SOFTWARE ENGINEERING

Fleet Management

Software Specification Requirement

Patrick Bair pjb4752@sru.edu

Anthony Christe ajc2245@sru.edu

**Modified starting 2/6/2013**

Kelly Smith kms3458@sru.edu

Mitch Nemitz mjn0588@sru.edu

Alex McCracken ajm3652@sru.edu

# Document Revisions

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Primary Authors | Description of Version | Date Completed |
| 1.0 | Patrick Bair, Anthony Christe | Initial specifications of the project, based on primary requirements. | 1/26/2010 |
| 1.1 | Patrick Bair, Anthony Christe | Updates to primary requirements based on new features needed. Merged with formal requirements of CpSc 488 groups from Spring 2009. | 2/1/2010 |
| 1.2 | Patrick Bair, Anthony Christe | Updated to primary requirements based on the migration of this software from a Desktop Application to an Online Application. | 2/15/2010 |
| 1.3 | Kelly Smith, Mitch Nemitz, Alex McCracken | Updated primary requirements to reflect a new iteration of this project. Moving forward this document will evolve to reflect the changes in requirements and implementation of the project. | 2/6/2013 |
| 1.4 | Kelly Smith, Mitch Nemitz, Alex McCracken | Added in a note on completion of portability | 2/20/13 |
| 1.5 | Kelly Smith, Mitch Nemitz, Alex McCracken | Added our UML diagram, Deployment Doc, and several other small changes to wording | 3/13/2012 |
| 1.5.1 | Kelly Smith | Updated Spring to version 3.x and added in a class diagram for action hierarchy and an activity diagram for user authentication. Reorganized document and added requirements to git respository. | 4/9/2013 |

# Contents

Contents

[1 Document Revisions 2](#_Toc351023521)

[2 Contents 3](#_Toc351023522)

[3 Introduction 7](#_Toc351023523)

[3.1 Description 7](#_Toc351023524)

[3.2 Broad Objectives 7](#_Toc351023525)

[4 Project Drivers 9](#_Toc351023526)

[4.1 Purpose of the Project 9](#_Toc351023527)

[4.2 Users 9](#_Toc351023528)

[4.2.1 Stakeholders 9](#_Toc351023529)

[4.2.2 Customers 9](#_Toc351023530)

[4.2.3 Administrative 9](#_Toc351023531)

[4.2.4 Asset Manager 10](#_Toc351023532)

[4.2.5 Operations Manager 10](#_Toc351023533)

[4.2.6 Safety Manager 10](#_Toc351023534)

[4.2.7 Maintenance personal 10](#_Toc351023535)

[4.2.8 Supervisors 10](#_Toc351023536)

[4.2.9 Developers 10](#_Toc351023537)

[4.2.10 Purchase 10](#_Toc351023538)

[5 Project Constraints 10](#_Toc351023539)

[5.1 Solution Constraints 10](#_Toc351023540)

[5.2 Implementation environment of the current system 10](#_Toc351023541)

[5.3 Anticipated workplace environment. 10](#_Toc351023542)

[5.4 Naming Conventions 11](#_Toc351023543)

[5.5 Relevant Facts and Assumptions 11](#_Toc351023544)

[5.5.1 Cost of lease 11](#_Toc351023545)

[5.5.2 Terms of the lease 11](#_Toc351023546)

[5.5.3 Purchase price of the vehicle 11](#_Toc351023547)

[5.5.4 Financing vehicle loans 11](#_Toc351023548)

[5.5.5 Disposal/Resale/Parts Recovery 11](#_Toc351023549)

[5.5.6 Cost of disposal 11](#_Toc351023550)

[5.5.7 Resell Value 11](#_Toc351023551)

[5.5.8 Parts Value 11](#_Toc351023552)

[5.5.9 Condition of Assets 12](#_Toc351023553)

[5.5.10 Value 12](#_Toc351023554)

[5.5.11 Contractual Agreements 12](#_Toc351023555)

[5.5.12 Vendor 12](#_Toc351023556)

[5.5.13 Service Provided 12](#_Toc351023557)

[Functional Requirements 12](#_Toc351023558)

[5.6 Current Situation 12](#_Toc351023559)

[**5.6.1** **UML Diagram** 13](#_Toc351023560)

[5.7 Working with the database. 13](#_Toc351023561)

[5.7.1 Policies 14](#_Toc351023562)

[5.7.2 Renewal Dates 14](#_Toc351023563)

[5.7.3 Covered Vehicles 14](#_Toc351023564)

[5.7.4 Active Claims 14](#_Toc351023565)

[5.7.5 Closed Claims 14](#_Toc351023566)

[5.7.6 Costs 14](#_Toc351023567)

[5.7.7 Vehicle Audits 15](#_Toc351023568)

[5.7.8 Unsatisfactory Vehicles 15](#_Toc351023569)

