Project Name
Acceptance Test Plan

Version 1.0

Revision History

Date	Version	Description	Author
2013-01-11	0.01	Fill the document	DK,DR,AP
2013-01-13	0.02	Fixed the document	AP
2013-01-20	Final	Final revision	DR

Table of Contents

1. Introduction
   - 1.1 Purpose of this document
   - 1.2 Intended Audience
   - 1.3 Scope
   - 1.4 Definitions and acronyms
     - 1.4.1 Definitions
     - 1.4.2 Acronyms and abbreviations
   - 1.5 References

2. Test-plan introduction

3. Test items
   - 3.1. Traffic Simulator
   - 3.2. Android Application
   - 3.3. Web Application

4. Features to be tested

5. Features not to be tested

6. Approach
   - 6.1. Installation and Configuration
   - 6.2. Documentation problems

7. Item pass/fail criteria

8. Suspension criteria

9. Environmental needs

10. Test procedure
    - 10.1 Traffic Simulator - Desktop
    - 10.2 Adroid Application - Android
    - 10.3 Web Application - Web

11. Risks and contingencies

12. Approvals

1. Introduction

1.1 Purpose of this document

The purpose of this document is to determine if the requirements specification is met by the developed product through a set of test cases described in the next sections.
This document is defined at the end of the implementation work and would be revised in case new functionalities were added before the final release. Its contents will be then made final when the product is delivered.

1.2 Intended Audience

The intended audience is:

- Supervisor: responsible to monitor the status of the project, its direction and outcomes.
- Project Team: to have an overview of what will be tested of the software developed.

1.3 Scope

This document describes the set of test cases for the Car Gossip Android application, web application and traffic simulator. Moreover, it contains the results coming from the execution of the test cases together with the steps needed to perform it.

1.4 Definitions and acronyms

1.4.1 Definitions

Keyword	Definitions
Gossip	Messages sent through a DSRC device which then in turn will be spread to other nearby cars. The basis is a Gossip algorithm.
Swiss Database	ETH Zurich Open Database is the database that is used to ETH Zurich and provides the traffic information as velocity, direction and GPS coordinates.

1.4.2 Acronyms and abbreviations

Acronym or abbreviation	Definitions
ICD	Internal Car Device
DSRC	Dedicated short-range communication
GPS	Global Positioning System

1.5 References

This document is based on the previously developed deliverables available at

http://fer.unizg.hr/rasip/dsd/projects/cars_talk/documents

- Project description
- Requirements definition
- Design description

2. Test-plan introduction

The Car Gossip product will be tested at different levels by means of acceptance and unit testing.
The requirement document will serve as a reference to individuate what to be verified, especially at acceptance testing level, on which this document focuses.
The final product is composed by three parts that have to be tested:

- Desktop Application
- Android Application
- Web Application

3. Test items

3.1: Desktop application

The Desktop application part will include the following parts that need to be tested:

- Parser
- Scenario creation
- Simulator
- DSRC messaging
- Bluetooth connection

In the Parser, we will test connecting to the database and importing traffic data from mov files.

The Scenario creation will be tested by verifiying if the user can choose a number of cars and also a limited time frame and produce the expected simulation with the given values.
In the simulator, we will test if the movement of the cars involved in the simulation is presented on the map, DSRC messages are being sent and the Bluetooth connection is established.

3.2: Android application
In the Android application we will test the following things:

- Bluetooth connection
- Visualizing alerts and sending them
- Visualizing traffic data (surrounding cars and own car)
- The user interface
- Sending requests to the web server

3.3: Web application
In the Web application part we will test if the web server receives message from the Android application and stores the messages in a database.

