**Testing Plan**

**for the**

**SIUE Department of Computer Science**

**CS425 / CS499 Senior Project**

**Software Design and Implementation Courses**

**by**

**Zach Benchley, Matt Lievens, Logan Maughan, Brian Olsen**

**of**

****

**Learning Outcomes-Based Assessment Database Team**

Revision 1.0

As Of: March 17, 2013

Change Log:

|  |  |
| --- | --- |
| Revision | Change Note(s) |
|  |  |
| 1.0 | * Initial Release |

Reviewed and Approved By:

Name Signature Date

Table of Contents

[1 Test Driven Development, Unit & Integration Testing 4](#_Toc383015297)

[1.1 Test Driven Development Description 4](#_Toc383015298)

[1.2 Test Driven Development Execution 6](#_Toc383015299)

[1.3 Stubbing Methods 6](#_Toc383015300)

[1.4 Unit Testing 7](#_Toc383015301)

[1.5 Integration Testing 7](#_Toc383015302)

[2 Prototype Summary 7](#_Toc383015303)

[2.1 Purpose of the prototype 7](#_Toc383015304)

[2.2 Efforts Summary 7](#_Toc383015305)

[2.3 Accomplishments 7](#_Toc383015306)

[3 Usability Test 8](#_Toc383015307)

[2.1 Determining our sample size 8](#_Toc383015308)

[2.2 Evaluation methods 9](#_Toc383015309)

[2.3 Resources needed 11](#_Toc383015310)

# 1 Test Driven Development, Unit & Integration Testing

## 1.1 Test Driven Development Description

The team will execute testing throughout each sprint, and more specifically before any functional code is written. The general model executed is going to follow Test Driven Development.

1. Test First Development – This is the core idea of Test Driven Development where test code is written before any functional code. A popular approach is to write functional code in small steps until the original test is satisfied. The main rule to be followed is, “No Test? No Code!” Figure 1 models the steps of Test First Development.



Figure 1 - Test First Development Flow chart

1. Refactoring – Test driven development is a composition of test first development and refactoring. There are different types of refactoring based on the role of the team member. The type of refactoring ultimately determines what type of test driven development is being applied. Below are the following types of refactoring and correlating roles.
   * 1. Behavior Driven Development – There is no test code nor functional code written during this development. This step is handled and maintained by the Owner Proxy by gathering information and flushing out requirements. These requirements are then flushed out into Agile use cases that can be used to develop acceptance tests. This step is handled mainly in the Concept phase and is continued each preparation phase in each iteration.
     2. Acceptance Driven Development – In this stage the first test code is written while the functional code is optional (since it is expected to fail.) The test code produced is called the acceptance test and its primary goal is to check the behavior of a given module. The important thing to remember in this stage is to avoid testing the functionality of the code (i.e. the methodology and efficiency of the code). The acceptance test executes black box testing as it gives a set of inputs and expects a specific output. The tests can be implemented either as a JUnit test to test behavior of a module or an integration test to check interaction between 2 or more modules. This can be the responsibility of the Team or the Scrum Master.
     3. Developer Test Driven Development – This is the stage that is iterated along with an acceptance test. A developer test is written to check for specific functionality. This test is analogous to a white box test as it has knowledge of the inner-workings of the module. This test can be written to fit all the requirements at once or can be iterated one method at a time. These tests are specifically JUnit tests to test specific functionality of separate modules. It is likely that once all developer tests have passed that the acceptance test will pass as well. If not make sure to check everything has been tested thoroughly.

