Department of Computer Science and Engineering
The University of Texas at Arlington

Team Autono-Mo
Jacobia

System Test Plan

Team Members:
Bill Butts
Darius Salemizadeh
Lance Storey
Yunesh Shakya

Last Updated: 20 July 2012 @ 5:45:00 AM
# Table of Contents

List of Figures .......................................................................................................................... v
List of Tables ........................................................................................................................... vi
Introduction .............................................................................................................................. 7
  Purpose .................................................................................................................................. 7
  Scope ..................................................................................................................................... 7
References ................................................................................................................................. 9
  Overview ................................................................................................................................. 9
System Requirements Specification ............................................................................................. 9
  Customer Requirements .......................................................................................................... 9
Architecture Design Specification ............................................................................................... 10
Detailed Design Specification ..................................................................................................... 10
Test Items ................................................................................................................................. 11
  Design Decomposition ........................................................................................................... 11
  Movement Layer Unit Testing ............................................................................................... 13
  Application Layer Unit Testing ............................................................................................ 13
  Navigation Layer Unit Testing ............................................................................................. 14
  Sensor Layer Unit Testing ..................................................................................................... 14
  Movement Layer Subsystem Testing ..................................................................................... 15
  Application Layer Subsystem Testing .................................................................................... 15
  Navigation Layer Subsystem Testing .................................................................................... 15
  Sensor Layer Subsystem Testing .......................................................................................... 16
  Jacobia Layer Testing ............................................................................................................ 17
Risks ......................................................................................................................................... 18
  Strategy ................................................................................................................................. 18
Features to Be Tested .............................................................................................................. 20
  Overview ................................................................................................................................. 20
  Features ................................................................................................................................. 20
    Obstacle Avoidance ............................................................................................................. 20
    Path Navigation .................................................................................................................. 20
Test Deliverables .................................................................................................................. 28
System Test Plan .................................................................................................................. 28
Test Case Specification ......................................................................................................... 28
Test Results .......................................................................................................................... 28
Fail Test Result ..................................................................................................................... 29
Test Code .............................................................................................................................. 29
Test Schedule ......................................................................................................................... 30
Approvals ............................................................................................................................... 31
### List of Figures

<table>
<thead>
<tr>
<th>Figure #</th>
<th>Title</th>
<th>Page #</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Project Scope Overview</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>System Decomposition</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>Example Test Result Form</td>
<td>23</td>
</tr>
<tr>
<td>4</td>
<td>Failed Test Result Documentation</td>
<td>24</td>
</tr>
</tbody>
</table>
### List of Tables

<table>
<thead>
<tr>
<th>Table #</th>
<th>Title</th>
<th>Page #</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Risk Test Plant Severity Table</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>Test Schedule</td>
<td>25</td>
</tr>
</tbody>
</table>
Introduction

Purpose
The purpose of the System Test Plan document is to produce a methodology by which the testing will be based on in order to judge the correctness of Project Jacobia. This methodology will encompass the details that will be followed at every level of abstraction for the system. From the layer level to the module layer, the methodology will provide precise metrics and procedures for ensuring that Project Jacobia adheres to all requirements outlined in the System Requirements Specification.

Scope
The scope of the project is to develop a generic control module that can fit any RC car that, when attached, will allow the RC car to be autonomous. The control module will be delivered along with a software application which will allow a user to configure the control module to the specific RC car it is being used on, as well as supply the RC car with a destination location. The RC car will then travel to the user-supplied destination while avoiding obstacles in its path by employing the use of obstacle avoidance algorithms. The system will be built in such a way as to accommodate all RC cars, however for demonstration and implementation purposes, only the Traxxas type RC car functionality will be used.

