direct applications in telematics systems. In addition to automotive sensor systems, telematics services make use of a variety of commercial sensors including GPS, radar, lasers, video cameras, microphones and inductive loops.

The telecommunications standards and technologies supporting vehicle telematics services include 802.11, WiMax, DSRC, cellular, Land Mobile Radio, Bluetooth, Satellite and infrared. Future systems will continue to expand coverage and availability of wireless connectivity. Cianca *et al.* show how high-altitude unmanned aerial platforms carrying communications relay payloads can supplement existing satellite services and provide efficient fleet management and traffic control services [32].

The evolution of the vehicle data network has been necessitated by the rapid expansion of electronic systems in the vehicle. Traditional point-to-point wiring methods for vehicle electronics were bulky and had a negative impact on performance and reliability. This led to the adoption of control networks for creating a shared infrastructure for device communication and control [33]. Leen and Heffernan provide an overview of recent advances in vehicle control networks and discuss how networking standards enable X-by-wire systems that replace traditional rigid mechanical components such as the steering column and hydraulic brakes with dynamically configurable electronic units performing the same functions [33].

P54 provides an example of how vehicle networking standards can be applied for the integration of aftermarket electronics [34]. The P54 system employs the IDB standard using version 2.0B of the CAN protocol to create a common interface for aftermarket police vehicle electronics such as lights, sirens, radios and radars. These devices typically provide a serial and/or parallel data connection for monitoring and controlling the device. The aftermarket devices are connected to the CAN network using the P54 common interface for the IDB. An embedded computer in the cruiser enables control of the networked devices via the P54 speech user interface (SUI) and touchscreen graphical user interfaces (GUIs).

For this thesis, an OBD-II scan tool and WTPMS was interfaced with the P54 IDB network. These devices support serial interfaces allowing direct connections to the IDB network. The use of standardized interfaces with the P54 system enables the integration of any device supporting serial data control. This means that new diagnostic hardware and services can be added to the system as they become available.

### ***2.2.2 Middleware***

Vehicle telematics middleware components manage communications resources and facilitate interaction between distributed components and applications. Middleware application requirements vary depending on the services they must support. Various middleware platforms are presented in [35-37]. Bisdikian *et al.* specify four requirements for a middleware platform supporting mobile commerce services: use of open and standard internet protocols, intelligent data filtering and abstraction, dynamic modification of set membership, and security and privacy management [35].

Reilly and Taleb-Bendiab select the Jini middleware to provide application services to remote in-vehicle computers and Palm devices [36]. This middleware architecture satisfied their requirements of suitability for wireless communication, fault tolerance and limited computing resource usage. A middleware platform based on international standards is discussed in [37]. This middleware architecture uses the Java based Open Service Gateway Initiative for device access, authentication and communications. The Automotive Multimedia Interface Collaboration is also employed to provide a uniform set of application programming interfaces that can be used to provide services in any vehicle.

There are two applications that function as middleware components in the P54 system: the Proxy Application [38] and the P25Proxy Application [2]. The Proxy Application allows P54 components to communication across distributed or remote computing platforms. Using the Proxy Application, data available on a local computer running P54 can be made available on any other computer running P54 so long as a network connection supporting UDP/IP packets exists between them. The ability to

communicate across distributed computing components enabled the development of a PDA version of the Vehicle Diagnostics application for this thesis. This application communicates with the in-vehicle computer via the Proxy Application to view and record the diagnostic and sensor data remotely. The P25Proxy Application provides a standardized interface for remote data transmission between P54 equipped cruisers and a central server when Project25 compatible digital radios are employed. These proxy applications are intended to render the underlying communications network invisible to P54 system components in order to facilitate data sharing across distributed components.

### ***2.2.3 Software***

The software supporting vehicle telematics must be flexible enough to accommodate many service types. Additionally it should be generic enough to be deployed on a variety of computing platforms. Karimi *et al.* describe a software architecture for supporting remote diagnostics and other services [39]. They identify a number of design principles for their application: distributed architecture, data replication, vehicle client data caching, carrier independency, filtering and resource-effectiveness.

The application of these design principles in [39] produced a layered software architecture composed of the four distinct components. The first component was a client application which allows users to interact with the system via a web interface. The second was a remote server which acted as a middleware application between the client application and the database server. The third component was the database server which stored and maintained all system data. The final component was a remote agent which provided the interface between the vehicle and the rest of the system [39].

Munson *et al.* introduce a rule-based programming framework for developing telematics applications based on a sense-and-respond model [40]. In this framework, a telematics event detection service monitors data within a telematics system and evaluates it against application defined rules. When a rule is triggered, the event is acted on accordingly by the application which defined the rule. Examples of event-driven telematics services supported by this framework include vehicle congestion detection, traffic monitoring and location based warnings.

P54 implemented a software architecture for controlling networked aftermarket electronics which supports a SUI and multiple GUIs [41,42]. The P54 system software is completely modular and is based on Microsoft's Component Object Model (COM). Application modules for interfacing new devices may be added to the system at any time. The P54 software messaging scheme allows one-to-one and one-to-many communication between applications. Support libraries for registry access, speech input/output, GUI interaction and IDB device communication provide a common framework for P54 application module development. These software components were used in the development of remote diagnostics and fleet management services for this thesis.

### **2.3 Implementation Issues**

The acceptance of vehicle telematics by the public depends on the ability of service providers to create trust by protecting user privacy. They must assure end-users that data collected in these systems will not be used for exploitative or malicious purposes. Service providers must also ensure their systems are secure physically and logically from tampering and misuse by end-users and third parties in order to provide reliable services. Additionally, these services often require user interfaces that are



accessible at high speeds and must be implemented without affecting vehicle reliability. The requirements for security, privacy, usability and reliability (SPUR) [43] as they relate to vehicle telematics are described by Giuli *et al.* in [44]. In addition to discussing the SPUR requirements, the authors introduce a service-oriented architecture providing standardized interfaces to vehicle data and in-vehicle services called the Vehicle Consumer Services Interface.

### ***2.3.1 Privacy and Data Security***

Issues regarding privacy and security in telematics systems are highlighted by the example application of probe vehicles for traffic monitoring in [27]. In this example the privacy of vehicle operators may be compromised because their movements can be tracked when they transmit location and identification data. If the data is transmitted anonymously, then the security of the system may be compromised by unauthenticated attackers. To provide for both security and privacy in the traffic monitoring system, Hoh *et al.* suggest an architecture where the authentication and the data analysis are handled by separate entities. Additionally they recommend the use of encryption, tamper-proof hardware and data sanitization techniques to ensure data integrity. The authors also investigate the possibility that data mining techniques known as inference attacks could be used to identify the homes and trace the movements of probe vehicle owners when anonymous location and speed data are available at the analysis entity. They recommend suppressing the amount of data collected by sampling at an interval of several minutes to reduce the effectiveness of data mining techniques [27].

Computational inference attack methods for home identification using location data and possible countermeasures were investigated by Krumm [45]. In this study,

anonymized GPS data was collected from 172 subjects over a two week period. Inference attack methods investigated included last destination, weighted median, largest cluster and best time attacks. The last destination algorithm performed best in terms of locating subject's home addresses. Countermeasures studied included: spatial cloaking in which data samples near the home location are deleted, adding Gaussian noise to location samples, and rounding whereby each location sample is snapped to its nearest location on a square grid. The analysis of these countermeasures shows that in all cases, significant data corruption is required to completely eliminate the threat of inference attacks.

Duri *et al.* at the IBM T.J. Watson Research Center have proposed a framework that focuses on data protection through user defined privacy policies and secure systems that enable data sharing in telematics [46,47]. In [47] they identify three key concepts implemented in their framework for the purpose of providing trust in telematics systems: defense-in-depth, data aggregation, and user defined privacy policies. Defense-in-depth is a comprehensive security measure where each hardware and software component in the system is as secure as possible from physical and logical attacks. Data aggregation is intended to minimize the amount of private data available outside of the trusted system. Two mechanisms are proposed to ensure proper operation of aggregation applications. First each data aggregation application is isolated from other applications and its access to systems resources such as files and sockets is restricted. Second, all local and network communication is managed by a data protection application which checks data against privacy policies and produces an audit trail. The final data protection framework component is the user defined privacy policy. This policy defines personal user data handling preferences and specifies how service providers will use collected data. Their

framework will insure compliance with the privacy policy by classifying data types and defining data handling rules in accordance with the policy.

Security concerns exist for the services proposed by this thesis. The wireless transmission of data from vehicles to a central server creates opportunities for attackers to compromise the system. For this reason, a system of encryption using private keys is employed in the wireless data collection infrastructure [1]. OpenSSL is used to encrypt all data transferred between servers and clients. Additionally 128-bit AES private key encryption can be applied to any sensitive data files transmitted wirelessly. The police officers using the system are able to choose when to send wireless updates and the wireless functions are only enabled during this time, helping to limit the ability of attackers to detect and exploit the connection.

### ***2.3.2 Human Factors Design***

The integration of telematics services in vehicles creates challenges in presenting the information provided by these services to drivers in a safe and efficient manner. In addition to the need for safe and efficient user interfaces, these services have the potential to increase the cognitive workload of drivers resulting in driver distraction [48]. Many projects have proposed flexible human-machine interfaces (HMI) and configurable multifunction displays (MFD) to address user interface issues [44,49,50]. In [49] Bernard Champoux proposes a user-configurable MFD that allows drivers to select how data is presented depending on whether they are driving in a city or on a highway. Drivers can create display presets that specify what information appears on the MFD as well as control the size and position of readouts. Drivers can toggle between display presets as needed using controls on the steering wheel.

A vehicular SUI allows drivers to interact with data sources and control in-car electronics without needing to take their hands off the wheel or eyes off the road. P54 has implemented a SUI and GUIs that work in parallel with traditional police hardware user interfaces to provide synchronized control and access to integrated electronic devices including lights and sirens, radars, and radios. In [51] Kun *et al.* evaluated officer use of the SUI, GUI and traditional hardware user interfaces while on patrol. They found that officers tended to use different interfaces to perform the same task depending on their situations.

The selection of an appropriate user interface depended on the balance of safety against the speed of the interaction. For example most record queries can be performed faster using the SUI versus the traditional interface and therefore the SUI was preferred for this task. Reviewing the data presented by the record queries is performed fastest using the GUI. However, this interaction is unsafe while driving and only one officer was found to use it. They also determined that the design of the interface should match the way it is used in the field. They noticed that officers consistently declined to listen to verbal feedback responses from the SUI designed to guide the interaction. Additionally they found that training officers in the operation and use of the SUI can improve SUI performance [51].

To assess the impact of a SUI on driving performance two experiments were conducted at P54 by Kun *et al.* [52,53]. In [52], a driving simulator was used to create a scenario where participants had to change channels on a police radio using a SUI while driving. Three aspects of the SUI were varied in order to observe their impact on driving performance: SUI accuracy, the use of a push-to-talk (PTT) button, and the type of dialog

repair employed. The results of the study indicate that the accuracy of the speech engine and its interaction with the use of the push-to-talk button does impact driving performance significantly, but the type of dialog repair employed does not. In [53] driving performance using the SUI to change police radio channels was compared with the performance using a visual/manual radio interface. The results of this study indicate that the use of a SUI provides a significant improvement in driving performance when compared to the traditional manual/visual interface for performing the same task.

The applications developed for this thesis provide vehicle operators the ability to monitor individual vehicle sensors, read tire pressures and observe any faults detected by the OBD-II system. This data is presented to NHDS personnel via a touchscreen based GUI. The information provided by these applications is intended to be viewed only while the vehicle is stationary. The GUI has not been designed for usability while driving because this is not how the system is intended to function. While driving, the diagnostics applications can run in the background collecting data that can then be analyzed in post-processing. The time-stamped diagnostic data will be stored in the vehicle and uploaded to a central server where it can be viewed by fleet managers. If a malfunction occurs, the driver can pull over and use the Vehicle Diagnostics application to assess the nature of the fault and then perform any needed maintenance.

## **CHAPTER 3**

### **PROJECT54 VEHICLE DIAGNOSTICS**

The implementation of a vehicle diagnostics service in the P54 system included hardware integration and software application development. The hardware integration effort involved interfacing an OBD-II scan tool and WTPMS with the P54 IDB network. Device-specific software applications were developed for configuring, monitoring and controlling the hardware and making the diagnostic data available to other applications within the P54 system. A Vehicle Diagnostics application was developed providing a user interface for viewing and logging diagnostic data on both a PC and PDA. These applications were deployed and tested in three trial vehicles. The information provided by these applications can be updated daily using Project 25 compliant digital data radios [2] and planned state operated WiFi hotspots [1]. The following sections describe the implementation and functionality of the P54 vehicle diagnostics service developed for this thesis.

#### **3.1 Diagnostic Hardware**

The first step to implement a diagnostics service in the P54 system was identifying diagnostic hardware supporting either serial or parallel data interfaces. This hardware needed to provide methods for configuration, control and access to diagnostic data. There is a great deal of diagnostic and status information stored in vehicle ECUs and accessible using the vehicle data bus. Unfortunately, most of this information is

proprietary to vehicle manufactures and access methods are not available to the public.

OBD-II provides a standardized interface for accessing a mandated subset of the available diagnostic data in a vehicle. OBD-II supports continuous monitoring of vehicle parameters using networked sensors and logging of fault conditions using Diagnostic Trouble Codes (DTCs) stored in vehicle ECUs. Each DTC corresponds to a pre-defined message describing the location and nature a fault. A commercial OBD-II scan tool supporting RS-232 communication was selected to provide access to this data.

Maintaining proper tire pressure can improve fuel economy, increase tread life and reduce stopping distances [9]. The passage of the TREAD Act [10] makes tire pressure monitoring systems mandatory for all vehicles manufactured after September 1, 2008. The format and methods for accessing tire pressure information in these systems are proprietary to vehicle manufactures and can vary by model and year. To implement a generic TPMS solution in the P54 framework, a commercial aftermarket WTPMS system supporting RS-232 data output was employed.

### ***3.1.1 WTPMS Hardware Description***

The selected WTPMS features two distinct advantages that made it particularly appropriate for this implementation: wireless valve stem cap pressure sensors and RS-232 data output. Valve stem cap pressure sensors are easier to install and maintain than sensors mounted inside the tire or on the rim and provide individual pressure readings for each tire. The ability to access tire pressure information via an RS-232 data feed means that this information can be made available within the P54 system by interfacing the monitor with the IDB network. The information available via RS-232 includes tire pressure, status/condition, matched sensor location and low battery indicator status.

The battery powered valve stem pressure sensors periodically transmit pressure information using RF signals sent to a receiving monitor featuring LED pressure readouts. During normal operation each sensor will provide a pressure update approximately once every five minutes. When low pressure conditions are detected the sensors will transmit alert messages once every 14 seconds. Under normal operation the sensor batteries are expected to last approximately three years.

The monitor produces auditory and visual alerts when tire pressure drops by 12.5% and 25% below the baseline normal pressure. When a tire is 12.5% below the baseline pressure, the monitor will flash and beep once every second. For 25% below baseline pressure conditions, the monitor will flash and beep twice every second. The baseline pressure for a sensor is automatically set to the pressure of the tire at the time the sensor is first installed.

### ***3.1.2 OBD-II Scan Tool Description***

Commercial OBD-II scan tools provide access to vehicle diagnostic services in conformance with the SAE J1978 OBD-II Scan Tool standard. These devices support standardized interfaces for communicating with the vehicle OBD-II network and performing the diagnostic services defined in SAE J1979/ISO 15031-5 [54,55]. There are numerous OBD-II network layer implementations that vehicle manufactures have employed in the past. For all vehicles manufactured from the 2008 model year forward the only protocol used will be ISO 15765-4 which is based on the CAN standard operating at 500 Kbps [6]. To implement a generic OBD-II solution in the P54 system a scan tool was selected which supports all currently defined OBD-II network protocols. A summary of the supported OBD-II protocols is provided in Table 3.1.



Supported Protocol	Band Rate	Comments
SAE J1850 PWM	41.6 Kbaud	Ford
SAE J1850 VPW	10.4 Kbaud	General Motors
ISO 9141-2	5 baud init, 10.4 Kbaud	Chrysler, European and Asian manufacturers
ISO 14230-4	5 baud init, 10.4 Kbaud	Physical layer identical to ISO 9141-2
ISO 14230-4	fast init, 10.4 Kbaud	
ISO 15765-4 (11 bit ID)	250 Kbaud	CAN standard, all US vehicles from 2008 on will use ISO 15765-4 at 500 Kbaud
ISO 15765-4 (29 bit ID)	250 Kbaud	
ISO 15765-4 (11 bit ID)	500 Kbaud	
ISO 15765-4 (29 bit ID)	500 Kbaud	

**Table 3.1 - Scan Tool Supported OBD-II Protocols**

This scan tool functions as a protocol interpreter for communicating between an RS-232 and an OBD-II interface. The scan tool receives bytes of hexadecimal data representing ASCII character strings from the RS-232 interface. There are two types of commands supported by the scan tool: internal configuration requests and OBD bus commands. All scan tool configuration requests begin with the characters “AT”. Any commands that do not begin with “AT” are treated as OBD messages and will be combined with appropriate header and checksum bytes before being transmitted to the vehicle OBD-II bus. Please see [56] for further details of the supported “AT” commands and the OBD messaging format. Since the scan tool is only a protocol interpreter, a software application implementing the appropriate OBD-II communications protocols is required to support the diagnostics services detailed in the SAE J1979/ISO 15031-5 standard. These services include querying vehicle OBD-II sensors and retrieving and clearing the list of stored DTCs.

### ***3.1.3 Diagnostic Hardware Integration***

Once the appropriate diagnostic hardware was identified, the next step was to integrate this hardware in the P54 system by connecting the devices to the P54 IDB network. For both devices, the only hardware required to connect to the IDB network is a

null modem cable and a P54 Common IDB Interface [34]. The null modem cable is only needed if older Version 4 IDB boxes are used and should be omitted when using Version 5 IDB Boxes. Please see APPENDIX A and APPENDIX B for detailed installation instructions for the WTPMS and OBD-II scan tool respectively. See the P54 Hardware Manual for instructions on setting up the IDB network [57].

### **3.2 Diagnostic Software**

The modular COM software architecture of P54 simplifies the integration and control of new devices within the system. For each device connected to the P54 IDB network there is a corresponding software module. These software modules are responsible for P54 communication and control of the device. Software modules are capable of one-to-one and one-to-many communication with other modules using text based inter-application messaging. Individual software modules can optionally include a GUI and SUI. Additionally, a client-server feedback mechanism exists that enables synchronized automated communication of hardware and application status information between client and server applications. Using this feedback mechanism, a single client application can provide a consolidated user interface for synchronized interaction with multiple hardware devices each controlled by a separate server module.

The third proposed step to implement a vehicle diagnostics service for this thesis was to develop device-specific software modules for the diagnostic hardware. These applications are responsible for configuring, monitoring and controlling the WTPMS and OBD-II scan tool and making their data available to other modules within the P54 system. These applications act as servers providing diagnostic data to client applications via feedback messaging. These server applications do not include user interfaces and are

solely responsible for the serial communication between the device and the IDB network. The OBD-II and Tire Pressure applications provide feedback messaging interfaces enabling client applications to access diagnostic data and control the hardware using P54 inter-application messages.

### 3.2.1 Tire Pressure Application

The P54 Tire Pressure application module, *TirePressure.dll*, acts as a data server providing P54 applications access to the WTPMS data. The Tire Pressure application uses the underlying P54 IDB COM component, *IdbCom.dll*, to establish a serial connection with the WTPMS over the IDB network. The serial connection is first opened and initialized using the connection settings shown in Table 3.2.

Device Address	Baud Rate	Handshaking
1E	38,400	Enabled

**Table 3.2 - Tire Pressure IDB Connection Settings**

Once the serial connection is established, the Tire Pressure application will wait to receive data from the monitor in a continual loop until the application is shut down. This application only reads data from the device because the WTPMS hardware does not provide any control functionality via the serial connection. When the tire pressure monitor receives a sensor update the information is read by the Tire Pressure application over the network connection. Sensor updates are provided in the form of an eight byte data packet (the structure of the packet is proprietary). The Tire Pressure application parses this data packet and uses the information to update a structure containing the status of each tire. P54 applications can request to receive or stop feedback from the Tire Pressure application at any time by sending one of the feedback request messages shown in Figure 3.1.

```

To Receive Feedback:
Message(L"ClientAppName", L"TirePressure", L"ClientAppName", L"FEEDBACK ON");

To Stop Feedback:
Message(L"ClientAppName", L"TirePressure", L"ClientAppName", L"FEEDBACK OFF");

```

**Figure 3.1 - Tire Pressure Client Feedback Request Messages**

As new tire pressure information is received it will be parsed by the Tire Pressure application and reported to all clients using a predefined wide character tire status message. The format of the Tire Status Message is shown in Figure 3.2.

```

L"STATUS <location>,<pressure>,<condition>,<battery>"

```

**Figure 3.2 - Tire Status Message Format**

Each parameter <param> in the tire status feedback message represents an integer value that must be interpreted by the client application. The range of values for these parameters and their corresponding interpretation is given in Table 3.3.

Parameter	Values	Interpretation
<location>	0	Front-Left Tire
	1	Front-Right Tire
	2	Rear-Left Tire
	3	Rear-Right Tire
<pressure>	10-150	Tire Pressure in psi
<condition>	1	Regular Update: Tire OK
	2	12.5% <= Pressure Loss < 25%
	3	Pressure Loss >= 25%
	4	New Baseline Pressure Set
	5	Hall Effect Trigger
<battery>	0	The battery is not low
	1	The battery is low

**Table 3.3 - Tire Status Message Parameters**

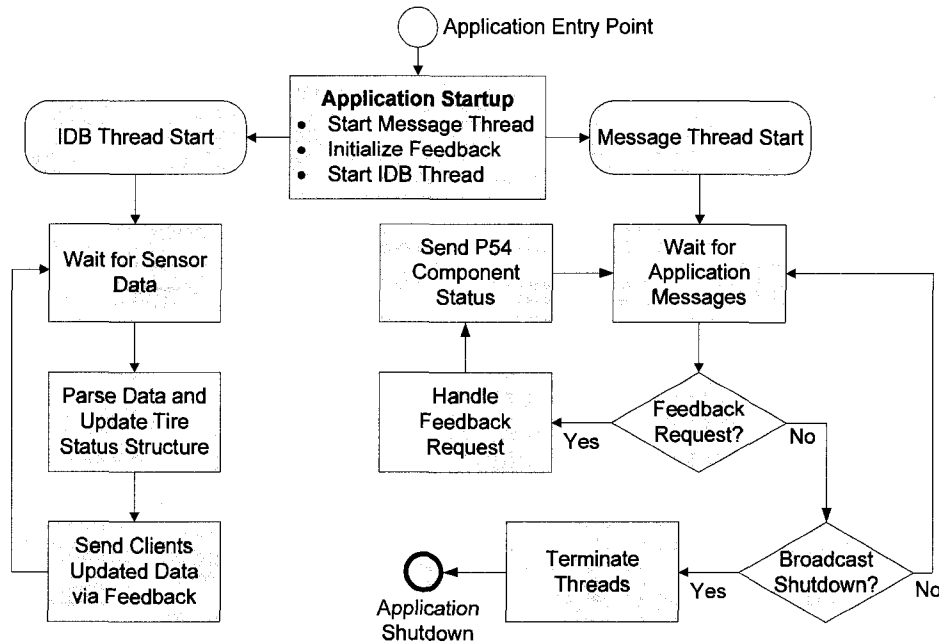
The Tire Pressure application provides clients a status message containing a list of any errors reported by P54 applications. These errors typically indicate a lack of communication with a device over the IDB network. The format of this message is shown

in Figure 3.3. This message contains an integer count of the number of errors, <n>, followed by comma delimited descriptions of each error.

**L“STATUS P54 <n>,description1,description2,...,description<n>”**

**Figure 3.3 - Project54 Component Status Message Format**

The purpose of this message is to provide an interface for distributed clients such as PDAs to receive the component status of a local P54 installation. This information is stored in the registry of the P54 host computer and is not otherwise accessible by remote clients. A flowchart of the Tire Pressure application is provided in Figure 3.4.



**Figure 3.4 - Tire Pressure Application Flowchart**

The Tire Pressure application also features a logging component that stores the most recent tire pressure status in the registry. This data is combined with OBD-II and P54 system component status information and used to generate daily vehicle status reports. These reports are intended for the Fleet Management service and will be described in greater detail in Chapter 4.

### 3.2.2 OBD-II Application

The P54 OBD-II application module, *OBD-II.dll*, is the data server providing interfaces for P54 client applications to access information on the OBD-II bus. The OBD-II application consists of an IDB thread which handles all serial communication functions and a message thread which processes P54 inter-application messages. The flowchart for the IDB thread is shown in Figure 3.5.

