TEST ENGINE SYSTEM

(TES)

Test Plan Specification

PHASE IV

April 29, 2003

Version 1.0

Team Members

Test Engine System - Test Plan Specification

Table of Contents

1.0 Introduction

The 'test engine' program provides test administrators an easy way to create exams, give exams, and monitor exam-taker responses. The 'test engine' will consist of an administration 'view' and a user 'view'. The administrator of the system will be the professor administering the exam. The user of the system will be the exam-taker or the student taking the exam. The exam-taker will scroll through each exam question of the exam he is taking and provide a response to the question before moving on to the next question.
Exam questions will be entered by the exam administrator prior to the exam time. Each exam will be 'activated' by an exam administrator. Once the exam is 'activated', the exam-takers will be able to begin the exam and post answers to exam questions. The 'test engine' program will be delivered as an intranet solution. Use of the program is envisioned to take place in one classroom at one given instance of time. Hence, if two exams are going on at the same time by two different professors, each classroom will have its own URL / intranet website to view the exam and enter responses to exam questions.

1.1 Goals and objectives

The goal of the system is to provide test administrators an easy way to create exams, give exams and monitor exam-taker responses. The system will also provide a uniform interface to exam-takers and an ability to immediately display exam results to the exam taker and compile exam reports to the system administrator.

1.2 Statement of scope

The presentation layer will be divided into two parts: administration and user. The 'logic engine' will consist of a library to process calculations required by the system administrator based on student responses to questions during the exam. The 'logic engine' will reside on the same server as the web-server. The data-store will retain all data related to the exam.
Inputs into the system will be by both the administrator and by the exam taker of the system. The administrator will enter an exam into the system. Exam type will be specified. Example exam types include 'real exam' and 'practice exam'. The administrator will enter an exam type. Example exam question types could be 'multiple choice' or 'true / false'. Based on the question type, the administrator of the system is prompted for an exam question and possible answers. The amount of answer choices the administrator enters will be dependent on the exam question type selected.
Another input into the system is answer responses entered by the user taking the exam. The user is prompted for the exam type to take. Example exam types include 'real exam' and 'practice exam'. The exam-taker is displayed an exam question and prompted for an answer to the exam question. Upon answering all questions for the exam, the user comes

to the last question in the exam. Upon clicking finish, the exam-taker sends all results of his exam to a persistent data store.
The administrator can also enter usernames / passwords into the system. The username / password will be used to authenticate the user in the logon process.
The administrator will also have the ability to view reports based on the exam responses. Sample reports include: List of current students for an administrators' classroom, exam results per student (group all student exam grades together) and student response report (showing a detailed response per student).

1.3 Software context

Currently, exams are offered by paper method. The administrator of the exam grades the exams and the next time the administrator meets with the test-takers, he gives the results. The administrator may also post exam results to a web page or some other interface easily accessed by large amounts of exam-takers. The 'test engine' seeks to fit within the existing process by automating the process and reducing duplicate processes performed by the administrator.
The 'test engine' fits into the overall process of the classroom test-taking process. The 'test engine' will attempt to automate the creation of exams and automate the evaluation of exam results. The 'test engine' will also provide the test administrator to re-use existing exam questions on future exams thus creating a 'question pool' from which to choose questions from.

1.4 Major constraints

A constraint of the system is that each user will need to have access to the system at the time specified. Ex: each user will need to have access to the system during the classroom time when the administrator is giving the exam. This may or may not be possible with current university computing constraints.
Another constraint of the system is that the 'test-engine' does not currently have a timer giving a certain time to the system. Hence, the administrator of the system will have to 'activate' an exam at the beginning of an exam and then make sure everyone is finished with the exam through an interface.

2.0 Test Plan

2.1 Overview

The rationale behind a solid test plan is to catch errors early in the software development process and reduce the cost of inevitable code errors. The idea being that the earlier a software error is identified, the less costly that error is.

2.3 Software to be tested

The presentation layer will be divided into two parts: administration and user. The 'logic engine' will consist of a library to process calculations required by the system administrator based on student responses to questions during the exam. The 'logic engine' will reside on the same server as the web-server. The data-store will retain all data related to the exam.
Hence two distinct areas are to be tested, the 'logic engine' libraries and the jsp pages that form the user interface to the system.

2.3.1 Interface to be tested

Exam Management
Allows the exam administrator the ability to enter a new exam into the system.