The scope of our testing detailed in this document shall cover the wide range of requirements that the system has been defined by in the Systems Requirements Specification. The exceptions to this will be the items described in our Non-Testable Features section.
Figure 1: Project Scope Overview
References

Overview
The references are the information from the SRS, ADS and DDS that are taken into consideration when making test cases for testing the framework outlined in this documentation. These references are the basis for testing whether the capabilities of the system meets the specifications of which the project was created for. Extracts from these documents shall be used for developing test criteria and formulating the severity of the aforementioned tests.

System Requirements Specification

Customer Requirements
The following are the customer requirements from the SRS which describe the requirements of the system per the customer’s input. These requirements are the references that serve as a basis for our formulation for the construction of the tests that will verify that the system meets the needs of the customer. Further information on said requirements can be obtained by referring to Section 3 of Team Autono-Mo’s SRS. The customer requirements are as follows:

- 3.1 Replacement of RC Vehicle Receiver
- 3.2 PPM Signal Channel Selection
- 3.3 PPM Signal Calculation
- 3.4 PPM Signal Communication
- 3.5 Maintain Standard Connections
- 3.6 Secure Module Mounting
- 3.7 Maintain Vehicle Stability
- 3.8 Operation Instructions
- 3.9 System Scale
- 3.10 System Power
- 3.11 Obstacle Avoidance and Navigation Startup
- 3.12 Input GPS Destination
- 3.13 RC Vehicle Trajectory
- 3.14 Start to Finish Travel
- 3.15 RC Vehicle Trajectory Corrections
- 3.16 Obstacle Avoidance
- 3.17 Obstacle Detection
- 3.18 Object-Detection Sensors
- 3.19 Path-finding Algorithm
- 3.20 System Navigation
- 3.21 Wireless Communication
- 3.22 Location Information Transmission
- 3.23 Local Object Information Transmission
• 3.24 Live Video Feed Transmission
• 3.25 Information Reporting
• 3.26 Location Information
• 3.27 Host Application Local Obstacle Display
• 3.28 Configuration File

Architecture Design Specification
The Architecture Design depicts a high to medium level overview of the Jacobia System. Therefore, the ADS shall be referenced for Integration and System testing phases. The testing shall be based on the layer, and subsequently subsystems, designs outlined in the ADS. The ADS shall serve as a guide as to specific desired outputs for respective inputs. This shall greatly enhance the efficiency and validity of our Integration and System testing phases.

Detailed Design Specification
The DDS shall be referenced for the Unit and Component Testing phases. The details conveyed in the DDS shall serve as the basis for the strategy with which the team shall employ while testing the units and components of the system. The Unit Test of each function mentioned in DDS shall be tested separately. Once verification has produced satisfactory results as defined in the Acceptance Criteria section of this document this will elicit the beginning of the Component Testing phase. To reiterate for emphasis, these tests will be created and judged based on the expected results established in the DDS.
Test Items

Design Decomposition
The following is a decomposition of the system design. From this diagram, the main system can be decomposed into testable features. The main system is decomposed into the layers and subsystems from the Architecture Design Specification. Then the layers and subsystems are decomposed into the units of the Detailed Design Specification. From the diagram below, it can be inferred that testing of the individual units will commence at the beginning of the system testing. The testing of integrating units into their respective modules will follow until satisfactory testing results are achieved. This same natural progression would take place at the subsystem and layer level as well.
Figure 2: System Decomposition
### Movement Layer Unit Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLUT1</td>
<td>Emergency Switch</td>
<td>Physical switch flip</td>
<td>RC Vehicle stops moving</td>
<td>Low</td>
</tr>
<tr>
<td>MLUT2</td>
<td>RC Vehicle Configurator</td>
<td>User’s RC vehicle type choice</td>
<td>Movement Command Interpreter variables set to configuration values</td>
<td>Low</td>
</tr>
<tr>
<td>MLUT3</td>
<td>Movement Command Interpreter</td>
<td>Movement commands</td>
<td>PPM signals sent to RC vehicle</td>
<td>Medium</td>
</tr>
</tbody>
</table>

