Virtual Classroom Tool

TEST SPECIFICATION REPORT

CENG 492 Spring 2005

May 8, 2005

Mehmet Akif TAŞKIRMAZ
Serhat MEŞE
Abdullah ÖZTÜRK
Fırat KIZILER

Computer Engineering Department
Middle East Technical University

TABLE OF CONTENTS

1. INTRODUCTION

This section gives a general overview of the test specification for the virtual classroom tool KLAS.

1.1. Goals and Objectives

A good product should work perfectly, doing the right thing at the right time. To do that, the software has to go through a series of tests before its final release. Error free software is extremely difficult to achieve. Especially for software developed in a short time frame but high quality can be achieved with a detailed test specification. All (or at least most) of the test case will be listed, the development team will follow it step by step, item by item, to test all the necessary objects, data flows, limits, boundaries, and constraints of the software.

4DsmarTech would like to have a test specification to counter any difficulties that may impact the development and the future performance of the software.

1.2. Scope

An overall plan for integration of the software and a description of specific tests are documented in this section. Below are the different kinds of tests that we will take to ensure the quality of the software.

- Unit Testing
  - Instructor Application
  - Student Application
  - Server Application

  Unit tests will be performed using black box testing methods.

- Validation Testing
  We will test software as a whole, so all the units of the software will be included
  - Instructor Application
  - Student Application
  - Server Application

- High-order Testing
  The software will be tested for several test methods. Units to be tested are.
  - Instructor Application
  - Student Application
  - Server Application

2. TESTING PLAN

We want the product to be bug free. Below is the description of the testing procedure and strategy that we will follow to achieve this goal.

2.1. Software to Be Tested

We have written the functionalities of instructor, student and server applications in our final design report. Therefore, we will not mention about these in this report again.

2.2. Testing Strategy

In the following section we will describe the testing strategy. We will user four different methods to test our product.

2.2.1. Unit Testing

In the unit test case we will be testing the separate modules of the software. We will carry out white box testing where each module or component of the software is tested individually. We will test the components by passing data through it and we will be monitoring data to find the errors.

We will be looking for entry and exit conditions of the data. We will make sure that all the components work without any troubles.

The test primarily be carried out by the programmer who designed and implemented the module.

2.2.2. Validation Testing

In this method of the test we will look at the software requirement document to see whether there is a conflict between the product and the specifications written in that report.

We will perform the black box testing where the software is completed and we will test all the software components together. We will have several input data or test data that we will derive results for. We will insert this data in the software and will get results form the software. We will compare the results from the software with the results that we derived. This way will check for the validation of the software.

In case there are problems with the software we will create a deficiency list and will record all the problems in there. We will test all the components and subcomponents of the software to perform validation test.

2.2.3. High-order Testing

In this test method we will combine two other types of the testing. We will test for several different conditions by following test methods.

- **Stress Testing:** In this test method we want to monitor stress caused to system and the software due to overuse of network by many clients (instructor and students). We want to make sure that the system does not break down under the extreme use conditions.

- **Performance Testing:** Performance bounds are set during the design part of the software development. These bounds will help us in determining the effectiveness of the software.

2.3. Testing Resources

We will use several different resources to carry out the test on our software. Following is the test resources

- MSDN Library
- Windows Media Server Help
- Windows Media Encoder Help
- Internet

2.4. Testing Tools and Environment

Testing tools includes our personal computers. The computers will connected with ethernet cards.

2.5. Test Schedule

Following is the tentative schedule for the testing of KLAS.

System Testing

16/05/2005 – 23/05/2005

Corrections According to the Test Results

23/05/2005 – 05/06/2005

Final Testing

06/06/2005 – 11/06/2005

3. TEST PROCEDURE

In this section we will describe the test procedures in detail.

3.1. Testing Procedures

In this section we will try to describe over all software specification. We will describe the methods for all the different tests to be performed.

3.1.1 Unit Testing

In this method of testing we will test the smallest unit of software called modules. We will be testing all the important paths to find any errors within the boundary of module. So here we will apply sort of white box search. We will be testing parts of the software rather than the entire software. The modules are as follows.

Login Window

We will make use several different names to log in to the system. We will use correct and incorrect user names and passwords to access the software and thus access database. We will not be allowed to log in using incorrect passwords and error message will be shown. When correct password is presented we will be able to log in to the course content window. We will also test Login and Cancel buttons on this window by performing test above.

Course Content Module

The interface depends on the user name coming from the login window. The intructor should be able to add course, edit course, remove course, add student, edit student, remove student, add course material, remove course material and start online lecture. While adding course, we will leave one or more course information text fields empty, try to add a course with same course code already taken by another course and enter course capacity more than allowed by the system. While removing course, we will try to remove a course with registered students. While adding a student, we will try to add a student who has an already registered student id for that course. For all these situations, we should get warning messages.

Video Module

In this part, there will be no error because of the usage (They do not control this module). We will only test the delivery of the video content.

Whiteboard Module

We will draw several figures on the whiteboard. We will select, erase, move, undo, redo, save to file and load from file these figures. We will also test whether these figures are sent to the students correctly.

Presentation Module

In this part, we will test the handling of the presentation events in the student part, performed by the instructor.

Chat Module

We will test whether the chat messages are transmitted among the users correctly.

3.1.2 Validation Testing

This test is performed to validate the software. The test is known as black box testing where the entire software will be created and will test all the component of the software together.

We will enter predetermined data with expected results. We will compare predicted results with those that software gives us and will determine if the software is valid or not.

Every button, tab or menus will be tested. We will test the correctness of the database and will make sure there are no errors regarding database updates.

We should not have in troubles with the software since we have already tested all the parts of the software before. Every thing here should work fine and we should be able to validate the software. We will be validated only if each and every thing works in the software. Any software in the error will force us to wait for the validation until we have fixed all the errors.

3.1.3 High-order Testing

High-order tests are combination of several different test methods. We will be performing two different tests on our software.

- **Stress Testing:** In this test method we are concerned with the software's ability to allow concurrent transaction. Too much of the work at the same time may cause system shutdown or frees. We what to test and to make sure that this does not happen. As test procedure we will try to create as much traffic for the software as we can by opening several student applications concurrently.

- **Performance Testing:** We will be testing the software to see if it meets the performance criteria set during design system specification.

3.2. Test Record Keeping and Log

We will use table to log all the test, describe them and record the results of the tests. Then we will use this table for final testing after doing the corrections.

4. REFERENCES

1. http://www.testing-solutions.com
2. http://www.mhhe.com/engcs/compsci/pressman