[5.7.9 Driver Deficiencies 15](#_Toc351023570)

[5.7.10 Driver Risk Assessment 15](#_Toc351023571)

[5.7.11 Scheduled Safety Training 15](#_Toc351023572)

[5.7.12 Driving Competency 15](#_Toc351023573)

[5.7.13 Verification 15](#_Toc351023574)

[5.8 Scope of the Project 16](#_Toc351023575)

[6 Functional Requirements and Diagrams 17](#_Toc351023576)

[6.1 Asset Manager Requirements 19](#_Toc351023577)

[6.2 Operations Manager Requirements 28](#_Toc351023578)

[6.3 Safety Manager Requirements 34](#_Toc351023579)

[6.4 Additional Use Cases 38](#_Toc351023580)

[7 Data Requirements 41](#_Toc351023581)

[7.1 Table Types 41](#_Toc351023582)

[Figure BusType Table 41](#_Toc351023583)

[Figure PurchaseInfo Table 42](#_Toc351023584)

[Figure Disposal Table 42](#_Toc351023585)

[Figure Appraisal Table 42](#_Toc351023586)

[Figure AppraiserInfo table 43](#_Toc351023587)

[Figure DealerInfo table 43](#_Toc351023588)

[Figure Warranty Table 43](#_Toc351023589)

[7.2 Class Design 45](#_Toc351023590)

[8 Non-Functional Requirements 46](#_Toc351023591)

[8.1 Look and Feel 46](#_Toc351023592)

[8.2 User Interface Design 46](#_Toc351023593)

[8.3 Interface design by function. 46](#_Toc351023594)

[8.4 Database diagrams 47](#_Toc351023595)

[Figure Class Diagram 47](#_Toc351023596)

[Figure Sequence Diagram 48](#_Toc351023597)

[Figure State Diagram 49](#_Toc351023598)

[8.5 Usability 50](#_Toc351023599)

[8.5.1 Learning 50](#_Toc351023600)

[8.6 Performance 50](#_Toc351023601)

[8.7 Operational 50](#_Toc351023602)

[8.7.1 Environment 50](#_Toc351023603)

[8.7.2 Release Requirements 50](#_Toc351023604)

[8.8 Maintainability and Support 50](#_Toc351023605)

[8.8.1 Changes or Additions 50](#_Toc351023606)

[8.8.2 Serviceability 50](#_Toc351023607)

[8.8.3 Security 51](#_Toc351023608)

[8.8.4 Recovery from failure 51](#_Toc351023609)

[8.8.5 Portability Script 51](#_Toc351023610)

[9 Project Issues 52](#_Toc351023611)

[9.1 Open Issues 52](#_Toc351023612)

[9.2 New Problems 52](#_Toc351023613)

[9.3 Tasks 52](#_Toc351023614)

[9.4 Migration to New Product 52](#_Toc351023615)

[9.5 Risks 52](#_Toc351023616)

[9.6 Costs 52](#_Toc351023617)

[9.7 Budget 52](#_Toc351023618)

[9.8 User Documentation 52](#_Toc351023619)

[9.8.1 User Documentation Requirements 52](#_Toc351023620)

[9.8.2 Deployment Instructions 52](#_Toc351023621)

[1 Git Installation 52](#_Toc351023622)

[2 Maven Installation 53](#_Toc351023623)

[3 Java Installation and Configuration 53](#_Toc351023624)

[4 Initial Eclipse Setup and Configuration 53](#_Toc351023625)

[5 MySQL Installation 53](#_Toc351023626)

[6 Tomcat 7 Installation 54](#_Toc351023627)

[7 Obtaining Project Source 54](#_Toc351023628)

[8 Obtaining Libraries and Building Binary 54](#_Toc351023629)

[9 Deploying WAR File 55](#_Toc351023630)

[10 Deploying MySQL Server 55](#_Toc351023631)

[10.1 Future Considerations 55](#_Toc351023632)

[10.2 Ideas for Solutions 55](#_Toc351023633)

# Introduction

## Description

The Enterprise Routing System is used to automate the task of scheduling transit times, routes, and maintenance of a fleet of vehicles.