### Application Layer Unit Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALUT1</td>
<td>Application Facilitator</td>
<td>Strings from Sketch(Arduino Code)</td>
<td>Display of the string in X-CTU Terminal</td>
<td>Low</td>
</tr>
<tr>
<td>ALUT2</td>
<td>Presentation</td>
<td>Buttons, text box, combo box, labels</td>
<td>Display of GUI component in the Screen</td>
<td>Low</td>
</tr>
<tr>
<td>ALUT3</td>
<td>Processing</td>
<td>Events for those GUI Components</td>
<td>Display of reaction due to the Events</td>
<td>Low</td>
</tr>
<tr>
<td>ALUT4</td>
<td>Logger</td>
<td>Logger Command</td>
<td>Generation of Log files</td>
<td>Low</td>
</tr>
<tr>
<td>ALUT5</td>
<td>DBManager</td>
<td>Database command</td>
<td>Updating, adding, deleting and accessing items from the database</td>
<td>Low</td>
</tr>
</tbody>
</table>
### Navigation Layer Unit Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLUT1</td>
<td>Localization Calculation</td>
<td>GPS input</td>
<td>GPS output</td>
<td>Low</td>
</tr>
<tr>
<td>NLUT2</td>
<td>Path Calculation</td>
<td>GPS, Threat Description, Heading</td>
<td>Movement Command</td>
<td>Low</td>
</tr>
<tr>
<td>NLUT3</td>
<td>Instruction Handler</td>
<td>Command</td>
<td>Specified Behavior</td>
<td>Low</td>
</tr>
<tr>
<td>NLUT4</td>
<td>Threat Detection</td>
<td>Sensor Information</td>
<td>Threat Summary</td>
<td>Low</td>
</tr>
<tr>
<td>NLUT5</td>
<td>Xbee Wireless Communication</td>
<td>Byte</td>
<td>Byte on Wireless Line</td>
<td>Low</td>
</tr>
<tr>
<td>NLUT6</td>
<td>Navigation Facilitator</td>
<td>Message</td>
<td>Xbee Code</td>
<td>Low</td>
</tr>
</tbody>
</table>

### Sensor Layer Unit Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLUT1</td>
<td>Android GPS Sensor</td>
<td>Satellite GPS Signal</td>
<td>Longitude and Latitude Data</td>
<td>Medium</td>
</tr>
<tr>
<td>SLUT2</td>
<td>Ultrasonic Sensor Array</td>
<td>Ultrasonic Feedback</td>
<td>Analog Signal Proportional to Object Distance</td>
<td>Low</td>
</tr>
<tr>
<td>SLUT3</td>
<td>Android Compass Sensor</td>
<td>Magnetic Reading</td>
<td>Orientation Information</td>
<td>Medium</td>
</tr>
<tr>
<td>SLUT4</td>
<td>getGPS</td>
<td>Longitude and Latitude Reading</td>
<td>Conditioned Longitude and Latitude Information</td>
<td>Low</td>
</tr>
<tr>
<td>SLUT5</td>
<td>getCompass</td>
<td>Orientation Reading</td>
<td>Conditioned Orientation Calculated for Degrees</td>
<td>Low</td>
</tr>
<tr>
<td>SLUT6</td>
<td>getSensorData</td>
<td>Analog Signal</td>
<td>5-Integer Array for Each Sensor Sorted in Descending Order</td>
<td>Low</td>
</tr>
<tr>
<td>SLUT7</td>
<td>sendSensorData</td>
<td>5-Integer Array for Each Sensor Sorted in Descending Order</td>
<td>Byte Code Generated Representing Obstacle Information</td>
<td>Low</td>
</tr>
</tbody>
</table>
### Movement Layer Subsystem Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLST1</td>
<td>Emergency Stop</td>
<td>Physical switch flip</td>
<td>RC Vehicle stops moving</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLST2</td>
<td>PPM Signal Generator</td>
<td>User’s RC vehicle type choice, movement commands</td>
<td>PPM signals sent to servos</td>
<td>Medium</td>
</tr>
</tbody>
</table>