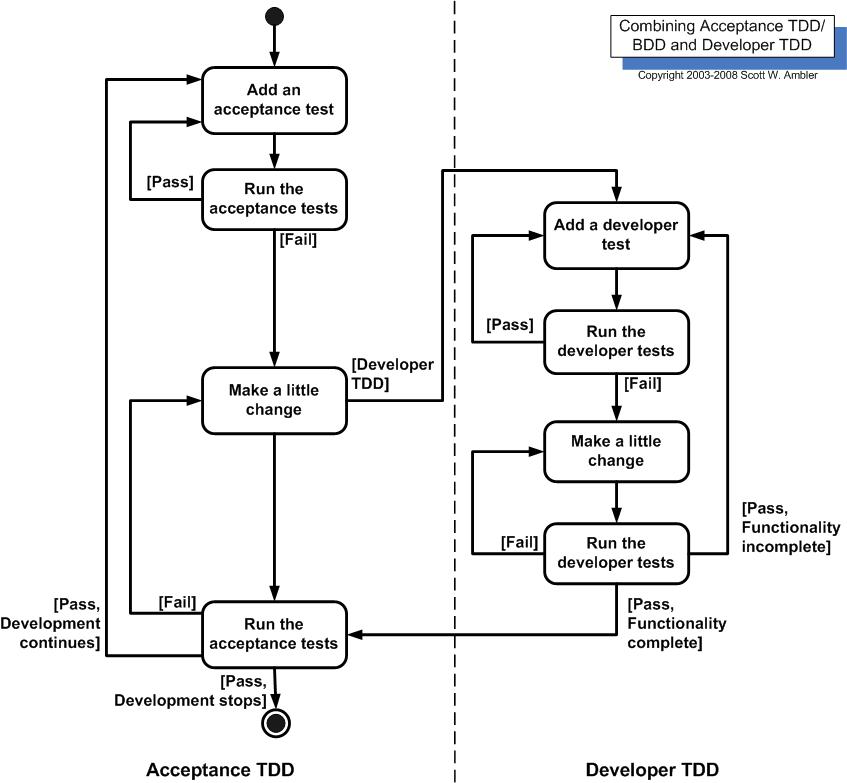


Figure 2 - Acceptance/Developer TDD Flow Chart

## 1.2 Test Driven Development Execution

* Sprint 1
  + Wrote stubbed unit tests to develop core database functionality via the façade (e.g. Calls being sent to the abstract query correctly, etc… ).
  + No TDD was applied to the prototype since the code was being thrown out.
  + Started development of utility class (QueryTestHelper.java) This is a stubbed instantiation of the models that can be used to simulate data coming from the
* Sprint 2
  + Wrote tests to develop core database functionality via the façade (e.g. Calls being sent to the abstract query correctly, etc… ).
  + The GUI was still prototype and still wasn’t being applied to the TDD standard.
* Sprint 3
  + Wrote tests to develop model class functionality. There is little to no need to apply Unit Testing to the view classes since the GUI elements are injected from the JavaFX fxml files. There is little to no need to apply unit testing to the controller classes since the controller is only reacting to events and passing information on to the models. The functionality of both view and controller will be tested during integration testing.
  + Continued TDD iterations to develop queries for the database.
* Sprint 4
  + After some consideration the Unit Testing methods used to test the database are being stripped out of the project as we have transitioned to prepared statements and there is no way to apply unit testing to this segment of the code. The important thing not to confuse is that this wasn’t a wast5e
  + Model unit tests were updated as needs changed.
* Sprint 5
  + Model unit tests were updated as needs changed.
  + Started development of integration tests
* Sprint 6
  + Model unit tests will be updated as needs change.
  + Integration tests
* Sprint 7
  + Model unit tests will be updated as needs change.
  + Integration tests

## 1.3 Stubbing Methods

For the QueryTestHelper.java we use a Google stubbing API called Mockito. It is useful when utilizing Test Driven Development since it doesn't require the objects to actually be implemented to test them.

## 1.4 Unit Testing

The team used the JUnit 4 API to execute our unit tests for individual classes. Having these tests allowed efficient ease of continual testing as Models changed and caught minor bugs being introduced throughout the development process.

## 1.5 Integration Testing

While the initial plan was to use JUnit 4 API to execute the integration tests, this proved to be less approachable as the database calls were created by using prepared statements and disabled verification through unit methods. The integration test was verified in a top to bottom fashion in a breadth first order that runs through the various interface dependencies and will ultimately make sure the calls from the MVC elements back to the database function correctly.