## Broad Objectives

A. Verify the correct operation of the system with full-featured testing

B. Improve reporting capabilities of the system using JasperReports

a. Allow per vehicle reports to be generated for most content

b. Provide a general overview of vehicle reports

c. Add reporting for the following scenarios

1. Fuel consumption for a given time range

2. Fuel consumption over a lifetime

3. MPG estimates

4. Fuel cost per vehicle per fleet

5. Repair costs for both a single vehicle and entire fleet over a period of time

6. Driver expense report

7. Insurance Claims (Active claims and closed claims)

8. Insurance claims per vehicle

9. Safety records per vehicle

10. Safety deficiencies for entire fleet

11. Vehicle summary report for single vehicle

12. Report of all transactions on each Fuel cared or expense report

E. Expand Vehicle conditions to allow user defined rules

F. Improve UI by expanding upon the display frames, including expense and maintenance

G. Add fuel and driver cards for specific accounts and enhance their functionality

H. Examine and complete safety needs

I. Remove hard coded options for Vehicle condition and allow user defined rules.

J. Remove hard coded options for fuel and driver cards and allow user defined rules.

K. User interface improvements

a. Fix frame sizes so that tables fit nicely.

b. Give expense and maintenance their own frames.

c. Add additional functionality to expense and maintenance frames.

d. Add separate buttons for on the side of the program for each section of the program so that the UI is not totally related to the Vehicle Add Form.

e. Format Date in the maintenance tables should only display the date.

f. Create a calendar with time slots for easier maintenance scheduling

M. Redesign expenses so that they reflect Generally Accepted Accounting Principles and so that reports can be generated to be consistent with accounting practices.

# Project Drivers

## Purpose of the Project

This project is designed to provide a simple interface that allows for full management of a fleet of vehicles. This type of management is complex task can require extensive paperwork and labor resources. A single solution that replaces older forms of management can improve the efficiency and effectiveness of the management of a vehicle fleet.

## Users

### Stakeholders

### Customers

Customers for this product would include any business/organization that maintains a fleet of vehicles for transportation or shipping purposes.

### Administrative

Administrators maintain the system and resources.

### Asset Manager

Asset Managers are in charge of overseeing the management of fleet assets.

### Operations Manager

Operations managers are in charge of daily fleet operations.

### Safety Manager

Safety managers oversee the safety of the fleet.

### Maintenance personal

Maintenance personal include anyone that helps maintain the fleet.

### Supervisors

Supervisors are concerned with all aspects of the management of the fleet. They need to access all parts of the project.

### Developers

### Purchase

# Project Constraints

## Solution Constraints

The system must be designed as a general purpose system that can be within multiple organizations.

The solution must be completed within a 14 week time frame.

## Implementation environment of the current system

The user may or may not have a current system in place. The system will be expected to run on any computer that has a modern web browser. Modern web browsers include Safari 4 +, Internet Explorer 7 +, Firefox 3 +, Opera 10 +, and Google Chrome 4 +.

The product will use and manage a relational database containing the pertinent information about the vehicles in the fleet, depots, and contractors. The interface should be easy and intuitive for the users, with tabs containing specialized interfaces for the type of work that the user needs to do with the database, depending on their level of access. Each of the four levels of access will have its own login and access to the database according the responsibility of the position.

The interface will be built upon an existing database. Users will be able to make changes, updates and expansions to the database through the web interface.

## Anticipated workplace environment.

Much of the use will be within an office environment where each user has access to personal computer. This will likely be the case for management personnel. Other user environments may include PC’s or terminals that are operated by one or more personnel in a shop or common area.

## Naming Conventions

Vehicle – may be a truck or a bus

Bus – large passenger vehicle

Truck – large cargo hauling vehicle

Interface – what the user uses to interact with the database

User – Asset, Operations, or Safety Managers or their Supervisors

Asset – bus or truck

Database – contains all the information regarding the vehicles

## Relevant Facts and Assumptions

Certain assumptions must be made when dealing with a fleet of vehicles regarding asset, operations, and safety management must be reflected in and dealt with in our product.

Assumptions regarding asset management that we must include within our product are:

### Cost of lease

Payment amount that may need to be incorporated into our product

### Terms of the lease

Length of the lease, allowable mileage, etc.

### Purchase price of the vehicle

Keep track of how much was originally paid for the vehicle

### Financing vehicle loans

What is due on the vehicle loan if it is not paid off, and how often the payments are due

### Disposal/Resale/Parts Recovery

If a vehicle is no longer in the fleet, what happened to it? Was it sold, junked, used for parts?

### Cost of disposal

Were any costs incurred while disposing the vehicle

### Resell Value

How much was the company able to sell the vehicle for

### Parts Value

If sold for parts, how much were they worth

### Condition of Assets

Is the vehicle damaged, is its mileage high, has it been involved in an accident

### Value

What is the current value of the asset given its age and condition

### Contractual Agreements

What sort of warranties if any are in effect covering the vehicle, and what sorts of things to they cover

### Vendor

What company sold the truck, what if any obligations do they have covering the vehicle

