26.4.2010

TEST SPECIFICATION REPORT

OCEAN'S 4

SEHIR COBANI

BURAK TIKNAZ 1502749
FATİH PEHLİVAN 1502590
OKAN ÖZMEN 1502947
YAKUP TURGUT 1502780

TABLE OF CONTENTS

1. INTRODUCTION

Sehir Cobani is a kind of GPS (Global Positioning System) software with extra capabilities. Our system is capable of communicating with other Sehir Cobani systems using ad-hoc network structure. This provides many capabilities to our system such as;

- Managing crossroads
- Detecting traffic density
- Transporting accident information
- Warning drivers about optimal following distance
- Assisting to overtake

Accomplishing all of the properties of the project described above requires implementing them properly. However, in order to check whether our project works properly or not, comprehensive testing of the project should be carried out after implementation. This document is intended for clarifying specifications of the testing strategy and methods we will use in our project.

1.1 Goals and objectives

The purpose of the testing phase is to make our software product an error-free, robust and time efficient as much as possible. This document organizes testing process and testing methodology of our project. In this document, the testing strategies which we will use are stated in this document. Additionally, one can find that which part of our system is going to be tested by whom.

1.2 Scope

This document briefly explains testing process. While explaining this process, these steps are going to be clarified in this document:

- Parts/modules that are going to be tested
- Constraints of testing process
- Methods to cope with the bugs, errors
- The relation of testing process with our schedule
- Responsibilities of members for testing different modules

1.3 Major Constraints

1.3.1 Time

Our project Şehir Çobanı is intended to be used on PDA (Personal Digital Assistant) and final software product we will produce will have many properties which should work quickly and respond to the user requests in a rapid manner, that is it should work in real time. Therefore, we should search for the

enhancements; make testing and debugging which speed up our system.

1.3.2 Data

In our project, necessary data will be transferred with ZigBee modules in ad-hoc structure. Since every car broadcasts the message it receives, there will be lots of data that should be handled. Because of this reason, we are implementing kind of algorithms to achieve data transfer we desire to provide the main properties of our project such as overtaking information, crossroad information, etc. Additionally, since our system will send accident information to emergency services and since this should be quick, we will be testing our communication modules and algorithms to accelerate the data transfer. Lastly, map data are stored on the device, thus testing process also includes check for correctness of the map data and its efficiency to gather information from it.

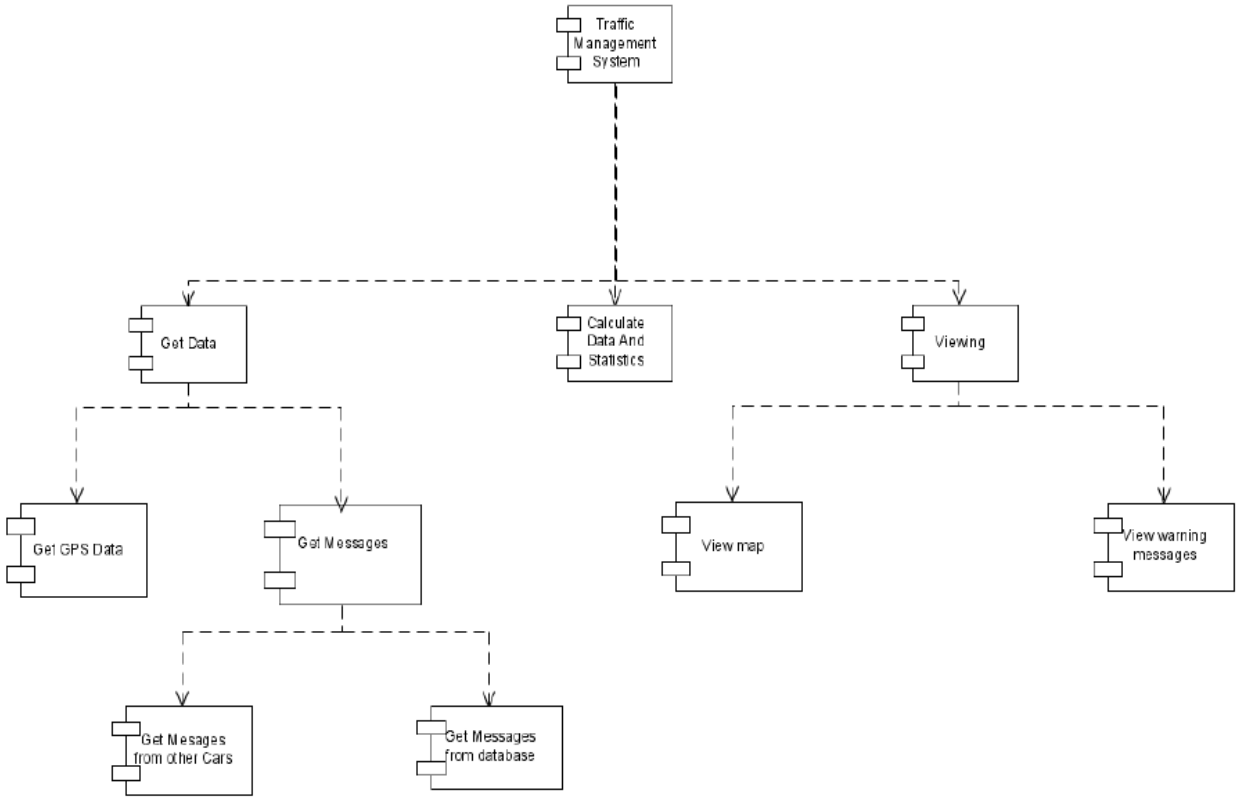
1.3.3 Hardware

Our project, Şehir Çobani, will be tested on GPS simulator and PDA with ZigBee modules for the performance and correctness constraints.