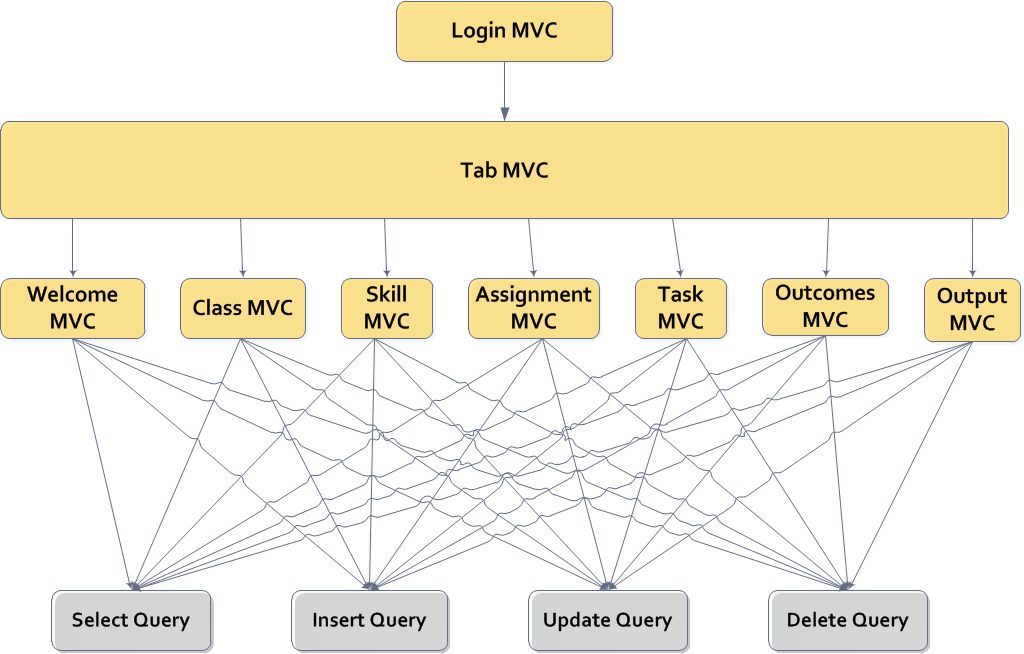


Figure 3 - Graph of Top Down Integration Testing Execution.

# 2 Prototype Summary

As stated in the LOBA-PP document Section 6.1 CS 425 we decided to do make LearnOBA’s user interface a prototype. The sections below summarize our efforts and accomplishments

## 2.1 Purpose of the prototype

The team built the prototype to demonstrate our vision of LearnOBA to the client through the first two iterations. This offers the ability to get quick feedback from the client without having to solidify a design for the user interface right away. Along with demonstrating what LearnOBA will look like, the prototype will also be used to test ways for the user interface to interact with the database (i.e. what is now the database façade). The last function the prototype was to give the team a sandbox to help learn JavaFX by creating an interface freely and not have to worry about making big changes while learning the language.

## 2.2 Efforts Summary

There were 48.5 person hours dedicated to the creation of the prototype not including the time spent learning JavaFX. This accounts for approximately ten percent of our entire development hours. We displayed the prototype twice to the client individually and once for the final presentation. Internally the prototype was used as a development tool to reveal inconsistencies with the database and the GUI, as well as, use for testing scenarios for the database.

## 2.3 Accomplishments

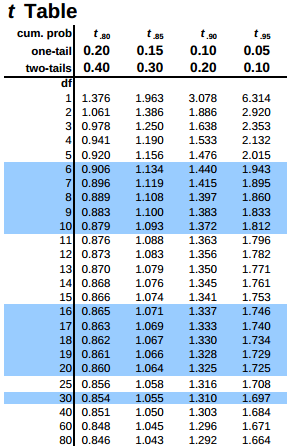
Reference LOBA-PP-TEST-RESULTS.docx

# 3 Acceptance / Usability Test

The team will conduct a acceptance test usability test to confirm that our software meets the requirements of the client. Since time is of the essence and there will need to be a suitable sample size to test the team will conduct a survey that will be taken by various faculty. This will be done in a two phases. The first phase will be a structured set of steps guiding the faculty member along various paths of the software. During this phase the faculty member will answer questions that measure the amount specific requirements were met. In the following phase there will be an open ended comments section requesting that the faculty member addresses any concerns he or she has that weren’t covered and any suggestions they may have.