### Service Provided

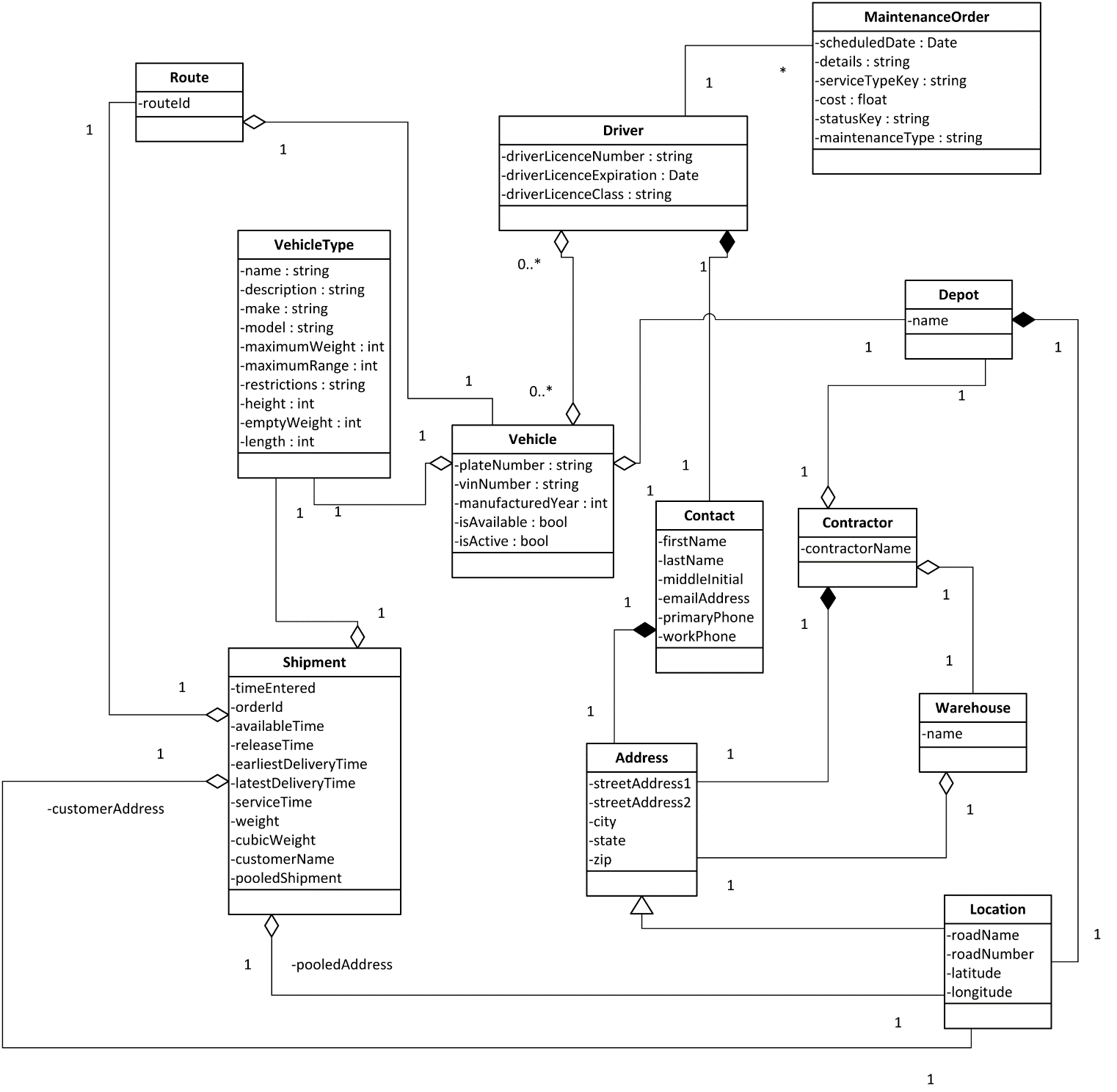
What sorts of service warranties are in effect covering the vehicle?

# Functional Requirements

## Current Situation

The targeted client for this system would include a company with a fleet of vehicles that has grown to a capacity that requires significant management capabilities. The client does not currently have fleet management software in use in their company, and most likely uses paper and pen methods to manage documents and information. A client may also have various separate methods for record keeping such as spreadsheet documents that are hard to manage and collaborate with other co-associates.

### **UML Diagram**



## Working with the database.

Users must keep track of certain events that must be reflected in the database. Every update to the database is driven by an event that occurs within the daily business of managing a fleet. The following is a list of events that the asset manager would use the product to track

|  |  |  |
| --- | --- | --- |
| Event Name | Input & Output | Summary |
| A vehicle is acquired | Enter the information regarding the asset | Certain information needs to be entered for every new bus purchased by the company. This is all entered and displayed on the asset tab. |
| A vehicle is disposed of | Flag the vehicle as disposed and on the inactive vehicles list. Add disposal information to the correct table. | Remove old record, and make a new record storing the required information about the cost of disposal, how much the asset was sold for, or whether it was used for spare parts. |
| A vehicle is appraised | Record results of the latest appraisal. | Record the current condition and the value of the asset. Calculate depreciation based on previous value. |
| Warranty information is entered | Warranty and dealer information is updated in the database. | Keep track of warranties and dealer contact information |

### Policies

The inputs will consist of the policy number, and vehicle ID. The outputs will be a policy description and details.