2. TEST PLAN

2.1 Software to be tested

Our project can be separated into 3 parts. As you can see here in this picture our project consists of 3 main modules “Get Data”, “Calculate Data and Statistics” and “Viewing” respectively.



Generally, it can be said that Get Data, Calculate Data And Statistics and Viewing modules are going to be tested separately and then their integration testing will also be done.

2.2 Testing Strategy

In this part of the document, the software testing methods and strategies we will use for our project will be explained.

2.2.1 Unit Testing

The main goal of unit testing in our project is taking the smallest piece of testable software in our project, isolating this unit from the remainder of the code, and determining whether it behaves exactly as we expect. In our project, every member is assigned to some implementations by trac system and then each member implements the task and before submitting into svn repositories he should test the code he write. The unit testing implementation of our project will consist of these steps (i.e testing before submitting to svn repositories).

2.2.2 Integration Testing

Integration testing is carried out in two steps. Firstly, the class with whole methods will be tested and finally, the combined class implementations will be tested together. With the help of integration testing, the problems that may occur during the integration of classes will be realized and tried to fix. After testing the classes, the integration of the packages we wrote will be tested. In other words we can say that Şehir Çobanı will be tested in a bottom-up manner. This approach enables a reliable testing scheme.

2.2.3 Validation Testing

Validation testing of our system will be done as follows. Firstly, whole system properties are separated into smaller parts and then these parts are assigned to trac and then all group members implement these parts. After implementation, each group member is responsible for validating the specified requirements assigned in trac, after that all implementations made by all group members are combined and they are validated as a whole. Finally, when we complete our project, our final software will be tested so that it ensures that the software which has been built is traceable to the client requirements.

2.2.4 High Order Testing

High-Order Testing is also known as system testing. System testing is applied on whole system to see whether the combination of modules suits to the requirements specified or not. Its purpose is to detect any errors within the modules of the system, connection between the modules and also examine the system as a whole.

As a high-order testing, two major types of system testing are performed.

Firstly, reliability testing is carried out. The reliability of the messages taken from ZigBee modules will be controlled. The accident information sent to the emergency services should be reliable therefore this information transferring process should be tested. Moreover, some scenarios are implemented and test whether accident information arrives to the emergency services or not. This testing will be based on ad-hoc network communication.

Secondly, performance testing is carried out. The speed of the data communication between cars in various distances and emergency services should be tested to improve the system if necessary.

2.3 Responsibilities

Every group member is responsible for testing the code he wrote before submitting into svn repositories. In this manner, we are developing the system by testing in each step and as a result of that we will have a system at the end mostly error-free. However, of course there will be many unexpected

errors in integration of the implementations of each group members. Therefore we are planing to test system elaborately again and again. In the table below, the distribution of testing responsibilities of three main modules described before (p.2.1 Software to be tested) can be seen.

Task	People
Get Data Module	Yakup & Burak
Viewing Module	Fatih & Okan
Calculate Data And Statistics	All

Tests are equally distributed among group members. However, there is no certain limitation. Every group member can test any part of the project if he has time. This is also encouraged since it will enable the group to understand every aspect of the code.

2.4 Test Record Keeping

Record Keeping is a significant part of the testing process to track down errors. Below the form of the records is given.

- The record files will have extension .txt
- Name of the records should be in the form %testedClass-%testType
- In the record file there must be a header which gives information on:
  - ✓ Date of the test process
  - ✓ Tester of the test process
  - ✓ Comments of the tester (satisfaction, errors, etc.)
- In the file, after the header there must be the output of the execution (If there were any output)

These record files will be arranged and gathered according to type of the testing process.

2.5 Test Metrics

When the properties of our project are considered, it can be said that there are 3 main software test metrics.

- Time

- Accuracy
- Responsiveness(turnaround time)to users

First metric is time. This time is related to the communication time between cars and also between cars and emergency services. The time of information transferring to the cars and emergency services will be a control value for our test processes.

Second metric is accuracy. This is about how accurate the address information of the accident occurred. In addition, the accuracy of the car's exact position is also important for us since we will use this information in crossroads and overtaking.

Last metric is responsiveness. The turnaround time to the users' requests is also important for our project. While testing, this metric is also used in many cases like whenever user press the overtake button also whenever user specifies the destination she/he wants to go...