## 2.1 Determining our sample size

In order to conduct an accurate study the team is going to propose a 90% or 95% confidence interval for how accurate the statistics suggest the team meeting our goal. The scores will be results out of 10 and for the first set of calculations will accept a marginal error of ± 1. Since no previous tests have been done the standard deviation of the population will have to be fabricated upon an agreed amount of the client and the team. The tentative amount used will be a standard deviation of 1.5. To evaluate n we will use the t-statistic formula and will be solved with respect to n. To obtain a starting point 10 will be used as the starting sample size to obtain the degrees of freedom for the t-statistic and will be run through variations as needed.



Based off the values above the optimum number of participants for the usability study is between 5 and 6 so we will choose the higher value of 6 faculty members. The table below contains the same calculation applied to a 95% confidence interval, as well as, different values for the marginal error. These values are tentative until accepted by the client.

|  |  |  |
| --- | --- | --- |
| d (marginal error from ) |  |  |
|  | 8 | 6 |
|  | 16 | 10 |
|  | 27 | 16 |

## 2.2 Evaluation methods

The faculty will run through a list of instructions each along the way asking them on a 1 – 10 scale how they feel a specific requirement was met. Below are the questions that will be used to evaluate if certain requirements are being met. The bullet points beneath each question is the requirements being tested and will not be included in the survey.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to create a login identity and log into the system?**

* The teacher wants be able to login and logout to the LearnOBA system to have access to its functionality and to leave the system.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to select, create and remove a class?**

* The teacher wants to be able to create a class so that all of the courses that they are teaching can be offered LearnOBA.
* The teacher wants to be able to remove a class when it is over so that the system does not get overcrowded.
* The teacher wants to be able to select a class so that they can manage it.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to add and remove a student from a class?**

* The teacher wants to be able to add a student to a class so that the roster can be kept up to date.
* The teacher wants to be able to remove a student from a class so that the roster can be kept up to date.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to view, add and remove the list of skills?**

* The teacher wants to be able to add specific skills to the skill list so that it can be kept up to date.
* The teacher wants to be able to remove specific skills from the skill list so that it can be kept up to date.
* The teacher wants to be able to view a list of skills for a class so that they can decide if it needs to be updated or not.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to view, add and remove assignments from a class?**

* The Teacher wants to be able to add an assignment to a class so that the class is keep the class up to date.
* The Teacher wants to be able to remove an assignment to a class so that the class is kept up to date.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to view, add and remove tasks from an assignment?**

* The teacher wants to be able to add a task to an assignment so that it can be kept up to date.
* The Teacher wants to be able to remove a task from an assignment so that it can be kept up to date.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to view, add and remove skills from a task?**

* The teacher wants to be able to add a skill to a specific question so that a student can be judged on their mastery for that skill.
* The teacher wants to be able to remove a skill from a specific question in case they do not want to test a student on mastery for that specific skill.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to set the pre filled outcome scores and update those scores?**

* The teacher wants to be able to add student scores so that the student’s progress can be accurately tracked.
* The teacher wants to be able to change student scores so that the student’s progress can be accurately tracked.
* The teacher wants to be able to pre-fill all of the outcomes for each task so that progress tracking and score updating is easier.

1. **On a scale of one to ten, one being the worst and ten being the best, how would you rate the ability to export individual assignment info sheets and the blackboard mastery sheets?**

* The students want to be able to look up their scores so that they can see how close they have come to mastering their skills.
* The teacher wants to be able to export student scores in a comma delimited form so that it can be uploaded to Blackboard.
* The students want to be able to view a breakdown of exams so that they can better see what skills they need to work on.

1. **To the best of your ability summarize your overall impression of the software and add any comments or suggestions.**

## 2.3 Resources needed

* Google forms document.
* 6 - 27 Faculty members (to include Dr. Foster).
  + Personal or School Computer.
  + 15-20 min of their time.
* Our finished executable.