4. Features to be tested

4.1: Desktop Application

ID	Feature	Status

DA01	Database connection to MongoDB	Tested
DA-02	Import of data from ns-2 movement format file	Tested
DA03	Select region for the scenario	Tested
DA04	Select time frame for the scenario	Tested
DA05	Create the scenario	Tested
DA06	Scenario loading from the database	Tested
DA07	Scenario saving to the database	Tested
DA08	ICD Selection in the simulation window	Tested
DA09	Start Simulation	Tested
DA10	Stop Simulation	Tested
DA11	Forward/Rewind simulation	Tested
DA12	Visualize selected ICD	Tested
DA13	Visualize surrounding cars	Tested
DA14	Establish Bluetooth connection with phone that runs the Android application	Tested
DA15	Receive alert messages from the Android application via BT	Tested
DA16	Visualize alert message and spreading to surrounding cars	Tested
DA17	Distribute messages through a simulated DSRC device	Tested

4.2: Android Application

ID	Feature	Status
AA01	Set up Bluetooth connection	Tested
AA02	Initialize a Map with a view of the current position	Tested
AA03	Receive position data from the selected ICD in the simulator	Tested
AA04	Visualize and update map view with the position data	Tested
AA05	Receive gossip messages from surrounding cars	Tested
AA06	Visualize surrounding cars on the map	Tested
AA07	Select an alert message	Tested
AA08	Visualize alert message	Tested
AA09	Send alert message via Bluetooth	Tested
AA10	Send alert message via REST-interface	Tested
AA11	Send gossip message via REST-interface	Tested
AA12	Receive confirmation via REST-interface	Tested

4.3: Web application

ID	Feature	Status

WB01	Receive gossip message	Tested
WB02	Store gossip message in a database	Tested
WB03	Receive alert message	Tested
WB04	Store alert message in a database	Tested
WB05	Verify alert message	Tested
WB06	Creation of dynamic queries	Tested
WB07	Visualization of query responses	Tested

5. Features not to be tested

The following features will not be tested:

- Resilience to MongoDB database crashes
- High network delays
- Server crashes
- Realistic car traces

6. Approach

The system will be tested manually through a series of detailed test cases. The tests will be described in detail in order to enable the interested reader to recreate and execute them. The tests will be carried out by the project team by dividing the tests into three categories; one for the Desktop Application, one for the Android Application and finally one for the Web Application.

6.1 Installation and Configuration

No configuration is needed for application deployment. Application is installed as java jars and as the Android package. This doesn't affect testing.

6.2 Documentation problems

Since the features that are planned for the web application are still not in a determined state, it is not clear what to test in these cases.

7. Item pass/fail criteria

7.1: Desktop Application

DESKTOP-001: Database connection to MongoDB

Pass: The connection is established and stable

Fail cases:

- Desktop Application cannot connect to the MongoDB database server.
- The MongoDB database is not available.

DESKTOP-002: Import of data from ns-2 movement format file

Pass: Traffic simulator successfully parses the data and stores the data in the MongoDB .
Fail cases:
- Application cannot parse the input data.
- Application cannot store parsed data in the database.

DESKTOP-003: Select region for the scenario
Pass: Application selects cars from the map and marks them as selected.
Fail cases:
- Application throws an unhandled exception.

DESKTOP-004: Select time frame for the scenario
Pass: Application selects time frame on the double slider and stores the time frame.
Fail cases:
- Application throws an unhandled exception.

DESKTOP-005: Create the scenario
Pass: The scenario is created and stored in the database
Fail cases:
- Scenario with the given name already exists.
- User didn't select any cars.
- Application cannot store data in the database.

DESKTOP-006: Scenario loading from the database
Pass: Application loads lists of scenarios from the database.
Fail cases:
- No cars are selected.
- Application cannot store data in the database.

DESKTOP-007: Scenario saving to the database
Pass: Application stores scenario to the database.
Fail cases:
- Application cannot store data in the database.

DESKTOP-008: ICD Selection in the simulation window
Pass: Application reacts to the user input, selects the selected ICD and starts to follow it on the map.
Fail cases:
- Application doesn't follow the ICD.

DESKTOP-009: Start Simulation
Pass: Application starts the simulation.
Fail cases:
- Application throws an unhandled exception.

DESKTOP-010: Stop Simulation
Pass: Application stops the simulation.
Fail cases:
- Application throws an unhandled exception.

DESKTOP-011: Forward / Rewind simulation
Pass: Application reacts to user input and forwards / rewinds the simulation by setting the new simulation time.
Fail cases:
- Application doesn't set the right time.

DESKTOP-012: Visualize selected ICD
Pass: Application uses the selected ICD and visualizes it on the map.

