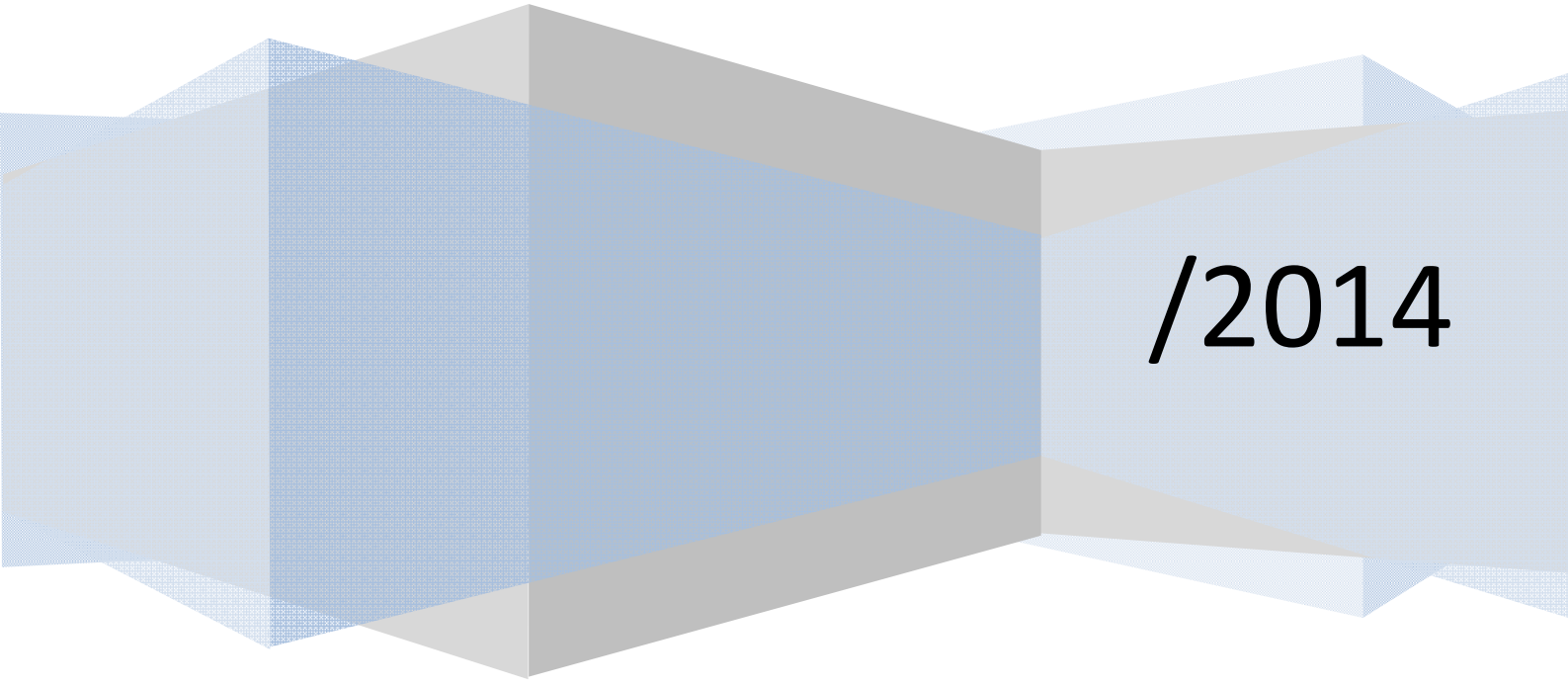
DROGBA INC.


| | |
|---|---|
| ALİ HOPYAR | 1746056 |
| FATİH HAFIZOĞLU | 1746049 |
| HALİM KAYA | 1746148 |
| VOLKAN GÜMÜŞ | 1746007 |


# SOFTWARE TEST DOCUMENTATION



/2014

# Contents

# 1.Heading

The following sections identify this document and its scope. They also give information about the context in which this document is prepared.

## 1.1. Document Identifier

This document is the Software Test Document of the project COW-3D carried out by the project group Drogba Inc. It is based on the IEEE Std 829-2008, IEEE Standard for Software and System Test Documentation.

The document lists the test cases that should be conducted to verify and validate the system's features.

This is the first version of the Software Test Document, it is written by the members of the project group.

## 1.2. Scope

The software product is a web based application developed in Unity3D game engine. It has one type of user, thus a one type of user interface. The features to be tested are mainly the user interfaces' functionalities.