Example Exam Management test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
Next page	Enter data in the form fields and click next	The new exam question interface should appear.

Enter New Exam Question User Interface
Allows the test administrator the ability to enter a new question for a selected exam.

Example Exam Question User Interface test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
Clear form	Select the reset button.	The form fields should be reset to blank values
Enter question	Enter data in the form fields and click submit	A blank form should appear ready to accept another exam question in the system.

Exam Home Page
User enters credentials into the username / password fields to be authenticated by the Test Engine system.

Example Exams Home Page Interface test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
Enter correct User Credentials	Enter a correct username / password combination and click submit	The user should be authenticated and the next page displayed should be the select exam type page.
Enter false User Credentials	Enter false username / password combination and click submit	The login page should re-appear with a message that indicates incorrect username / password values were entered.

Select Exam Type
User selects the exam type when starting an exam.

Example Exam Type User Interface test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
'Real' Exam Type	Select the 'real' exam type from the screen.	The user should be directed to the Exam question page for a 'real' exam.
'Test' Exam Type	Select the 'test' exam type from the screen.	The user should be directed to the Exam question page for a 'test' exam.

Exam Question Type #1
User is displayed with exam choices in multiple-choice format.

Example Exam Question Type #1 User Interface test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
Next Exam question with no exam response	Do not select an exam response and click 'Next' on the form.	The same exam question should appear with an error message indicating that a exam response selection is required.
Next Exam question with exam response	Select an exam response and click 'Next' on the form.	The next question in the exam should appear.

Exam Question Type #2
User is displayed with exam choices in True / False format.

Example Exam Question Type #1 User Interface test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
Next Exam question with no exam	Do not select an exam response and click 'Next' on	The same exam question should appear with an error message

response	the form.	indicating that a exam response selection is required.
Next Exam question with exam response	Select an exam response and click 'Next' on the form.	The next question in the exam should appear.

2.3.2 Components to be tested and Test Plan

DataAccess component

The DataAccess component is an encapsulation of all database access and is called from jsp pages or other classes.

To test the DataAccess component, we will create a 'driver' program which will test each method of the DataAccess component.

Example driver test methods include:

Test Case Name	Test Case Actions	Test Case Expected Results
DataAccess Insert	Insert record into table, pass insert statement to DataAccess ExecuteInsert() method.	Check the database, make sure the inserted record is in the table with the expected values.
DataAccess Update	Update a record in the database, pass the update statement to DataAccess ExecuteUpdate() method.	Check the database, make sure the updated record has the expected values.
DataAccess Delete	Delete a record in the database, pass the delete statement to DataAccess ExecuteDelete() method.	Check the database, make sure the record that was marked for deletion has actually been deleted.
DataAccess Select	Select a record in the database and display it on the 'driver screen interface.	Check the database against the values displayed on the 'driver' screen interface and make sure they are consistent.

User component

A superclass that will have common methods and properties which will be inherited by more specific implementations of a user such as professor and student.

Example User component test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
SaveNewUser Profile	Pass the appropriate fields to the SaveNewUser() method from the jsp page or from a component.	Check the database against the values that were passed through the SaveNewUser() method.
GenerateReport	Call the GenerateReport() method of the User object	Check the values displayed on the screen against the expected

	from a jsp page or from a component.	results in the database.

Exam component

The exam object will be a data structure that will be passed around the test engine system.

Example Exam component test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
Set / Get property	Set a property value on the Exam datastructure. Get the property value from the object.	Ensure the value retrieved from the object is the same as the value set.

Question component

The question object will be a data structure that will be passed around the test engine system.

Example Exam component test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
Set / Get property	Set a property value on the Question datastructure. Get the property value from the object.	Ensure the value retrieved from the object is the same as the value set.

Utility

The Utilty object will contain miscellaneous functions used throughout the test engine system.

Example Exam component test cases include.

Test Case Name	Test Case Actions	Test Case Expected Results
CreateTable Method	Test the CreateTable() method of the Utilty object. Pass a ResultSet object to the CreateTable and display the resulting html on a web page.	Display the data retrieved from the database in the resultset as an html table on the web page.

3.0 Test Procedure

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

3.1 Testing Procedures

3.1.1 Unit Testing

Unit testing is the initial testing done after coding. In this process, the program logic is verified and the code is checked for conformance to design specification. In this method of testing we will test the smallest unit of software called modules. The modules are as follows.