Fail cases:
- ICD doesn't appear on the map.

DESKTOP-013: Visualize surrounding cars
Pass: Application shows ICDs on the map with selected ICD in the center.
Fail cases:
- ICDs don't appear on the map.

DESKTOP-014: Establish Bluetooth connection with phone that runs the Android application
Pass: Application connect to Android application via the Bluetooth device and displays it in the status bar.
Fail cases:
- Bluetooth device isn't available.
- Bluetooth connection cannot be established.

DESKTOP-015: Receive alert messages from the Android application via BT
Pass: Application is connected to Android application via the Bluetooth device and receives alert messages.
Fail cases:
- Bluetooth connection isn't established.
- Alert messages aren't received.

DESKTOP-016: Visualize alert message and spreading to surrounding cars
Pass: Application draws the followed car on the map and draws all the surrounding cars. Cars to which the messages are spread are drawn differently.
Fail cases:
- The cars are not shown on the map.
- The spreading of the messages is not shown.

DESKTOP-017: Distribute messages through a simulated DSRC device
Pass: Application simulates the ICDs and DSRC message spread via the DSRC cloud. The DSRC cloud should accept and distribute messages to surrounding ICDs.
Fail cases:
- The messages are not spread.

7.2: Android Application

ANDROID-001: Set up Bluetooth connection
Pass: Bluetooth connection is set up.
Fail cases:
- Bluetooth device isn't available.
- Bluetooth connection cannot be established.

ANDROID-002: Initialize a Map with a view of Switzerland
Pass: Shows the map of Switzerland.
Fail cases:
- Map of Switzerland is not shown.

ANDROID-003: Receive position data from the selected ICD in the simulator
Pass: ICD data is received from the simulator via Bluetooth.
Fail cases:
- Bluetooth connection isn't established.
- ICD data isn't received.

ANDROID-004: Visualize and update map view with the position data

Pass: Center the map to the current ICD data.
Fail cases:
- The map isn't updated.

ANDROID-005: Receive gossip messages from surrounding cars
Pass: Receives gossip messages from the simulator.
Fail cases:
- Gossip messages are not received.

ANDROID-006: Visualize surrounding cars on the map
Pass: Application visualizes the received data from the cars on the map.
Fail cases:
- Surrounding cars on the map are not shown.

ANDROID-007: Select an alert message
Pass: Application reacts on user input and selects alert from the alert overlay.
Fail cases:
- Application doesn't react on user input.

ANDROID-008: Visualize alert message
Pass: Alert messages icons appear on the map.
Fail cases:
- Alert messages are not shown on the map.

ANDROID-009: Send alert message via Bluetooth
Pass: Alert messages are sent to the simulator application.
Fail cases:
- Alert messages are not received by the simulator application.

ANDROID-010: Send alert message via REST-interface
Pass: Alert messages are sent via REST-interface to the server.
Fail cases:
- Internet connection isn't available
- Messages cannot be sent.

ANDROID-011: Send gossip message via REST-interface
Pass: Gossip messages are sent via REST-interface to the server.
Fail cases:
- Internet connection isn't available
- Messages cannot be sent.

ANDROID-012: Receive confirmation via REST-interface
Pass: Confirmation messages are received by the application.
Fail cases:
- Internet connection isn't available
- Messages cannot be received.

ANDROID-013: Remove alert popup
Pass: After an alert on a map is clicked, a popup to remove the alert is shown.
Fail cases:
- No popup is shown
- The alert is not clickable

ANDROID-014: Remove alert “yes”

Pass: If the user clicks “yes” and the alert is removed and the popup disappears

Fail cases:

- The alert is not removed
- The popup is not disappearing
- The wrong alert is removed

ANDROID-015: Remove alert “no”

Pass: If the user clicks “no” popup disappears

Fail cases:

- An alert is removed
- The popup is not disappearing

ANDROID-016: Message is lost

Pass: No alteration on the map occurs

Fail cases:

- The map produces glitches
- The program crashes
- The map is updated in a wrong way

ANDROID-017: Message gets corrupted

Pass: No alteration on the map occurs

Fail cases:

- The map produces glitches
- The system crashes
- The map is updated in a wrong way

ANDROID-018: The user “spams” alert messages

Pass: The user is prevented from spamming and gets an error message

Fail cases:

- The user can stil spam
- The user gets not notified
- The system crashes

7.3: Web application

WEB-001: Receive gossip message

Pass: Receives gossip messages from the Android device.