### Renewal Dates

The inputs will be Vehicle ID and policy number. The outputs will be the length of the policy, when was started, and when should it be renewed. The computation will be the Renewal date and any possible change of policy.

### Covered Vehicles

The inputs will Vehicle ID and Policy number. The outputs will be Type of vehicle and Coverage, which is if the vehicle is covered and for how much and the depreciation of coverage. The computation of the depreciation of coverage will be performed.

### Active Claims

The inputs would vehicle ID, Driver Name, and Driver ID. The outputs will be type of claim, status of claim, and cost of claim. A computation of claims cost will be performed.

### Closed Claims

The inputs will vehicle ID, Driver ID, and Driver Name. The outputs will be total cost of claim, type of claim, any legal penalties of civil or criminal nature, on the company or individuals such as drivers and any changes to company insurance policies resulting from the claim. A computation on the total cost versus insurance deductibles, and any increases in insurance rates will be performed

### Costs

The inputs will be Claim number, type of claim, and gross cost of claim. The outputs will be the results of legal fees, penalties, totals of any increases in insurance rates, and the total cost of all such costs. A computation of insurance rates increases and totaling of all costs will be performed.

### Vehicle Audits

The Inputs will vehicle ID and or type of vehicle. The outputs will be inspection results, Time of Inspection, cost of inspection and time of next Inspection. A computation of when the next inspection will be carried out, as well the cost of the Inspection.

### Unsatisfactory Vehicles

The Inputs will be vehicle ID, Vehicle Type. The Outputs will be Inspection results, reason for failure, and corrective measure to be taken, as well as Cost incurred due to failure and correction. A computation of Cost of Corrective Measures vs. Cost of vehicle will preformed.

### Driver Deficiencies

The inputs will be Driver Name, Driver ID. The outputs will be Driver Handicaps, Driver Criminal records, Disciplinary Actions.

### Driver Risk Assessment

The inputs are Driver Name, Driver ID. The outputs should whether Drivers employment is covered under company policy, legal policy, and insurance. The computation is Driver Risk vs. Policy Coverage.

### Scheduled Safety Training

The inputs are Safety Training schedule, attendance mandatory or voluntary, and type of training. The outputs are Safety Trainer, time of training, attendance of training, and next training event. The computations are number of workers vs. number of attendees at event, and calculation of next safety training session.

### Driving Competency

The inputs are Driver Name, Instructors Name, Driver ID, and Instructor ID. The outputs are Competency test results, time of testing, requirements for competency and location of drivers test. The computation is time of next testing after aforementioned qualification, and a grading scale of success to failure.

### Verification

The inputs are names of training facility or trainer qualifications, type qualification standards such as diploma or certificate, etc. The outputs would name of Driver, Driver ID, success or failure of safety training, and what vehicles that driver is qualified to drive. The computation should be application of success or failure of training and type of training to what vehicles of the fleet driver is qualified to drive.

## Scope of the Project

The project will focus on the roles of three primary users for the management of a fleet. The roles consist of the Asset Manager, Operations Manager and Safety Manager.

Also it’s important to note the how the database is set up.

The current database is set up in the following way. There are four main tables: Vehicles, BusType, Contractors, and Depot.

The following illustration shows how the database relates to itself. Vehicles and BusType have a one-to-many relationship with Depot, since many trucks or busses can be in one depot. Contractors to Depot has a many-to-one relationship, there are more than one contractor that work out of the same depot. BusTypes and Vehicles have a one-to-one relationship because only one contractor can work on a vehicle at a time.

Tables will need to be added to the database that reflect how the vehicle was acquired with cost and terms of the loan or lease. We may also want to add a table for vehicles no longer in the fleet and how they were disposed of. A table will be needed with fields for the condition and appraisal of the assets, this will include the current value, condition, and the condition of individual parts for example the engine or the transmission.