### Application Layer Subsystem Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALST1</td>
<td>Application Communication</td>
<td>Connect Command</td>
<td>Connection Successful with Arduino</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALST2</td>
<td>GUI</td>
<td>GUI Components</td>
<td>Display of those GUI Components</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALST3</td>
<td>App Processing</td>
<td>Events</td>
<td>Successful Event Handler</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALST4</td>
<td>Database</td>
<td>Datas</td>
<td>Update, edit, access and store in database</td>
<td>Low</td>
</tr>
</tbody>
</table>

### Navigation Layer Subsystem Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLST1</td>
<td>Localization</td>
<td>GPS</td>
<td>GPS</td>
<td>Medium</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NLST2</td>
<td>Path Processing</td>
<td>GPS, Heading, Threat Level</td>
<td>Movement Command</td>
<td>Medium</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NLST3</td>
<td>Obstacle Avoidance</td>
<td>Sensory Information</td>
<td>Threat Level</td>
<td>Medium</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NLST4</td>
<td>Navigation Communication</td>
<td>Message</td>
<td>Message</td>
<td>Low</td>
</tr>
</tbody>
</table>
## Sensor Layer Subsystem Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLST1</td>
<td>GPS Locator</td>
<td>Satellite GPS Signal</td>
<td>Longitude and Latitude Data</td>
<td>Medium</td>
</tr>
<tr>
<td>SLST2</td>
<td>Visual Sensing</td>
<td>Ultrasonic Feedback</td>
<td>Analog Signal Proportional to Object Distance</td>
<td>Low</td>
</tr>
<tr>
<td>SLST3</td>
<td>Heading Detection</td>
<td>Magnetic Reading</td>
<td>Orientation Information</td>
<td>Medium</td>
</tr>
<tr>
<td>SLST4</td>
<td>Signal Conditioner</td>
<td>Longitude and Latitude Reading, Orientation Reading, Analog Signal(sensors), 5-Integer Array for Each Sensor Sorted in Descending Order</td>
<td>Conditioned Longitude and Latitude Information, Conditioned Orientation Calculated for Degrees, 5-Integer Array for Each Sensor Sorted in Descending Order, Byte Code Generated Representing Obstacle Information</td>
<td>Medium</td>
</tr>
</tbody>
</table>
## Jacobia Layer Testing

<table>
<thead>
<tr>
<th>ID</th>
<th>Module</th>
<th>Input</th>
<th>Output</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>JST1</td>
<td>Sensor Layer</td>
<td>Sensory data from GPS, compass, ultrasonic devices</td>
<td>Bytecode explaining obstacles surrounding the RC vehicle, and RC vehicle’s heading/current GPS location</td>
<td>Medium</td>
</tr>
<tr>
<td>JST2</td>
<td>Navigation Layer</td>
<td>Sensory layer data, user’s input from application layer</td>
<td>Output of movement commands to movement layer and to application layer</td>
<td>Medium</td>
</tr>
<tr>
<td>JST3</td>
<td>Application Layer</td>
<td>Data exchange between arduino and application</td>
<td>Display of exchanged data between arduino and application</td>
<td>Low</td>
</tr>
<tr>
<td>JST4</td>
<td>Movement Layer</td>
<td>Movement commands, emergency stop switch flipped, user’s RC vehicle type</td>
<td>Sends PPM signals to RC vehicle, stops sending of PPM signals to RC vehicle, servo variables set to configuration values</td>
<td>Medium</td>
</tr>
</tbody>
</table>
Risks

The risks detailed in this section pertain to the risks that will be encountered when Team Autono-Mo undergoes the testing phase of the project. The purpose of the following table, Table 1, is to define each risk, pose a suitable method for minimizing each risk, the team’s strategy in regards to each risk, as well as the severity that each risk poses to the overarching success of the project.