Login Window

We will make use different student names to log in to the test engine system. We will use valid and invalid student id and passwords to access the test engine system. If it is invalid user name or password error message will be pop up. When correct user name and password is entered student will be able to log in to next window (Exam Tester window). Also test Submit and Reset buttons on this window by performing test above.

Exam Tester window

In this window has two radio buttons one for Real Test and other for Practice Test and one Next button. Student will be able to select any one radio button and click Next button. Now student will be get next window which is Question page.

3.1.2 Integration Testing

This is system and sub system test is the set of define and controlled activities for verifying the complete functionality of system and across sub system. The purpose of this testing is to determine a system and sub systems interact properly. The test will verify the sequence of calls to and from a module to its descendent modules.

At the end of the test all results should be positive. All of the software component should be work properly. All errors will be record them in the error document and fixed at the letter time.

3.1.3 Validation Testing

This test is performed to validate the software. The test is known as block box testing. Software structure, usability and the business rules are applies when testing in black box, which is exterior and internal component cannot be seen. Black box testing implies that the selections of test data as well as the interpretation of the test results are performed on the basis of the functional properties of a piece of software.

3.1.4 High-order Testing

High-order tests are combination of several different test methods.

Recovery Testing
Testing return to normal operation. Recovery testing assesses the ability of a system to return to normal levels of operation after a failure occurred. Here we are concerned with ability of the software to retrieve lost data.

Security Testing

We will be testing the Test Engine system to see if unauthorized students/users are able to access to the system. This testing we want to make sure that the security checks are working and no one is able to temper with the data.

1. Password Login
   We will use valid or invalid student id and invalid passwords to access the test engine system.
2. Modular Access
   Test engine system identifies the student and allows him or her to access only certain modules. Professor has permission to access all modules. We will test to see if the system restricts unauthorized users from accessing certain modules of the system. Also we want to make sure that the student cannot access modules that are only for the professor.
3. Priority Access
   We will test to make sure that student is not able to delete or update data entered by him or another student.

We should not have any problems with any sections of the security testing. Security measures are serious issue and we have made all attempts to keep it from any bug.

Stress Testing
We want to make sure that the system does not break down under the extreme use conditions.

Performance Testing
Performance testing verifies that your program meets specific performance and efficiency objectives such as response times under certain workload and configuration condition. The system has to process huge amount of data or perform many function calls within a short Period of times. Belo are just some of the performance bounds we will be look for.

Response time for login
Response time of browse function

Response time for one module to another module

Alpha Testing

Before the software release it is thoroughly tested within and among the developers group. Purpose of this testing is specially to ensure software structure, business rules and functionality of the software.

Beta Testing

This pre-release testing is done by the educated user group.

The purpose of Beta testing is to validate:-

a) User requirements/feature
b) End to End functionality
c) Work center methods and procedure
d) O A & M (operation ,administration, maintenance) procedure
e) Data quality
f) System security
g) Measures of success
h) Recovery procedure

3.2 Testing Resources and Staffing

Resources

This section should include a listing of all the resources (e.g., test environment, application executable, test tools, bug reporting utility, etc) necessary to perform the testing.

To ensure the quality of testing it is important to have a clean machine (i.e. clean formatted hard drive).
This application runs IE and Netscape browser.
Intel Pentium or compatible processor
Excel will be used for this process to report the bug.

Staffing

Unit Testing Coordinator- Faizul Haque
System Testing Coordinator- Artan Rukaj
Integration Testing Coordinator-Joseph Shoback

3.3 Test Record Keeping and Test Log

Test Record Keeping

MS Excel will be used to record the results of the tests. A test log is kept to monitor the tests that have been applied. An error or bug log is kept to monitor any problems that have arisen during testing.

Test Log

Test No.	Description	Expected Result
1	Checking the system security with user name and password	When correct user name and password is entered student will be able to log in to next window (Exam Tester window).
2	Exams managements page After entering all data we click the "Next" button	Enter Question Page Should be appear
3	In this page two radio button, one is practice test and other Real test. Student should select any one default select Real test.	When page appear Real test is selected. Other one not.
4	In Student page After selecting exam type we click the "Next" button	Questions page appear.
5	In Questions page student should be select any one radio button and click the Next button.	Next question appear or if it is last question than pop-up message exam finished.