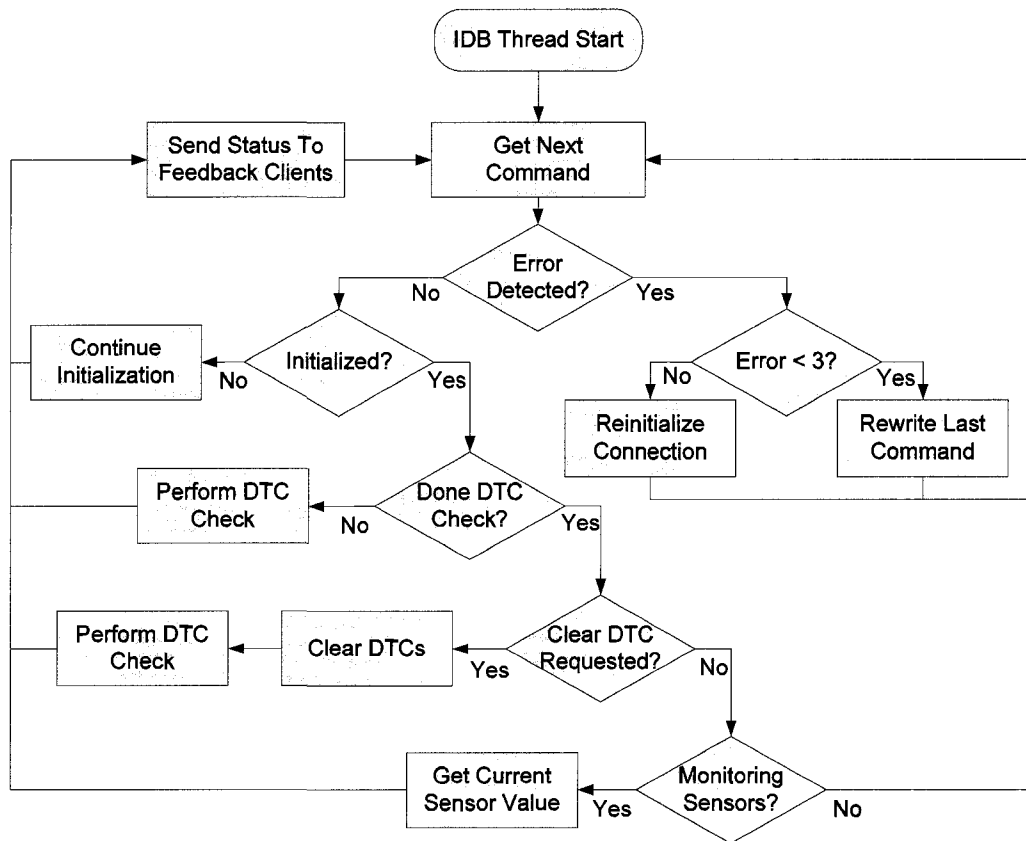


Figure 3.5 - OBD-II IDB Thread Flowchart

The IDB Thread operates in a continual loop checking to see which command should be written and processing the resulting response when appropriate. This loop will continue until the application is terminated. The information gathered in the IDB thread can be provided to clients using feedback messages. Client applications can request to

receive or stop feedback from the OBD-II application at any time by sending one of the feedback request messages in Figure 3.6.

**To Receive Feedback:**  
**Message(L“ClientAppName”, L“OBDII”, L“ClientAppName”, L“FEEDBACK ON”);**

**To Stop Feedback:**  
**Message(L“ClientAppName”, L“OBDII”, L“ClientAppName”, L“FEEDBACK OFF”);**

Figure 3.6 - OBD-II Client Feedback Request Messages

Clients can receive OBD-II status information and issue commands to activate functions of the OBD-II application using messaging interfaces. These requests are processed by the OBD-II message thread shown in Figure 3.7.

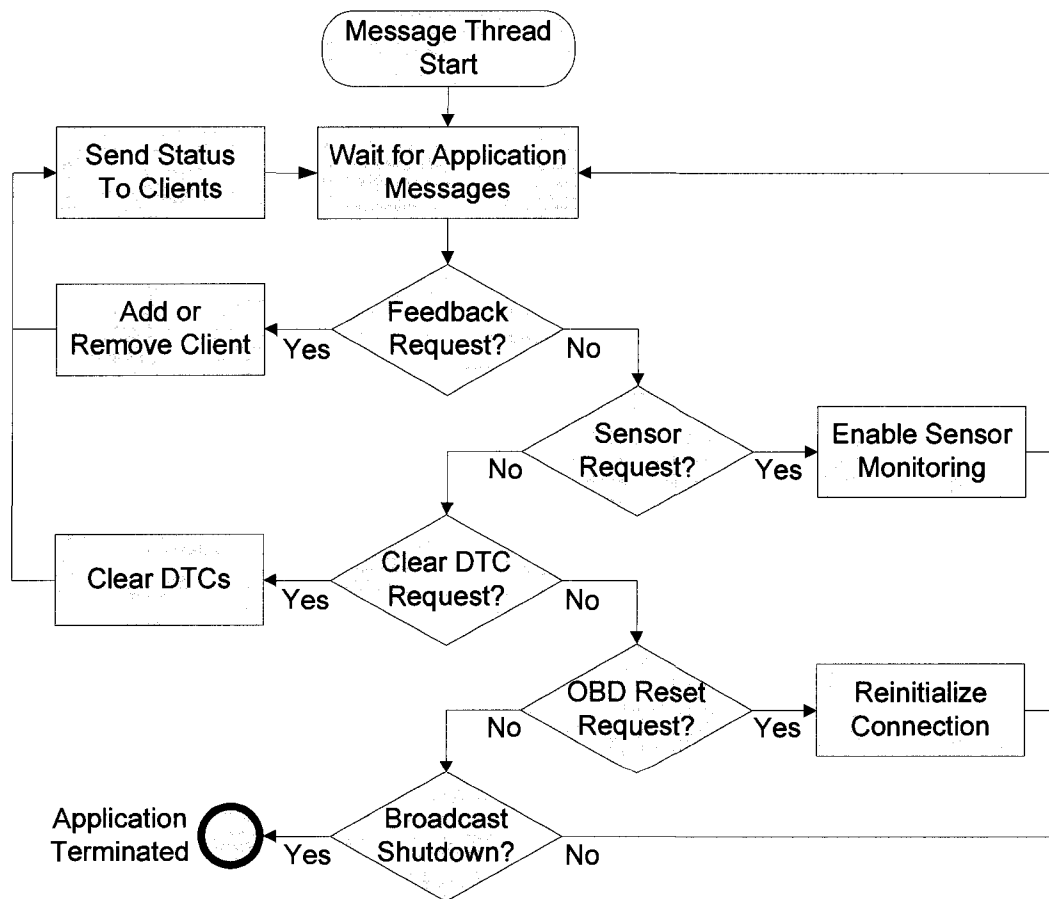


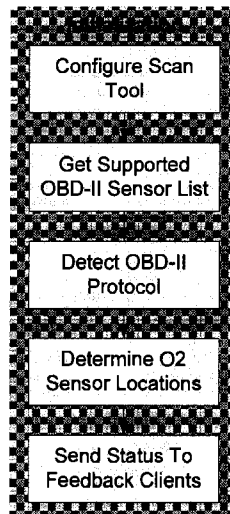
Figure 3.7 - OBD-II Message Thread Flowchart

The OBD-II IDB thread establishes a serial connection with the scan tool device over the P54 IDB network using the settings shown in Table 3.4. The serial commands issued by the OBD-II application are carriage-return terminated character strings. These can be either the “AT” commands used to change the scan tool device settings or OBD command request messages. The scan tool will automatically add appropriate header and checksum bytes to any OBD commands.

Device Address	Baud Rate	Handshaking
1C	9,600	Disabled

**Table 3.4 - OBD-II IDB Connection Settings**

The OBD-II connection is initialized when the IDB thread starts. The steps of the OBD-II initialization process are outlined in Figure 3.8.



**Figure 3.8 - OBD-II IDB Thread Initialization Process**

First a series of “AT” commands are issued to reset the scan tool and configure its operation. The vehicle OBD-II bus is then queried to determine which sensors are available. There are a total of 93 different sensors supported by the P54 OBD-II application. The complete list of P54 supported sensors and their corresponding indices can be found in APPENDIX C. Most vehicles only provide a limited subset of these



sensors and this will vary depending on the make, model and year of the vehicle. A message containing the comma delimited list of vehicle-supported sensor indices is provided to feedback clients so that they will know which sensors are available for monitoring. The format for this message is shown in Figure 3.9.

**L“STATUS SUPPORTED PIDS <index>,<index>,…,<index>”**

**Figure 3.9 - Available Vehicle Sensor Status Message Format**

In Figure 3.9 and other messages, PID stands for Parameter Identification and is the abbreviation used to refer to sensor parameters within the OBD-II standard. The <index> in OBD-II messages refers to the index of a sensor as given in APPENDIX C.

For the next stage in the initialization process, the vehicle OBD-II protocol is detected and the O2 sensor location labels are determined. All OBD-II equipped vehicles employ one of two label formats for describing the locations of O2 sensors: using up to two banks of cylinders, each supporting up to four sensors, or using up to four banks of cylinders, each supporting up to two sensors. The labels describe the location of the sensor in relation to the engine cylinders. The physical interpretation of the location labels can vary by manufacturer and by the make, model and year of the vehicle. The format for the O2 location definition message is shown in Figure 3.10.

**L“STATUS O2 13” or L“STATUS O2 1D”**

**Figure 3.10 - O2 Location Definition Status Message Format**

The O2 location definition is provided to client applications so that they can assign the correct labels to O2 sensor readings. The correct sensor location labels for each definition are shown in Table 3.5.

O2 Sensor Indices (SEE APPENDIX)	Location Label	
	L"STATUS O2 I3" Definition	L"STATUS O2 I4" Definition
21, 35, 36, 59, 60	Bank 1 Sensor 1	Bank 1 Sensor 1
22, 37, 38, 61, 62	Bank 1 Sensor 2	Bank 1 Sensor 2
23, 39, 40, 63, 64	Bank 1 Sensor 3	Bank 2 Sensor 1
24, 41, 42, 65, 66	Bank 1 Sensor 4	Bank 2 Sensor 2
25, 43, 44, 67, 68	Bank 2 Sensor 1	Bank 3 Sensor 1
26, 45, 46, 69, 70	Bank 2 Sensor 2	Bank 3 Sensor 2
27, 47, 48, 71, 72	Bank 2 Sensor 3	Bank 4 Sensor 1
28, 49, 50, 73, 74	Bank 2 Sensor 4	Bank 4 Sensor 2

**Table 3.5 - O2 Sensor Location Labels**

A detailed description of all initialization steps is given in Table 3.6.

Initialization Stage	Step	Command	Description
Configure Scan Tool	1	"ATWS\r"	Reset OBD Interface
	2	"ATE0\r"	Turn Echo Off
	3	"ATSP0\r"	Set Automatic Protocol Detection
Get Supported OBD-II Sensor List	4	"0100\r"	Check Supported Sensors [0-31]
	5	"0120\r"	Check Supported Sensors [32-78]
	6	"0140\r"	Check Supported Sensors [79-92]
Detect OBD-II Protocol	7	"ATDPN\r"	Determine OBD Protocol
Determine O2 Locations	8	"0113\r" or "011D\r"	Get O2 Sensor Locations

**Table 3.6 - OBD-II Initialization Steps**

If the OBD connection becomes unresponsive a client application may request to re-initialize the connection by sending the Reset Request Message shown in Figure 3.11. This will cause the OBD-II application to begin the initialization process at Step 1 and continue through Step 8.

```
Message(L"ClientAppName", L"OBDII", L"ClientAppName", L"RESET");
```

**Figure 3.11 - OBD-II Client Reset Request Message**

The OBD-II IDB thread handles the serial communication for implementing diagnostic services defined in the SAE J1979/ISO 15031-5 standard. These services include monitoring current OBD-II sensor values, retrieving stored DTCs and clearing

DTCs and emissions related vehicle data. DTCs provide indications of faults detected in the body, chassis, powertrain and network vehicle systems. There are 2025 standardized DTC definitions supported by the P54 OBD-II application. The DTC format is defined in the SAE J2012/ISO 15031-6 Diagnostic Trouble Code Definition standard. A DTC consists of a three digit numeric code preceded by a two-character alphanumeric designator. Table 3.7 provides a summary of the DTC formats for standardized and manufacturer defined trouble codes.

Code Format and Range	Description
B0XXX	SAE J2012/ISO 15031-6 Body Trouble Codes
B1XXX - B3XXX	Manufacturer Body Trouble Codes
C0XXX	SAE J2012/ISO 15031-6 Chassis Trouble Codes
C1XXX - C3XXX	Manufacturer Chassis Trouble Codes
P0XXX	SAE J2012/ISO 15031-6 Powertrain Trouble Codes
P1XXX - P3XXX	Manufacturer Powertrain Trouble Codes
U0XXX	SAE J2012/ISO 15031-6 Network Trouble Codes
U1XXX - U3XXX	Manufacturer Network Trouble Codes

**Table 3.7 - Standardized and Manufacturer Defined DTC Formats**

When a DTC check is performed, either at initialization or after a client-requested reset, the information is communicated to clients using the DTC Status Message as shown in Figure 3.12.

**L“STATUS DTC <n>,DTC1,description1,...,DTC<n>,description<n>”**

**Figure 3.12 - OBD-II DTC Status Message Format**

The parameter <n> represents an integer count of the number of stored DTCs. This count is followed by a comma delimited list of each DTC followed by its description. Descriptions are only available for the DTCs defined in the SAE J2012/ISO 15031-6 standard. For manufacturer defined trouble codes the description will be given as “UNKNOWN”. Client applications may send a message requesting the OBD-II

application to erase stored DTCs and clear all emissions-related diagnostic data from the vehicle. This is accomplished by sending the DTC clear request shown in Figure 3.13.

```
Message(L"ClientAppName", L"OBDII", L"ClientAppName", L"CLEAR DTC");
```

**Figure 3.13 - OBD-II Client DTC Clear Request Message**

To clear the DTCs the vehicle ignition key must be in the "ON" position and the vehicle engine must be off. When the OBD-II application attempts to clear the DTCs a status message will be sent to client applications indicating whether the clear attempt was successful. The format for this message is shown in Figure 3.14.

```
Success:      L"STATUS DTC CLEAR OK"  
Failure:     L"STATUS DTC CLEAR ERROR"
```

**Figure 3.14 - Clear DTC Status Message Format**

In addition to the DTC check, the OBD-II application provides an interface for client applications to request continuous updates of vehicle sensor values. To request sensor monitoring, clients send the OBD-II application a message containing a list of sensor indices. The format of the Client Sensor Request Message is shown in Figure 3.15

```
Message(L"ClientAppName", L"OBDII", L"ClientAppName", L"MONITOR PIDS <index>, ..., <index>");
```

**Figure 3.15 - OBD-II Client Sensor Request Message**

The indices <index> in the list correspond to those shown in APPENDIX C. The OBD-II application will combine the list of requested sensors from each client into a single monitoring list and remove any duplicate requests. The sensors will then be sequentially sampled in a continual loop. The elapsed time between sensor readings is approximately 250-350ms. When new sensor values are detected, client applications will be provided with the information using the Sensor Update Status Message shown in Figure 3.16.

**L“STATUS PID <index>,Label,Value”**

**Figure 3.16 - OBD-II Sensor Update Status Message Format**

The OBD-II application does not filter sensor update messages to the individual requesting clients. All updates will be provided to all clients regardless of whether the client requested the sensor. Sensor reading should be filtered at the client end by checking the <index> so that only requested values are processed. This ensures a fast and uniform update cycle for all requested sensors by avoiding potential duplicate queries. As more sensors are requested, the refresh rate of individual sensors will decrease.

The OBD-II application provides clients with the P54 Component Status using the message shown in Figure 3.3. This application also supports a logging component that stores the DTC status in registry. This information is then combined with the Tire Pressure and P54 component status to produce status logs for the Fleet Management service. A summary of the messaging interfaces supported by the OBD-II application is provided in Table 3.8 and Table 3.9.

Message	Description
L“STATUS SUPPORTED PIDS <index>,<index>,...,<index>”	List of available sensors
L“STATUS O2 13” or L“STATUS O2 1D”	The O2 sensor definition
L“STATUS DTC <n>,DTC1,description1,...,DTC<n>,description<n>”	The count and list of DTCs
L“STATUS DTC CLEAR OK”	DTCs were successfully cleared
L“STATUS DTC CLEAR ERROR”	DTC were not cleared
L“STATUS PID <index>,Label,Value”	Updated sensor data
L“STATUS P54 <n>,description1,description2,...,description<n>”	P54 component errors

**Table 3.8 - Messaging Interfaces from OBD-II Application to Clients**

Message	Description
L“FEEDBACK ON”	Receive feedback from OBD-II application
L“FEEDBACK OFF”	Stop feedback from OBD-II application
L“RESET”	Re-initialize the OBD-II connection
L“CLEAR DTC”	Clear stored DTCs and emissions data
L“MONITOR PIDS <index>,<index>,...,<index>”	Monitor the sensors at the listed indices

**Table 3.9 - Messaging Interfaces from Clients to OBD-II Application**

### 3.2.3 Vehicle Diagnostics Application

The fourth proposed step to implement a diagnostics service in the P54 system was to create an application providing a user interface for viewing and logging diagnostic data. The P54 Vehicle Diagnostics application, *VehicleDiagnostics.dll*, is a client application employing a SUI and GUI to provide a consolidated interface for viewing and requesting diagnostic information from the Tire Pressure and OBD-II server applications. A PDA version of the Vehicle Diagnostics application, *Diagnostics.dll*, was developed to provide remote access to diagnostic data. The PC application screenshot is shown in Figure 3.17 and the PDA version is shown in Figure 3.18.

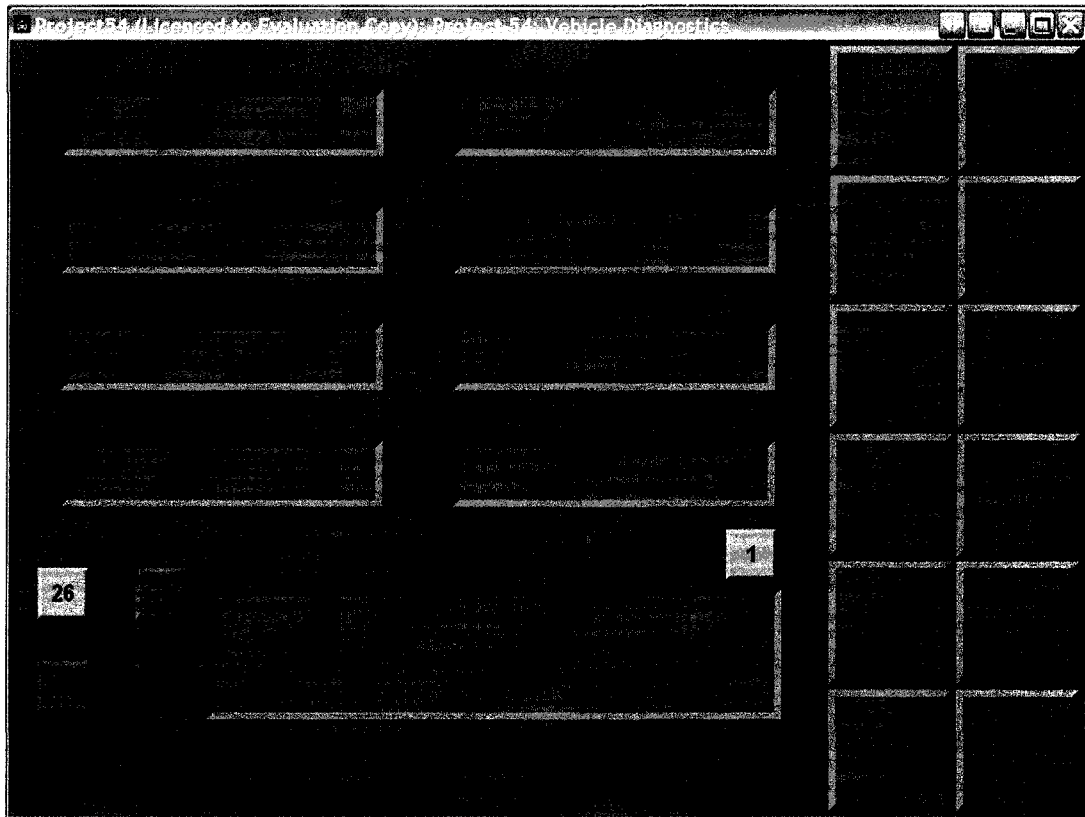
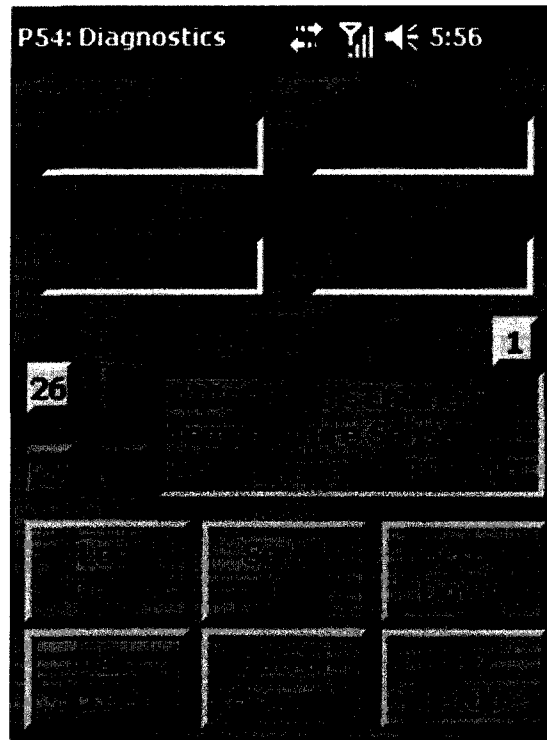










Figure 3.17 - Project54 Vehicle Diagnostics Application Screenshot



**Figure 3.18 - Project54 PDA Diagnostics Application Screenshot**

The PC and PDA versions of the Vehicle Diagnostics application implement identical functionality. The difference between them is that the PDA Diagnostics application communicates with the Tire Pressure and OBD-II applications via the P54 Proxy middleware application [38] over a wireless ad-hoc network. These applications sign up to receive feedback from the Tire Pressure and OBD-II server applications by sending the “FEEDBACK ON” messages shown in Figure 3.1 and Figure 3.6. When the Vehicle Diagnostics application receives new status messages from the Tire Pressure and OBD-II applications the data is parsed and the GUI is updated to show the vehicle condition. Users can interact with the Vehicle Diagnostics application by pressing the GUI buttons or by using speech commands corresponding to button labels. The speech enabled GUI button functions of the Vehicle Diagnostics application screen are described in Table 3.10.

GUI Button	Function	GUI Button	Function
	Switch to the Main Screen Application		Log all diagnostic data
	Switch to the Patrol Screen Application		Scroll up the status message text area
	Go to the OBD Setup Screen		Scroll down the status message text area
	Switch to the Maintenance Reports Application		Hide the Vehicle Diagnostics Button on the Mainscreen at next startup

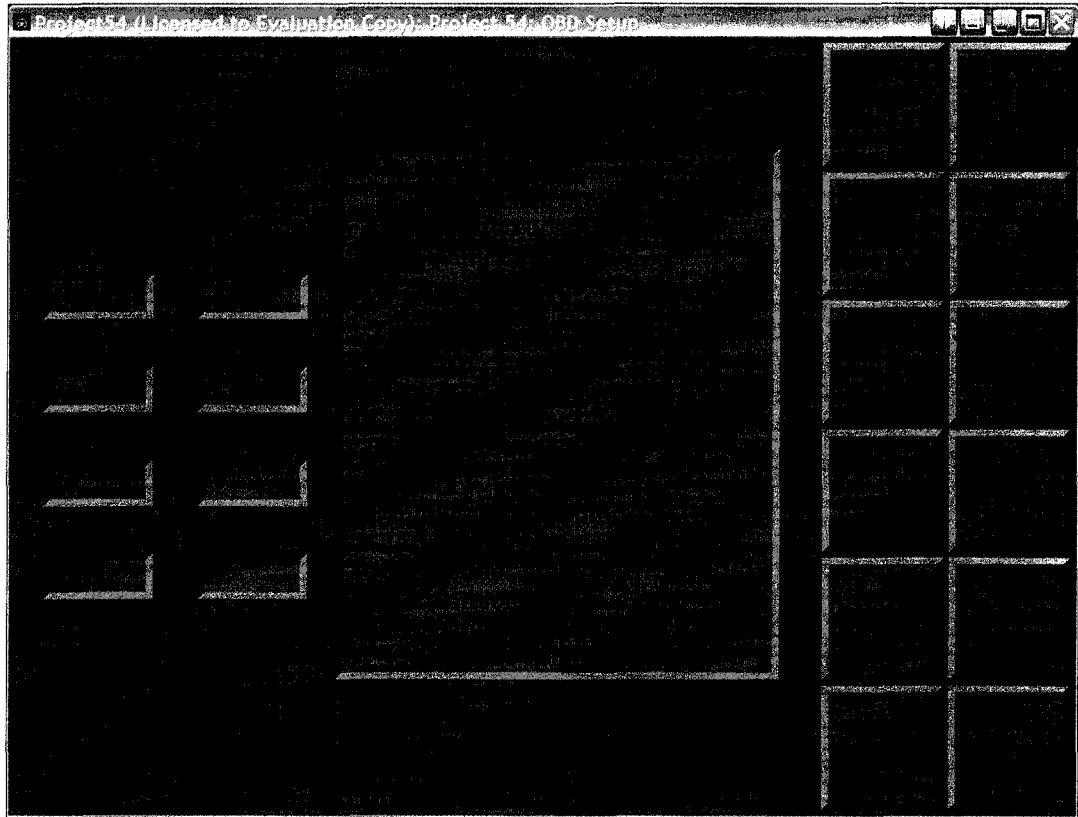
**Table 3.10 - Vehicle Diagnostics GUI Button Functions**

To activate GUI button functions with speech commands a user presses the PTT and says the name of the GUI button [41]. Toggle style buttons such as “LOG DATA” will remain depressed once activated. Saying the name of the button followed by “OFF” will deactivate toggle style button functions. For example saying “LOG DATA OFF” would turn off logging once it has been activated.

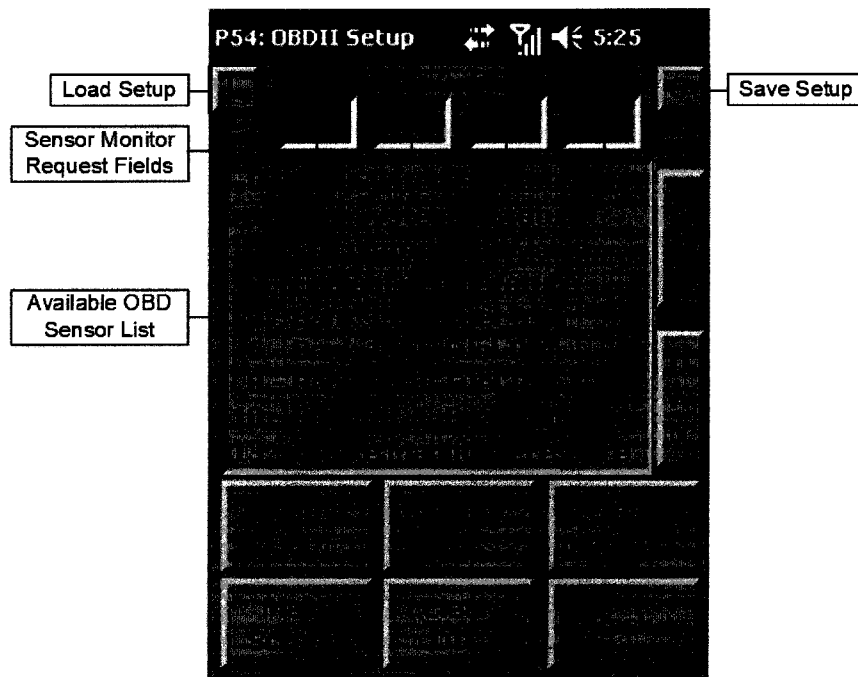
The tire pressure, DTC and P54 component status will be updated and displayed without any user action. If logging is enabled all status messages received by the Vehicle Diagnostics application will be time-stamped and recorded to the Vehicle Diagnostics Log File. By default a new log file will be created for each day the Vehicle Diagnostics application is loaded. The name of the file will be the date it was created. All data received during the day will be appended to the file. It is possible to change the log file name using the Diagnostic Logger Setup application described in section 3.2.4.