This software is a small scale product. It is not part of a larger system, rather a stand-alone application. Hence, there is only one level of test. The test of the product itself.

## 1.3. References

- IEEE Std 829-2008, IEEE Standard for Software and System Test Documentation
- Software Design Description of COW-3D
- Software Requirement Specification of COW-3D

## 1.4. Level in the Sequence

As mentioned in section *1.2*, this level is not part of a larger test hierarchy or sequence. There is only one level of test.

## 1.5. Test Classes and Overall Test Conditions

This level of test covers the whole product. The features to be tested are the functionalities desribed in the Software Requirement Specification, the functionalities described by means of use cases. The product is tested to verify and validate that it operates as expected.
Thus, this system test focuses on meeting the system's requirements.

# 2. Details for System Test Plan

The following sections describe the test items, the features to be tested and not tested, a traceability matrix, and the approach that is followed through this testing phase.

## 2.1. Test Items and Their Identifiers

The test items are the functionalities specified in Software Requirement Specification document's Functional Requirements(3.2) section. These functionalities are decribed by means of use cases. The test items are listed below with their use case identifiers;

- UseCase No:1 – Start
- UseCase No:2 – Move Forward
- UseCase No:3 – Move Backward
- UseCase No:4 – Move Right
- UseCase No:5 – Move Left
- UseCase No:6 – Camera Up
- UseCase No:7 – Camera Down
- UseCase No:8 – Camera Right
- UseCase No:9 – Camera Left
- UseCase No:10 – Set Starting Point
- UseCase No:11 – List Starting Points
- UseCase No:12 – Set Destination Point
- UseCase No:13 – List Destination Points
- UseCase No:14 – Find the Path

The use cases listed above have their corresponding explanations in Software Requirement Specification document's Product Perspective(2.2) section. The use case

identifiers can be use to find detailed information about the corresponding functionality in SRS.

## 2.2. Test Trecability Matrix

The test traceability matrix is given below. The first row specifies the corresponding use case. Use cases are denoted by UC and use case number. They correspond to the use cases listed in section 2.1. The test cases are denoted by TC and test case number and are listed in the first column.

| | UC 1 | UC 2 | UC 3 | UC 4 | UC 5 | UC 6 | UC 7 | UC 8 | UC 9 | UC 10 | UC 11 | UC 12 | UC 13 | UC 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TC 1 | X | | | | | | | | | | | | | |
| TC 2 | | X | | | | | | | | | | | | |
| TC 3 | | | X | | | | | | | | | | | |
| TC 4 | | | | X | | | | | | | | | | |
| TC 5 | | | | | X | | | | | | | | | |
| TC 6 | | | | | | X | | | | | | | | |
| TC 7 | | | | | | | X | | | | | | | |
| TC 8 | | | | | | | | X | | | | | | |
| TC 9 | | | | | | | | | X | | | | | |
| TC 10 | | | | | | | | | | X | | | | |
| TC 11 | | | | | | | | | | | X | | | |
| TC 12 | | | | | | | | | | | | X | | |
| TC 13 | | | | | | | | | | | | | X | |
| TC 14 | | | | | | | | | | | | | | X |

**TABLE 1: Test Traceability Matrix**

## 2.3. Features to be Tested

The features to be tested are listed in section 2.1. They are the functionalities that the user is provided with. All the features are part of the user interface. These features are powered by scripts written by the developers, they are neither part of the development environment nor a third party library.

## 2.4. Features not to be Tested

The features that are provided and managed by the Unity3D game engine will not be tested. A few of these features might be the process of rendering the model,

or mapping the textures. As the product is run on web browsers with Unity Web Player plugged in, the features that the web player provides will not be tested either.

## 2.5. Approach

The approach that is followed in this system test can be described as Black box. In this black box method, the inputs are the user's interaction with the user interface, and the outputs are the system's responses to these interactions. The interactions are button clicks and mouse movements.

## 2.6. Item pass/fail Criteria

The tested items are considered to pass the test if the output is exactly the same as the specified output in SRS document.

On the other hand, the items that are being tested are considered to fail if the output is not the expected one or an anomaly is detected in the system's response.

## 2.7. Test Deliverables

The test deliverable of this system test is the document itself. This document is Software Test Document and it includes the combination of the context of the following documents;

- Level Test Plan(s)
- Level Test Case
- Level Test Report

Section 1, 2 and 3 cover the Level Test Plan(s), section 4 includes the context in Level Test Case and section 5 covers the Level Test Report.