Fail cases:

- Internet connection isn’t available
- Messages cannot be received.

WEB-002: Store gossip message in a database

Pass: Messages are successfully stored in the database.

Fail cases:

- Cannot store the messages in the database.

WEB-003: Receive alert message

Pass: Receives alert messages from the Android device.

Fail cases:
- Internet connection isn't available
- Messages cannot be received.

WEB-004: Store alert message in a database

Pass: Messages are successfully stored in the database.

Fail cases:
- Cannot store the messages in the database.

WEB-005: Verify alert message

Pass: Alert messages are reported as spam or not spam by the application.

Fail cases:
- Application misreports message as spam.

8. Suspension criteria and resumption requirements

We test the code development and if a bug is found, the test is stopped and the bug fixed.
Afterwards the tests are restarted from the beginning.
If unwanted problems with the connections via Bluetooth and the Internet persist, the test is stopped until there is a solution found.
Is the problem resolved, the tests can be restarted.

9. Environmental needs

In this chapter we list which hardware and software we need to test the Android, Desktop and Web Application.

9.1. Hardware

This is the list of the hardware that we need to test each part of the project, so to test:
- the Android Application, we need to have ether a Smartphone or a tablet that have integrated the Bluetooth and Wi-Fi device;
- the Desktop Application, we need to have a PC that has the Bluetooth device and network card to connect via Internet;
- the Web Application, we need to have a server.

9.2. Software

This is the list of the software that we need to test each part of the project, so to test:
- the Android Application we need to have the Android OS.
- the Desktop Application we need to have the Window or Linux OS.
- the Web Server we need to the Window Server or Linux Server OS.

10. Test procedure

10.1. Desktop Application

10.1.1. Database connection to MongoDB - DESKTOP-001

Description: Testing database connection.
Test type: Positive
Preconditions:

- The application is running.
- MongoDB server is runing.

Input definition:

1. User clicks on the connect to database button in the database tab.

Output definition:

1. Application connects in the background to the MongoDB server.
2. Database statistics are shown in the database tab.

10.1.2. Import of data from ns-2 movement format file - DESKTOP-002

Description: Testing importing data to database.
Test type: Negative
Preconditions:

- The application is running.
- MongoDB server is running.
- Application is connected to MongoDB server.

Input definition:

1. User clicks on the import mov file button in the database tab, and selects a corrupted file from the dialog.

Output definition:

1. Parser searches for the valid entries in the file and it finds none.
2. The old database is dropped as announced before importing.
3. The new database is empty.

10.1.3: Select region for the scenario - DESKTOP-003

Description: Testing region selection for the scenario setup.
Test type: Positive
Preconditions:

- The application is running.
- Application is connected to MongoDB server.
- Database data is loaded.

Input definition:

1. User selects with the left mouse button wanted region on the map in the scenario setup tab.

Output definition:

1. Cars in the selected region are marked as selected.

10.1.4: Select time frame for the scenario - DESKTOP-004

Description: Testing time frame selection for the scenario setup.
Test type: Positive
Preconditions:

- The application is running.
- Application is connected to MongoDB server.
- Database data is loaded.

Input definition:

1. User selects time frame on the double slider in the scenario setup tab.

Output definition:

1. Time frame is stored in the scenario setup model.

10.1.5: Create the scenario - DESKTOP-005

Description: Testing creating scenarios.
Test type: Positive
Preconditions:

- The application is running.

- MongoDB server is runing.
- Database data is loaded.

Input definition:

1. User selects car region or selects cars from the table.
2. User selects time frame or leaves the default one.
3. User clicks on the create scenario button.

Output definition:

1. Scenario is created.

10.1.6: Scenario loading from the database - DESKTOP-006

Description: Testing loading scenarios from the database.
Test type: Positive

Preconditions:

- The application is running.
- MongoDB server is runing.

