Requirements Analysis

Drop the Needle Test

Summer 2012

Nicholas Marks – nickloop@gmail.com

Project Description:

This program, Drop the Needle, will simulate the kind of test that a music student would be given. The program will take in a user-defined playlist and using specified options the program will randomize the playlist and choose 30 second clips from each piece. The user will be prompted to state the title, movement, opus, performers, composer, and relevant information. Once all the questions have been completed the program will grade the user against the information stored in Itunes, or other text based answer key. If Itunes does not have the correct information stored then the user can input their own answers before the test. Once graded the program will display a grade(percent and letter) and the correct answers. Wherever the user made a mistake the answer will be highlighted. The scorecard will be saved to a database so the user can review another time.

-Vision-

Problem Statement:

Students taking music classes which require listening and memory recall on tests need a study tool. Currently there are not many ways to study for the listening portion of exams, especially alone. By using an mp3 or media player it is easy to study the beginnings of pieces, but if you will be tested on any part of the piece it is more difficult. When using the mp3 or media player you have to look at the screen to fast forward which gives away the answer. This program allows you to study alone and uses media and information already stored on your Itunes. This way you can focus on listening and answering the questions without looking at the media player and seeing the answers.

High-level goals:

1. Create multiple GUI's that take in user specified options, simulate the test based of the options, and display the score and correct answers.
2. Correct testing mechanics that simulate a test environment.
3. Players can start a test and play smoothly till finish without bugs or errors.
4. Test is fun and effective as a learning tool.

Problems/Concerns:

1. Creating a fun and engaging looking GUI.
2. Writing code that is easy to read and follow.
3. Parts of the algorithm dealing with option, randomizing and playing 30 second clips of sound, saving state mid-test, and saving completed tests for future reference.

User Goals:

1. Player: To use as a fun game to play with friends, or as a study guide for Drop the Needle exams.
2. An easy to use and understand interface which promotes quick and fun gameplay.

System Features:

1. Easy to set-up test with many different settings as options.
2. Playback of each question allowed twice.
3. Randomization of test so each one is unique. This way the user gets the most exposure to the pieces required for the test in different ways.
4. Test questions formatted the same way as one would see on an actual exam.

Other Requirements and Constraints:

1. The possibility to favorite questions to use on another test.
2. The possibility to target difficult sections of pieces.
3. The possibility to create test questions and answer keys.

Use Cases:

Name: Define Options

Primary Actor: User

Main Success Scenario: Default options are set. 10 Questions, Random sections of each piece, All Movements, No Composer limits, No priority for pieces, and File path is through Itunes so enter a playlist name.

Extensions:

1. User can change the number of questions, sections of piece, movements.
2. User can type in limits for composers and pieces.
3. User can use the browse button to find playlist or folders for the File Path.
4. User can type in the full path.

Name: Initialize Options

Primary Actor: System

Main Success scenario: The Player has chosen the default options, so the system only searches for the playlist/folder specified for the file path. The option state is saved and sent as input for the test initializer.

Extensions:

1. User changed the default settings, the system changes the option state and sends it as input for the test initializer.
2. Easy, Medium, Hard modes.

Name: Initialize Test

Primary Actor: System

Main Success scenario: The option state is an input from the options menu. The Test is created as specified by the options.

Extensions:

1. The Option state is loaded from a previous test simulation and a new test is created with the same specifications.
2. A Test state is loaded from a previous test and the same test is administered.
3. A Test state is loaded from a previous test and the same test is administered, in a different order.

4. An incomplete Test is loaded from a previous test and the same test is continued.

Name: Take Test

Primary Actor: User

Secondary Actor: System

Main Success Scenario: The user listens to each clip twice and answers all questions. After each clip is played the user indicates to go onto the next question. Answers are saved by the System to be used in scoring or reloading an incomplete test.

Extensions:

1. The test is timed and once the two clips are played the user has a specified amount of time before the next question will begin.
2. The user submits his answer early and indicates to move onto the next question early.
3. Easy, Medium, Hard modes... Hard: answers must be typed in by memory. Medium: answers are given as multiple choice based on the other items in the playlist as well as a random sampling of other answers. Easy: answers are given as multiple choice based on the other items in the playlist.

Name: Score Test

Primary Actor: System

Main Success Scenario: The answers that have been collected by the system are compared to the answer key prepared by the system from the text saved in the Itunes library. A score is calculated and sent to the scorecard display along with the correct answers.

Extensions:

1. Part of the answer is unknown and the user is prompted to fill in the correct answer.

Name: Display Score

Primary Actor: System

Main Success Scenario: The score is taken as input along with the options state. The scorecard GUI is displayed with the options first, the final score second, and all the correct answers with mistakes highlighted.

Extensions:

1. The option to replay any of the clips is given next to each answer.

Schedule:

Iteration 1:

1. Implement GUI’s to display Options menu, Test simulation, and Score Display.

Iteration 2:

1. Implement interactions between the GUI’s to change state and setup the mechanisms by which the listeners will be used.
2. Begin developing algorithms for retrieving playlist, test initialization, random sound clips, answer key generation, scoring test...

Iteration 3:

1. Implement algorithms for retrieving playlist, test initialization, random sound clips, answer key generation, scoring test...
2. Update GUI’s to reflect changes made during algorithm development.

Glossary:

1. bla
2. blah
3.