# 3.Test Management

## 3.1. Planned Activities and Tasks; Test Progression

In this section how the system can be tested whether it performs well in terms of stability, reliability and responsiveness will be given.

Firstly, goals and system boundaries should be decided before starting actual test. Making clear what the boundaries and the goals are in the system is very important in terms of receiving exact information.

Parameters can be divided into system parameters and workload parameters. System parameters include hardware and software parameters. The workload parameters are the special parameters which are determined according to users' requests.

Parameters can be divided into two parts: those that will be changed during the evaluation and those that will not. The parameters to be changed are called factors.

Design different scenarios for test process. It is rational to decide of sequence of tests that offer maximal information with minimal effort.

## 3.2. Environment / Infrastructure

All environments are explained at 5.1.1. of STD document.

# 4.Test Case Details

In this section test case identifiers, objectives, inputs and outcomes,environmental needs,special procedural requirements,intercase dependencies for each functional test case are provided as tables.

## 4.1.Start Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:1 | Successful Case | Test Case with all the pre-condition is true. | 1.User enters unity-3D supported web browser.<br><br>2.User presses start button<br><br>3.User selects the mode from 2 available modes. | 1.The system becomes active. |

**TABLE 2: Start Test Case**

## 4.2. Move Forward Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:2 | Successful Case | Test Case with all the pre-condition is true. | 1.User presses “forward key” | 1.The system becomes active.<br><br>2. System moves the viewing point to forward |

**TABLE 3: Move Forward Test Case**

## 4.3. Move Backward Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:3 | Successful Case | Test Case with all the pre-condition is true. | 1.User presses "backward key" | 1.The system becomes active.<br><br>2. System moves the viewing point to backward |

**TABLE 4: Move Backward Test Case**

## 4.4. Move Right Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:4 | Successful Case | Test Case with all the pre-condition is true. | 1.User presses "right key" | 1.The system becomes active.<br><br>2. System moves the viewing point to right |

**TABLE 5: Move Right Test Case**

## 4.5. Move Left Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:5 | Successful Case | Test Case with all the pre-condition is true. | 1.User presses "left key" | 1.The system becomes active.<br><br>2. System moves the viewing point to left |

**TABLE 6: Move Left Test Case**

## 4.6. Camera Up Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:6 | Successful Case | Test Case with all the pre-condition is true. | 1.User presses "up arrow key" | 1.The system becomes active.<br><br>2. System changes the viewing direction to upward. |

**TABLE 7: Camera Up Test Case**

## 4.7. Camera Down Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:7 | Successful Case | Test Case with all the pre-condition is true. | 1.User presses "down arrow key" | 1.The system becomes active.<br><br>2. System changes the viewing direction to downward. |

**TABLE 8: Camera Down Test Case**

## 4.8. Camera Right Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:8 | Successful Case | Test Case with all the pre-condition is true. | 1.User presses "right arrow key" | 1.The system becomes active.<br><br>2. System changes the viewing direction to rightward. |

**TABLE 9: Camera Right Test Case**

## 4.9. Camera Left Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. | | | | |
| TestCase No:9 | Successful Case | Test Case with all the pre-condition is true. | 1.User presses "left arrow key" | 1.The system becomes active.<br><br>2. System changes the viewing direction to leftward. |

**TABLE 10: Camera Left Test Case**

## 4.10. Set Starting Point Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. "Path Finding Mode" must be selected. | | | | |
| TestCase No:10 | Successful Case | Test Case with all the pre-condition is true. | 1.User selects "Set Start Point" in input form. | 1.The system becomes active.<br><br>2. System sets the selected point to start point. |

**TABLE 10: Set Starting Point Test Case**

## 4.11. List Starting Points Test Case

| **Test Case ID** | **Test Case Name** | **Test Case Description** | **Test Inputs** | **Outcomes** |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. “Path Finding Mode” must be selected. | | | | |
| TestCase No:11 | Successful Case | Test Case with all the pre-condition is true. | 1.User selects “Select From Starting Points” in input form.<br><br>2.User selects one of the points from list. | 1.The system becomes active.<br><br>2. System sets the selected point to start point. |

**TABLE 12: List Staritng Points Test Case**

## 4.12. Set Destination Point Test Case

| **Test Case ID** | **Test Case Name** | **Test Case Description** | **Test Inputs** | **Outcomes** |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. “Path Finding Mode” must be selected. | | | | |
| TestCase No:12 | Successful Case | Test Case with all the pre-condition is true. | 1.User selects “Set Destination Point” in input form. | 1.The system becomes active.<br><br>2. System sets the selected point to destination point. |