The Vehicle Diagnostics OBD Setup Screen is used to access the functions of the OBD-II application. The PC version of the OBD Setup screen is shown in Figure 3.19 and the PDA version is shown in Figure 3.20.


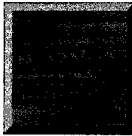
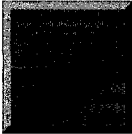
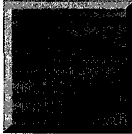

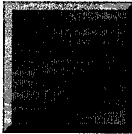


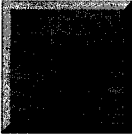
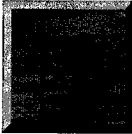



**Figure 3.19 - PC OBD Setup Screenshot**



**Figure 3.20 - PDA OBD Setup Screenshot**

The OBD Setup screen allows the user to select OBD sensors for monitoring, reset the OBD connection and clear DTCs. The list of available vehicle sensors is displayed along with text fields that allow the user to request sensors for monitoring. Sensors can be assigned to individual monitoring locations by entering a comma delimited list of sensor indices in a monitor request field as shown in Figure 3.19 and Figure 3.20. Although there are only eight monitor request fields available on the PC Diagnostics application and four on the PDA version, multiple sensors may be assigned to each monitor allowing all available sensors to be monitored. The “MONITOR ALL” function will automatically select all available sensors and assign each sensor to a monitor field. When the user exits the OBD Setup Screen the Vehicle Diagnostics application will send the Client Sensor Request Message shown in Figure 3.15 to enable monitoring of the indicated sensors. The speech enabled GUI button functions of the OBD Setup Screen are described in Table 3.11.

GUI Button	Function	GUI Button	Function
	Switch to the Main Screen Application		Scroll up the list of available OBD sensors (“UP” on PDA version)
	Switch to the Patrol Screen Application		Scroll down the list of available OBD sensors (“DWN” on PDA version)
	Return to the Vehicle Diagnostics Screen (“Back” on PDA version)		Load a previously defined Sensor Monitoring Setup
	Clear all Sensor Monitor Request Fields		Save the current Sensor Monitoring Setup
	Monitor all available OBD sensors		Re-initialize the OBD connection
	Clear all emissions-related diagnostic information from the vehicle		

**Table 3.11 - OBD Setup Screen GUI Button Functions**

Selecting the “CLEAR DTC” function will cause the Vehicle Diagnostics application to send the DTC Clear Request Message shown in Figure 3.13. This function will erase all vehicle emission-related diagnostic information including DTCs stored in ECUs. To clear this information the ignition key must be in the “ON” position and the engine must be off. In accordance with SAE specification the Vehicle Diagnostics application will prompt the user to verify that this is the desired action. The confirmation dialog is shown in Figure 3.21. The Vehicle Diagnostics application will receive one of the two status messages shown in Figure 3.14 to indicate whether the DTC clear attempt was successful. If the attempt succeeds, the Malfunction Indicator Lamp (MIL) or

“Check Engine” light on the dashboard should turn off and the Vehicle Diagnostics application should indicate that there are no DTCs stored in the vehicle.

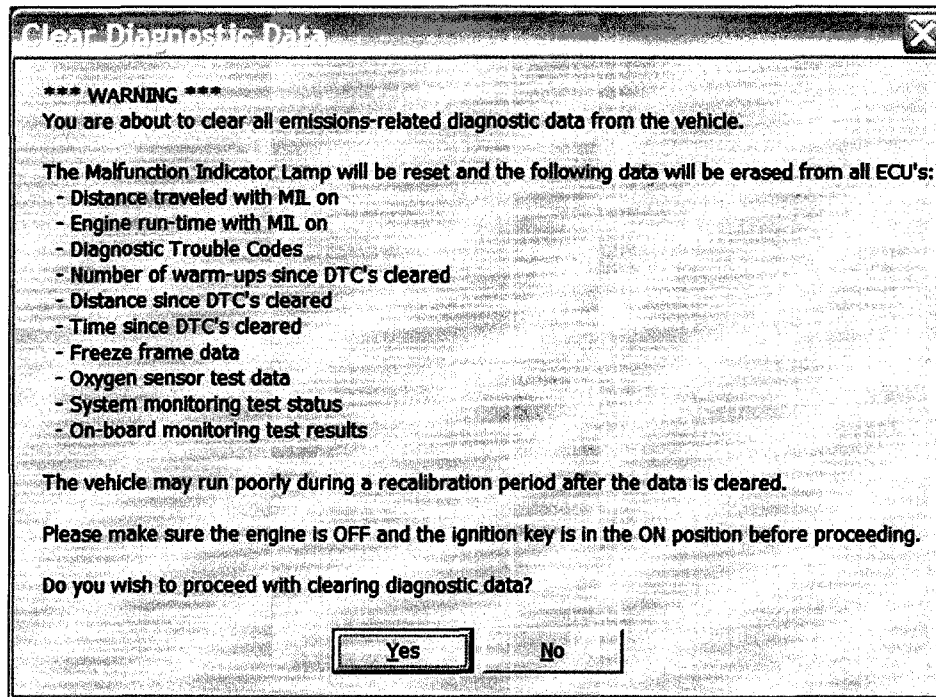


Figure 3.21 - DTC Clear Request Confirmation Dialog

Using the “SAVE SETUP” and “LOAD SETUP” functions it is possible to create and use pre-defined sensor monitoring configurations. When these functions are activated, the user will be prompted with a dialog asking them to select a file for loading or specify a filename for saving the configuration. This simplifies the monitoring process by skipping the step of entering each desired sensor index into a monitor request field.

### 3.2.4 Diagnostic Logger Setup Application

The Diagnostic Logger Setup application, *Diagnostic Logger Setup.exe*, provides a means to configure the logging and monitoring functions of the Vehicle Diagnostics application without using the Vehicle Diagnostics GUI. With this application the user can: select sensors for monitoring, enable logging, specify the log file name, load and

save pre-defined monitoring setups and set whether the Vehicle Diagnostics GUI is displayed in P54. The screenshot for this application is shown in Figure 3.22. This application sets various registry keys which are checked by the Vehicle Diagnostics application when it first starts.

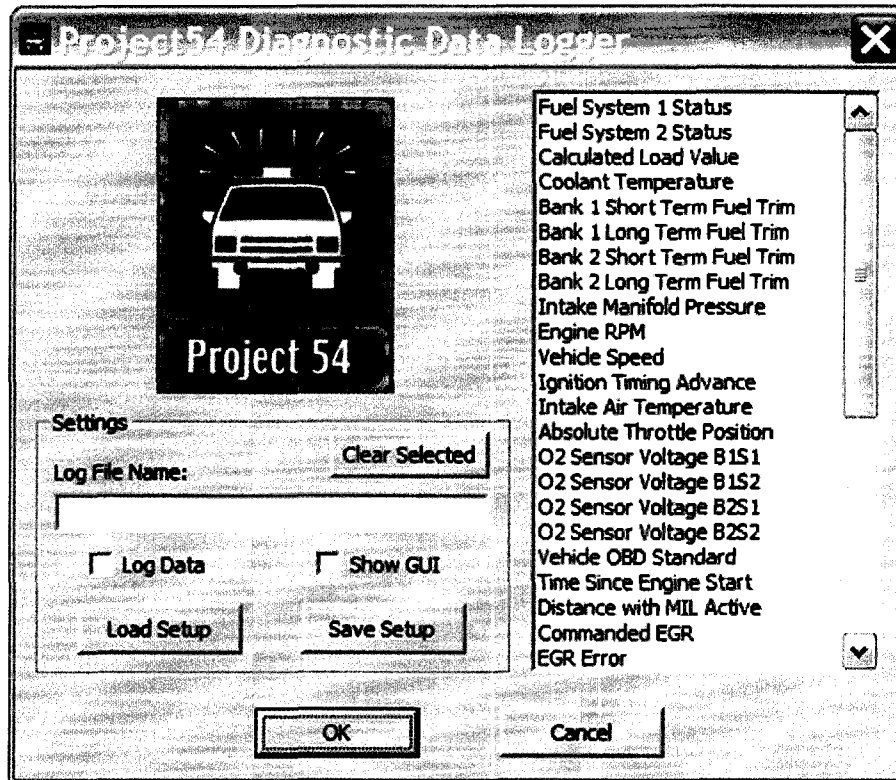


Figure 3.22 - Diagnostic Data Logger Application Screenshot

If the P54 OBD-II application has successfully been initialized there will be a list of supported vehicle sensors in the registry. When the Data Logger application is first started it will check for the available sensors and display them in the list box. If the available sensors are unknown, the application will instead list all P54 supported sensors shown in APPENDIX C. The user can select sensors for monitoring by clicking on their names in the list box and exiting the application by selecting *OK*. If logging is enabled any highlighted sensors in the list box will be requested for monitoring the next time the

Vehicle Diagnostics application is started. The *Clear Selected* button will deselect all sensors in the list box.

This application can create and load predefined sensor setups using the *Load Setup* and *Save Setup* options. When either of these options is selected, the user will be prompted to specify a filename for loading or saving. If the *Show GUI* option is not selected, the Vehicle Diagnostics application will still load when P54 is started but the user will not be able to access the GUI or any of its functions.

The default logging behavior of the Vehicle Diagnostics application is to create daily logs using the date as the filename. Leaving the *Log File Name* field blank will maintain the default logging behavior. When a log file name is specified, all data received by the Vehicle Diagnostics application will be logged to that file until the log file name is changed. Logging is still available even if the Vehicle Diagnostics GUI is not shown.

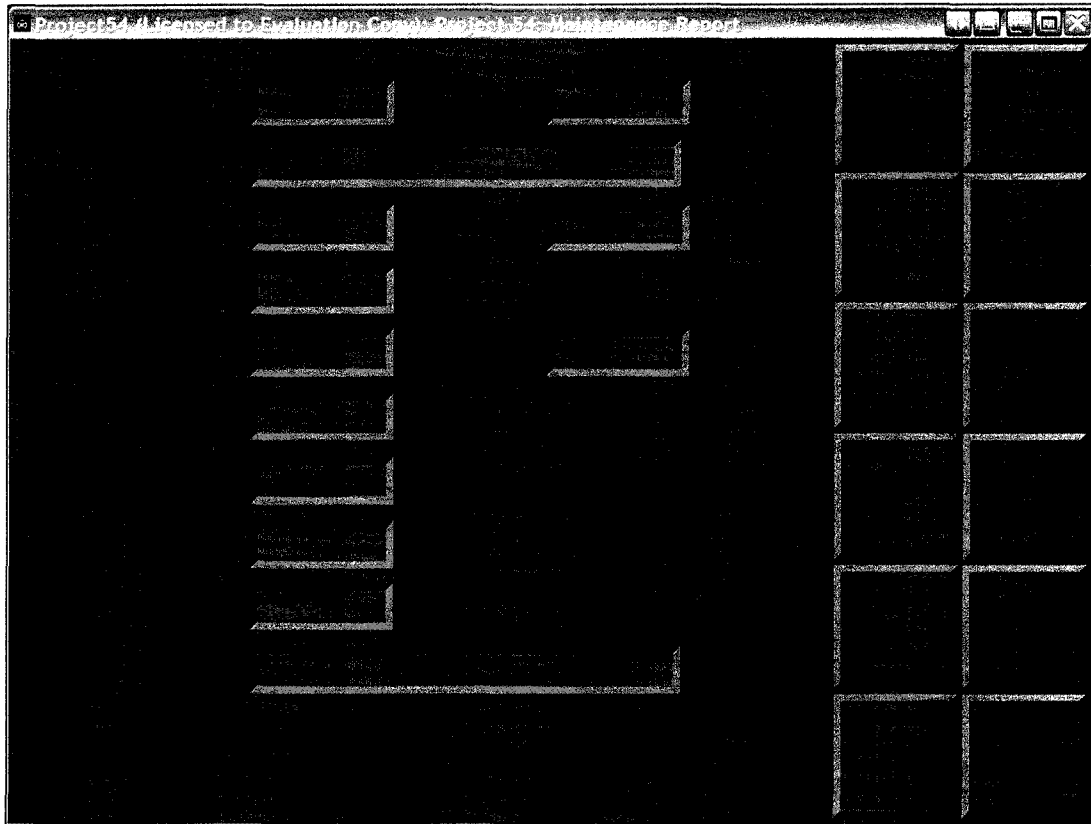
## CHAPTER 4

### FLEET MANAGEMENT

The fleet management service collects and combines pertinent vehicle information allowing fleet managers to track operating expenses and monitor the status of fleet vehicles. An electronic data entry form was developed to record daily vehicle maintenance expenses. This information is combined with data from the diagnostics service to provide fleet managers an overview of the operational status and costs for the vehicle fleet. The in-vehicle information is stored in a series of text files which can be transmitted daily using P25 digital data radios and state operated WiFi hotspots. Fleet managers will be able to access this information using a web-based interface. A data plotting application was created for visualizing recorded vehicle diagnostic information. Additionally a MATLAB post-processing script was developed to determine fuel economy using recorded OBD-II sensor data.

#### **4.1 Maintenance Reports Application**

The NHDS Division of State Police currently uses hardcopy forms to create daily logs of routine vehicle maintenance expenses. These forms are used to track expenses such as gasoline, oil changes and car washes. The hardcopy forms must later be manually transcribed into an electronic database. The P54 Maintenance Reports Application, *MaintenanceReports.dll*, is an electronic data entry form developed to facilitate the recording, transmission and storage of this data. This application is shown in Figure 4.1.



**Figure 4.1 - Project54 Maintenance Reports Application Screenshot**

The Maintenance Reports application allows officers to record the maintenance expense information currently collected using the hardcopy form. To save time and eliminate redundant effort the identification information including the equipment number, plate number, trooper name, trooper ID and unit will automatically be loaded into the form after the first time this information is recorded. To create a new report, the officer enters information in the appropriate fields. This information will be retained in the form until the report is submitted. A log file will be created when the officer presses the *Submit Report* button. Only one Maintenance Report log may be created per day. If multiple reports are submitted during the same day then only the most recent entry will be retained. The formats for the fleet management log files are described in section 4.2.



## 4.2 Fleet Management Log Files

The fleet management service includes three logs providing an overview of the diagnostic status and operating expense information for the vehicle. The three log types are: Maintenance and Mileage Reports, P54 and OBD Status Reports and Vehicle Diagnostics Logs. By default the fleet management logs will be stored in the P54 log file folder using the directory structure shown in Figure 4.2.

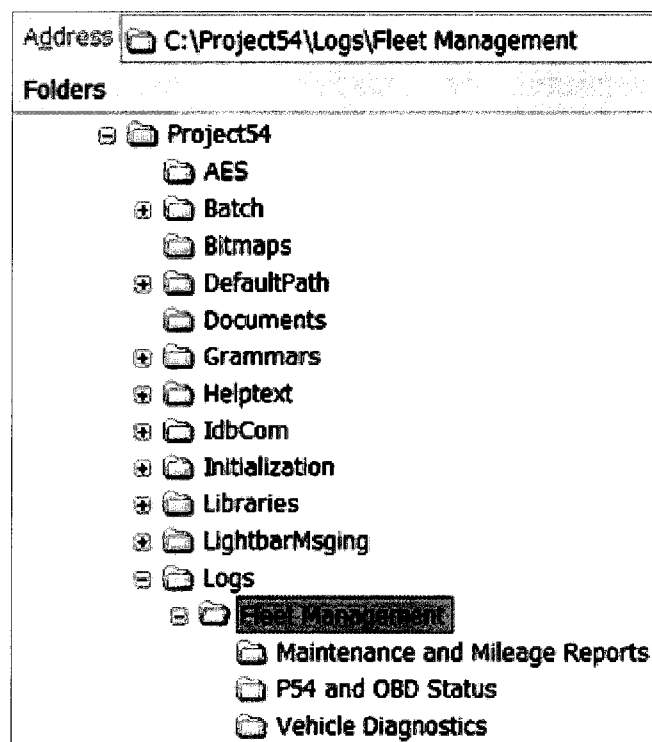


Figure 4.2 - Fleet Management Log File Directory

### 4.2.1 Maintenance and Mileage Reports Format

The maintenance and mileage reports contain the information entered in the fields of the Maintenance Reports Application shown in Figure 4.1. Each file in the Maintenance and Mileage Reports folder will be named for the date on which it was created. There will only be one report for each day. These text files contain a comma delimited list of fields describing the vehicle operator ID information as well as the

vehicle mileage and maintenance expense information. The formats for these fields are given in Table 4.1 and an example report is shown in Figure 4.3. It should be noted that if a field does not contain any data then that entry in the report will be left blank as is shown for field 13 in the example of Figure 4.3.

Field	Description	Format
1	Date/Time report created	YYYY-MM-DD HH:MM:SS
2	Equipment Number	Four digit number
3	Plate Number	Seven alpha-numeric characters
4	Trooper ID Number	Four digit number
5	Trooper Name	-
6	Troop/Unit	Three alphabetic characters
7	Odometer reading	Integer number (miles)
8	Gas 1 quantity	Decimal number (gallons)
9	Gas 2 quantity	Decimal number (gallons)
10	Gas cost	Decimal number (\$)
11	Oil added	Decimal number (quarts)
12	Car wash cost	Decimal number (\$)
13	Miscellaneous costs	Decimal number (\$)
14	Remarks	-

Table 4.1 - Maintenance and Mileage Report Fields

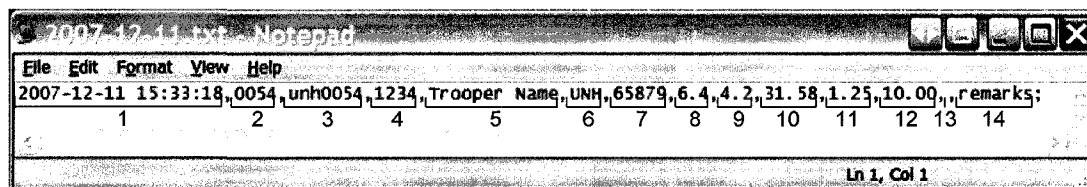


Figure 4.3 - Example Maintenance and Mileage Report

#### 4.2.2 P54 and OBD Status Report Format

The P54 and OBD Status reports provide a summary of the vehicle status by combining information from the Maintenance Reports, Tire Pressure and OBD-II applications. These reports will automatically be created if either the Tire Pressure or OBD-II application is installed. Each file in the P54 and OBD Status folder will be named for the date it was created. There will only be one report created per day. These

reports consist of three semi-colon delimited sections with each section containing a variable number of comma delimited data items. The three sections can include the information from the OBD DTC status report shown in Figure 3.12, the P54 Component Status shown in Figure 3.3 and the pressure of each tire. The number of data items included in these reports depends on whether the OBD-II and Tire Pressure applications are installed as well as whether any DTC or P54 component errors are present. The format for these reports is described in Table 4.2. Two examples of the P54 and OBD Status report are given in Figure 4.4 and Figure 4.5.

Section	Data Item	Format/Comments
Operator ID and OBD-II DTC Status	1. Date/Time report created	YYYY-MM-DD HH:MM:SS
	2. Equipment Number	Four digit number
	3. Plate Number	Seven alpha-numeric characters
	4. Trooper ID Number	Four digit number
	5. Trooper Name	-
	6. Troop/Unit	Three alphabetic characters
	7. OBD-II installed?	True ("t") or False ("f") If False, this is the last data item in this section
	8. Number of DTCs, <n>	Integer count of stored DTCs
	9. First DTC	See Table 3.7 for DTC formats. These items will only be included if <n> does not equal 0
	10. First DTC description	
↓		
7+2n. n <sup>th</sup> DTC		
8+2n. n <sup>th</sup> DTC description		
P54 Component Status	1. Number of errors, <m>	Integer count of P54 component errors
	2. First error description	These items will only be included if <m> does not equal 0
	3. Second error description	
	↓	
1+m. m <sup>th</sup> error description		
Tire Pressure Status	1. Tire Pressure installed?	True ("t") or False ("f") If False, this is the last data item in this section
	2. Front-left tire pressure	The integer tire pressure readings
	3. Front-right tire pressure	
	4. Rear-left tire pressure	
	5. Rear-right tire pressure	

Table 4.2 - P54 and OBD Status Report Format

The screenshot shows a diagnostic report window with a menu bar (File, Edit, Format, View, Help) and a title bar. The main content area displays the following information:

2007-12-12 18:54:58, 0054, unh0054, 1234, Trooper Name, UNH, 1, 0, 1, AppManager ERROR: No IDB Main Connection Response, f;										
1	2	3	4	5	6	7	8	9	10	
Operator ID and OBD-II DTC Status								P54 Component Status		Tire Pressure
Ln 1, Col 1										

Figure 4.4 - P54 and OBD Status Report Example 1

The screenshot shows a diagnostic report window with a menu bar (File, Edit, Format, View, Help) and a title bar. The main content area displays the following information:

2007-12-12 18:47:20, 0054, unh0054, 1234, Trooper Name, UNH, 1, 1, P0195, Engine Oil Temperature Sensor, 0, 1, 25, 30, 29, 10;										
1	2	3	4	5	6	7	8	9	10	
Operator ID and OBD-II DTC Status								P54 Component Status		Tire Pressure
Ln 1, Col 1										

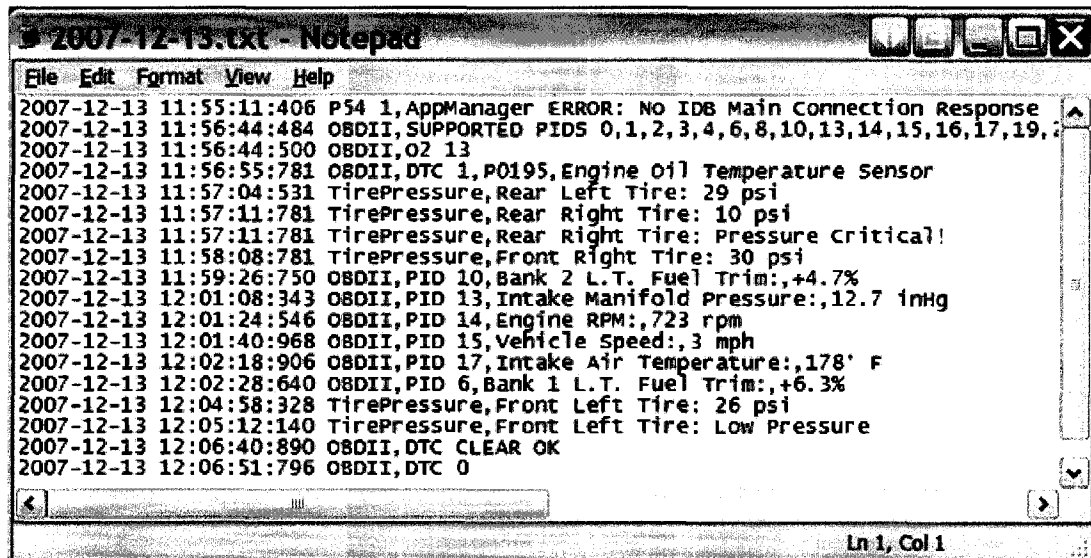
Figure 4.5 - P54 and OBD Status Report Example 2

These two examples illustrate how the report format is affected depending on the number of errors and the installed diagnostic hardware. Example 1 shown in Figure 4.4 represents a report for a vehicle with the OBD-II application installed where no DTCs and one P54 error were detected and the Tire Pressure application is not installed. Example 2 in Figure 4.5 represents the case where the OBD-II application has detected one DTC, there are no P54 component errors and the Tire Pressure application is installed. These reports will continually be updated to reflect the most recent status information for a given day.

#### 4.2.3 Vehicle Diagnostics Log Files

The vehicle diagnostics log files contain the data recorded by the Vehicle Diagnostics application when logging is enabled. These files will include all status messages received by the Vehicle Diagnostics application from the Tire Pressure and OBD-II applications. In addition to the DTC, P54 component and tire status information, these files will include the time-stamped sensor readings of individual OBD sensors. The information included in the vehicle diagnostics log files can be used to interactively plot individual sensor recordings and display the vehicle condition using the Diagnostic Data

Plotter application which will be described in section 4.4.1. An example vehicle diagnostic log file is shown in Figure 4.6.



```
File Edit Format View Help
2007-12-13 11:55:11:406 P54 1,AppManager ERROR: No IDB Main Connection Response
2007-12-13 11:56:44:484 OBDII,SUPPORTED PIDS 0,1,2,3,4,6,8,10,13,14,15,16,17,19,
2007-12-13 11:56:44:500 OBDII,O2 13
2007-12-13 11:56:55:781 OBDII,DTC 1,P0195,Engine Oil Temperature Sensor
2007-12-13 11:57:04:531 TirePressure,Rear Left Tire: 29 psi
2007-12-13 11:57:11:781 TirePressure,Rear Right Tire: 10 psi
2007-12-13 11:57:11:781 TirePressure,Rear Right Tire: Pressure Critical!
2007-12-13 11:58:08:781 TirePressure,Front Right Tire: 30 psi
2007-12-13 11:59:26:750 OBDII,PID 10,Bank 2 L.T. Fuel Trim:,+4.7%
2007-12-13 12:01:08:343 OBDII,PID 13,Intake Manifold Pressure:.,12.7 inHg
2007-12-13 12:01:24:546 OBDII,PID 14,Engine RPM:,723 rpm
2007-12-13 12:01:40:968 OBDII,PID 15,Vehicle Speed:.,3 mph
2007-12-13 12:02:18:906 OBDII,PID 17,Intake Air Temperature:.,178 F
2007-12-13 12:02:28:640 OBDII,PID 6,Bank 1 L.T. Fuel Trim:,+6.3%
2007-12-13 12:04:58:328 TirePressure,Front Left Tire: 26 psi
2007-12-13 12:05:12:140 TirePressure,Front Left Tire: Low Pressure
2007-12-13 12:06:40:890 OBDII,DTC CLEAR OK
2007-12-13 12:06:51:796 OBDII,DTC 0
Ln 1, Col 1
```

Figure 4.6 - Vehicle Diagnostic Log File Example

### 4.3 Communication Infrastructure

The proposed fleet management service will use a combination of P25 compliant digital data radios and Wifi hotspots to transmit collected data to a central server. The Maintenance and Mileage Reports and the P54 and OBD Status Reports can be transmitted daily using the P25Proxy Application [2]. The proxy application can accept messages from the Vehicle Diagnostics application and transmit these messages to a server using P25 compliant data packets from the in-vehicle radios.