<table>
<thead>
<tr>
<th>Risks</th>
<th>Control Method</th>
<th>Strategy</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware components are damaged as a result of testing</td>
<td>Carefully check wiring of hardware components</td>
<td>Monitor</td>
<td>High</td>
</tr>
<tr>
<td>Test scope fails to encompass all the project requirements</td>
<td>Compare test plan to system requirements</td>
<td>Research</td>
<td>High</td>
</tr>
<tr>
<td>Testing fails to find all bugs in the system</td>
<td>Wide ranges of test input</td>
<td>Monitor</td>
<td>Medium</td>
</tr>
<tr>
<td>Unable to complete all tests</td>
<td>Test according to priority of requirements</td>
<td>Accept</td>
<td>High</td>
</tr>
<tr>
<td>Gold plating of system during testing phase</td>
<td>Do not allow any new features during the testing phase</td>
<td>Monitor</td>
<td>High</td>
</tr>
</tbody>
</table>

Table 1: Risk Test Plan Severity Table

Strategy

The strategy column explicitly defines the plan that the team has in regards to each risk. The following breakdown defines the meaning of each strategy approach outlined in the Risk Table.

Monitor

This strategy means that the team, specifically, the Risk Manager will closely monitor the risk in order to make sure that their control method is indeed working as intended. However, in the event that the control method fails to stymie the risk, the team shall opt to create another control method.

Research

This strategy takes a more academic approach to the risk and pertains to a Risk Manager-led effort to research the effectiveness of the risk and take the actions required to make
sure that the team adheres to the chosen control method. This method differs from the Monitor method due to the fact that there exists only available one control method for the risk, and therefore the team must follow through with it.

Accept

This strategy means that the team will accept the risk and not take any actions on the part of the Risk Manager in order to research or monitor the effectiveness of the control method.
Features to Be Tested

Overview
The following is a list of all of the features that Team Autono-Mo will perform extensive testing on. All of these are essential features to Jacobia, and as such, Jacobia will not function without these features all being complete and working correctly. Each feature will have its testing procedure explained, as well as an explanation of the risk assigned to it.

Features

Obstacle Avoidance
Risk: High
The Obstacle Avoidance feature deals with all things related to avoiding obstacles detected by the ultrasonic sensors of Jacobia. This feature will require a large amount of cases in which it will make a decision on how the car should turn, what speed it should drive at, or whether or not it should just stop altogether. All of these decisions are based around the obstacles surrounding the vehicle, and as such, there are a great deal of test cases that have to be run through in order to find all configurations that Jacobia will have to respond to. Since there will be a large amount of time devoted to figuring out what Jacobia will be deciding to respond to, this feature has been assigned a high risk.

Path Navigation
Risk: High
The Path Navigation feature deals with all things related to determining what final path Jacobia should send the attached RC Vehicle on. All of these decisions are based on things decided by the Obstacle Avoidance subsystem, as well as by the current GPS location and the destination GPS location. This system will require a large amount of testing in order to account for all of the different decisions that should be made based on where the car currently is, as well as the input from the Obstacle Avoidance subsystem. Due to these reasons and the large amount of testing required, this feature has been assigned a high risk.

Wireless Communication
Risk: Medium
The Wireless Communication feature deals with all things related to communicating data wirelessly between the Application Layer of Jacobia and the Navigation Layer. This is the communication between the Jacobia module and the host application. While Team Autono-Mo has already done some testing in this area, there still is a lot of work to be done, and many more
tests need to be completed. However, they will not take a large amount of time. Due to these reason, this feature has been assigned a medium risk.

**Host Application**  
Risk: Medium

The Host Application feature deals with all things related to the user inputting commands and connecting Jacobia to their RC vehicle. This GUI has already been made as a prototype and shows functionality. Most of this feature has already been tested, and as such will not require too much more work. Due to these reasons, this feature has been assigned a medium risk.