# Functional Requirements and Diagrams



Figure User Roles

## Asset Manager Requirements



Figure : Disposal of Vehicle



Figure : Acquisition of Vehicle



Figure : Vehicle Services



Figure : Vehicle Appraisal



Figure Asset Acquisition User Case



Figure Vehicle Disposal User Case

Figure Vehicle Operations Interaction



Figure



Figure Submit data action



Figure Populate action



Figure Appraisal User Case

## Operations Manager Requirements



Figure Vehicle Operations characteristics User Case



Figure Records and Registration User Case



Figure Expense Management User Case



Figure Maintenance User Case



Figure Driver Reporting User Case



## Safety Manager Requirements



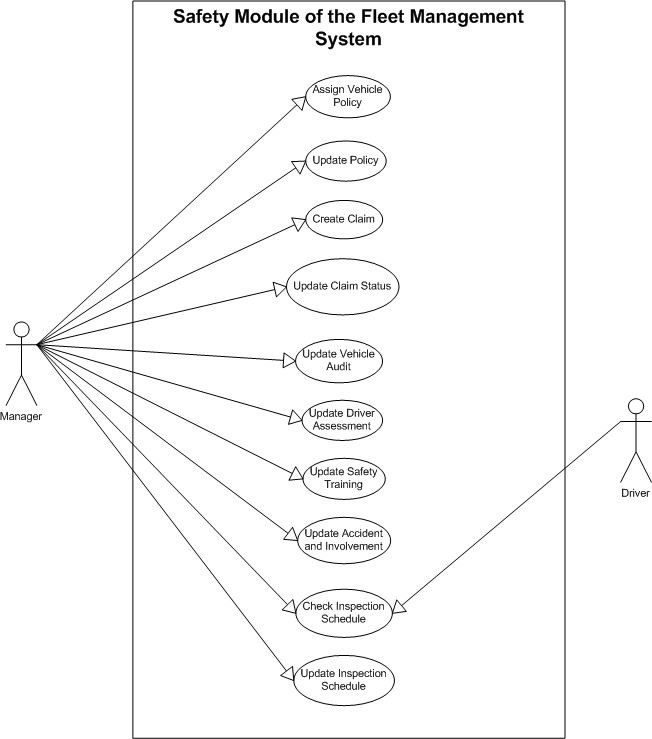
Figure Insurance Management User Case



Figure Claims Management User Case Diagram



Figure Safety Control Measures



## Additional Use Cases

Goals: Give a vehicle an insurance policy  
Preconditions: The policy is not already assigned to the vehicle  
Summary: Manager enters information of policy and vehicle to create connection  
Related Use Cases: Update Policy  
Steps:   
Actor Actions: System Response:   
1. Enter Vehicle Identification 2. Verify vehicle is in system  
3. Enter policy information 4. Show confirmation of success  
Postconditions: Vehicle and policy have a connection

Name: Update Policy  
Actors: Manager  
Goals: Keep the policy info up to date  
Preconditions: The policy information is out of date  
Summary: Manager enters current policy information  
Related Use Cases: Assign Vehicle Policy  
Steps:   
Actor Actions: System Response:   
1. Enter policy information 2. Check for conflicts  
 3. Confirm success  
Postconditions: The policy information is up to date

Name: Create Claim  
Actors: Manager  
Goals: Put insurance claim on file  
Preconditions: The claim is not on file  
Summary: Manager enters claim info into system  
Related Use Cases: Update Claim Status  
Steps:   
Actor Actions: System Response:   
1. Enter Claim information 2. Check for duplicate claim  
 3. Confirm success  
Postconditions: The claim is on file

Name: Update Claim Status  
Actors: Manager  
Goals: Keep the claim information up to date  
Preconditions: Claim information is out of date  
Summary: Manager enters up to date claim information  
Related Use Cases: Create Claim  
Steps:   
Actor Actions: System Response:   
1. Enter Claim Identification 2. Show Claim information  
3. Enter changes 4. Confirm success  
Postconditions: The claim information is up to date

Name: Update Vehicle Audit  
Actor: Manager  
Goals: Keep a log of the vehicle audits  
Preconditions: Vehicle is in system  
Summary: Manager puts most recent audit in log  
Related Use Cases: None  
Steps:   
Actor Actions: System Response:   
1. Enter Vehicle Identification 2. Show audit history  
3. Enter new audit information 4. Confirm success  
Postconditions: The most recent report is in the log

Name: Update Driver Assessment  
Actor: Manager  
Goals: Keep a record of the driver’s abilities  
Preconditions: Driver is in the system  
Summary: Put information of driver’s abilities in system  
Related Use Cases: None  
Steps:   
Actor Actions: System Response:   
1. Enter driver identification 2. Show driver information  
3. Enter new information 4. Confirm success  
Postconditions: Record is up to date

Name: Update Safety Training  
Actor: Manager  
Goals: Prepare the training offered  
Preconditions: The purpose of the training must be known  
Summary: Keeps the training schedule up to date  
Related Use Cases: Update Training Assessment  
Steps:   
Actor Actions: System Response:   
1. Enter training information 2. Confirm success  
Postconditions: Training information is up to date

Name: Update Accident and Involvement  
Actor: Manager  
Goals: to have a record of all accidents  
Preconditions: Accident has occurred  
Summary: Update the accident log  
Related Use Cases: None  
Steps:   
Actor Actions: System Response:   
1. Enter driver identification 2. Show driver information  
3. Enter new competency information 4. Confirm success  
Postconditions: Accident has been recorded