The vehicle data distribution and collection system developed at P54 by Jacob LeBlanc *et al.* [1] will be used to collect the larger and more detailed Vehicle Diagnostic Log files. There are three layers to this system. At the top layer is the P54 master update server. This server will function as the central repository for all collected fleet management reports. The middle layer consists of remote servers which are typically

located at NHDS fueling and troop stations. These sites are chosen to eliminate the need for officers to make special stops in order to perform a data update. Using these locations, the uploading of fleet management logs can be incorporated into the normal daily routines of the NHDS. At the bottom layer are the individual vehicles in which the fleet management log files are created. When cruisers visit NHDS fueling or troop stations an ad hoc 802.11b wireless connection is established between the vehicle and the remote update server. This connection is initiated by the officer in charge of the vehicle, giving them control of when data will be uploaded. The fleet management data will then be transferred over the wireless connection and propagated to the master update server. The architecture for the data update system is depicted in Figure 4.7.

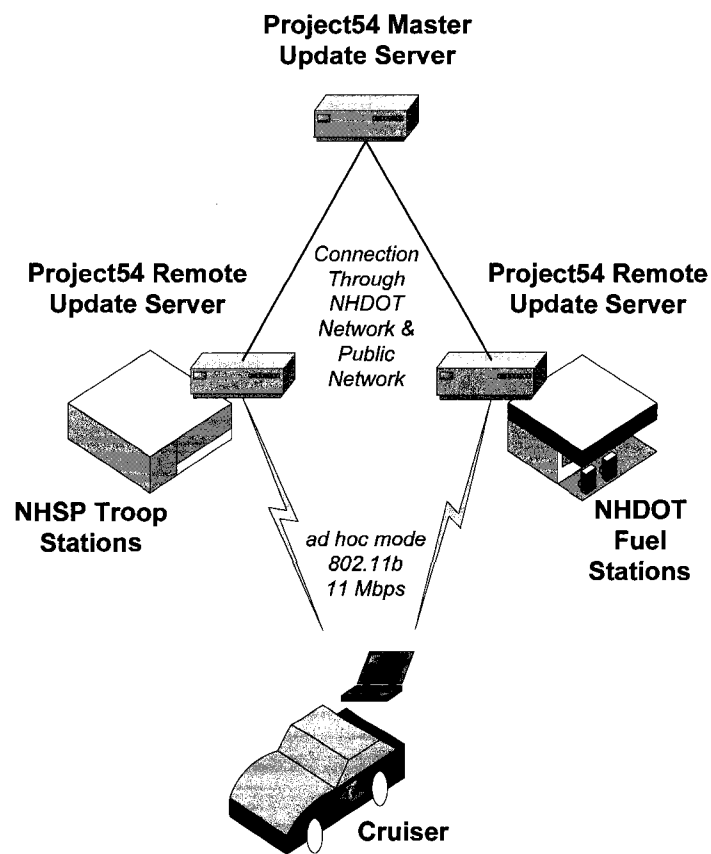


Figure 4.7 - Project54 Data Update System Physical Network Architecture

A web interface is currently under construction that will allow a fleet manager to analyze the collected vehicle data at the master server. Other wireless communications such as cellular data modems can also be used to transfer the fleet management files to the master server. However, due to cost and bandwidth limitations, the wireless update system described above will initially be the preferred method.

#### **4.4 Data Analysis and Post-Processing**

Two applications were developed to demonstrate how information collected in the fleet management service can be analyzed and employed to provide benefits to fleet managers. A Diagnostic Data Plotter application was developed that offers a summary overview of the condition of fleet vehicles and supports plotting of recorded OBD sensor values. A MATLAB post-processing script was developed to determine a vehicles fuel economy using the combined measurements from multiple OBD sensors.

##### ***4.4.1 Diagnostic Data Plotter Application***

The Diagnostic Data Plotter application, *Diagnostic Data Plotter.exe*, parses the data contained in the Vehicle Diagnostics Log files and uses this information to display an overview of the vehicle condition. The application screenshot is shown in Figure 4.8.

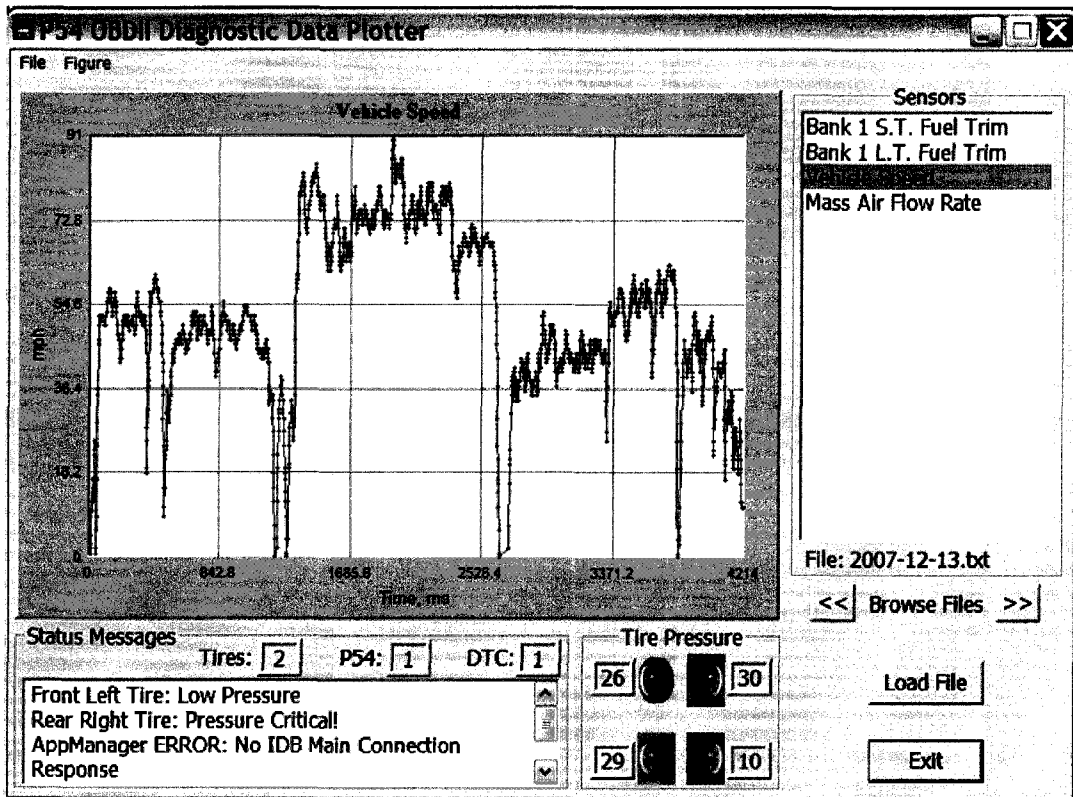


Figure 4.8 - Diagnostic Data Plotter Application Screenshot

Pressing the *Load File* button will launch a dialog box prompting the user to select a Vehicle Diagnostics log file. The application will then analyze the file and update the Data Plotter GUI to display the vehicle condition as it was recorded at the end of the selected file. This overview will include the tire pressures, DTC errors, and any P54 component errors. By default all Vehicle Diagnostics log files in a particular folder will be named for the date on which they were created. Using the *Browse Files* options will allow the user to quickly load the next (>>) or previous (<<) file in the current folder selected from the *Load File* dialog box. This allows the user to quickly check the day-to-day status of a vehicle to observe when problems occur and when they are corrected.

The values of recorded OBD sensors in the Vehicle Diagnostics log files may be plotted by clicking on the name of a sensor in the *Sensors* list. The sensor data will then



be plotted against a millisecond time scale. The user can zoom in on a specific section of the graph by left clicking and dragging on an area in the plot window. Right-clicking will then return the graph to its default view. Additional data and graphing options are available using the menu commands described in Table 4.3.

Menu Item	Option	Description
File	Load	Launch the Load File Dialog box to select a Vehicle Diagnostics Log file within a particular folder.
	Export Data	Select a folder to export the recorded OBD sensor data to. Individual CSV style text files will be created for each sensor in the <i>Sensors</i> list containing the millisecond time-stamped sensor readings.
	Exit	Close the Diagnostic Data Plotter application.
Figure	Properties	Adjust the properties of the figure window. Includes line colors, data point marker styles and annotations.
	Print	Print the plot of the currently selected OBD sensor.
	Copy	Copy the figure window containing the sensor plot.
	Save	Save the sensor plot figure window bitmap.

**Table 4.3 - Diagnostic Data Plotter Menu Commands**

#### **4.4.2 Fuel Economy Estimation**

All OBD-II equipped vehicles use a Mass Air Flow (MAF) or Manifold Absolute Pressure (MAP) sensor in regulating vehicle emissions. By combining the data from these sensors with data from intake air temperature, engine rpm and long and short term fuel trim sensors it is possible to estimate the amount of fuel consumed for a given time interval. Equation 4.1 is used when a MAF sensor is employed and Equation 4.2 applies for MAP sensor based systems [58]. The distance traveled by the vehicle may also be estimated using the recorded values from the vehicle speed sensor.

$$Fuel\ Consumed\ (L) = \frac{1}{AF \times \rho \times 1000} \sum_{i=1}^N MAF_{t_i} \times \overline{LongFT_{t_i}} \times \overline{ShortFT_{t_i}} \times \Delta t_i$$

**Equation 4.1 - Mass Air Flow Based Fuel Consumption Equation**

$$\text{Fuel Consumed (L)} = \frac{VE \times ED \times M}{120 \times R \times AF \times \rho \times 1000} \sum_{i=1}^N \frac{RPM_i \times MAP_i \times \overline{LongFT}_i \times \overline{ShortFT}_i}{IAT_i + 273.12} \Delta t_i$$

**Equation 4.2 - Intake Manifold Pressure Based Fuel Consumption Equation**

The coefficients and variables used in these equations are described in Table 4.4.

Variable or Coefficient	Description	Value Range	Units
AF	Ideal air to fuel stoichiometric ratio	14.64	-
$\rho$	Fuel density	0.74	g/mL
MAF	Mass air flow rate	[0 655.35]	g/s
LongFT	Average long term fuel trim	[0 1.9922]	-
ShortFT	Average short term fuel trim	[0 1.9922]	-
$t_i$	Time during interval, i	[0 $\infty$ ]	s
VE	Volumetric efficiency	0.8 (assumed value)	-
ED	Engine displacement	5.7 (Dodge Charger)	L
M	Molar mass of air	28.97	g/mol
R	Ideal gas constant	8.314	kPa
RPM	Engine revolutions per minute	[0 16383.75]	min <sup>-1</sup>
MAP	Manifold absolute pressure	[0 255]	kPa
IAT	Intake air temperature	[-40 215]	°C

**Table 4.4 - Fuel Consumption Calculation Variables and Coefficients**

Combining the estimated fuel consumption and distance traveled allows for a miles-per-gallon estimate of the vehicle fuel economy based solely on information from the OBD-II sensors. There are three steps involved in obtaining the estimated vehicle fuel economy using the applications developed for this thesis. First, the appropriate sensors are selected for monitoring and the data is logged using the Vehicle Diagnostics application. Next the Diagnostic Data Plotter application is used to export the recorded sensor data to a folder containing individual files of the data from each sensor. Finally, a MATLAB post-processing script is run which loads the time stamped sensor data from the individual sensor files and performs the numerical calculations for estimating the distance traveled and fuel consumed. This MATLAB post-processing script has been included in APPENDIX D.

A sudden decrease in fuel economy could serve as an indicator of a malfunctioning vehicle. The ability to estimate vehicle mileage from speed measurements enables mileage based routine maintenance scheduling. The mileage and fuel consumption estimation has only been performed during post-processing of collected sensor data and this method will not presently be deployed in the P54 system for use by the NHDS. This is intended solely as a demonstration of how the information available from OBD-II sensors may be combined to provide indirect measurements of useful vehicle parameters.

## **CHAPTER 5**

### **TESTING AND DEPLOYMENT**

The diagnostic hardware and software described in this thesis were tested in multiple vehicles during development and successfully deployed in a NHDS vehicle. The data collected during the trial deployment includes P54 application status messages, tire pressure indicators, DTC checks and recorded OBD-II sensor values. Over 7 million data samples were logged in the vehicle over the course of 242 days of testing. Early analysis of the collected data led to modification of the log file formats. The diagnostic information collected during this trial deployment is the same information that will be available to the fleet management service as described in Chapter 4. The following sections describe the testing and deployment of this system.

#### **5.1 Development Testing**

The diagnostic hardware and software described in this thesis was tested in multiple vehicles at different times during development. Early testing was essential for debugging the OBD-II application and ensuring that it provided a generic interface for communicating with all OBD-II equipped vehicles. There are slight variations between the five OBD-II protocols supported by the scan tool (Table 3.1) that affect the way the P54 OBD-II software communicates with the vehicle. The purpose of development testing was to discover these differences and account for them in the OBD-II application.

During development, the P54 OBD-II application was tested on many different makes and models of vehicles. These included: 1998 Acura Integra, 1998 Eagle Talon, 2002 Chevrolet Impala, 2006 Dodge Charger, 2001 Ford Focus, 2002 Nissan 350Z, 1999 Saab 9-5 Aero, 2003 Toyota Highlander and more. One notable vehicle not included is the Ford Crown Victoria. Prior work at P54 by S. Y. Kim [11] focused on the Crown Victoria using hardware and software similar to that developed for this thesis. While this vehicle was not tested directly here, the results from testing the Ford Focus and the wide selection of other makes and models suggests that the all features of the software should be supported for the Crown Victoria as well.

Occasionally, testing on a new vehicle revealed a flaw that required modification of the software. One example was discovered while testing on the Dodge Charger. The Charger employs the ISO 15765-4 CAN based OBD-II protocol. This protocol supports multi-line responses from multiple vehicle ECUs. When the P54 OBD-II application queried the Charger for supported OBD sensors, it received a multi-line response message containing information from two ECUs. Each ECU supported different sensors and the OBD-II application was unable to parse the message. When this was discovered, the OBD-II application was modified to accommodate these types of multi-line responses.

Early testing was also performed to verify that the P54 OBD-II application could detect and report vehicle DTCs. This functionality was first tested on the 2002 Impala which had an active “Check Engine” light indicating the presence of one or more DTCs. The early version of the software was able to detect and retrieve the stored DTC but

could not provide the description of the error. As a result, the application was modified and it now supports 2025 standardized DTC definitions.

An additional test of the DTC reporting system was performed by creating a DTC on the Dodge Charger test vehicle. A spark plug wire was disconnected from one cylinder of the engine. When the engine was started, an error condition was detected and the 'Check Engine' light was illuminated on the dashboard. The OBD-II application was able to identify the DTC and provide the description of the error as shown in Figure 5.1.

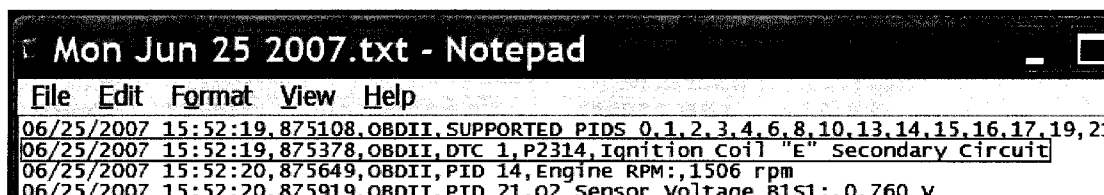


Figure 5.1 – DTC Detected by the Project54 OBD-II Application

The WTPMS and P54 Tire Pressure application were also tested during development. The pressure sensors were installed for several months on a 1998 Eagle Talon. There was one problem discovered when a sensor was installed incorrectly in cold temperatures. The rubber stopper in the sensor was not seated properly and a slow leak developed, resulting in a flat tire. This also became an issue during the NHDS deployment test. It is important to follow the manufacturer's recommendations for installing the sensors to make sure that there are no air leaks resulting from over or under-tightening the valve stem sensors.

## 5.2 Deployment Testing

The applications described in this thesis have been deployed in three trial vehicles. Two of these vehicles are used only for testing and demonstration purposes by P54. These are a 2002 Chevrolet Impala and a 2006 Dodge Charger. The third

deployment vehicle was a 2006 Dodge Charger which was employed in the active service of the NHDS. Data was collected in this vehicle over the course of 242 days spanning July 30, 2007 – April 28, 2008. In total there were 7,006,275 data samples collected which included P54 application status checks, DTC checks, OBD sensor readings and tire pressure messages. The collected data can be split into two phases. Phase one represents the first 88 days during which data was collected. Based on the analysis of the data from these first 88 days, changes were made to the log file naming convention and the format for certain messages. For phase two during the remaining 154 days of data collection, the log files and formats used were those described in Chapter 4 of this thesis.

### ***5.2.1 NHDS Deployment - Phase One***

For the initial NHDS deployment, the diagnostic hardware and software applications were installed within the existing P54 system in the vehicle. Logging was enabled and OBD-II sensors were selected for monitoring. Vehicle data was logged continuously whenever the vehicle was running with the P54 system active. The collected vehicle diagnostic log files were parsed using the script given in APPENDIX F. The results of this analysis are summarized in Table 5.1 Table 5.2 and Table 5.3 for the Tire Pressure, P54 and OBD-II data respectively. These tables provide an overview of the types and volume of data collected during this trial deployment.

<b>Item</b>	<b>Count</b>
Tire Pressure Status Messages	8494
Regular Pressure Updates	8403
Error Messages	91
Low Sensor Battery Warnings	2
Low Tire Pressure Warnings	83
Critical Tire Pressure Alerts	6
Number of Days with Errors Detected	14

**Table 5.1 - Phase One Tire Pressure Data Summary**

Item	Count
Status Messages	202
Number of Errors	176
Number of Days with Errors Detected	22
Unique Errors	6: W4Radio ERROR: No IDB Interface Response AppManager ERROR: No IDB Main Connection Response Golden Eagle ERROR: No IDB Interface Response PushTalk ERROR: No IDB Interface Response WhelenSerial ERROR: No IDB Interface Response TirePressure ERROR: No IDB Interface Response

**Table 5.2 - Phase One Project54 Status Data Summary**

Item	Count
Status Messages	2060683
O2 Label Messages	1323
Supported OBD Sensor Messages	675
OBD Sensor Readings	2058008
PID 0 - Fuel System Status	1220
PID 4 - Bank 1 S.T. Fuel Trim	377281
PID 6 - Bank 1 L.T. Fuel Trim	143472
PID 8 - Bank 2 S.T. Fuel Trim	377114
PID 10 - Bank 2 L.T. Fuel Trim	141174
PID 13 - Intake Manifold Pressure	283798
PID 14 - Engine RPM	4167630
PID 15 - Vehicle Speed	187528
PID 17 - Intake Air Temperature	129691
PID 92 - Battery Voltage	0
DTC Error Checks Performed	677
DTCs Detected	3
Number of Days with DTCs	1
Unique DTCs	1: P0138 02 Sensor Circuit High Voltage Bank 1 Sensor 2

**Table 5.3 - Phase One OBD-II Data Summary**

The 91 tire pressure error messages over the course of 88 days and 176 P54 system errors detected from 202 status messages may seem like high error rates at first. The reason for these high numbers is that once an error condition was present, the error was often repeatedly logged. The system operates in a continuous monitoring cycle. For



this reason it is important to consider the number of unique errors detected and the number of days during which errors were present.

In general, the Project54 system errors indicate a lack of communication between the P54 software and the networked IDB interfaces. The most common cause for these errors is when the P54 software is running while the IDB network is not connected to power due to the vehicle ignition being switched off. In this situation, the computer and software can be functioning normally but errors will be reported by the software applications as they attempt to communicate with their associated IDB boxes and are unable to receive responses.

After the initial deployment phase, small modifications were made to the log file formats and the file naming convention. The first alteration to the log files was to insure that each data item was time stamped and placed on a single line in the log file. Initially, there were certain messages such as tire pressure and P54 status that included newline characters that caused individual messages to be spread across multiple lines. This created problems for the Diagnostic Data Plotter application when attempting to parse the data.

The second problem involved the naming of the collected files. The initial file naming system consisted of a "DAY MON DD YYYY" format. This format has the unfortunate side effect of causing all files to be sorted according to what day of the week they were created when viewing the files with Windows Explorer. To remedy this, the ISO 8601 date and time representation was adopted as the naming convention for all log files and data samples. This standard specifies a "YYYY-MM-DD" format that insures

files are sorted in sequential, ascending or descending order when sorted in windows explorer. This makes it easier to identify specific dates in a large collection of files.

### 5.2.2 NHDS Deployment - Phase Two

After the initial analysis of the data from the phase one deployment, the software was modified and updated in the cruiser. All data logged during phase two of the deployment adheres to the formats and naming conventions described in Chapter 4 of this thesis. The software deployed here represents full-functional code for the services detailed in this thesis and has not been modified since the deployment. The phase two deployment spanned 154 days. The collected vehicle diagnostics log files have been parsed and the results are summarized in Table 5.4, Table 5.5 and Table 5.6.

Item	Count
Tire Pressure Status Messages	11983
Regular Pressure Updates	10544
Error Messages	1439
Low Sensor Battery Warnings	0
Low Tire Pressure Warnings	1303
Critical Tire Pressure Alerts	136
Number of Days with Errors Detected	57

Table 5.4 - Phase Two Tire Pressure Data Summary

Item	Description
Status Messages	11084
Number of Errors	1952
Number of Days with Errors Detected	33
Unique Errors	8: TirePressure ERROR: No IDB Interface Response OBD-II ERROR: No IDB Interface Response WxRadio ERROR: No IDB Interface Response AppManager ERROR: No IDB Main Connection Response Golden Eagle ERROR: No IDB Interface Response PushTalk ERROR: No IDB Interface Response WhelenSerial ERROR: No IDB Interface Response WhelenSerial ERROR: Control Head Communication Failure

Table 5.5 - Phase Two Project54 Status Data Summary

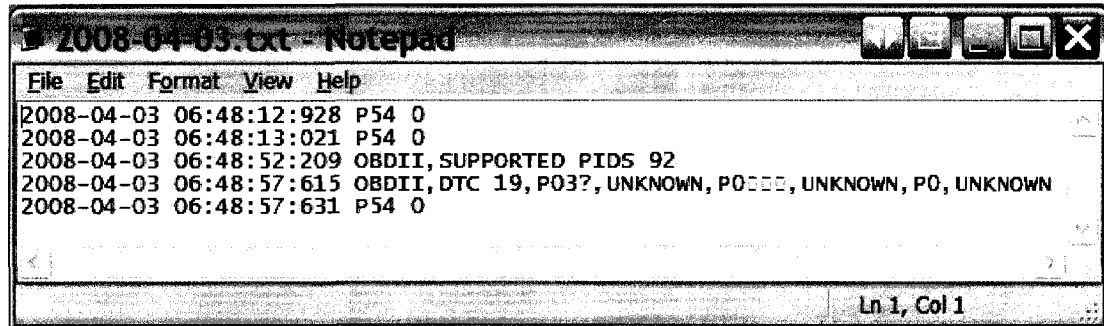
Here again the total P54 and tire pressure error numbers are high due to the repeated log messages. There was also a recurring error with the tire pressure monitoring system installation during this phase of deployment. The tires were not cold when the system was first installed. This caused the baseline pressure to be set higher than it should have been. When P54 was started when the tires were cold, low pressure warnings were generated. After the vehicle was driven a short distance, the tires would heat up and the pressure would rise to the baseline level, removing the error condition. This happened over the course of several weeks and is reflected in the 1439 tire pressure error messages logged.

Item	Description
Status Messages	4913829
O2 Label Messages	2569
Supported OBD Sensor Messages	1297
OBD Sensor Readings	4908667
PID 0 - Fuel System Status	1
PID 4 - Bank 1 S.T. Fuel Trim	964020
PID 6 - Bank 1 L.T. Fuel Trim	298841
PID 8 - Bank 2 S.T. Fuel Trim	961042
PID 10 - Bank 2 L.T. Fuel Trim	294679
PID 13 - Intake Manifold Pressure	652158
PID 14 - Engine RPM	1039085
PID 15 - Vehicle Speed	417792
PID 17 - Intake Air Temperature	281048
PID 92 - Battery Voltage	1
DTC Error Checks Performed	1296
DTCs Detected	359
Number of Days with DTCs	40
Unique DTCs	4: P0138 O2 Sensor Circuit High Voltage Bank 1 Sensor 2 P0573 Brake Switch "A" Circuit High P03? UNKNOWN P0□□□ P0

**Table 5.6 - Phase Two OBD-II Data Summary**

The DTC P0138 was a recurring error over the course of this deployment. This is most likely an indication of a malfunctioning oxygen sensor. Additionally, there was one

day which saw a report of 19 DTCs including the last two items shown in the Unique DTC list of Table 5.6. The log file for that day is shown in Figure 5.2.



```
2008-04-03 06:48:12:928 P54 0
2008-04-03 06:48:13:021 P54 0
2008-04-03 06:48:52:209 OBDII, SUPPORTED PIDS 92
2008-04-03 06:48:57:615 OBDII, DTC 19, P037, UNKNOWN, P0300, UNKNOWN, P0, UNKNOWN
2008-04-03 06:48:57:631 P54 0
```

**Figure 5.2 – Vehicle Diagnostics Log for 19 DTC Error Report**

Observing the log file, it appears the 19 DTCs for that day represent a false-positive report due to either a bug in the software or some abnormal operating condition. The average file size for a daily vehicle diagnostics log report for this deployment was between 1-3 MB depending on the amount of time the vehicle was running and the Project54 system was active during the day. The file size for this anomalous report was 218 B for the entire day, much smaller than a standard days worth of records. Also, there are no tire pressure reports and the list of vehicle supported PIDs contains only the battery voltage sensor and that is a function of the scan tool itself and is always available. These factors indicate that the system was not operating correctly due to either a hardware or software error during the day this log file was created.