Input definition:

1. User clicks on the load scenarios button in the scenario selection tab.

Output definition:

1. Scenarios are loaded from the database.
2. Loaded scenarios are shown in the scenarios table.

10.1.7: Scenario saving to the database - DESKTOP-007
Description: Testing saving scenarios to the database.
Test type: Positive

Preconditions:

- The application is running.
- MongoDB server is runing.
- Scenario is crated.

Input definition:

1. User previously created the scenario by pressing create scenario button.

Output definition:

1. Scenario is stored to the database.

10.1.8: ICD Selection in the simulation window - DESKTOP-008
Description: Testing selecting ICDs in the simulation.
Test type: Positive

Preconditions:

- The application is running.
- The simulation is created.

Input definition:

1. User selects ICD from the ICD selection table in the simulation tab.

Output definition:

1. The selected ICD is marked as primary in the simulation.

10.1.9: Start Simulation - DESKTOP-009
Description: Testing simulation starting.
Test type: Positive

Preconditions:

- The application is running.
- The simulation is created.

Input definition:

1. User preses start button in the simulation tab.

Output definition:

1. Simulation is started, time track starts to move, car movement is visualized.

10.1.10: Stop Simulation DESKTOP-010
Description: Testing stopping the simulation.
Test type: Positive
Preconditions:
- The application is running.
- The simulation is created.
- The simulation is started.

Input definition:
1. User presses the stop button in the simulation tab.

Output definition:
1. Simulation is stopped and reset to the start state.

10.1.11: Forward / Rewind simulation - DESKTOP-011
Description: Testing simulation time manipulation.
Test type: Positive
Preconditions:
- The application is running.
- The simulation is created.
- The simulation is started.

Input definition:
1. User selects new time in the time slider in the simulation tab.

Output definition:
1. Simulation is set to the selected time and all of the corresponding models behind the simulation are updated to the new time.

10.1.12: Visualize selected ICD - DESKTOP-012
Description: Testing ICD visualization.
Test type: Positive
Preconditions:
- The application is running.
- The simulation is created.
- The simulation is started.

Input definition:
1. User selects ICD from the ICD selection table in the simulation tab.

Output definition:
1. Simulation map is cantered to the selected ICD. Selected ICD is drawn red on the map.

10.1.13: Visualize surrounding cars - DESKTOP-013
Description: Testing visualizing traffic on the map.
Test type: Positive
Preconditions:
- The application is running.
- The simulation is created.
- The simulation is started.

Input definition:
Output definition:
1. ICDs appear on the map and they are drawn blue.

10.1.14: Establish Bluetooth connection with phone that runs the Android application - DESKTOP-014
Description: Testing Bluetooth connectivity.
Test type: Positive
Preconditions:

- The application is running.
- The Android application is running.
- Bluetooth device is available.

Input definition:

1. User selects connect button in the Android application.

Output definition:

1. Bluetooth connection is established.
2. Bluetooth status bar is updated and shows the connected message.

10.1.15: Receive alert messages from the Android application via BT - DESKTOP-015
Description: Testing Bluetooth communication.
Test type: Positive
Preconditions:

- The application is running.
- The Android application is running.
- Bluetooth device is available.
- Android application is connected to the simulator.
- The simulation is started.

Input definition:

1. User presses the alert button on the Android device.

Output definition:

1. Alert messages are received by the ICD.

10.1.16: Visualize alert message and spreading to surrounding cars - DESKTOP-016
Description: Testing visualizing the alerts.
Test type: Positive
Preconditions:

- The application is running.
- The simulation is started.
- Alert messages are received.

Input definition:

1. User previously pressed alert button on the Android device and alert messages are received.

Output definition:

1. Alert message icons are shown on the simulation map.

10.1.17: Distribute messages through a simulated DSRC device DESKTOP-017
Description: Testing gossip algorithm.
Test type: Positive
Preconditions:

- The application is running.
- The simulation is started.

Input definition:

1. User selects ICD from the ICD selection table.

Output definition:

1. Received messages are shown in the message table, ICDs that received messages from the selected ICD turn green on the map.