**TABLE 13: Set Destination Test Case**

## 4.13. List Destination Point Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. “Path Finding Mode” must be selected. | | | | |
| TestCase No:13 | Successful Case | Test Case with all the pre-condition is true. | 1.User selects “Select From Destination Points” in input form.<br>2.User selects one of the points from list. | 1.The system becomes active.<br>2. System sets the selected point to destination point. |

**TABLE 14: List Destination Point Test Case**

## 4.14. Find The Path Test Case

| Test Case ID | Test Case Name | Test Case Description | Test Inputs | Outcomes |
|---|---|---|---|---|
| **Pre-condition:** User has already entered the system via a browser and user has unity-3D web plugin. “Path Finding Mode” must be selected and destination and start point must be entered. | | | | |
| TestCase No:14 | Successful Case | Test Case with all the pre-condition is true. | 1.User selects “Find the Path” in input form<br>2.User selects one of the points from list. | 1.The system becomes active.<br>2. System shows the path to the user. |

**TABLE 15: Find the Path Test Case**

## 4.15. Environmental Needs

In this section software and hardware requirements for “COW-3D” will be described.

### 4.15.1. Software Needs

- The program can be run at any computer which has browser and internet connection.
- System does not require high performance.
- The application can be run on every web browser regardless of the operating system environment.

### 4.15.2. Hardware Needs

There is no specific hardware constraints.

## 4.16. Special Procedural Requirements

For our system there is no any specific,special requirement to run application.

## 4.17. Intercase Dependencies

All our use-cases are independent from each other.Therefore to control any test-case we do not have to do other test-cases.

# 5. System Test Report Details

The following sections describe the overview of the test results, the detailed test results, the retionale for decisions and the conclusions and recommendations.

## 5.1. Overview of the Test Results

All the items specified in Table-1 is tested during the test procedure.

### 5.1.1. Test Environments

- Windows 7, 4 GB RAM, 1.6 GHz Processor and 1 GB video card
- Windows 8/8.1, 8 GB RAM, 2.2 GHz Processor and 2 GB video card

## 5.2. Detailed Test Results

Detailed test results summary has been given in the following sections.

### 5.2.1. TC1 – Start

This functionality runs without any problem. The user can start and can select explore or path finding mode.

**Result : Succeed**

### 5.2.2. TC2 – Move Forward

This functionality runs without any problem. The user can move forward successfully.

**Result : Succeed**

### 5.2.3. TC3 – Move Backward

This functionality runs without any problem. The user can move backward successfully.

**Result : Succeed**

### 5.2.4. TC4 – Move Right

This functionality runs without any problem. The user can move right successfully.

**Result : Succeed**

### 5.2.5. TC5 – Move Left

This functionality runs without any problem. The user can move left successfully.

**Result : Succeed**

### 5.2.6. TC6 – Camera Up

This functionality runs without any problem. The user can move the camera up successfully.

**Result : Succeed**

### 5.2.7. TC7 – Camera Down

This functionality runs without any problem. The user can move the camera down successfully.

**Result : Succeed**

### 5.2.8. TC8 – Camera Right

This functionality runs without any problem. The user can move the camera right successfully.

**Result : Succeed**

### 5.2.9. TC9 – Camera Left

This functionality runs without any problem. The user can move the camera left successfully.

**Result : Succeed**

### 5.2.10. Set Starting Point

This functionality runs without any problem. The user can set the starting point successfully.

**Result : Succeed**

### 5.2.11. List Starting Point

This functionality runs without any problem. The user can list the starting point options successfully.

**Result : Succeed**

### 5.2.12. Set Destination Point

This functionality runs without any problem. The user can set the destination point successfully.

**Result : Succeed**

### 5.2.13. List Destination Point

This functionality runs without any problem. The user can list the destination point options successfully.

**Result : Succeed**

### 5.2.14. Find the Path

This functionality runs without any problem. The FPS(First Person Shooter) moves from the starting point to the destination point succesfully.

**Result : Succeed**

## 5.3. Rationale for Decisions

Rationale for decisions is that maximize the sequence of tests with the minimal effort.

## 5.4. Conclusion

Each test case is tested one by one with a proper of time. For example; user and camera moves are tested about 25 – 30 seconds and finding the path is about 3 – 4 minutes. According to out test, each test passed and never failed.