### **5.3 Fleet Management Testing**

The fleet management service described in this thesis combines the diagnostic log file information with daily reports of vehicle operating expenses using the Maintenance Reports application. For the NHDS deployment, the vehicle status summary reports contained in the P54 and OBD Status files were logged. There were a total of 242 files

logged, one for each day of the deployment. These logs contain a summary of the diagnostic data as described in Chapter 4 of this thesis. The Maintenance Reports application has not been deployed at this time.

## **CHAPTER 5**

### **TRAFFIC MONITORING**

Exploring the use of radar equipped NHDS vehicles for traffic congestion monitoring was the third and final service proposed for this thesis. Radar data was collected from a stationary vehicle observing oncoming traffic for rural and highway road segments. A time stamped count of the observed vehicles was maintained during the collection of radar data. The information was analyzed to determine if the radar data alone can be used to provide an indication of traffic conditions. A scoring method was developed which provides a measure of traffic congestion based on the average speed of traffic and the number of radar readings received during a given time interval.

#### **6.1 Traffic Data Collection**

The P54 system software was employed to collect traffic data for rural and highway road segments at different times during the day. The data was collected from a stationary vehicle observing oncoming traffic. The data collected for this investigation consisted of time-stamped traffic speed samples and markers indicating when individual vehicles were observed on the roadway. A Stalker DSR traffic radar unit was used to collect traffic speed samples. When the traffic radar unit detected a vehicle, the speed information was transmitted over the IDB network and processed by the P54 Radar application. When the Radar application detected a change in the measured traffic speed, a status message was generated to indicate the change.

A Griffin AirClick wireless USB remote control was used mark the observation of individual vehicles on the roadway. Each time a button was pressed on the AirClick remote, a status message was generated to indicate the presence of a vehicle. The status messages generated by the traffic radar unit and the AirClick remote were time stamped and logged to a file using the P54 Data Logger application developed by Edward Bourbeau [59]. The traffic data collection setup is depicted in Figure 6.1.

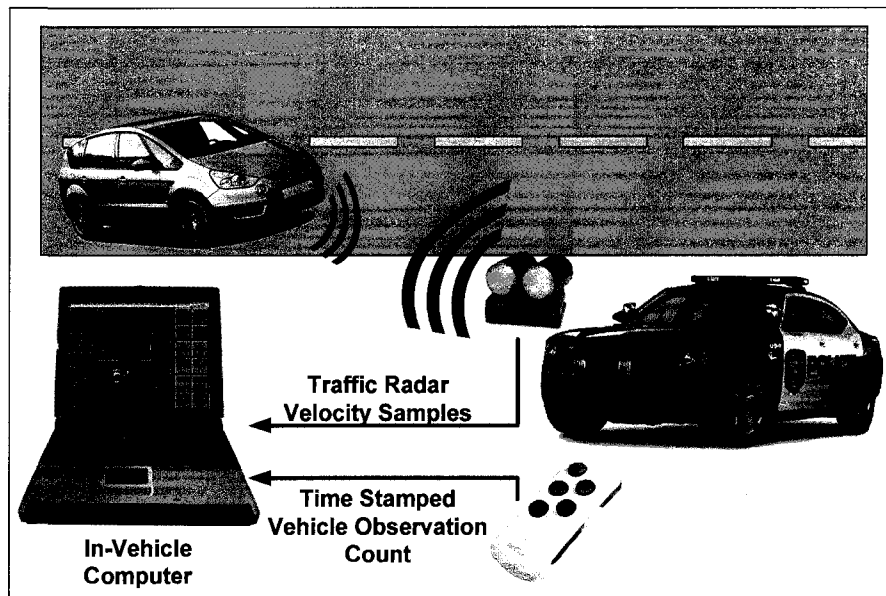


Figure 6.1 - Traffic Data Collection Setup

The collected status message log files were then parsed using the traffic data parser application, *TrafficDataParser.exe*, to separate the time-stamped vehicle observations and the traffic speed samples into individual CSV style log files. A total of six datasets were collected at three different locations for this investigation. Three of the datasets were for a rural road segment with a 35 mph speed limit. Two datasets were for a single oncoming lane, 55 mph highway and one was for a multilane highway with a speed limit of 55 mph during rush hour traffic. A complete description of the locations and the details of the collected datasets can be found in Table 6.1.

	Dataset Number					
	1	2	3	4	5	6
Description	Rural			Single Lane Highway		Multilane Highway
Speed Limit (mph)	35			55		55
Date	3/28/07	3/30/07	4/4/07	5/9/07	6/15/07	6/8/07
Start Time	11:55	11:18	13:13	16:57	15:12	16:53
End Time	12:40	12:30	14:12	17:56	16:41	18:57
Latitude	43.163648			43.152766		43.11014
Longitude	-70.935829			-70.94661		-70.815087
Vehicle Count	142	229	215	555	868	6296
Radar Hits	2439	3604	3429	3823	6352	24802
Travel Time (min)	17.18	15.74	15.95	6.89	7.32	3.94
Avg. Speed (mph)	37.13	35.75	36.26	54.33	53.04	39.17

Table 6.1 - Description of Traffic Monitoring Datasets

## 6.2 Traffic Data Analysis and Observations

The datasets collected for this investigation were analyzed using a MATLAB script to determine if a relationship could be identified between the radar data and the observed traffic conditions. The MATLAB script used to perform this analysis can be found in APPENDIX E. When run, the script first prompts the user to select a folder containing the parsed speed and car count log files. The data in those files is then loaded into arrays for processing and plotting. There were three items available in the data which were relevant to this analysis: the count of observed vehicles, the number of radar readings, or hits, and the measured traffic speed.

To identify trends in traffic conditions each dataset was divided into equally spaced observation time intervals. In [60] it was determined that 15 minutes is the optimal time interval for adequately capturing traffic congestion trends. For the initial analysis, observation intervals of 5, 10, 15 and 20 minutes were employed to assess the impact of the observation interval. The resulting datasets for each of these observation intervals are depicted in Figure 6.2 through Figure 6.7.