Name: Check Inspection Schedule  
Actors: Manager, Driver  
Goals: Check date of inspection  
Preconditions: Actor has access  
Summary: Check the future dates of inspection for particular vehicles  
Related Use Cases: Update Inspection Schedule  
Steps:   
Actor Actions: System Response:   
1. Log into system with credentials. 2. System grants access to necessary data  
3. Actor looks up the schedule  
Postconditions: The data has remained unchanged

Name: Update Inspection Schedule  
Actor: Manager  
Goals: Make sure inspections are done on time  
Preconditions: Intended schedule is known  
Summary: Manager makes a date for the scheduled inspection   
Related Use Cases: Check Inspection Schedule  
Steps:   
Actor Actions: System Response:   
1. Log into system with credentials. 2. System grants access to necessary data  
3. Actor inputs schedule information 4. Confirm Success  
Postconditions: The schedule has been updated

## Database diagrams



Figure Class Diagram



Figure Sequence Diagram



Figure State Diagram

# Data Requirements

There will be a user interface for the managers of the data to use to manipulate the database. There will be specific tables within the database for each of the managers, and there will be interactions between these tables. All of this will be handled in classes within the program, and accessed via methods.

The first table the acquisition manager needs to interact with is the BusType table.

## Table Types



Figure BusType Table

This is table is where all of the initial information about the newly acquired vehicle is stored. All managers will need to access this table. This is also when the new vehicle is assigned a unique ID number. This ID number is the primary key and will link the separate tables.



Figure PurchaseInfo Table

This table contains all the information about the purchase of the vehicle regarding whether it was a lease or bought out right, loan information and links to the dealer, warranty and appraiser table.



Figure Disposal Table

This table holds all the final information about a vehicle’s disposal. If a vehicle has a disposal record, it should be flagged inactive in the other tables.



Figure Appraisal Table

Periodically the vehicles will need to be inspected and appraised so that the company may keep track of the value of its assets. The appraisal information will be held in this table.



Figure AppraiserInfo table

This table keeps track of all the information about the individual appraisers



Figure DealerInfo table

This table will contain all the information on the dealers that vehicles are purchased from.



Figure Warranty Table

Vehicles may be under warranty or contract to be serviced periodically either by the manufacturer, the dealer, or an outside contractor. That information will be stored in the warranty table.



Figure Data Tables and Relationships

## Class Design

# Non-Functional Requirements

## Look and Feel

The interface should be a simple and easy to use. The use of pop up windows should be avoided so the interface does not appear cluttered and confusing. A modular design is recommended where each user’s responsibilities are grouped into tabs consisting of common functions. The user should be able to log in. Role determination should be made at login and unauthorized roles should be disabled. Buttons on the side of the screen will allow the user to assume any authorized role. When a role is assumed by a user, tabs should change to reflect the functions of that role.

## User Interface Design

## Interface design by function.

## Usability

### Learning

The project should be easy to learn for all users. Any user with basic PC user skills should be able to use the software in a very short amount of time. The user documentation should serve as an adequate reference for most common PC users during the learning process. It is expected that a very entry level PC user could learn to efficiently use the software in a very short amount of time. Administration skills may require advanced PC knowledge or additional support based on the background of the administrator and the actual operating environment.

## Performance

Response time should be fast enough to ensure user efficiency and satisfaction. Maximum response should be less than 2 seconds.

The system should be capable or serving multiple user’s without significant decrease in performance. Therefore, the system must not consume more than 40% of the CPU at any one moment and it also must remain less than 205 in RAM usage.

## Operational

### Environment

This project will be written in Java and tested on Microsoft Windows XP machines. The user will need a computer with access to internet and a modern internet browser.

### Release Requirements

The product will be accessible via and computer with an internet connection and a modern web browser.

## Maintainability and Support

### Changes or Additions

The system design should use an object oriented approach. The design should focus on flexible and reusable components that can be easily modified to change functionality. However scaling the system up may result in the need to replace or rewrite significant portions of the code.

### Serviceability

Updates to the software will be automatic since the software is a web application. The updates process should be transparent to the user. The user will then be notified of the list of changes.

### Security

The first level of security is the system itself it prevents users from accessing the database directly by using encapsulation. The next level will be a login that will prevent users from having access to parts of the database they have no need to alter.

### Recovery from failure

The system runs off a database and therefore will not lose more than the currently processing task in the event of a failure. Any tasks which have completed will not be affected.

### Portability Script

System contains a deployment script that makes it completely portable and able to run under any system regardless of location on the machine.

# Project Issues

## Open Issues

## New Problems

## Tasks

## Migration to New Product

## Risks

## Costs

## Budget

Project limited to currently available software and hardware environments. No budget for additional purchases.