10.2: Android application - ANDROID

10.2.1: Set up Bluetooth connection - ANDROID-001
Description: The test to verify that a Bluetooth connection between Simulator and Android application can be established.
Test type: Positive
Preconditions:

- The Simulator is running and the Android application is started.
- The system where the Simulator is run needs to be in range of the mobile phone, where the Android application is executed on.
- Both sides of the connection need to turn on their Bluetooth module and be visible to the other end.

Input definition:

1. User selects “Connect to device on Android application.
2. User selects the corresponding system there the simulator is run on.

Output definition:

1. The video component has loaded the video.
2. Navigation controls are enabled.

10.2.2: Initialize a Map with a view of Switzerland - ANDROID-002

Description: The test to verify that a map view is opened after the Bluetooth connection is set up.

Test type: Positive

Preconditions:

- Simulator and Android application are connected via Bluetooth

Output definition:

1. A map with an overview of Switzerland will be presented to the user

10.2.3: Receive position data from the selected ICD in the simulator - ANDROID-003

Description: The test to verify if the own car position is transmitted to the phone

Test type: Positive

Preconditions:

- Bluetooth connection is active
- Simulation is running and the selected ICD is moving

Output definition:

1. The user is presented with a view of the current position of the car in a zoomed map

10.2.4: Visualize and update map view with the position data - ANDROID-004

Description: The test to see if the own car position is visualized on the map.

Test type: Positive

Preconditions:

- Bluetooth connection is active
- Simulation is running and the selected ICD is moving
- ANDROID-003 is fulfilled

Output definition:

1. The user is presented with a view of the current position of the car in a zoomed map

10.2.5: Receive gossip messages from surrounding cars - ANDROID-005

Description: The test to see if other car positions are received through gossip.

Test type: Positive

Preconditions:

- Bluetooth connection is active
- Simulation is running and the selected ICD is active
- Nearby car position data is received via Bluetooth

Output definition:

1. The user sees the surrounding cars marked on the map

10.2.6: Visualize surrounding cars on the map - ANDROID-006

Description: The test to verify that gossip position messages are visualized on the map.

Test type: Positive
Preconditions:
- Bluetooth connection is active
- Simulation is running and the selected ICD is active
- ANDROID-005 is fulfilled

Output definition:
1. The user sees the surrounding cars marked on the map

10.2.7: Select an alert message - ANDROID-007
Description: The test to check if alert messages are selectable.
Test type: Positive
Preconditions:
- Bluetooth connection is active
- Alert overlay is activated

Input definition:
User presses an alert button
Output definition:
1. The user sees the surrounding cars marked on the map

10.2.8: Visualize alert message - ANDROID-008
Description: The test to see if produced alert messages are visualized on the map.
Test type: Positive
Preconditions:
- Bluetooth connection is active
- Alert overlay is activated

Input definition:
ANDROID-007 is fulfilled
Output definition:
1. The user sees the surrounding cars marked on the map

10.2.9: Send alert message via Bluetooth - ANDROID-009
Description: The test to see if produced alerts are distributed via Bluetooth.
Test type: Positive
Preconditions:
- Bluetooth connection is active

Input definition:
1. ANDROID-007 is fulfilled

Output definition:
1. ICD inside simulator receives alert

10.2.10: Send alert message via REST-interface - ANDROID-010
Description: The test to see if produced alerts are distributed via Internet.
Test type: Positive
Preconditions:
- Internet connection is active

Input definition:
1. ANDROID-007 is fulfilled

Output definition:
1. Web server receives the alert message

10.2.11: Send gossip message via REST-interface - ANDROID-011

Description: The test to see if received gossip is distributed via Internet.
Test type: Positive
Preconditions:

- Internet connection is active

Input definition:

1. ANDROID-005 is fulfilled

Output definition:

1. Web server receives the gossip message

10.2.12: Receive confirmation via REST-interface
Description: The test to see if confirmation messages from the web server can be received.
Test type: Positive
Preconditions:

- Internet connection is active

Input definition:

1. ANDROID-010 is fulfilled

Output definition:

1. Alert messages are changed visibly to a verified state

10.2.13 Remove alert popup- ANDROID-013
Description: When the user clicks on an alert icon on the map a delete message appears.
Test type: Positive
Preconditions:

- Android application is started
- The alert icon is presented on the map

Input definition:

1. User clicks on an alert icon.

Output definition:

1. The popup appears on the screen with the message:”Are you sure to remove the icon message?” and the buttons “Yes” and “No”

10.2.14 Remove alert “yes”- ANDROID-014
Description: When the user clicks the "Yes" button to remove the alert
Test type: Positive
Preconditions:

- Android application is started
- The alert icon is presented on the map
- The user has clicked on the alert icon

Input definition:

1. User clicks on ”Yes” button.

Output definition:

1. The popup disappears.
2. The alert icon disappears.

10.2.15 Remove alert “no”- ANDROID-015
Description: When the user clicks "No” button
Test type: Positive
Preconditions:

- Android application is started
- The alert icon is presented on the map
- The user has clicked on the alert icon

Input definition:

1. User clicks on ”No” button.

Output definition:

1. The popup disappears.

10.2.16 Message is lost - ANDROID-016

Description: A message is lost during Bluetooth transmission.

Test type: Negative

Preconditions:

- Bluetooth connection is active

Input definition:

- ICD sends message

Output definition:

1. No visible change on the map
2. No error is thrown

10.2.17 Message gets corrupted - ANDROID-017

Description: A message gets corrupted during Bluetooth transmission.

Test type: Negative

Preconditions:

- Bluetooth connection is active

Input definition:

1. ICD sends message

Output definition:

1. No visible change on the map
2. No error is thrown

10.2.18 The user “spams” alert messages - ANDROID-018

Description: Test to see if a spam protection inside the application is active.

Test type: Negative

Preconditions:

- Bluetooth connection is active
- Alert overlay is active

Input definition:

1. User presses alert buttons consecutive times in a short period

Output definition:

1. An error message is shown to the user
2. The user is blocked for a short amount of time

10.3: Web application - WEB

10.3.1: Receive gossip message - WEB-001

Description: Test if the Web application receives gossip messages.

Test type: Positive

Preconditions:

- Web server needs to be running.
- Android application is started.

Input definition:

1. User selects connect button in the Android application.
2. User drives the car and waits for the gossip to be sent.

Output definition:

1. Gossip is received. User receives notification on his Android application.

10.3.2: Store gossip message in a database - WEB-002

Description: Test if the gossip message is stored in a database.

Test type: Positive

Preconditions:
- Web server needs to be running.
- Android application is started.

Input definition:
1. User selects connect button in the Android application.
2. User drives the car and waits for the gossip to be sent.

Output definition:
1. User receives notification on his Android application that the message is sent and the ID of the message.
2. User goes to the webpage /cargossip/gossip/{id} where the {id} is the ID that he received to his Android application

10.3.3: Receive alert message - WEB-003
Description: Test if the Web application receives alert messages.
Test type: Positive
Preconditions:
- Web server needs to be running.
- Android application is started.

Input definition:
1. User selects connect button in the Android application.
2. User types and sends the alert via Android application

Output definition:
1. Alert is received. User receives notification on his Android application.

10.3.4: Store alert message in a database - WEB-004
Description: Test if the alert message is stored in a database.
Test type: Positive
Preconditions:
- Web server needs to be running.
- Android application is started.

Input definition:
1. User selects connect button in the Android application.
2. User drives the car and waits for the gossip to be sent.

Output definition:
1. User receives notification on his Android application that the message is sent and the ID of the message.
2. User goes to the webpage /cargossip/alert/{id} where the {id} is the ID that he received to his Android application

10.3.5: Verify alert message - WEB-005
Description: Test if the alert message is valid.
Test type: Positive
Preconditions:
- Web server needs to be running.
- Android application is started.

Input definition:
Output definition:

11. Risks and contingencies

All developers have to be ready to make quick fixes if a bug is caught in their responsible parts of the development. During the testing process the developers need to document what they used as input and what came out as output. If an error occurred this should be documented as well so the responsible developer has all

information needed to fix the issue. The developer has to make sure that every requirement is fulfilled for the test procedure for the test to be positive. When one test is done the developer has to put in the result in the test report.

12. Approvals

Name	Title	Date yyyy-mm-dd	**Signature**
Federico Ciccozzi	Supervisor		