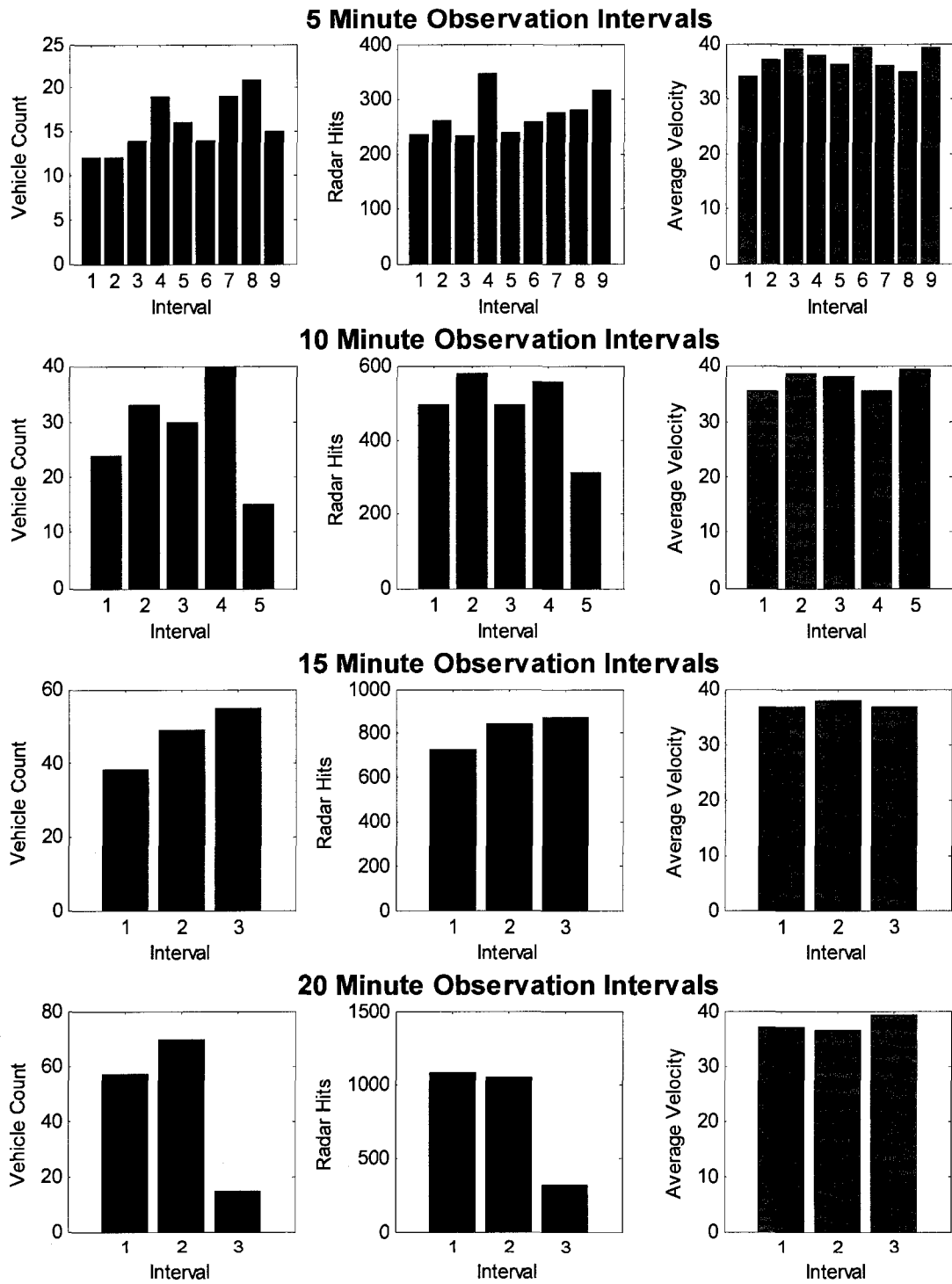


Figure 6.2 - Traffic Dataset #1, 35 mph Rural Road

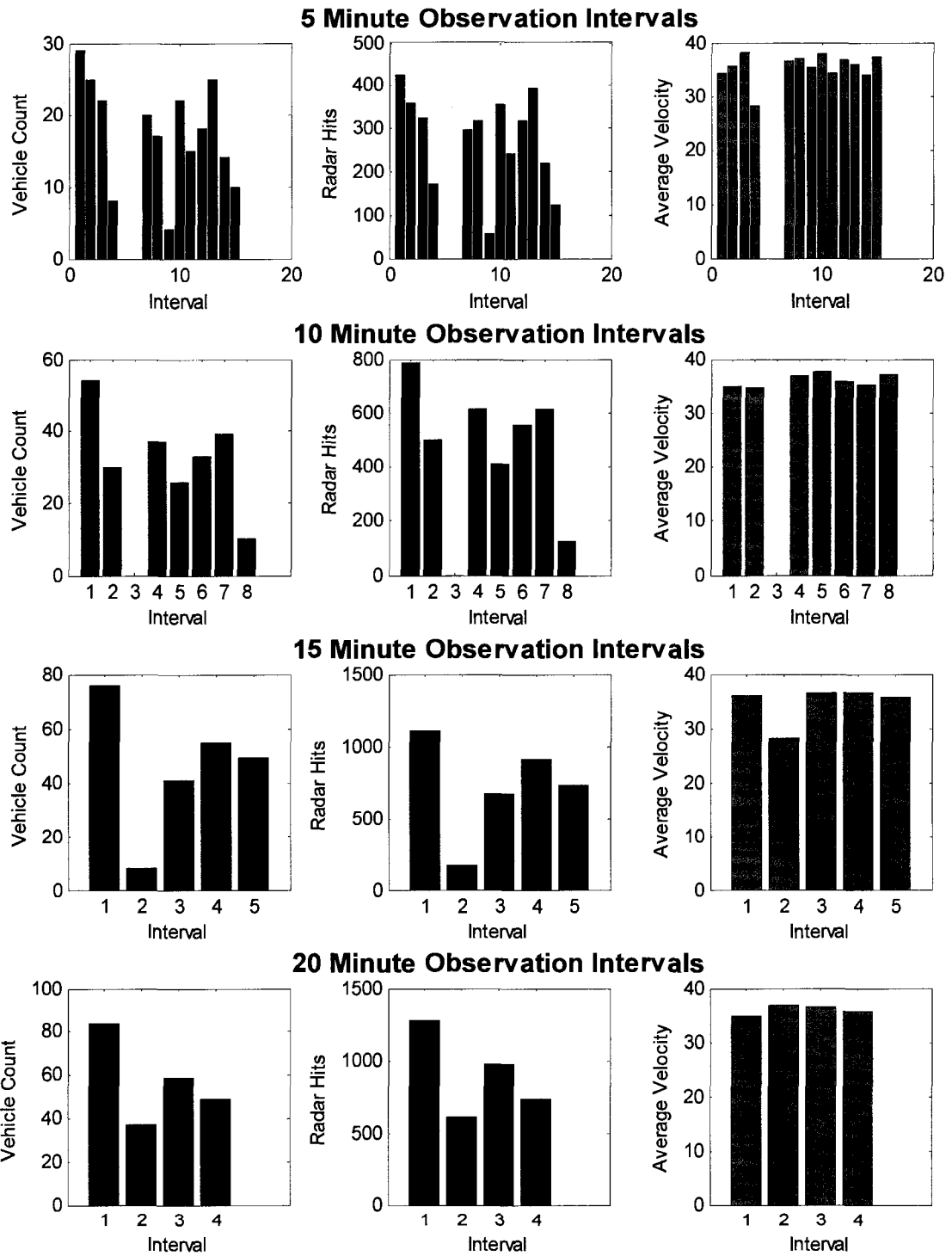


Figure 6.3 - Traffic Dataset #2, 35 mph Rural Road

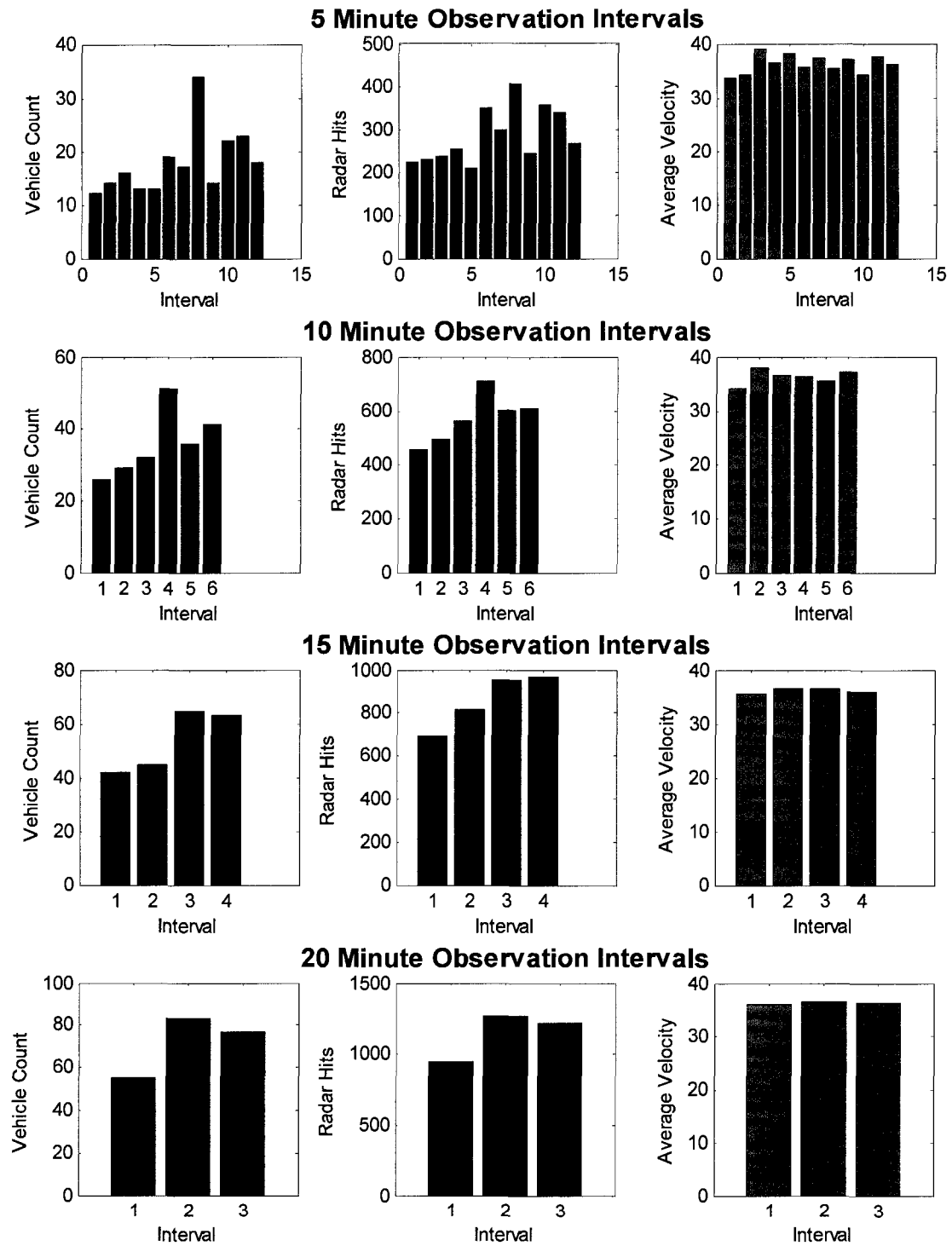


Figure 6.4 - Traffic Dataset #3, 35 mph Rural Road

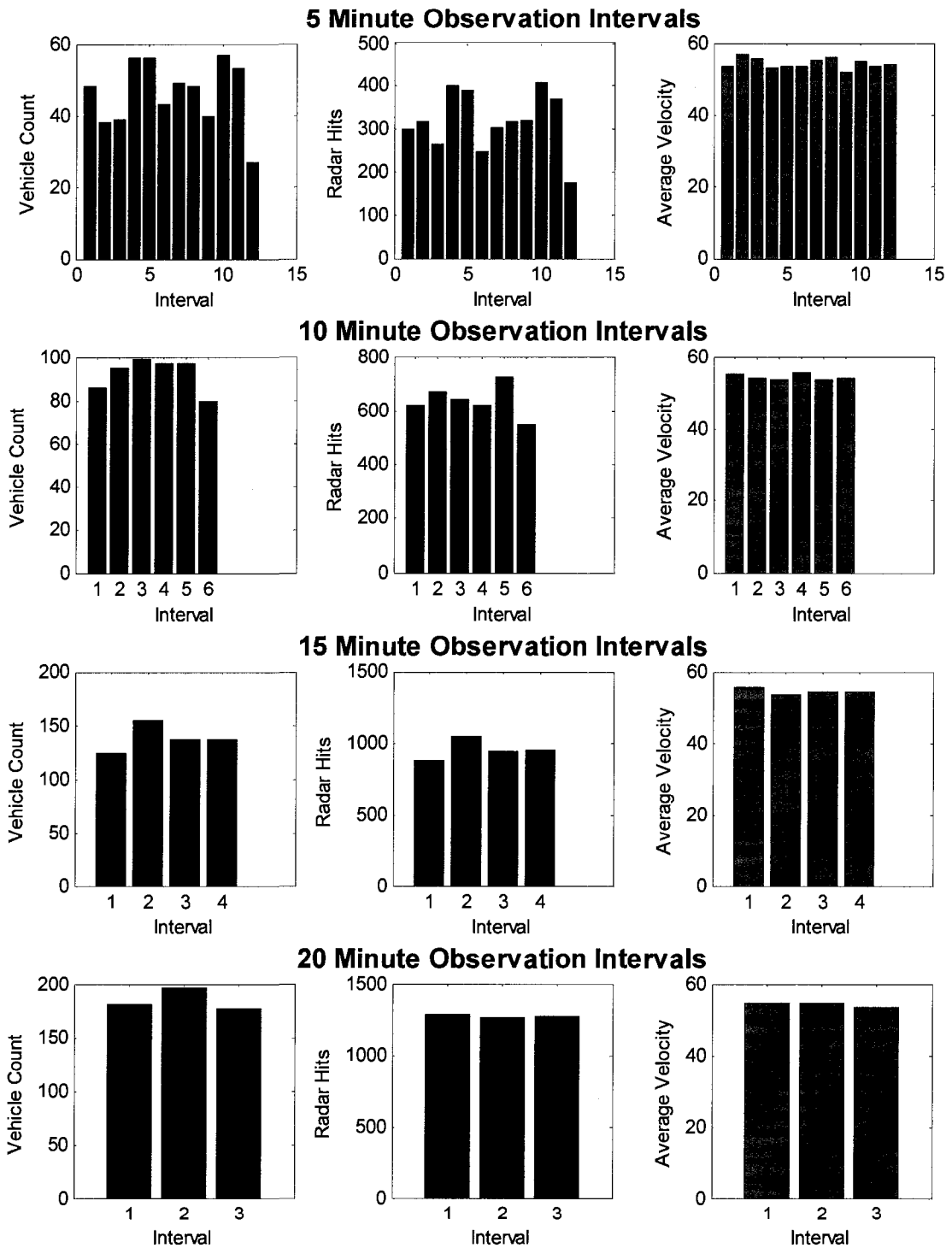


Figure 6.5 - Traffic Dataset #4, 55 mph Single Lane Highway

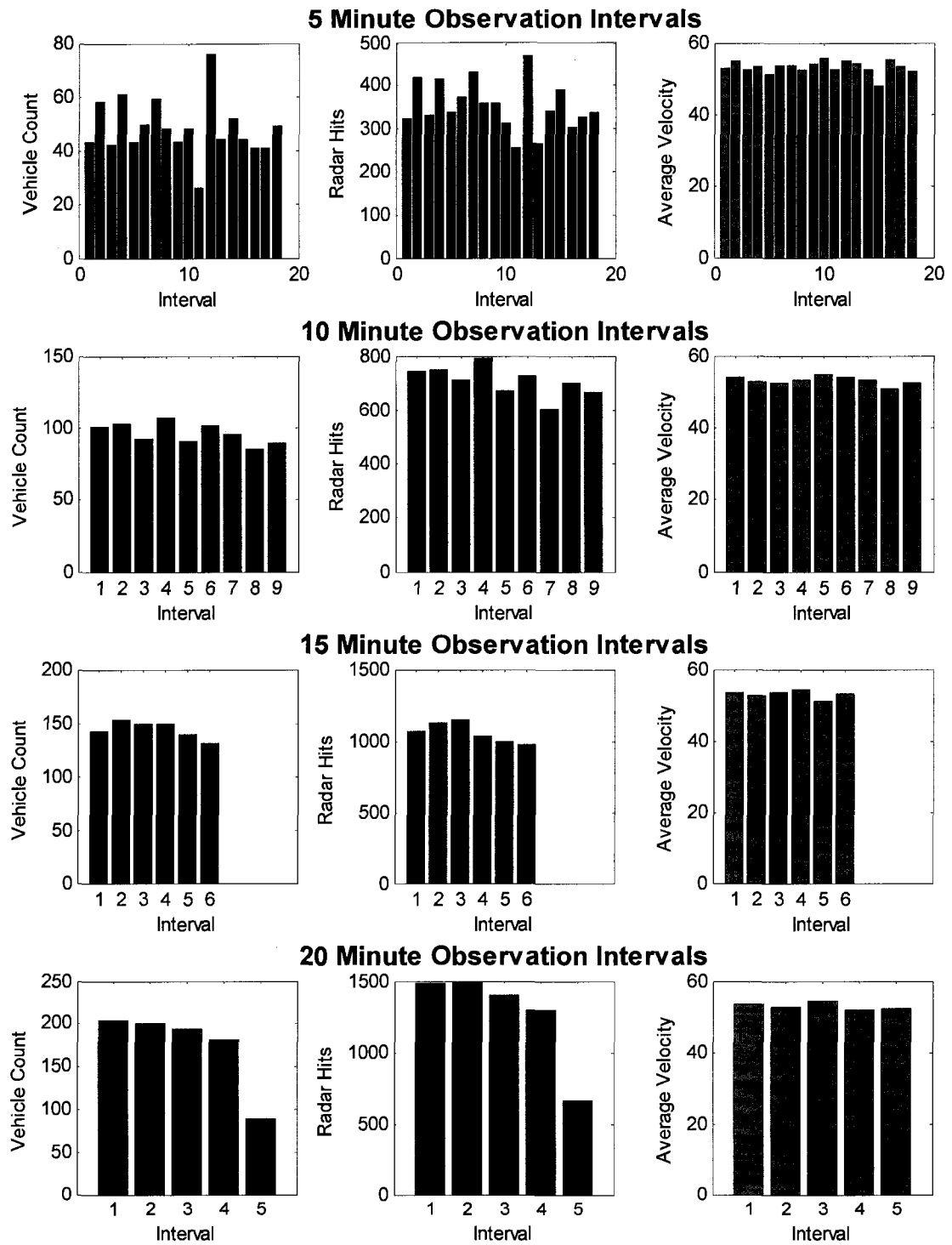


Figure 6.6 - Traffic Dataset #5, 55 mph Single Lane Highway

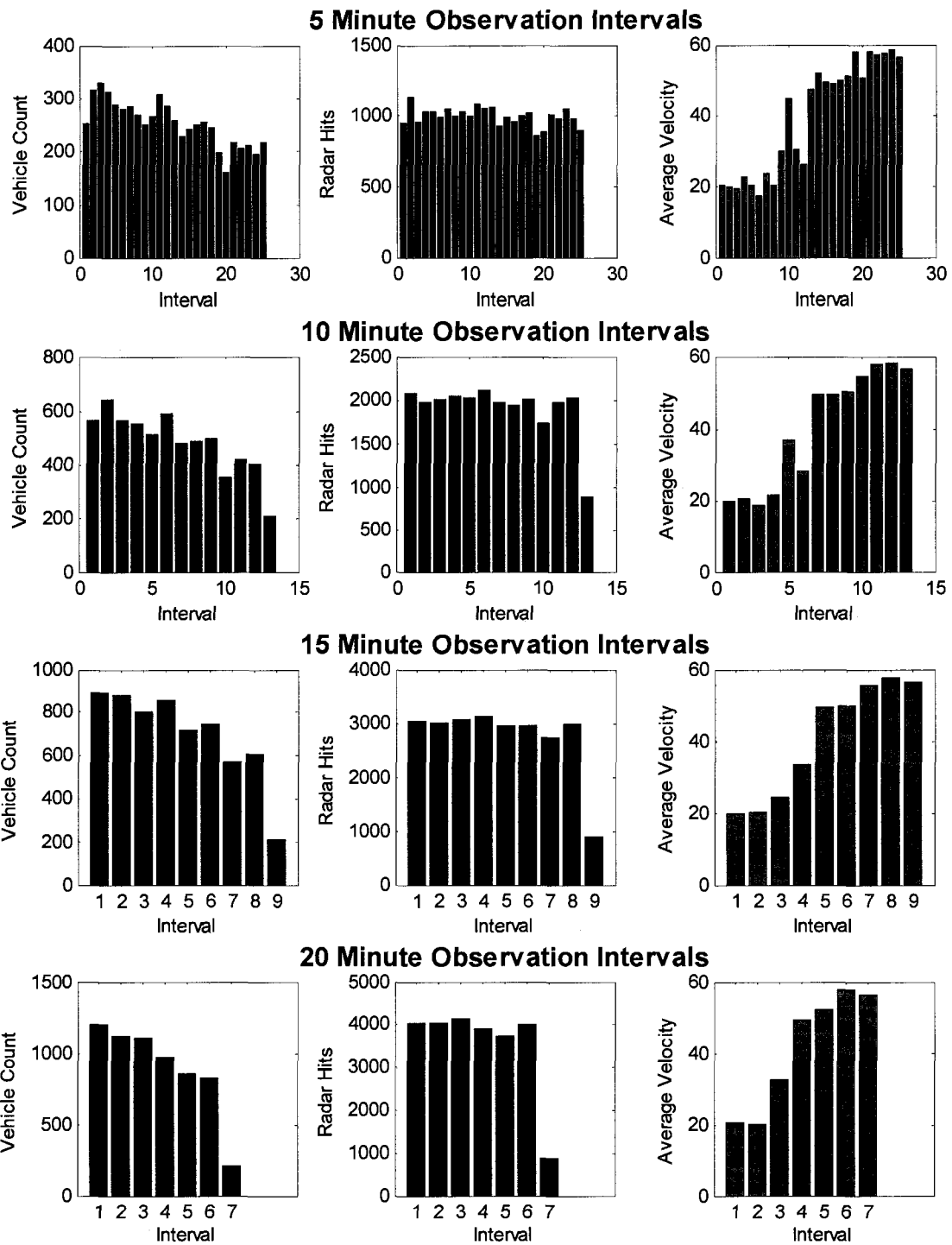


Figure 6.7 - Traffic Dataset #6, 55 mph Multilane Highway

Analysis of the data revealed what appeared to be a correlation between the count of observed vehicles and the number of radar hits recorded during a given time interval. A radar hit was generated each time the P54 Radar application detected a change in the speed measured by the traffic radar unit. The presence of additional vehicles on the roadway tended to generate additional radar hits due to small differences in their measured speed as they moved into and out of the field of observation of the radar unit. Additionally, the number of radar hits recorded per vehicle observed remained relatively constant for a given location as is shown in Table 6.1.

It was desired to determine if a linear relationship exists between the count of vehicles on the roadway and the number of hits recorded from the radar unit. To accomplish this, the correlation coefficient between the two quantities was calculated with MATLAB for each dataset using the various observation time intervals. The results of these calculations are show in Table 6.2.

Dataset Number	Correlation Coefficient			
	5 Minute Intervals	10 Minute Intervals	15 Minute Intervals	20 Minute Intervals
1	0.5362	0.8938	0.9834	0.9677
2	0.9833	0.9915	0.9934	0.9926
3	0.9033	0.9685	0.9513	0.9991
4	0.8938	0.7432	0.9967	-0.7332
5	0.8632	0.6376	0.8318	0.9977
6	0.6098	0.8091	0.8981	0.9301
Average	0.7983	0.8406	0.9426	0.6923

**Table 6.2 - Correlation between Radar Hits and Vehicle Count**

These results suggest that there is a correlation between the number of vehicles observed on the roadway and the number of radar hits received within a given time interval. On average, fifteen minute intervals produced the highest correlation. This

suggests that 15 minutes is the optimum interval for assessing the number of vehicles on the road based on the number of hits received by the radar unit.

The average speed of traffic per observation interval appears to trend towards the speed limit for the road as the interval increases. This was the case for all datasets except for dataset six shown in Figure 6.7. This dataset was for a multilane highway during stop-and-go rush hour traffic conditions. For these conditions, there is a noticeable deviation from the road speed limit even as the observation interval is increased to 20 minutes. As the volume of traffic lessened, the average interval speed again trended towards the speed limit for the road. This suggests that for cases when traffic volume exceeds roadway capacity, the average speed can provide a clear congestion indicator.

It was desired to determine if a relationship exists between the average traffic speed and the number of vehicles observed per interval. Intuitively, as the number of vehicles on the road increases for a given interval it was expected that there would be a decrease in the average speed, resulting in a negative correlation between these two quantities. To test this, the correlation coefficient between the average speed and the vehicle count was calculated for each dataset at each interval length. The results of these calculations are shown in Table 6.3.

Dataset Number	Correlation Coefficient			
	5 Minute Intervals	10 Minute Intervals	15 Minute Intervals	20 Minute Intervals
1	-0.1984	-0.5370	0.2374	-0.9992
2	0.7131	0.6397	0.8199	-0.8170
3	-0.1080	0.2351	0.2300	0.9770
4	-0.1449	-0.0848	-0.9787	0.6556
5	0.2917	0.4332	0.2263	0.3770
6	-0.8210	-0.8091	-0.7598	-0.7401
Average	-0.0446	-0.0205	-0.0375	-0.0911

**Table 6.3 - Correlation between Average Speed and Vehicle Count**



In general the results of this correlation analysis indicate a negative correlation between the vehicle count and the average speed. For all cases except for dataset six, the correlation was inconsistent for changing interval lengths and tended to be weak. Dataset six demonstrates that for the case of congestion, where traffic volume is near or exceeds roadway capacity, a relatively strong negative correlation does exist between the average speed and the vehicle count per interval. This negative correlation was more pronounced for smaller intervals with 5 minute intervals providing the strongest negative correlation.

The analysis of the collected traffic data suggests that the information available from the radar unit can provide an indication of traffic conditions. This indication is based on the number of radar hits received and the average speed of traffic during a given observation interval. Using the number of radar hits, it is possible to determine whether traffic at a given location is sparse or dense during a 15 minute observation interval. From the average speed, it is possible to detect whether the volume of traffic has exceeded the capacity of the roadway, resulting in congestion.

### **6.3 Traffic Congestion Scoring**

The information required to provide a traffic monitoring service is currently available in all radar equipped NHDS vehicles. For vehicles employing P54, a software module could be added to the system to implement this service. By gathering daily statistics of radar data at specific locations, baseline models of traffic patterns can be created. As new radar data is received, it can be processed by the in-vehicle software and scored against the baseline model to provide an automated assessment of traffic conditions. Congestion data could then be collected using the statewide NHDS communications infrastructure and rebroadcast to the public via datacasting [22].

The purpose of scoring the received radar data is to detect changes in traffic conditions that indicate congestion. Using the radar data, individual intervals can be scored against the entire dataset using two criteria: average traffic speed and traffic density. The average observed speed during a 15 minute interval at a given location can be scored as fast or slow based on the deviation from the mean average speed for the entire dataset at that location. Traffic for a given location during a fifteen minute interval can be scored as sparse or dense by determining the deviation of the radar hit count from the mean for that location.

The z-score is employed to perform the scoring of the interval traffic radar data in terms of speed and density. The z-score is a statistical tool used to evaluate a given observation against a larger population in terms of how many standard deviations the observation is above or below the population mean. The formula for the z-score is given in Equation 6.1.

$$Z = \frac{X - \mu}{\sigma}$$

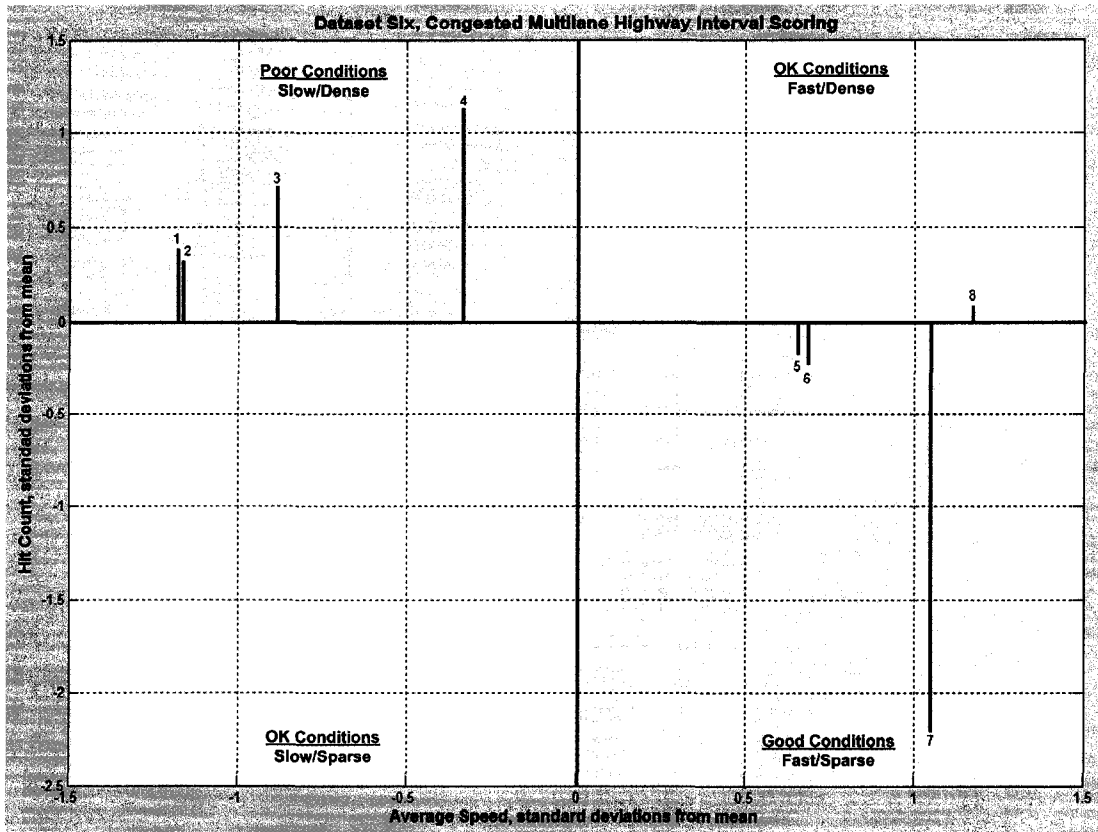
X - the observation to be scored

$\mu$  - population mean

$\sigma$  - population standard deviation

**Equation 6.1 - The Z-Score**

The z-score was used to score the radar data collected for the case of traffic congestion shown in dataset six in Figure 6.7. The average speed and the cumulative number of radar hits during each fifteen minute interval were scored in terms of standard deviation against the mean value of these quantities for the entire dataset. The results are shown in Figure 6.8.



**Figure 6.8 - Congested Multilane Highway Interval Z-Score Results**

In these results only the first eight intervals from dataset #6 have been scored. The reason for this is that interval number nine does not contain a full 15 minutes worth of observations and would therefore skew the resulting mean and standard deviation for the population. This scoring method was able to accurately detect that intervals 1-4 represented congested traffic conditions and that traffic improved for intervals 5-7. For interval number 8, traffic was fast moving but there was a slight increase in the number of radar hits received. These results match the traffic patterns that can be observed for the 15 minute intervals in Figure 6.7.

## **CHAPTER 7**

### **CONCLUSIONS**

This thesis has sought to utilize information resources available within P54 equipped vehicles to support the operations of the NHDS. Telematics services have been implemented for remote diagnostics and fleet management within the P54 system. Additionally this thesis has investigated using traffic radar equipped NHDS vehicles to perform traffic congestion monitoring. These services apply the sensing, computing, and communications capabilities of P54 equipped vehicles to provide benefits to the vehicle operators, fleet managers and the public.

#### **7.1 Vehicle Diagnostics**

A P54 vehicle diagnostics service was implemented for this thesis. Aftermarket diagnostic hardware consisting of an OBD-II scan tool and WTPMS was integrated with the P54 IDB network. The OBD-II scan tool supports all current OBD-II protocols and will provide the diagnostic services detailed in Chapter 3 for all OBD-II equipped vehicles. The valve stem cap pressure sensors of the WTPMS are easy to install and this system will provide early notification of low tire pressures.

P54 software modules were created to configure, monitor and control the diagnostic hardware and make the data available to other P54 application modules using messaging interfaces. Additionally, GUI applications were created for monitoring the diagnostic data using the in-vehicle touchscreen display or from outside of the vehicle

using a PDA. These applications provide vehicle operators with an overview of the status of their vehicle including the tire pressures, OBD sensor readings, descriptions of DTCs, and the status of P54 system components. The vehicle operator can use the GUI applications to clear DTCs once the vehicle has been repaired.

The diagnostic hardware and software has been installed in three test vehicles. Two of the test vehicles are for research and development at P54 and one vehicle has been actively deployed in the service of the NHDS. The diagnostic service has helped to identify DTCs for two of the test vehicles and has provided early warning of low tire pressure conditions. Based on the success of these test deployments, the diagnostic service is ready for full deployment on all P54 equipped vehicles.

The diagnostic software automatically creates the P54 and OBD status log files containing the overview of the vehicle status. These log files include the tire pressures, P54 component errors and any DTCs detected by the system. These files are included in the fleet management reports to provide fleet managers an overview of the vehicles condition. Using the diagnostic service, individual OBD sensors can also be monitored and their values logged for later analysis.

## **7.2 Fleet Management**

A fleet management service was developed for this thesis. Daily records of pertinent data regarding the status and operating expenses for NHDS vehicles can be collected in a series of text files. Vehicle status information is provided by the diagnostics service. Daily operating expenses will be recorded by the vehicle operator using an electronic data entry form. The information will be uploaded from the vehicles on a daily basis using a combination of P25 radios and Wifi hotspots.

Two post-processing applications have been developed to analyze and make use of the collected data. A data plotting application was developed that allows fleet managers to quickly assess the day-to-day status the vehicle fleet and plot recorded OBD-II sensor data. Additionally a MATLAB script has been developed that can extrapolate vehicle fuel economy using the recorded OBD-II sensor values. A web interface is currently under construction which will allow fleet managers to log in and view the information collected at the central NHDS server.

### **7.3 Traffic Monitoring**

This thesis has investigated the use of traffic radar equipped NHDS vehicles to perform traffic congestion monitoring. First, a series of tests were performed to collect radar data for different types of roads and traffic conditions. The data was collected from a stationary vehicle observing oncoming traffic for rural and highway road segments. A count of the observed vehicles was maintained during these tests.

The collected data was then analyzed and it was determined that a correlation exists between the number of radar hits received during a fifteen minute interval and the number of vehicles observed on the roadway. Additionally it was determined that the average observed speed trends toward the speed limit for the road as the observation interval increases, except in cases where congestion is present. Using the z-score, the collected data for each 15 minute observation interval can be evaluated against the cumulative dataset at the given location to score the data in terms of average speed and traffic density. This scoring method accurately identified intervals of congestion for dataset six in Figure 6.7.

## CHAPTER 8

### SUGGESTIONS FOR FUTURE WORK

The implementation of remote diagnostic, fleet management and traffic monitoring services for this thesis has entailed a broad effort requiring the development of multiple software applications. The diagnostics hardware and software has been successfully deployed in the service of the NHDS. The fleet management service is ready for deployment and will rely on the implementation of the P54 vehicle data distribution and collection system [1] and the creation of a web interface for viewing the collected data. The preliminary analysis of the collected traffic radar data suggests that it should be possible to use radar equipped NHDS vehicles to perform congestion monitoring. Future work can expand and improve upon these efforts.

#### 8.1 Vehicle Diagnostics Improvements

##### *8.1.1 Multiple OBD-II Sensor Request Messaging*

The P54 OBD-II application, *OBD-II.dll*, currently performs sequential queries of individual OBD-II sensor values. According to the ISO 15765-4 CAN OBD-II standard, up to six different sensors may be requested simultaneously. This means that for vehicles employing the CAN OBD-II standard (all vehicles manufactured from 2008 on) a significant increase in sensor update frequency can be realized by implementing the multiple OBD-II sensor request messaging functionality. A detailed description of this

messaging format can be found in the *Diagnostic Service Definition for ISO 15765-4 Service \$01* in [54,55]. This will require modification of the IDB thread for *OBD-II.dll*.

### **8.1.2 Adjustable OBD-II Sensor Sampling Rates**

When querying the various OBD sensors, the measured values of some sensors change more frequently than others. For example, the value of the engine RPM will change more frequently than the value of the intake air temperature. Currently, all requested sensors are updated sequentially in a continual loop. To improve the resolution of rapidly varying sensors, a dynamic sensor query delay mechanism should be implemented that will vary the querying frequency of each sensor depending upon how often the sensor value changes. Sensor values that change frequently should be queried more often than sensors that change slowly. This will require modification of the IDB thread for *OBD-II.dll*.

### **8.1.3 Additional OBD-II Diagnostic Service Modes**

There are nine diagnostic service modes defined in [54,55]. The current P54 OBD-II application does not implement all of the available OBD-II diagnostic services. Table 8.1 provides a summary of the diagnostic services currently supported.

	<b>Service</b>	<b>Supported</b>
\$01	Request Current Powertrain Diagnostic Data	Yes
\$02	Request Powertrain Freeze Frame Data	No
\$03	Request Emission-Related Diagnostic Trouble Codes	Yes
\$04	Clear/Reset Emission-Related Diagnostic Information	Yes
\$05	Request Oxygen Sensor Monitoring Test Results	No
\$06	Request On-Board Monitoring Test Results for Specific Monitored Systems	No
\$07	Request Emission-Related Diagnostic Trouble Codes Detected During Current or Last Completed Driving Cycle	No
\$08	Request Control of On-Board System, Test or Component	No
\$09	Request Vehicle Information	No

**Table 8.1 - Summary of Supported OBD-II Diagnostic Services**



Supporting additional diagnostic services would require modification of the IDB thread of *OBD-II.dll*. Additionally, client-server messaging interfaces would need to be created in the OBD-II application so that clients could take advantage of these services. The Vehicle Diagnostics application would need to be modified to support the messaging interfaces provided by the OBD-II application and the GUI would need to present the information from these services to the vehicle operator.

#### ***8.1.4 Inferred Vehicle Parameters***

The values from OBD-II sensors can be combined to infer useful parameters such as fuel economy and distance traveled. The PC and PDA Vehicle Diagnostics applications, *VehicleDiagnostics.dll* and *Diagnostics.dll* respectively, can be modified to support the calculation of these quantities using the sensor readings from the OBD-II application. This would involve adding these options to the user selectable list of OBD sensors in the OBD Setup screen and then implementing the logic to request the appropriate sensors from the OBD-II application and perform the calculations on the received data.

## **8.2 Fleet Management Improvements**

### ***8.2.1 Real Time Updates and Additional Data***

The current fleet management system will feature daily asynchronous updates using the P54 vehicle data distribution and collection system. It is also possible to transmit these reports using the digital police radio network. However, the police radio network is not suitable for real-time updates of all data due to the limited bandwidth. Current wireless technologies such as cellular data modems can provide the required bandwidth to perform real-time updates but the cost would be prohibitive for a large

vehicle fleet such as that employed by the NHDS. With future improvements in the cost, availability and performance of wireless communications, real-time updates should become practical.

Once real-time updates become a viable option, the data collected in the fleet management system can be expanded to include the location of NHDS vehicles and the reports of traffic conditions from the traffic monitoring application. The addition of location information would enable an AVL service that could aid dispatching operations. The combination of traffic reports and an AVL service would allow dispatchers to route NHDS personnel around congested areas.

### ***8.2.2 Web Based Interface***

The fleet management service web interface is currently under construction. This web interface will use the information collected in the log files described in chapter four to provide a status overview of the vehicle fleet and a method for tracking vehicle expenses. Additionally, this interface can be combined with the Diagnostic Data Plotter and other post-processing applications to enable in-depth analysis of recorded vehicle sensor data.

### ***8.2.3 Data-Driven Diagnostics and Maintenance***

The maintenance and status information available in the fleet management system can be used to refine maintenance schedules and develop models for predicting faults. Over time, a large amount of data can be collected about operating conditions of the vehicle. This data can then be analyzed for trends that can help to identify existing or developing problems. The data should be used to optimize the preventative maintenance schedule and provide early notification when corrective maintenance is required.

## **8.3 Traffic Monitoring**

### ***8.3.1 Expanded Data Collection***

The radar data collected up to this point has been limited by the need to have a human observer present to maintain the count of the vehicles. This thesis has demonstrated that a correlation exists between the number of vehicles on a road and the number of radar hits received during a 15 minute observation interval. This implies that future radar data collection can be performed without the need to have a human observer present to count the vehicles. It is desirable to collect data over a 24 hour period at each location in order to observe daily trends. Once a daily baseline has been established, new 15 minute interval data can be scored against this baseline to produce a more accurate assessment of the traffic conditions. Automated test sites consisting of a traffic radar unit, a computer and a power source should be employed for this purpose.

In addition to automating the radar data collection for the stationary on-coming traffic tests, tests should be performed for various observation conditions such as when vehicles are moving away from the radar unit. Tests should also be performed to observe if traffic congestion scoring can be performed from a moving vehicle. If the radar data can be used to detect traffic congestion in these situations, then a traffic monitoring service could be automated within the P54 system without any need for NHDS personnel to modify their normal patrol routines.

### ***8.3.2 Project54 Traffic Monitoring Application***

A P54 Traffic monitoring application should be implemented. This application should combine the information from the traffic radar unit with vehicle speed information provided by the OBD-II application and GPS location samples. Using this data, NHDS

vehicles can act as mobile traffic probes and perform automated traffic congestion scoring using the radar data and the elapsed travel time for road segments based on GPS readings. The P54 Traffic Monitoring application should implement a real-time congestion scoring algorithm and provide an interface to report congestion scores to the central NHDS server. From there the data can be used to route NHDS vehicles around traffic or be provided to the public using a statewide broadcasting system such as datacasting [22].

## BIBLIOGRAPHY

- [1] J. LeBlanc, T. Hurton, W. T. Miller, III, and A. Kun, "Design and evaluation of a vehicle data distribution and collection system," *Fifth International Conference on Pervasive Computing (Adjunct Proceedings) 2007*.
- [2] K. Wilson-Remmer, "The P25Proxy Application for remote messaging," University of New Hampshire, Technical Report ECE.P54.2005.2, 2005.
- [3] N. M. Vichare and M. G. Pecht, "Prognostics and health management of electronics," *Components and Packaging Technologies, IEEE Transactions on [see also Components, Packaging and Manufacturing Technology, Part A: Packaging Technologies, IEEE Transactions on]*, vol. 29, no. 1, pp. 222-229, 2006.
- [4] S. You, M. Krage, and L. Jalics, "Overview of remote diagnosis and maintenance for automotive systems," *SAE Technical Paper Series*, 2005-01-1428 ed. Presented at SAE 2005 World Congress & Exhibition, Vehicle Diagnostics Session, SAE, 2005.
- [5] United States Environmental Protection Agency, "On-Board Diagnostics," 2007.
- [6] United States Environmental Protection Agency, "Control of Air Pollution From New Motor Vehicles and New Motor Vehicle Engines; Modification of Federal On-Board Diagnostic Regulations for: Light-Duty Vehicles, Light-Duty Trucks, Medium Duty Passenger Vehicles, Complete Heavy Duty Vehicles and Engines Intended for Use in Heavy Duty Vehicles Weighing 14,000 pounds GVWR or less," *Federal Register Volume 70, Number 243, FR Doc. 05-23669, Dec.2005*.
- [7] H. O. Marcy, J. R. Agre, C. Chien, L. P. Clare, N. Romanov, and A. Twarowski, "Wireless sensor networks for area monitoring and integrated vehicle health management applications," Presented at AIAA Space Technology Conference and Exposition, Albuquerque, NM: AIAA, 1999.
- [8] J. A. M. Polar, D. S. Silva, A. L. Fortunato, L. A. C. Almeida, and C. A. Dos Reis Filho, "Bluetooth sensor network for remote diagnostics in vehicles," in *Industrial Electronics, 2003. ISIE '03. 2003 IEEE International Symposium on Industrial*

Electronics, 2003.ISIE '03.2003 IEEE International Symposium on, 1 ed. 2003, pp. 481-484.