## User Documentation

### User Documentation Requirements

### Deployment Instructions

This project has been successfully built on both Windows and Linux, but this guide will be for Windows. Linux users should be able to follow most of the same steps to setup the following software stack for development and deployment:

Software Stack

* Java 7 JDK

○ Struts 2.x

○ Spring 3.x

○ Hibernate 3.x

* Apache Tomcat 7
* Maven 2.x
* Git
* MySQL 5.x
* Eclipse 4.x

# Git Installation

Git is an open source version control system used throughout the industry to manage large code bases.

Grab and install the latest version of Git: http://git­scm.com/.

For more information about Git, please refer to their getting started guide: http://git­scm.com/book/en/Getting­Started.

# Maven Installation

Download and install the latest version of Maven. If you are on Windows, download the Maven binary and place it somewhere permanent on your computer. Then, set the PATH variable for your system to include the bin directory of the maven install. http://maven.apache.org/download.cgi

# Java Installation and Configuration

At this point it is necessary to pick either 32bit or 64bit for your platform. The version of Java that you choose will determine the versions of all the software to be installed next.

1. http://www.oracle.com/technetwork/java/javase/downloads/jdk7­downloads­1880260.htm

l

1. Download the latest version of the Java SE JDK version 7.x+
2. On Windows, this will install to C:/Program Files or C:/Program Files (x86) depending on whether you chose the 32bit or 64bit version. Make a note of this.
3. Modify your system path so that the bin folder of your Java JDK installation is included (Windows only).
4. Set your JAVA\_HOME environment variable to the root of your Java installation (Windows only).

# Initial Eclipse Setup and Configuration

1. Download the latest version of Eclipse IDE for Java EE Developers. This project was constructed using version 4.2 Juno, however any newer version should work.
   1. http://www.eclipse.org/downloads/.
   2. This project has been successfully built using both 32 and 64 bit versions of eclipse.
   3. If you don’t already have a Java JDK installed, make sure you grab the latest version that matches your eclipse (32 or 64 bit).
2. Install the JBoss Tools plugin for your version of Eclipse.
   1. http://www.jboss.org/tools/download
   2. Find your correct version under Update Sites.
   3. Follow instructions for “Add New Software” in Eclipse.

# MySQL Installation

1. MySQL 5.x is needed for this project. Installation of MySQL is dependent on your operating system, but if you are using Windows a binary installer can be found here:

1. http://www.mysql.com/downloads/installer/
2. This particular installer comes with other useful tools and GUIs for interacting with the database.

# Tomcat 7 Installation

1. Grab the latest version of Tomcat 7 from here:

1. http://tomcat.apache.org/download­70.cgi
2. Unpack this folder somewhere permanent (probably in the same directory as your maven install.
3. This Tomcat server will be integrated with the Eclipse development environment.

Setting Up Project

# Obtaining Project Source

This project is contained within a Git repository. This respository will be made available to you.

(TODO: Edit this to reflect where the next groups repo will be ­ potentially git hub.)

1. On windows use open the **git bash** application that was installed during setup.
2. Navigate (in bash) to the workspace directory that was created when Eclipse was run for the first time. On windows this defaults to: C:\Users\(Your name)\workspace
3. Pull the latest source from the git repo. On windows use git bash: git clone (path to repository)

ex: git clone ssh://git@24.144.172.10:4000/home/git/ers.git

1. This will create a folder of the same name as the repository in your eclipse workspace folder.

# Obtaining Libraries and Building Binary

1. Open up Comand Prompt and navigate to the root directory of your project.
2. Run the following three commands in order: mvn clean, mvn compile, mvn war:war.
   1. Note: if the mvn command cannot be found you have not set up your PATH variable correctly for maven.
   2. Note: if maven says that your JAVA\_HOME variable is missing or invalid, please check that you have set up Java correctly as per the instructions in this document.
3. You now have all of the required libraries downloaded to your target folder and have created a Java WAR binary of the latest source. At this point, this .war file can be deployed to any Tomcat 7 server.

Eclipse Integration

Coming soon.

Deploying Project

# Deploying WAR File

1. Navigate to the *target* folder in your root project directory and locate the .war file created with maven. Rename this file to ERS.war (caps­sensitive).
2. Place ERS.war in the webapps directory of a deployed Tomcat 7 server.
3. Tomcat will automatically extract this to an ERS folder containing the entire web application.

# Deploying MySQL Server

This will only need to be done once (or when the database schema changes).

1. Obtain the ERS.sql file distributed with every release.
2. Import the ERS.sql file into a MySQL database called *ERS* (caps­insensitive).

*Note: The application will by default try to connect to a MySQL database at localhost:3306 with username and password of root and root respectively.*

## Future Considerations

Provide functionality for additional types of fleets.

Provide functionality for the use of an alternate database system other than MS Access.

Integration of Expense Management with accounting system.

## 

## Ideas for Solutions