**Emergency Stop Button**  
Risk: Medium

The Emergency Stop Button feature deals with the emergency switch that will be mounted on Jacobia in order to allow immediate stopping of the RC vehicle. This feature is only to serve as a physical back-up for the stop command on the host application, but is a necessary safety feature. While this is a fairly simple circuit to construct, and all it will do is cut all power to the RC vehicle servos, it is pertinent that it functions correctly and is thoroughly tested. Due to these reasons, this feature has been assigned a medium risk.

**GPS Destination Input**  
Risk: Low

The GPS Destination Input feature is a subset of the Host Application feature, and allows the user to input the GPS Destination that they wish the RC vehicle to travel to. While this feature is relatively simple, it is important enough to the functioning of the system to warrant an individual feature to test. Due to these reasons, this feature has been assigned a low risk.

**PPM Signal Generation**  
Risk: Low

The PPM Signal Generation feature deals with all tasks required to take a Movement Command input from the Navigation Layer and turn it into something that the RC vehicle can use to actually move. The majority of this feature has already been tested and implemented, and as such will not require much more testing, despite it being an important feature of the Jacobia system. Due to these reasons, this feature has been assigned a low risk.

**Configuration File**  
Risk: Low

The Configuration File feature deals with the configuration file that will be used to store all preset values for generating PPM signals. These configuration files will all be specific to the type
of RC vehicle that the user picks at the beginning of the host application. The configuration file is not an extremely important part of the Jacobia system, and as such, will not require extensive testing. Due to these reasons, this feature has been assigned a low risk.
Features Not To Be Tested

Overview

The following section contains a myriad of features that will not be testing for a various set of reasons. Those reasons range from not being user-visible to being encompassed by another requirement’s test plan. By reading the following section, it will become evident as to what is actually being tested in the System Test Plan. Any Requirements listed in the SRS with a future priority will not be tested, due to the fact that these Requirements will not be implemented.

Requirements Not To Be Tested

3.1.1 Description: The system shall replace the RC vehicle’s receiver as a communication device for the vehicle. An RC vehicle’s initial method of communication is the receiver which communicates with the transmitter (remote control), this receiver will be removed and replaced by connections to the control module.

By testing other features of the system, this will inherently be tested. Therefore, it isn’t necessary to conduct a test just for this requirement.

3.2.1 Description: The system shall choose appropriate channels to send PPM signals to. The module will be connected to several motors and servos which drive the vehicle. Each motor or servo will be on a separate channel. The control module will need to make decisions on which channel to send the PPM signals.

This is not a user-visible requirement. Also, this will be tested through the testing of other requirements that are user-visible.

3.3.1 Description: The system shall calculate the correct PPM signal to send to each of the vehicle’s motors and servos. After the module makes a new trajectory calculation, it will then need to calculate the correct PPM signals to send to the motors and servos.

This is not a user-visible requirement. Also, this will be tested through the testing of other requirements that are user-visible.

3.11.1 Description: The system shall contain software that initializes the Obstacle Avoidance and Navigation System. The system shall contain software that allows the system to calculate where it currently is, what direction it is facing, and several other start-up checks that need to be
run before the RC vehicle starts its journey. The system shall always perform the same checks every time it starts up.

This is not a user-visible requirement. Also, this will be tested through the testing of other requirements that are user-visible.

3.19.1 Description: The system shall use an efficient path-finding algorithm that will find a quick and safe route to the destination GPS coordinates. This algorithm will take into account obstacles found with the on-board sensors.

Having an efficient, quick and safe route is an opinion, and can thus not be logically tested. However, obstacles will be found via other testing.

3.28.1 Description: The host application shall upload the configuration file to the system in order to operate within the bounds of the chosen RC vehicle. This configuration file will include all the pertinent information needed to configure to a certain RC vehicle type.

