Middle East Technical University

Computer Engineering Department

CENG 492

SENIOR PROJECT

Test Specifications Report

Çağlar Ata 134714-5

Aysun Başçetinçelik 129752-2

Cemal Acar Erkek 127712-8

Mennan Güder 129784-5

Çağıl Öztürk 134782-2

TABLE OF CONTENTS

1. INTRODUCTION 3
1.1 Goals and Objectives 3
1.2 Document Scope 3
1.3. Testing Plan Scope 4
1.4. Constraints for Testing 4
1.4.1 Time 4
1.4.2 Staff 5
2. TESTING STRATEGY AND PROCEDURES 5
2.1 Testing Procedure 5
2.2 Test Items 6
2.3 Unit Testing 7
2.4 Integration Testing 8
2.5 Interface Testing 8
2.6 Validation Testing 9
2.6.1 Requirements Validation 9
2.6.2 Design Validation 10
2.7 Stress Tests 10
2.8 Alpha Tests 10
3. TEST SCHEDULE 11

1. INTRODUCTION

This document is released version of the "ANKA Yazılım" Test Specifications. Document is going to be used for

- Planning the testing phase of our project
- To explain test scenarios
- Constructing and Following well defined procedures for testing
- Defining an overall strategy for testing
- Increasing efficiency of testing

1.1 Goals and Objectives

Once source code has been generated, software must be tested to uncover and correct as many errors as possible before the delivery to our customer. Our goal is to design a series of test cases that have a high likelihood of finding errors. Because of the time limitation high efficiency and good planning of testing phase become our main goals.

As a result of the testing phase and corrections after that step we expect our system is

- Containing minimum number of errors
- Having high performance
- Satisfying the user requirements

1.2 Document Scope

This test specification document contains

- Ways of testing our software
- Plan of the testing phase,
- Procedures that we will follow during that testing
- Ways to keep records for testing phase

1.3. Testing Plan Scope

In order to ensure the quality of our final product we will carry out the following kinds of tests.

- **Unit Testing:** We will make no extra effort on unit testing because each function is nearly tested during its implementation; another reason for that decision is the time limitation.
- **Integration Testing:** Most of the effort for testing will be supplied for that kind of testing because there are so many modules from different libraries in our software. The communication between the modules has to be analyzed and tested carefully.
- **Stress Testing:** In order to avoid crushes in the final product we are going to test the system for the stress conditions.
- **Performance Testing**: We are testing the performance during the development of each function. We compare the frame rates before and after the function added to the system.
- **Alpha and Beta Testing:** Some possible end users are going to use the system and we are going to examine the system while they are using.
- **Interface Testing:** Interfaces are going to be tested for their functionalities

1.4. Constraints for Testing

We considered the following constraints during the planning of the testing schedule.

1.4.1 Time

We have nearly five weeks for the project, and development of the project is not finished yet. As a result of this time limitation we are going to do the test in parallel with the development phase.

1.4.2 Staff

Each of the group members has also a role in development phase thus it is another constraint to consider for the testing plan.

2. TESTING STRATEGY AND PROCEDURES

In this part of the test specification document, the procedures that are going to be followed during the testing are mentioned. The ways and strategies for testing phase are also determined and stated.

2.1 Testing Procedure

Our testing procedure is defined below.

Before the test is done

The following corresponding data have to be determined:

Step Number

In order to keep record for each test step give a unique number to each test step

Expected Results

Expected results are the valid outputs of the systems, under the supplied environment and conditions. This data must be acquired correctly because it affects the efficiency of the system directly.

Conditions for the test case

In order to get the expected result the environment that is going to result in the wanted affect has to be satisfied.

After the test is written and run

The followings will be done:

Actual Result

At the end of the test execution we got the test results either by observing the system or by examining the output. The data that we got in this way is the actual result.

Pass or Fail

In order to determine whether the system works correctly, compare the expected result with the actual result.

After all of these steps correct the error if the step fails. In both cases: pass or fail fill a form which is as follows:

Step Number &&Tester	**Conditions And Description**	**Expected Results**	**Actual Result**	**Status**	**Correction Status**	**Tested Code revision**

2.2 Test Items

All base modules in the project are going to be tested by considering individuals functions and overall interactions of these modules with each other.