2.6 Test Schedule

Task	**Start Date**	**End Date**
Unit tests		14/06/10
Integration Tests		14/06/10
Test Specification Report		26/04/10
Performance and Stress Tests	13/05/10	14/06/10
Alpha and Beta Tests	06/02/10	06/10/10
Bug Tracking and Fixes		14/06/10

3. TEST PROCEDURE

3.1 Software to be tested

In section 2, general description of software which is supposed to be tested was made. In this section, the elaboration of modules and parts of the system are going to be stated. These are:

- Get Data Module
  - ✓ GetGpsData Class Implementation
  - ✓ DeviceInputReader Class Implementation
  - ✓ Message Class Implementation

- Calculate Data And Statistics Module
  - ✓ Direction Class Implementation
  - ✓ CalculateStatistics Class Implementation
  - ✓ TrafficDensity Class Implementation

- Following Distance Module
  - ✓ LocationInformationMessage Class Implementation
- Overtaking Module
  - ✓ LocationInformationMessage Class Implementation

- Crossroads Module
  - ✓ LocationInformationMessage Class Implementation
- Emergency Services Module
- Viewing Module
  - ✓ GUI Design & Interfaces
  - ✓ Map Class Implementation
- Simulation
  - ✓ Simulation Engine Implementation

These modules will be explained within the testing procedure.

3.2 Testing Procedure

3.2.1 Unit Testing

The purpose of unit test is to check the verification and validity of all the methods in all classes. Since all the methods of our project cannot be written here, some cases which will be unit tested is stated.

Getting data correctly from the GPS method, and transferring this data (location information) to other cars method, calculating the following distance between two cars method which is done according to the location information of them, also the accident information transfer to the emergency services methods and simple GUI design methods such as button implementations, text boxes, also map implementations such as adding street names to the map ,adjusting zoom levels, these methods are all unit tested independent from each other. According to the results, if expected results are not obtained, error fixing process is carried out.

The expected result of each unit testing can be categorized like this:

Module	**Expected results**
Get Data Module	Taking the data information from GPS and being able to deliver it to the other cars.
Calculate Data And Statistics Module	Calculating the traffic density and accident statistics according to the location
Following Distance Module	Calculating the distance between two cars and warning the drivers if it is under optimal value
Overtaking Module	Calculating distance and speed of the closest car in opposite lane, giving this information to user
Crossroads Module	Calculating the relative locations of each car coming to the crossroad and warning all of them
Emergency Services Module	Transferring accident information to emergency services
Viewing Module	Implementing all the necessary interfaces for both users and emergency services
Simulation	Implementing simulation environment

3.2.2 Integration Testing

After making unit test to all modules, integration testing is conducted by using the bottom-up procedure. Every method will be integrated with the other methods in same module (in our case package) then all of the modules specified above will be integrated into a final system.

Expected results of the integration tests are modules' working properly together. The integrations will be mainly:

- Map-GUI Integration
- Get Data Module -Viewing Module Integration
- Get Data Module -Crossroads Module Integration
- Get Data Module-Overtaking Module Integration
- Get Data Module-Following Distance Module Integration
- Simulation Module Integration-All Remaining Module Integration

The purpose of the integration test is to detect possible errors when integrating two or more module into one.

3.2.3 Validation Testing

Validation of methods and modules is performed within the unit and integration testing phases

3.2.4 High Order Testing

3.2.4.1 Stress Testing

The purpose of stress testing is to check the extreme conditions. For our project, an extreme condition to be tested will be an environment in which very few cars exist (this will make difficult to transfer location and accident information in ad-hoc system). We will be trying to solve this problem in our simulation environment by producing similar scenarios to this. Another extreme condition that needs to be tested is that there are so many cars in simulation environment that data rate is very high. In this condition, system should behave properly.

3.2.4.2 Performance Testing

Performance testing is important since our project should meet the overtake requirements of users in real time. No delay in this overtaking process is accepted. Getting the GPS data in real time is also important. Therefore, many tests will be applied on this subject.

3.2.4.3 Security Testing

Since our system includes wireless communication, we need to ensure that undesirable connections are discarded. We also ensure reliability of the data system receives. Therefore, some tests will be done to our software in security subject.

3.2.4.4 Alpha/Beta Testing

The alpha tests of our project will be carried out by our group members. Every member is going to test whole system against any cases. After alpha tests, beta tests will be carried out by our classmates and friends after informing them about the system. This will provide us various looking aspects and possible errors to fix.

4. RISK AND CONTINGENCIES

The most possible risk for our project is facing some errors which cannot be fixed easily. Therefore, in order to prevent this situation, every module is to be implemented as error-free so that integration will be easier and the alpha and beta versions should be released reasonable amount of time before final release.

5. REFERENCES

- Software testing, Presentation prepared by METU Computer Engineering Department for the course Ceng 492
- Pressman, R. S., Software Engineering: A Practitioner's Approach, McGraw Hill, 2008, 6th Edition.
- Our Detailed Design Report
  - http://senior.ceng.metu.edu.tr/2010/oceans4/firstSemester.html
- MSDN library- Testing-Unit Testing
  - http://msdn.microsoft.com/en-us/library/aa292197(VS.71).aspx