This is not a user-visible requirement. In addition, it will be tested through the testing of other requirements. Therefore, it is not imperative to create a test for this requirement.

5.3.1 Description: The vehicle will only be designed to operate on solid ground. Mud, water, extreme inclines, and other unfavorable landscapes will not be covered by the vehicle’s capabilities.

We will not test this requirement because the only way to test it would be to operate the RC vehicle in the very environments detailed in the requirement. This could result in physical harm to the system and the RC vehicle.

7.4.1 Description: Team Autono-Mo shall be available to fix software and hardware issues. Should the system require maintenance that the end-user cannot do through the use of the Maintenance Manual, Team Autono-Mo should be contacted in order to fix the issue(s). This service will be provided free of charge.

We will not have a way to test this requirement due to the scope of the project. Therefore, we will not create a test for this requirement.
Testing Strategy and Deliverables

Overall Test Strategy
Team Autono-Mo will conduct extensive testing of the system to ensure that all requirements laid out in the SRS have been met and adhere to the customer’s requests. The System Test Plan will serve as a guide for testing all parts of Jacobia and then Jacobia as a completed system. All results from the tests ran will be recorded in order to complete documentation deliverables.

Configurations
Team Autono-Mo will perform different hardware and software configurations in order to create a system that is satisfactory. All critical and high priority requirements will be implemented and tested first. The team will then also try to implement all medium and low priority requirements, with more importance being placed on the former. Future requirements will most likely not be implemented and will not be tested as such.

Methodology
Team Autono-Mo will go through all testing in this order: Unit, Component, Integration, and System Verification testing. All testing will be done in team meetings, or in the presence of either the hardware or software manager of the team, depending on what is being testing. Autono-Mo will first concentrate on Unit Testing and making sure that no hardware or software by itself is defective and that it performs the correct role set out for it. Once all Unit Testing has been completed, Autono-Mo will then move on to Component Testing in order to test units when combined. After this, Integration Testing will occur, combining all components into their respective layers of the DDS and combining all layers. After this, the system will be tested as a whole to ensure that it is complete and fulfills all requirements set out in the SRS.

Each test will require that certain information is recorded:

- Test Name and Test ID Number
- Test Date
- Tester’s Name
- Input(s)
- Expected Output(s)
- Actual Output(s)
- Test Result (Pass/Fail)
- Problems Encountered (if failed)
- Additional Notes and Suggestions
**Metrics**

The metrics used to determine the priority of each test case will be based upon the priorities of the requirements to be implemented as set forth in the SRS. The order of testing priority will be as follows: Acceptance Criteria, Critical Priority Requirements, High Priority Requirements, Medium Priority Requirements, and finally, Low Priority Requirements.

The metrics used to determine the severity of each test component is based upon the number of other components that would be affected if the component failed to pass its test. The following three levels will be used to designate what test risk severity each component will have:

High: A failure of this component greatly hinders the most important functionalities of the system. An immediate replacement of the component or a complete design change may be required to make the system function correctly.

Medium: A failure of this component slightly hinders the most important functionalities of the system. A replacement of the component or a minor design change may be required to make the system function correctly.

Low: A failure of this component does not hinder the most important functionalities of the system. A substitution or replacement of the component may be required to make the system function correctly.

**Regression Testing**

Normal testing will be performed by Team Autono-Mo until an error or bug is found with a component. Each time one of these is found and addressed, tests will be conducted again on the component. Any components or units that could be affected by this error or bug will be tested again using normal testing methods. Regression Testing will be performed each time a new component is added to the system during Integration Testing. To do this, Team Autono-Mo will test all other related components to the component added. Regression Testing will allow Team Autono-Mo to ensure that the system is still completely operational.
Item Pass/ Fail Criteria

Test Plans and Procedures

Overview
This section assigns a quantifiable definition for the passing or failing of every test that Team Autono-Mo will conduct throughout the system. Each test can be categorized as a Hardware Unit Test, Software Unit Test, Integration Test, or a System Verification Test.