The main parts of our project that are going to be tested are:

- User Interface
  - Input handling
  - Output integration
- Graphics Engine
- Sound Engine
- AI Engine
- Editor Engine
- Viewer Engine
- File Manager
  - Save
  - Load
  - Export Animation

We will also test the individual classes which are not covered during the testing phases of above parts.

2.3 Unit Testing

We are going to do unit testing when there is no way to cover some part of the project by any of the high level tests. In such a situation we will write tests for the class that have to be tested. We are going to write one test case for each function in the class that needs to be tested. We are going to consider the inheritance cases also.

Data Tests

There are so many data items to be tested in the system. Main data items that will be tested are:

- **Static Models:** Static models will be tested for their integration to the program. Since during the exporting step of the model problems related to the material properties, textures occur, each model must be examined for correctness of these items.
- **Animated Models:** In order to avoid
    - Problems in defined animations of each Model
    - Texture problems
    - Material properties related problems

  All of these must be examined for each model that is going to be in the final package.
- **Sound data**: The sound is going to be tested for the integration to the system
- **Saved Game and animation files**: Test to detect whether there is any problem during the load of a previously saved game configuration
- **Results:** Examine whether the results are recorded correctly, by considering format and data correction

2.4 Integration Testing

Our aims for using integration testing are as follows:

- Check whether components are work together properly or nor
- Find out the defects that could not be identified by the other testing strategies

The modules are constructed independently, and then when all the functions and fields are prepared the module is integrated to all ready related modules. Before the integration step all required test steps are prepared and performed. In that way there will be no need to do extra tests after the integration up to new module added to the system. As a result, at the end of the project we are going to have fully integrated and tested software in terms of module integration. By means of integration testing, which is broad story-level tests, we are able to verify the interactions between the various actions supported by the application across all controllers.

2.5 Interface Testing

In this testing strategy we are going to set the required condition and observe whether the expected result is got or not. Some examples of our interface tests are mentioned below:

Step Number &&Tester	**Conditions And Description**	**Expected Results**	**Tested Code Revision**
1- Acar	Open a new map in editor menu	A new map is created	1.2
2-Aysun	Save a constructed map in editor menu	The map is saved to a file	1.2
3-Mennan	Select an object	Object properties can be seen in "Özellikler Menü"	1.2

2.6 Validation Testing

All other test strategies up to now are related to the whether something is done correctly or not. However, the functionality may be wrong itself so we have to verify the functionalities to obey the requirements and design. As a result of this need we are going to perform two validation steps which are:

2.6.1 Requirements Validation

This type of validation is for testing whether the functionalities mentioned in the requirements are satisfied or not. The reference document for validations is Requirement Analysis Report.

EDITOR MODE

Animation Editor: The following functionalities to verify in this module are:

- Add, delete, select and move operations of static objects like houses, trees, parked cars etc.
- Mouse drag and drop actions
- Add the dynamic objects to the scene
- Go to any frame of the animation
- Add screen messages to the animation at a chosen frame.
- Record and bound sound clips to screen messages
- Create and save new animations, load and edit existing on

Game Editor: The following functionalities to verify in this module are:

- Add, delete, select and move operations of static objects like houses, trees, parked cars etc.
- Mouse drag and drop actions
- Add the dynamic objects to the scene
- Choosing the playable character
- Create and save new animations, load previously saved game to edit and add invisible action triggers to the game

VIEWER MODE

Watching Animations

- Load, save the animation and exit from the animation
- Getting a message
- Replying a message

Playing Games

- Load, save the game and exit from the game
- Change camera view
- Get message
- Move main character

2.6.2 Design Validation

In this validation step we are going to examine the difference between the implementation way and the design document. In the case of any difference we are going test the correctness of the change.

2.7 Stress Tests

In order to test the system on the borders,

- Add a lot of static and dynamic objects to the system
- Scale objects very much
- Place the objects in extreme positions
- Set traffic density to the highest and lowest value
- Write message string in the longest possible way

2.8 Alpha Tests

The system is going to be tested by the people outside the group members. These people are going to give feedback and also we are going to examine during their usage to recognize the possible crushes.

3. TEST SCHEDULE

Task	End Date
Test Plan Delivery	23.04.2006
Unit Test	5.05.2006
Integration Tests	12.05.2006
Validation Tests	12.05.2006
Stress Tests	12.05.2006
Correction	12.05.2006