- [9] United States Department of Transportation, "Tire Pressure Monitoring System FMVSS No. 138," Office of Regulatory Analysis and Evaluation, National Center for Statistics and Analysis, Final Regulatory Impact Analysis, Mar.2005.
- [10] United States Department of Transportation, "Federal Motor Vehicle Safety Standards; Tire Pressure Monitoring Systems; Controls and Displays; Final Rule," National Highway Traffic Safety Administration, Federal Register / Vol. 70, No. 67,49 CFR Parts 571 and 585, Docket No. NHTSA 2005-20586, Apr.2005.
- [11] S. Y. Kim, "Prototype remote vehicle diagnostics for police cruisers." Masters Thesis in Electrical Engineering, University of New Hampshire, 2006.
- [12] G. A. Giannopoulos and G. Boulougaris, "European transport telematics and implications for the development of advanced logistics," Vehicle Navigation and Information Systems Conference, 1994 Proceedings 1994, pp. 271-276.
- [13] A. Goel and V. Gruhn, "Integration of telematics for efficient management of carrier operations," e-Business Engineering, 2005.ICEBE 2005.IEEE International Conference on 2005, pp. 404-408.
- [14] Z. Liao, "Taxi dispatching via Global Positioning Systems," *Engineering Management, IEEE Transactions on*, vol. 48, no. 3, pp. 342-347, 2001.
- [15] W. Jenkins, R. Lewis, G. Lazarou, J. Picone, and Z. Rowland, "Real-Time Vehicle Performance Monitoring Using Wireless Networking," Proceedings of 3rd IASTED International Conference on Communications, Internet, and Information Technology 2004, pp. 375-380.
- [16] P. D. Faas and J. O. Miller, "Impact of an Autonomic Logistics System (ALS) on the sortie generation process," Simulation Conference, 2003.Proceedings of the 2003 Winter, 1 ed 2003, pp. 1021-1025.
- [17] S. Y. Kim, K. Wilson-Remmer, A. L. Kun, and W. T. Miller, III, "Remote fleet management for police cruisers," Intelligent Vehicles Symposium, 2005.Proceedings.IEEE 2005, pp. 30-35.

- [18] R. Struble, J. D'Angelo, J. McGannon, and D. Salemi, "AM & FM's digital conversion: how HD radio TM will spur innovative telematics services for the automotive industry," *Vehicular Technology Magazine, IEEE*, vol. 1, no. 1, pp. 18-22, 2006.
- [19] A. Jameel, M. Stuempfle, D. Jiang, and A. Fuchs, "Web on wheels: toward Internet-enabled cars," *Computer*, vol. 31, no. 1, pp. 69-76, 1998.
- [20] S. D. Maclean and D. J. Dailey, "MyBus: helping bus riders make informed decisions," *Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]*, vol. 16, no. 1, pp. 84-87, 2001.
- [21] K. Krstovski, A. Kun, and W. T. Miller, III, "Distributed components for retrieval of driver's license data using a handheld computer," 8th IAESTED International Conference on Software Engineering and Applications (SEA'04) 2004.
- [22] S. Valcourt, K. Chamberlin, B. McMahon, and A. Kun, "Systems engineering of datacasting for public safety vehicles," *Technologies for Homeland Security, 2007 IEEE Conference on 2007*, pp. 45-50.
- [23] P. Vidales and F. Stajano, "The sentient car, context-aware automotive telematics," *Proceedings of First European Workshop on Location Based Service, LBS-2002 London, UK: 2002*.
- [24] J. C. McCall, O. Achler, and M. M. Trivedi, "Design of an instrumented vehicle test bed for developing a human centered driver support system," *Intelligent Vehicles Symposium, 2004 IEEE, 2004*, pp. 483-488.
- [25] N. Utamaphethai and S. Ghosh, "Dicaf: a distributed architecture for intelligent transportation," *Computer*, vol. 31, no. 3, pp. 78-84, 1998.
- [26] H. H. Cheng, B. D. Shaw, J. Palen, B. Lin, C. Bo, and W. Zhaoqing, "Development and field test of a laser-based nonintrusive detection system for identification of vehicles on the highway," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 6, no. 2, pp. 147-155, 2005.
- [27] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Enhancing Security and Privacy in Traffic-Monitoring Systems," *Pervasive Computing, IEEE*, vol. 5, no. 4, pp. 38-46, 2006.

- [28] A. Chen, N. Jain, A. Perinola, T. Pietraszek, S. Rooney, and P. Scotton, "Scaling real-time telematics applications using programmable middleboxes: a case study in traffic prediction," *Consumer Communications and Networking Conference, 2004.CCNC 2004.First IEEE 2004*, pp. 388-393.
- [29] T. Ishizaka, A. Fukuda, and S. Narupiti, "Evaluation of Probe Vehicle System by using Micro Simulation Model and Cost Analysis," *Journal of the Eastern Asia Society for Transportation Studies*, vol. 6, pp. 2502-2514, 2005.
- [30] R. Grossman, M. Sabala, A. Aanand, S. Eick, L. Wilkinson, P. Zhang, J. Chaves, S. Vejcek, J. Dillenburg, P. Nelson, D. Rorem, J. Alimohideen, J. Leigh, M. Papka, and R. Stevens, "Real Time Change Detection and Alerts from Highway Traffic Data," *Proceedings of the 2005 ACM/IEEE conference on Supercomputing* IEEE Computer Society, 2005, p. 69.
- [31] W. J. Fleming, "Overview of automotive sensors," *Sensors Journal, IEEE*, vol. 1, no. 4, pp. 296-308, 2001.
- [32] E. Cianca, R. Prasad, M. De Sanctis, A. De Luise, M. Antonini, D. Teotino, and M. Ruggieri, "Integrated satellite-HAP systems," *Communications Magazine, IEEE*, vol. 43, no. 12, p. suppl, 2005.
- [33] G. Leen and D. Heffernan, "Expanding automotive electronic systems," *Computer*, vol. 35, no. 1, pp. 88-93, 2002.
- [34] M. E. Martin, F. C. Hludik, and W. T. Miller, "The Project54 common interface for the intelligent transportation systems data bus," *Vehicular Technology Conference, 2002.VTC Spring 2002.IEEE 55th, 2 ed 2002*, pp. 679-683.
- [35] C. Bisdikian, I. Boamah, P. Castro, A. Misra, J. Rubas, N. Villoutreix, D. Yeh, V. Rasin, H. Huang, and C. Simonds, "Intelligent pervasive middleware for context-based and localized telematics services," *Proceedings of the 2nd international workshop on Mobile commerce* Atlanta, Georgia, USA: ACM Press, 2002, pp. 15-24.
- [36] D. Reilly and A. Taleb-Bendiab, "A service-based architecture for in-vehicle telematics systems," *Distributed Computing Systems Workshops, 2002.Proceedings.22nd International Conference on 2002*, pp. 741-742.



- [37] M. Kim, Y. Choi, Y. Moon, S. Kim, and O. Kwon, "Design and implementation of status based application manager for telematics," *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, 2 ed 2006, p. 3.
- [38] W. T. Miller, III, "Remote Project54 application messaging via the Proxy Application," University of New Hampshire, Technical Report ECE.P54.2003.4, 2003.
- [39] A. Karimi, J. Olsson, and J. Rydell, "A Software Architecture Approach to Remote Vehicle Diagnostics." Master Thesis in Informatics, IT University of Göteborg, Göteborg University and Chalmers University of Technology, 2004.
- [40] J. Munson, SW. Lee, D. Lee, D. Wood, G. Thompson, and A. Cole, "A rule-based system for sense-and-respond telematics services," *Proceedings of the 2005 workshop on End-to-end, sense-and-respond systems, applications and services* Seattle, Washington: USENIX Association, 2005, pp. 31-36.
- [41] A. L. Kun, W. T. Miller, III, A. Pelhe, and R. L. Lynch, "A software architecture supporting in-car speech interaction," *Intelligent Vehicles Symposium 2004*, IEEE, 2004, pp. 471-476.
- [42] A. Pelhe, A. L. Kun, and W. T. Miller, III, "Project54 system software architecture," *Proceedings of the winter international symposium on Information and communication technologies* Cancun, Mexico: Trinity College Dublin, 2004, pp. 1-6.
- [43] D. A. Patterson, "20<sup>th</sup> century vs. 21<sup>st</sup> century C&C: the SPUR manifesto," *Communications of the ACM*, vol. 48, no. 3, pp. 15-16, 2005.
- [44] T. Giuli, D. Watson, and K. V. Prasad, "The Last Inch at 70 Miles Per Hour," *Pervasive Computing, IEEE*, vol. 5, no. 4, pp. 20-27, 2006.
- [45] J. Krumm, "Inference Attacks on Location Tracks," *Fifth International Conference on Pervasive Computing*, Toronto, Ontario, Canada: 2007.
- [46] S. Duri, J. Elliott, M. Gruteser, X. Liu, P. Moskowitz, R. Perez, M. Singh, and J.-M. Tang, "Data protection and data sharing in telematics," *Mobile Networks and Applications*, vol. 9, no. 6, pp. 693-701, 2004.

- [47] S. Duri, M. Gruteser, M. Liu, P. Moskowitz, R. Perez, M. Singh, and J.-M. Tang, "Framework for security and privacy in automotive telematics," *Proceedings of the 2nd international workshop on Mobile commerce* Atlanta, Georgia, USA: ACM Press, 2002, pp. 25-32.
- [48] J. L. Harbluk and Y. I. Noy, "The impact of cognitive distraction on driver visual behavior and vehicle control," Transport Canada, Ottawa, Canada, TP#13889 E, 2002.
- [49] B. Champoux, "A mode of interaction for driver vehicle interface (DVI)," *Intelligent Vehicles Symposium, 2005 Proceedings IEEE*, 2005, pp. 795-800.
- [50] M. Kumar and T. Kim, "Dynamic speedometer: dashboard redesign to discourage drivers from speeding," *CHI '05 extended abstracts on Human factors in computing systems* Portland, OR, USA: ACM Press, 2005, pp. 1573-1576.
- [51] A. L. Kun, W. T. Miller, III, and W. H. Lenharth, "Evaluating the user interfaces of an integrated system of in-car electronic devices," *Intelligent Transportation Systems, 2005 Proceedings. 2005 IEEE* 2005, pp. 953-958.
- [52] A. L. Kun, T. Paek, and Z. Medenica, "The Effect of Speech Interface Accuracy on Driving Performance," *Interspeech 2007* Antwerp, Belgium, August 27-31: 2007.
- [53] Z. Medenica and A. L. Kun, "Comparing the influence of two user interfaces for mobile radios on driving performance," *Driving Assessment 2007*, Stevenson, WA, July 9-12: 2007.
- [54] International Organization for Standardization, "Road vehicles -- Communication between vehicle and external equipment for emissions-related diagnostics -- Part 5: Emissions-related diagnostic services," ISO 15031-5:2006, Nov.2007.
- [55] Society of Automotive Engineers, "E/E Diagnostic Test Modes," *Vehicle E/E System Diagnostic Standards Committee*, SAE J1979, May2007.
- [56] Elm Electronics, "ELM327 OBD to RS232 Interpreter," *ELM327 Datasheet, ELM327DSA*, 2007.

- [57] Project54, "Project54 Hardware Manual Version 3.0," Consolidated Advanced Technologies for Law Enforcement Program (CATLab), University of New Hampshire, <http://www.project54.unh.edu/Documents/Brochures,2007>.
- [58] G. T. Pepper, "Methods and Systems for Determining Consumption and Fuel Efficiency in Vehicles," United States Patent US 2007/0129878 A1, June 7, 2007.
- [59] E. Bourbeau, "A Prototype System for Human-Computer Interaction Logging, Post-Processing, and Data Visualization for the Project54 System." Masters Thesis in Electrical Engineering, University of New Hampshire, 2007.
- [60] J. Fawcett and P. Robinson, "Adaptive routing for road traffic," *Computer Graphics and Applications, IEEE*, vol. 20, no. 3, pp. 46-53, 2000.
- [61] F. T. Campos, W. N. Mills, III, and M. L. Graves, "A reference architecture for remote diagnostics and prognostics applications 1," in *AUTOTESTCON Proceedings, 2002. IEEE AUTOTESTCON Proceedings, 2002. IEEE 2002*, pp. 842-853.

## **APPENDICES**

# APPENDIX A

## WIRELESS TIRE PRESSURE MONITORING SYSTEM INSTALLATION

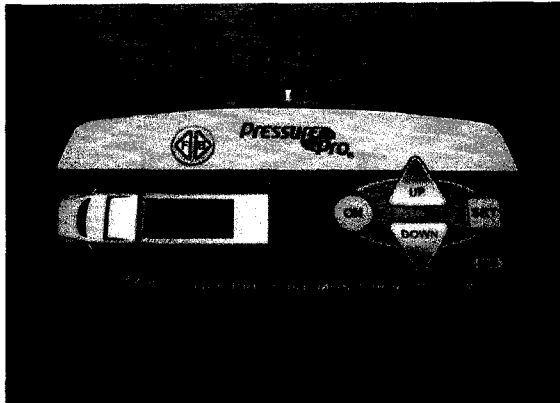


Figure A.1 - WTPMS Device Image

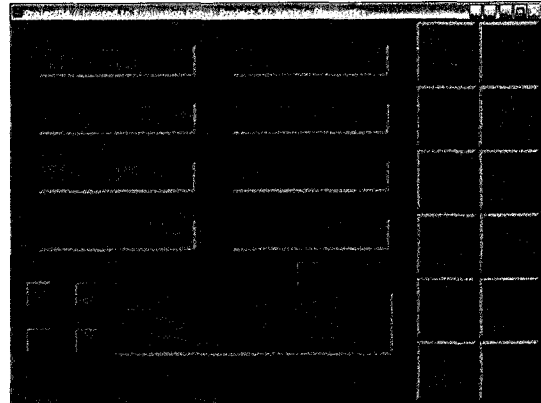


Figure A.2 - WTPMS User Interface

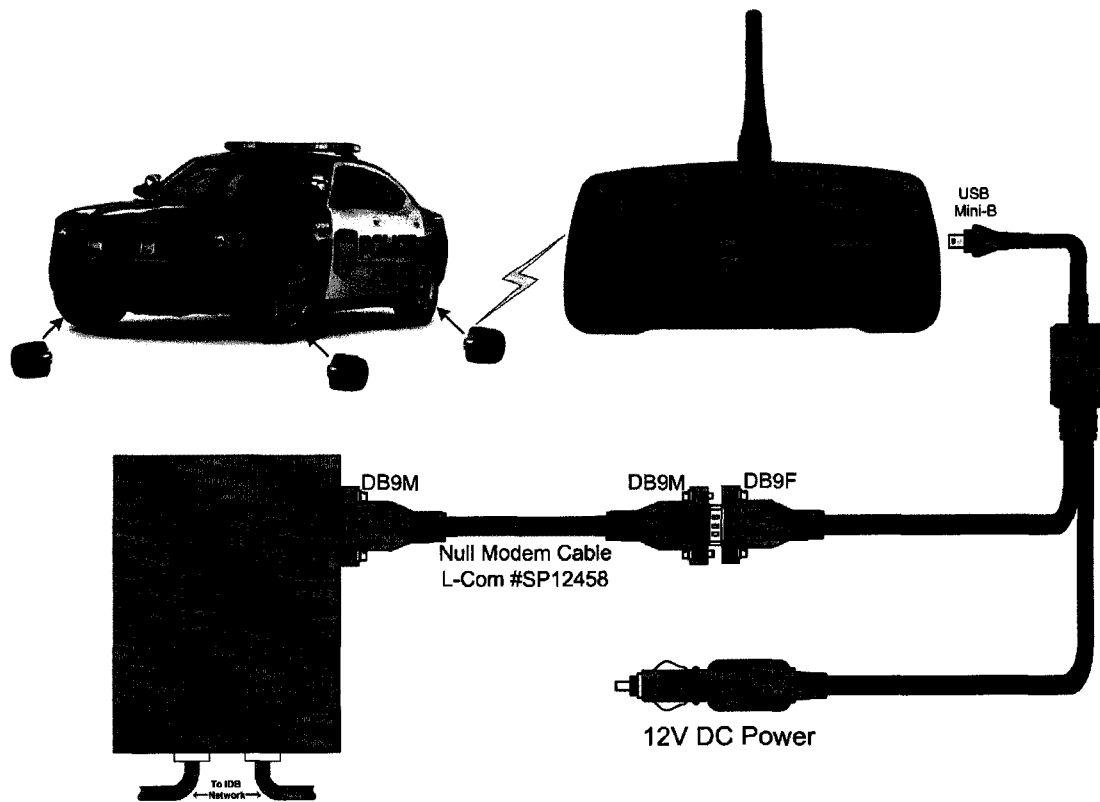


Figure A.3 - WTPMS Connection Diagram

Component	Qty.	Manufacturer	Model No.	Distributor Website
RS-232 Data Feed Capable Monitor	1	PressurePro	HDBPM34-RS232	www.advantagepressurepro.com
Tire Pressure Sensor	4	PressurePro	APS1	www.advantagepressurepro.com
RS-232 Fitted Power Cord	1	PressurePro	AAPC7	www.advantagepressurepro.com
3.5" Monopole Antenna	1	PressurePro	ABPPMA	www.advantagepressurepro.com
1 ft. DB9 null modem cable (male-male)	1	L-Com	SP12458	www.l-com.com
Tire Pressure IDB Box	1	P54	-	www.P54.unh.edu

**Table A.1 - Tire Pressure Monitoring System Parts List**

### Installation Instructions

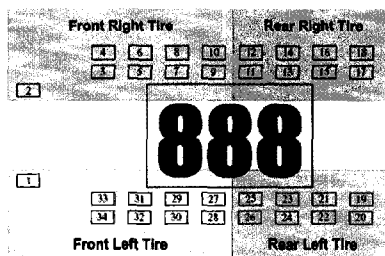
Hardware Connections:

1. Attach the antenna to the monitor.
2. Connect the RS-232 Fitted Power Cord to the monitor using the USB style connector.
3. Connect the null modem cable between the Tire Pressure IDB interface and the RS-232 Fitted Power Cord. Omit the null modem cable if using a new Version 5 style IDB box.
4. Connect the monitor to an ignition switched 12V DC power source using the cigarette lighter adaptor or by hardwiring (green wire to +12V, black wire to chassis/ground).

Sensor Installation:

1. Remove the valve stem caps from all tires.
2. Inflate all tires to the manufacturer's recommended pressure. This should be done when the tires are cold.
3. Place the PressurePro monitor into Program Mode by holding the square SET button for 5 seconds. The small green light below the ON button should stop flashing and turn solid green.
4. A red light will begin flashing to indicate an available sensor location. Pressing the UP or DOWN button will change the selected location.
5. Screw a sensor onto the valve stem of the tire at the location corresponding to the flashing red light.

NOTE: This system is intended to support 18 wheel tractor trailer trucks; there are 34 available locations on the monitor and only 4 are needed for a typical police vehicle installation. The flashing red light should be within the area corresponding to the tire location shown on the diagram below:



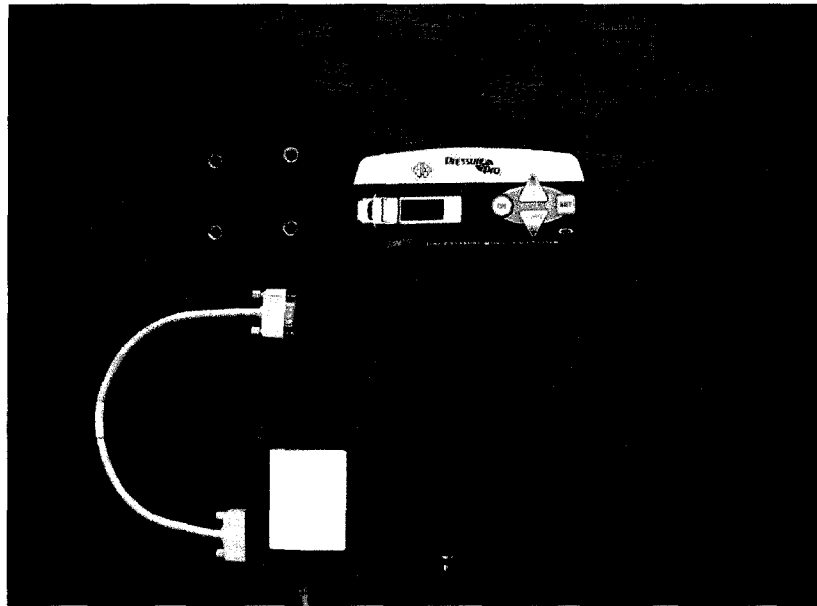
**Figure A.4 - Tire Locations for Monitor Programming**

It does not matter which specific location is selected on the monitor so long as it is within the correct quadrant for the tire where the sensor is being installed.

6. Wait for the monitor to display a pressure reading. This reading will become the baseline pressure for the sensor that was just installed.
7. Hold the SET button until the flashing red location light on the monitor moves to the next tire location.
8. Use the UP or DOWN button to move the flashing red light to the location where the next sensor will be installed. Go to step 5 and repeat this process until all 4 sensors have been installed.
9. Once all four sensors have been installed press the ON button to exit the setup.

**Deleting a sensor location:**

1. Press the UP or DOWN button until the desired sensor location is selected on the monitor.
2. Press and hold the SET button until the monitor displays “Del” (approx. 10 seconds). Deletion is complete.
3. To delete all sensors at once, press and hold the SET button for 45 seconds.



**Figure A.5 - Installed WTPMS Device**

IDB Box Settings			
Name	Address	DIP Switch Configuration	Handshaking
Tire Pressure	1E		

**Figure A.6 - WTPMS Common IDB Interface Settings**

## APPENDIX B

### OBD-II SCAN TOOL INSTALLATION INSTRUCTIONS

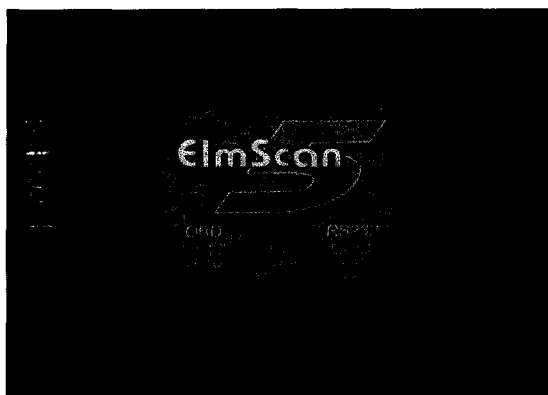


Figure B.1 - OBD-II Scan Tool Device Image

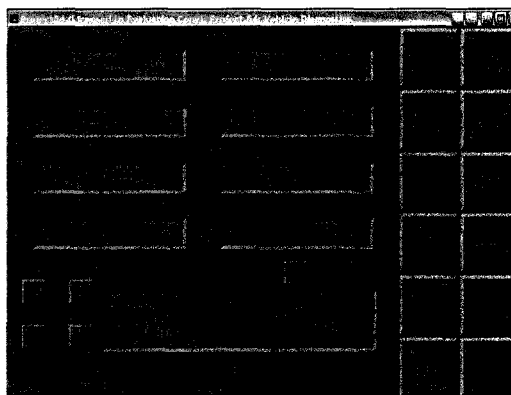


Figure B.2 - OBD-II User Interface

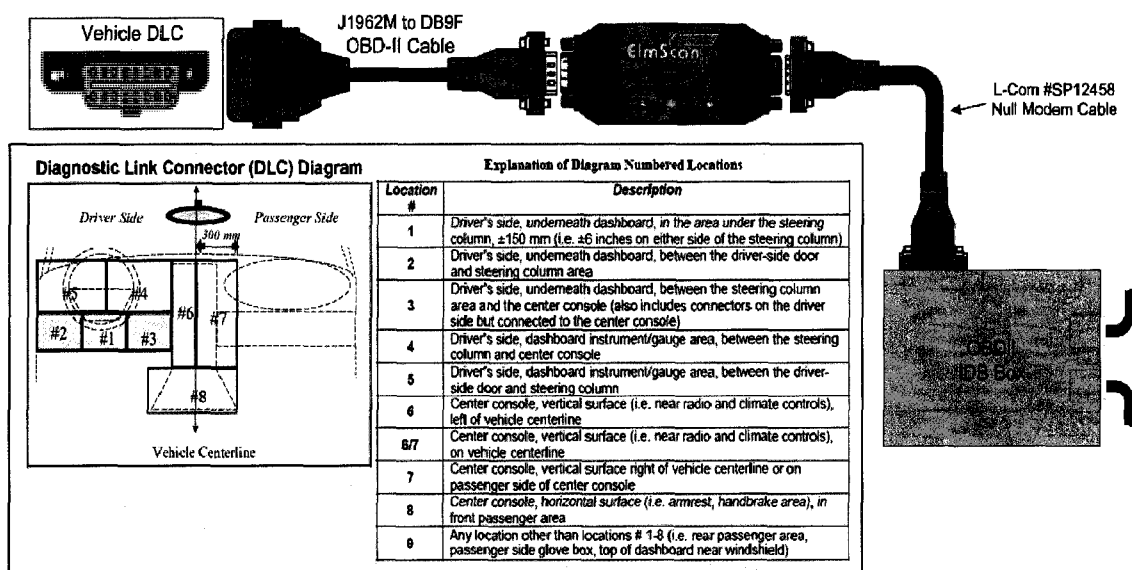


Figure B.3 - OBD-II Connection Diagram

Component	Qty.	Manufacturer	Model No.	Distributor Website	Distributor Part No.
ElmScan 5 Scan Tool	1	ScanTool.net	-	www.scantool.net	421100
1 ft. DB9 null modem cable (male-male)	1	L-Com	SP12458	www.l-com.com	-
OBD-II IDB Box	1	P54	-	www.P54.unh.edu	-
6 ft. J1962M to DB9F, Type D Cable (included)	1	OBD2Cables.com	143301	www.scantool.net	143301

Table B.1 - OBD-II Scan Tool Parts List



### Installation Instructions

1. **OBD-II Cable:** Connect the J1962M end to the vehicle Diagnostic Link Connector (see Figure B.3 for a list of possible DLC locations) and the DB9F end to the OBD side of the ElmScan 5 interface.
2. **IDB Interface Cable:** Connect the null modem cable to the RS-232 side of the ElmScan 5 interface and to the OBD-II IDB Box. Omit the null modem cable and use the standard RS-232 DB9 (male-female) connection if using a new V.5 IDB Box.