Hardware Unit Test
Pass: The hardware components return the correct outputs based on the valid input provided. In case of invalid input the hardware components shall return an error message or exception.

Fail: The hardware components returns the incorrect outputs based on the valid input provided. In case of invalid input the hardware components shall not return an error message or exception.

Software Unit Test
Pass: The software units return the correct outputs based on the valid input provided. In case of invalid input the software units shall return an error message or exception.

Fail: The software units return the incorrect outputs based on the valid input provided. In case of invalid input the software units shall not return an error message or exception.

Integration Test
Pass: The modules, subsystems, and layers communicate as described in the ADS and DDS.

Fail: The modules, subsystems, and layers do not communicate as described in the ADS and DDS.

System Verification Test
Pass: The system meets the requirements as described in the SRS.

Fail: The system does not meet all the requirements as described in the SRS.
Test Deliverables

Team Autono-Mo will provide the following documentation regarding the testing of Project Jacobia.

System Test Plan

This document will be provided to the reader as a framework for all the test cases which shall be included in the documentation as a test deliverable. The details of test cases, results, as well as a testing guide shall be provided in the accompanying documentation of Project Jacobia.

Test Case Specification

Test cases shall include the following.

- List of all valid inputs and expected outputs.
- List of inputs categorized under normal cases, threshold cases, and exception cases.
- List of expected output tested using inputs categorized under normal cases, threshold cases, and exception cases.

Test Results

Results for all the test cases shall be recorded in the following manner:

- Test Name and Id
- Date of test
- Tester’s name
- Test Items
- Input Specification
- Expected Output Specification
- Actual Output specification
- Test Result(Pass/Fail)
- Test Related Notes

Figure 3: Example Test Result Form
Fail Test Result

In case of a test failure, the following pieces of information shall be recorded in the documentation:

- Test Name and Id
- Date of Test
- Tester’s Name
- Test Failure Description
- Severity (low, medium, high)
- Status (Fixed/Ignored)
- Risk if not resolved

<table>
<thead>
<tr>
<th>Test Name</th>
<th>Test Id</th>
<th>Date of Test</th>
<th>Tester’s Name</th>
<th>Test Failure Description</th>
<th>Severity</th>
<th>Status</th>
<th>Risk if not resolved</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>**</td>
</tr>
</tbody>
</table>

**Figure 4: Failed Test Result Documentation**

Test Code

Test Code which includes test cases that uses the actual code to be delivered in the system shall be included in the product documentation.
## Test Schedule

<table>
<thead>
<tr>
<th>Test</th>
<th>Start</th>
<th>Finish</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit Testing</td>
<td>7/13/2012</td>
<td>7/18/2012</td>
<td>40hrs</td>
</tr>
<tr>
<td>Subsystem Testing</td>
<td>7/18/2012</td>
<td>7/22/2012</td>
<td>40hrs</td>
</tr>
<tr>
<td>Layer Testing</td>
<td>7/22/2012</td>
<td>7/24/2012</td>
<td>40hrs</td>
</tr>
<tr>
<td>System Testing</td>
<td>7/24/2012</td>
<td>7/26/2012</td>
<td>40hrs</td>
</tr>
</tbody>
</table>

Table 2: Test Schedule
The following people approve the testing procedure provided within this document.

<table>
<thead>
<tr>
<th>Name</th>
<th>Role</th>
<th>Signature</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike O'Dell</td>
<td>CEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chris McMurrough</td>
<td>Sponsor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bill Butts</td>
<td>Team Leader</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lance Storey</td>
<td>Hardware Manager</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Darius Salemizadeh</td>
<td>Risk Manager, Software Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yunesh Shakya</td>
<td>Software Manager</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>