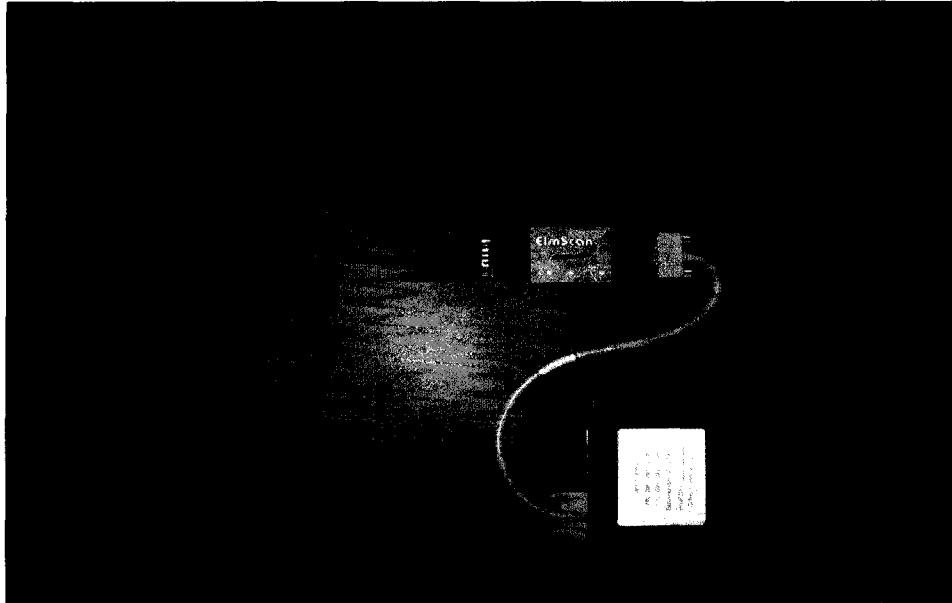


Figure B.4 - Installed OBD-II Scan Tool Device

IDB Box Settings			
Name	Address	DIP Switch Configuration	Handshaking
OBDII	1C		

Figure B.5 - OBD-II Common IDB Interface Settings

## APPENDIX C

### PROJECT54 SUPPORTED OBD-II SENSORS

Index	Sensor	Units
0	Fuel System 1 Status	-
1	Fuel System 2 Status	-
2	Calculated Load Value	%
3	Coolant Temperature	°F
4	Bank 1 Short Term Fuel Trim	%
5	Bank 3 Short Term Fuel Trim	%
6	Bank 1 Long Term Fuel Trim	%
7	Bank 3 Long Term Fuel Trim	%
8	Bank 2 Short Term Fuel Trim	%
9	Bank 4 Short Term Fuel Trim	%
10	Bank 2 Long Term Fuel Trim	%
11	Bank 4 Long Term Fuel Trim	%
12	Fuel Rail Pressure (gauge)	psi
13	Intake Manifold Pressure	inHg
14	Engine RPM	min <sup>-1</sup>
15	Vehicle Speed	mph
16	Ignition Timing Advance	°
17	Intake Air Temperature	°F
18	Mass Air Flow Rate	lb/min
19	Absolute Throttle Position	%
20	Commanded Secondary Air Status	-
21	O2 Sensor Voltage Bank 1 Sensor 1	V
22	O2 Sensor Voltage Bank 1 Sensor 2	V
23	O2 Sensor Voltage Bank (1/2) Sensor (3/1)*	V
24	O2 Sensor Voltage Bank (1/2) Sensor (4/2)*	V
25	O2 Sensor Voltage Bank (2/3) Sensor (1/1)*	V
26	O2 Sensor Voltage Bank (2/3) Sensor (2/2)*	V
27	O2 Sensor Voltage Bank (2/4) Sensor (3/1)*	V
28	O2 Sensor Voltage Bank (2/4) Sensor (4/2)*	V
29	Vehicle OBD Standard	-
30	Power Take-Off Status	-
31	Time Since Engine Start	sec.
32	Distance Traveled with MIL Active	miles
33	Fuel Rail Pressure Relative to Manifold Vacuum	psi
34	Fuel Rail Pressure	psi

Table C.1 - OBD-II Sensors Supported by Project54

Index	Sensor	Units
35	Wide Range O2 Sensor Voltage Bank 1 Sensor 1	V
36	Wide Range O2 Sensor Lambda Bank 1 Sensor 1	-
37	Wide Range O2 Sensor Voltage Bank 1 Sensor 2	V
38	Wide Range O2 Sensor Lambda Bank 1 Sensor 2	-
39	Wide Range O2 Sensor Voltage Bank (1/2) Sensor (3/1)*	V
40	Wide Range O2 Sensor Lambda Bank (1/2) Sensor (3/1)*	-
41	Wide Range O2 Sensor Voltage Bank (1/2) Sensor (4/2)*	V
42	Wide Range O2 Sensor Lambda Bank (1/2) Sensor (4/2)*	-
43	Wide Range O2 Sensor Voltage Bank (2/3) Sensor (1/1)*	V
44	Wide Range O2 Sensor Lambda Bank (2/3) Sensor (1/1)*	-
45	Wide Range O2 Sensor Voltage Bank (2/3) Sensor (2/2)*	V
46	Wide Range O2 Sensor Lambda Bank (2/3) Sensor (2/2)*	-
47	Wide Range O2 Sensor Voltage Bank (2/4) Sensor (3/1)*	V
48	Wide Range O2 Sensor Lambda Bank (2/4) Sensor (3/1)*	-
49	Wide Range O2 Sensor Voltage Bank (2/4) Sensor (4/2)*	V
50	Wide Range O2 Sensor Lambda Bank (2/4) Sensor (4/2)*	-
51	Commanded EGR	%
52	EGR Error	%
53	Commanded Evaporative Purge	%
54	Fuel Level Input	%
55	Warm-ups since DTC cleared	-
56	Distance since DTC cleared	miles
57	Evaporative System Vapor Pressure	in H2O
58	Barometric Pressure	inHg
59	Wide Range O2 Sensor Current Bank 1 Sensor 1	mA
60	Wide Range O2 Sensor Lambda Bank 1 Sensor 1	-
61	Wide Range O2 Sensor Current Bank 1 Sensor 2	mA
62	Wide Range O2 Sensor Lambda Bank 1 Sensor 2	-
63	Wide Range O2 Sensor Current Bank (1/2) Sensor (3/1)*	mA
64	Wide Range O2 Sensor Lambda Bank (1/2) Sensor (3/1)*	-
65	Wide Range O2 Sensor Current Bank (1/2) Sensor (4/2)*	mA
66	Wide Range O2 Sensor Lambda Bank (1/2) Sensor (4/2)*	-
67	Wide Range O2 Sensor Current Bank (2/3) Sensor (1/1)*	mA
68	Wide Range O2 Sensor Lambda Bank (2/3) Sensor (1/1)*	-
69	Wide Range O2 Sensor Current Bank (2/3) Sensor (2/2)*	mA
70	Wide Range O2 Sensor Lambda Bank (2/3) Sensor (2/2)*	-
71	Wide Range O2 Sensor Current Bank (2/4) Sensor (3/1)*	mA
72	Wide Range O2 Sensor Lambda Bank (2/4) Sensor (3/1)*	-
73	Wide Range O2 Sensor Current Bank (2/4) Sensor (4/2)*	mA
74	Wide Range O2 Sensor Lambda Bank (2/4) Sensor (4/2)*	-
75	Catalyst Temperature Bank 1 Sensor 1	°F
76	Catalyst Temperature Bank 2 Sensor 1	°F

Table C.1 Continued - OBD-II Sensors Supported by Project54

Index	Sensor	Units
77	Catalyst Temperature Bank 1 Sensor 2	°F
78	Catalyst Temperature Bank 2 Sensor 2	°F
79	Control Module Voltage	V
80	Absolute Load Value	%
81	Commanded Equivalence Ratio	-
82	Relative Throttle Position	%
83	Ambient Air Temperature	°F
84	Absolute Throttle Position B	%
85	Absolute Throttle Position C	%
86	Accelerator Pedal Position D	%
87	Accelerator Pedal Position E	%
88	Accelerator Pedal Position F	%
89	Commanded Throttle Actuator	%
90	Engine Run-time with MIL Active	hrs, min
91	Time Since DTCs Cleared	hrs, min
92	Battery Voltage	V

**Table C.1 Continued - OBD-II Sensors Supported by Project54**

\* All OBD-II equipped vehicles employ one of two label formats describing the locations of O2 sensors: using up to two banks of cylinders, each supporting up to four sensors, or using up to four banks each supporting up to two sensors. The labels describe the location of the sensor in relation to the engine cylinders. The physical interpretation of the location labels can vary by manufacturer and by the make, model and year of the vehicle. A maintenance manual or the manufacturer can provide further information about the interpretation of these labels for a specific vehicle.

**NOTE:** The OBD-II sensors available for a given vehicle will vary by make and model year. Most vehicles will be equipped with a limited subset of the OBD-II sensors supported by Project54.

## APPENDIX D

### MATLAB SCRIPT FOR MPG ESTIMATION

```
% defined constants
VE = 0.8;           % volumetric efficiency (assumed value)
ED = 5.7;           % Dodge Charger engine displacement (L)
M = 28.97;          % molar mass of air (g/mol)
R = 8.314;          % ideal gas constant (kJ/kmol-K)
AF = 14.64;         % ideal air-to-fuel stoichiometric ratio
P = 0.74;           % fuel density (g/ml)
% fuel trim variables
b1st=0;
b1lt=0;
b2st=0;
b2lt=0;
% average fuel trim variables
alt=0;
ast=0;
% other recorded sensor value variables
iat=0; % intake air temp
rpm=0; % engine RPM
map=0; % Manifold Absolute Pressure
maf=0; % Mass Air flow rate
combo = [0 0 0]; % combine all sensor data in sequential order
% get the directory containing the sensor readings
dname = uigetdir('C:\');
if dname ~= 0
    % Load recorded fuel trim values into arrays
    if exist(strcat(dname, '\Bank 1 S.T. Fuel Trim.txt'),'file')
        ablst = load(strcat(dname, '\Bank 1 S.T. Fuel Trim.txt'));
        ablst(:,3) = 1;
        ablst(:,2) = ablst(:,2)/100+1;
        combo = sortrows(cat(1,combo,ablst),1);
        b1st = combo(find(combo(:,3)==1,1),2);
    end
    if exist(strcat(dname, '\Bank 1 L.T. Fuel Trim.txt'),'file')
        ab1lt = load(strcat(dname, '\Bank 1 L.T. Fuel Trim.txt'));
        ab1lt(:,3) = 2;
        ab1lt(:,2) = ab1lt(:,2)/100+1;
        combo = sortrows(cat(1,combo,ab1lt),1);
        b1lt = combo(find(combo(:,3)==2,1),2);
    end
    if exist(strcat(dname, '\Bank 2 S.T. Fuel Trim.txt'),'file')
        ab2st = load(strcat(dname, '\Bank 2 S.T. Fuel Trim.txt'));
        ab2st(:,3) = 3;
        ab2st(:,2) = ab2st(:,2)/100+1;
        combo = sortrows(cat(1,combo,ab2st),1);
        b2st = combo(find(combo(:,3)==3,1),2);
    end
    if exist(strcat(dname, '\Bank 2 L.T. Fuel Trim.txt'),'file')
        ab2lt = load(strcat(dname, '\Bank 2 L.T. Fuel Trim.txt'));
        ab2lt(:,3) = 4;
```

```

        ab2lt(:,2) = ab2lt(:,2)/100+1;
        combo = sortrows(cat(1,combo,ab2lt),1);
        b2lt = combo(find(combo(:,3)==4,1),2);
    end
    % Load the recorded Intake Air Temperature values into arrays
    if exist(strcat(dname,'\Intake Air Temperature.txt'),'file')
        aiat = load(strcat(dname,'\Intake Air Temperature.txt'));
        aiat(:,3) = 5;
        aiat(:,2) = 5/9*(aiat(:,2)-32); % convert degrees F to C
        combo = sortrows(cat(1,combo,aiat),1);
        iat = combo(find(combo(:,3)==5,1),2);
    end
    % Load the recorded vehicle velocity values into array
    if exist(strcat(dname,'\Vehicle Speed.txt'),'file')
        av = load(strcat(dname,'\Vehicle Speed.txt'));
    end
    % Load the recorded engine RPM values into arrays
    if exist(strcat(dname,'\Engine RPM.txt'),'file')
        arpm = load(strcat(dname,'\Engine RPM.txt'));
        arpm(:,3) = 6;
        combo = sortrows(cat(1,combo,arpm),1);
        rpm = combo(find(combo(:,3)==6,1),2);
    end
    % Load the recorded Manifold Absolute pressure values into arrays
    if exist(strcat(dname,'\Intake Manifold Pressure.txt'),'file')
        amap = load(strcat(dname,'\Intake Manifold Pressure.txt'));
        amap(:,3) = 7;
        amap(:,2) = amap(:,2)*3.386; % convert inHg to kPa
        combo = sortrows(cat(1,combo,amap),1);
        map = combo(find(combo(:,3)==7,1),2);
        MAPCM = (VE*ED*M)/(120*R*AF*P*1000); % MAP constant multiplier
    end
    % Load the Mass Air Flow Rate values into arrays
    if exist(strcat(dname,'\Mass Air Flow Rate.txt'),'file')
        amaf = load(strcat(dname,'\Mass Air Flow Rate.txt'));
        amaf(:,3) = 8;
        amaf(:,2) = amaf(:,2)*453.59237/60; % convert lb/min to grams/s
        combo = sortrows(cat(1,combo,amaf),1);
        maf = combo(find(combo(:,3)==8,1),2);
        MAFCM = 1/(AF*P*1000); % MAF constant multiplier
    end
    % Calculate the average Fuel Trim values
    if exist('abl1t','var') && exist('ab2lt','var')
        alt = (b1lt+b2lt)/2; % average long term trim
    else
        alt = b1lt;
    end
    if exist('abl1st','var') && exist('ab2st','var')
        ast = (b1st+b2st)/2; % average short term trim
    else
        ast = b1st;
    end
    % determine the time difference between samples
    combo(1,:) = [];
    combo(2:length(combo),4) = diff(combo(:,1));

```

```

% determine the amount of fuel consumed
fuel = 0;
time = 0;
for i = 1:length(combo)
    if combo(i,3)==1
        blst=combo(i,2);
    end
    if combo(i,3)==2
        b1lt=combo(i,2);
    end
    if combo(i,3)==3
        b2st=combo(i,2);
    end
    if combo(i,3)==4
        b2lt=combo(i,2);
    end
    if exist('ab1lt','var') && exist('ab2lt','var')
        alt = (b1lt+b2lt)/2;    % average long term trim
    else
        alt = b1lt;
    end
    if exist('ablst','var') && exist('ab2st','var')
        ast = (blst+b2st)/2;    % average short term trim
    else
        ast = blst;
    end
    if combo(i,3)==5
        iat=combo(i,2);
    end
    if combo(i,3)==6
        rpm=combo(i,2);
    end
    if time < 120000 % if elapsed time is less than two minutes
        % calculate the amount of fuel consumed
        if combo(i,3)==7    % if a MAP sensor is used
            map=combo(i,2);
            time = time/1000;
            fuel = fuel + (rpm*map*alt*ast)/(iat+273.12)*time;
            time = 0;
        elseif combo(i,3)==8    % else if a MAF sensor is used
            maf=combo(i,2);
            time = time/1000;
            fuel = fuel + (maf*alt*ast)*time;
            time = 0;
        end
    else
        time = 0;
    end
    % determine the elapsed time between MAP or MAF readings
    time = time+combo(i,4);
end
% apply the appropriate constant multiplier
if exist('MAPCM','var')
    fuel = fuel * MAPCM;
else
    fuel = fuel * MAFCM;
end
end

```

```
% convert the amount of fuel consumed (L) to gallons
fuel = fuel * 0.264172051;
% determine the distance traveled
vt = diff(av(:,1));
dist = sum(av(1:length(av)-1,2)/3600000.*vt);
% determine fuel economy in mpg
mpg = dist/fuel;
end
```



## APPENDIX E

### MATLAB TRAFFIC DATA ANALYSIS SCRIPT

```
clear all;
clc;
close all;
% get the name of the folder containing the traffic data files
dname = uigetdir('C:\Documents and Settings\Ian C_2\Desktop\Thesis
Work\Radar Traffic Tests\');
if dname ~= 0
    % load the velocity and vehicle count data into arrays
    velocity = load(strcat(dname, '\VelocityLog.txt'));
    clicks = load(strcat(dname, '\ClickLog.txt'));
    %remove all velocity = 0 records
    velocity( find(velocity(:,2) == 0),: ) = [];
    % create a combined time index
    time = unique(cat(1,velocity(:,1),clicks(:,1)));
    time(1,2) = .000001;
    time(:,2) = cat(1,time(1,2),diff(time(:,1)));
    figure
    p = 1;
    % observation intervals of 5, 10, 15 and 20 minutes
    for q = 15
        clear ('clickcount','vtotal','vavg','hitcount');
        interval = q * 60000; % min. to ms.
        % determine the number of bins = total_time/interval
        bins = ceil(sum(time(:,2))/interval);
        clickcount(1:bins) = 0;
        vtotal(1:bins) = 0;
        vavg(1:bins) = 0;
        hitcount(1:bins) = 0;
        if(bins>0)
            % bin the data at each time sample
            for n = 1:length(time)
                % figure out which bin we are in
                bin = ceil(sum(time(1:n,2))/interval);
                % if there was a click at the current time sample
                if(~isempty(find(clicks(:,1) == time(n))))
                    clickcount(bin) = clickcount(bin)+1;
                end
                % if there was a radar hit at the current time sample
                if(~isempty(find(velocity(:,1) == time(n))))
                    hitcount(bin) = hitcount(bin)+1;
                end
            end
            vtotal(bin)=vtotal(bin)+velocity(find(velocity(:,1)==time(n),1),2);
            end
            if(hitcount(bin)>0)
                vavg(bin) = vtotal(bin)/hitcount(bin);
            end
        end
        % plot the data
        subplot(4,3,p)
```

```

        bar([clickcount'],'grouped','r');
        xlabel 'Interval'
        ylabel 'Vehicle Count'
        p=p+1;
        subplot(4,3,p)
        bar([hitcount'],'grouped');
        title (strcat([num2str(q),' Minute Observation
Intervals']),'FontWeight','bold','FontSize',14)
        xlabel 'Interval'
        ylabel 'Radar Hits'
        p=p+1;
        subplot(4,3,p)
        bar([vavg'],'grouped','g');
        xlabel 'Interval'
        ylabel 'Average Velocity'
        p=p+1;
    end
    % determine the correlation coefficient for each interval
    d = corrcoef(hitcount,clickcount);
    corcco(q) = d(1,2);
end
%% CONGESTION SCORING, set q to the desired interval prior to this
zv = zscore(vavg);
zh = zscore(hitcount);
figure
bar(zv,zh)
grid ON
end

```

## APPENDIX F

### LOG DATA PARSING TCL SCRIPT

```
# This script will parse the collected vehicle logs and provide a breakdown of the data.
# To run:      1. install ActiveState ActiveTcl
#             2. from the command prompt, chage the directory to the location of this script
#             3. type: tclsh LogParser.tcl
#####
set nday 0
set lines 0
# P54 variables
set p54count 0
set p54err 0
set p54day 0
set p54days 0
set p54list 0
# OBD-II variables
set obdcoun 0
set o2count 0
set supportcount 0
set dtccoun 0
set dtcerr 0
set dtcday 0
set dtcdays 0
set dtclist 0
set pidcount 0
set pid0 0
set pid4 0
set pid6 0
set pid8 0
set pid10 0
set pid13 0
set pid14 0
set pid15 0
set pid17 0
set pid92 0
# Tire Pressure
set tirecount 0
set tireerr 0
set tireday 0
set tiredays 0
set tirebat 0
set tirelow 0
set tirecrit 0
set tirepress 0
set tirelist 0
set n 0
# set the directory where the log files are located
set indir "C:\\lee_charger_logs\\Logs\\Fleet Management\\Vehicle Diagnostics\\"
# !!!! FLAG THIS FOR OLD STYLE LOGS !!!!
set oldstyle 0
```

```

set outdir "$indir\Summary\\"
set ofName "$outdir\summary.txt"
file mkdir $outdir
cd $indir
puts "\nAnalyzing log files in directory:\n$indir\n"
# make sure there are log files in the directory
if { [catch { set days [glob -directory $indir *.txt] } err ] } {
    puts "ERROR: No log files found.\n"
    exit
}
set numDays [length $days]
set prevDay ""
# for each log file (day), open it, look for key phrases and collect results
foreach day $days {
    set dtcday 0
    set p54day 0
    set tireday 0
    if { $day != $prevDay } {
        set newDay 1
        set $prevDay $day
    } else {
        set newDay 0
    }
    set day [lindex $days $nday]
    set fp [open $day r]
    foreach line [split [read $fp] \n] {
        if { [regexp {P54} $line] } {
            incr p54count
            if { [regexp {P54 (.*)} $line match num] } {
                set p54err [expr { $p54err + $num }]
                set j 0
                foreach err [split $line ,] {
                    if { $oldstyle } {
                        if { $j > 3 } {
                            if { [lsearch $p54list $err] == -1 } {
                                lappend p54list $err
                            }
                        }
                    } else {
                        if { $j > 0 } {
                            if { [lsearch $p54list $err] == -1 } {
                                lappend p54list $err
                            }
                        }
                    }
                    incr j
                }
            }
            set p54day 1
        }
        } elseif { [regexp {OBD-II} $line] } {
            incr obdcount
            if { [regexp {OBD-II,PID} $line] } {
                incr pidcount
                if { [regexp {OBD-II,PID 4,} $line] } {
                    incr pid4
                } elseif { [regexp {OBD-II,PID 6,} $line] } {

```

```

        incr pid6
    } elseif { [regexp {OBD-II,PID 8,} $line] } {
        incr pid8
    } elseif { [regexp {OBD-II,PID 10,} $line] } {
        incr pid10
    } elseif { [regexp {OBD-II,PID 13,} $line] } {
        incr pid13
    } elseif { [regexp {OBD-II,PID 14,} $line] } {
        incr pid14
    } elseif { [regexp {OBD-II,PID 15,} $line] } {
        incr pid15
    } elseif { [regexp {OBD-II,PID 17,} $line] } {
        incr pid17
    } elseif { [regexp {OBD-II,PID 0,} $line] } {
        incr pid0
    } elseif { [regexp {OBD-II,PID 92,} $line] } {
        incr pid92
    } else {
        puts "Error:\t$line\n"
    }
} elseif { [regexp {OBD-II,SUPPORTED} $line] } {
    incr supportcount
} elseif { [regexp {OBD-II,O2} $line] } {
    incr o2count
} elseif { [regexp {OBD-II,DTC} $line] } {
    incr dtccount
    if { [regexp {DTC (.*)} $line match num] } {
        set dtcerr [expr { $dtcerr + $num }]
        set dtcday 1
        set j 0
        foreach err [split $line ,] {
            if { $oldstyle } {

                if { $j > 3 } {
                    if { [lsearch $dtclist $err] == -1 } {
                        lappend dtclist $err
                    }
                }
            } else {
                if { $j > 1 } {
                    if { [lsearch $dtclist $err] == -1 } {
                        lappend dtclist $err
                    }
                }
            }
            incr j
        }
    }
}
} elseif { [regexp {Tire} $line] } {
    incr tirecount
    if { [regexp {Low Battery} $line] || [regexp {Battery Low} $line] } {
        incr tirebat
        incr tireerr
        set tireday 1
    } elseif { [regexp {Low Pressure} $line] } {

```

```

        incr tirelow
        incr tireerr
        set tireday 1
    } elseif { [regexp {Pressure Critical!} $line] } {
        incr tirecrit
        incr tireerr
        set tireday 1
    } elseif { [regexp {psi} $line] } {
        incr tirepress
    } else {
        puts "Error:\t$line\n"
    }
} elseif { $line == "" } { # EOF
    incr n
} else {
    puts $line
}
#count the total number of lines
incr lines
}
close $fp
incr nday
if { $p54day } {
    incr p54days
}
if { $dtcday } {
    incr dtcdays
}
if { $tireday } {
    incr tiredays
}
}
set p54u ""
for [61] { $i < [llength $p54list] } { incr i } {
    set p54u $p54u[lindex $p54list $i]\n
}
set dtcu ""
for [61] { $i < [llength $dtclist] } { incr i 2 } {
    set dtcu "$dtcu[lindex $dtclist $i] [lindex $dtclist [expr { $i+1 } ]]\n"
}
set summary "
Days Logged:\t$numDays
Data Points:\t[expr { $lines - $n }]

***** OBD-II RESULTS *****
Data Points:\t\t\t$obdcount
O2 Label messages:\t\t$o2count
Supported PID messages:\t\t$supportcount
Sensor Readings:\t\t$pidcount
PID 0 Fuel System Status:\t$pid0
PID 4 Bank 1 S.T. Trim:\t$pid4
PID 6 Bank 1 L.T. Trim:\t$pid6
PID 8 Bank 2 S.T. Trim:\t$pid8
PID 10 Bank 2 L.T. Trim:\t$pid10
PID 13 Intake Manifold Press.:\t$pid13
PID 14 Engine RPM:\t\t$pid14

```

PID 15 Vehicle Speed:\t\t\$pid15  
PID 17 Intake Air Temperature:\t\t\$pid17  
PID 92 Battery Voltage:\t\t\$pid92  
DTC Checks:\t\t\t\$dtccount  
DTC Errors:\t\t\t\$dtcerr  
Days with DTCs:\t\t\t\$dtcdays  
Unique Errors:\t\t\t[expr { [expr [length \$dtclist]-1]/2 }]  
\$dtcu

\*\*\*\*\* P54 RESULTS \*\*\*\*\*

Data Points:\t\t\$P54count  
Errors:\t\t\t\$P54err  
Days with Errors:\t\t\$P54days  
Unique Errors:\t\t[expr { [length \$P54list]-1 }]  
\$P54u

\*\*\*\*\* TIRE PRESSURE RESULTS \*\*\*\*\*

Data Points:\t\t\$tirecount  
Errors:\t\t\t\$tireerr  
Battery Low:\t\t\$tirebat  
Low Pressure:\t\t\$tirelow  
Pressure Critical:\t\t\$tirecrit  
Tire PSI Updates:\t\t\$tirepress  
Days with Errors:\t\t\$tiredays  
\n"  
puts \$summary  
puts [open \$ofName w] \$summary