Custom IEEE Standard Test Plan for Movies R US version 1

by Jason N Mansfield

Regis University

Custom IEEE Standard Test Plan for Movies R US version 1

(ANSI/IEEE Standard 829-1983)

Test Plan Identifier:

Movies v1.

Introduction

Due to the limited number of Tests available for UML based syntax and models this Test will be based purely on the creation of models implemented directly with the intention of including a Testing Framework.
Updated versions of this Test Plan will include history of Tests and improvements here.
The Testing Framework has two models in place as a goal for a permanent testing feature in the end product released to the customer:

IDtestHandler::TestTicket
IDtestHandler::TestGiftCard

In the future this bullet should reference previous archived test plans.

Test Items

Currently only the data writing and retrieving are of the highest concern to the testing process due to the solid use and previous testing done to other objects which are involved in this project.
Identification data objects Tickets.
Identification data objects Gift Cards.
Android Data Storage
No bugs specific have given us issues at this time. For future bugs found use this reference before troubleshooting: ISSUES
This test plan does not include issue related to the graphical interface, touch screen interface, nor server related issues beyond data communication.

Features to be Tested

Our primary concerns lie with the proper data communication.
Information retrieved from newly created Tickets and Gift Cards will be tested for accuracy. Data entities Payment::Ticket will be compared to IDTestHandler::TicketVerify. Data entities Payment::GiftCard will be compared to IDTestHandler::GiftCardVerify.

Features Not to Be Tested

Currently a large portion of features fall under test plans from previous projects and they should be consulted for bugs and issues related to: Haptic Touchscreen, Server Processes, Client web-based interface.
These features are already being used and have Test Plans in a large portion of other products already supported by us. Look to the Master Test Plan for issues related to these products.

Approach

We will be using a combination of JUnit and Monkeyrunner for Test Handling.
Initial problematic issues may be tested with Monkeyrunner by the development

team.
JUint will be used to create Test Handler Classes.
First review this hello world for a very basic Class used for testing: Hello, testing
Class should allow for creation of data object correctly followed by calling the data object correctly.
Concerns related to time it takes to search for and retrieve data should be documented.
Monkeyrunner should be used initially while code is in it beginning stages. The JUnit Classes should be created after all debugging and issues have been resolved.
Review of updated structural/security concerns with the Android API and JUnit should be monitored throughout project. Time frame has not been determined yet. Historically projects have a 3 - 8 month window.

Item Pass/Fail Criteria

Any corruption of data, omission of data, communication errors, etc will be considered a failure.

Suspension Criteria and Resumption Requirements

Testing will eventually be a built in feature. However, during the majority of the development process we will be using Monkeyrunner to determine data loss and communication. If we can confirm proper communication and write/reading of data objects development will move past testing.
At every level we will kickoff our custom Monkeyrunner script to test data objects. Even once JUnit Classes are created.

Test Deliverables

Deliverable documents: The SRS, Use Case Realizations, Class Models, Sequence Diagrams, Test Plan.
Test input will use MonkeyRunner scripts: TestTicket.py and TestGiftCard.py
MonkeyRunner and built in JUnit Classes will be used.

JUnit 4.8.2
Standard Monkeyrunner API within android API

Testing Tasks

Proper server side mysql database should be setup and verified. Client side software a machine should be verified for proper communication to Server.
Network connectivity, Mysql communication with our application is our biggest concern at this time.
An understanding a java communication with MySql is highly valued for this project, python is also needed for Monkeyrunner and should be no problem for our experienced team.

Environmental Needs

For Initial stages we need at minimum the Kiosk simulator, access to web interface, Fedora Server with LAMP setup, apache pre-configured for web interface.
Specify the level of security: root privileges for all developers. ("Tread lightly girls and boys.")
Identify special test tools needed: Using standard ADB for debugging, Developers may use different flavors for debugging if they feel it will improve both their performance and the product.

Identify any other testing needs: possibly GCC 4.6.0, GDB v 7.2.90.
Currently we have meet all our physical needs.

Responsibilities

Philip Bueschen has been managing, designing, preparing, executing, witnessing, checking and resolving issues using MonkeyRunner and will be in charge of scripting and teaching other members how to use these implementations.
Crystal Scott's team will be handling bugs that arrive using Philip's scripts.
Jason Mansfield's group will be handling environmental concerns.

Staffing and Training Needs

Our highest priority is to have at least 5 solid experienced Android Developer with at least one year of experience.
Philip, Jason and Crystal will be training less experienced members in Development specific to the following:

Philip: Python
Jason: MySQL and Java environment
Crystal: GDB, ADB, and other debugger training.

Schedule

Milestones: Apache Verified, MySQL verified, Java server/client test, Android API setup, more to come.
No forms are required for Apache, MySQL. Java, Android setup should have some light and to the point documentation related to any issues discovered. All Monkeyrunner tests should follow Philips template and be signed of by him as team lead. All JUnit tests do not require documentation.
Depending on the complexity of bug, errors or issues our goal is to solve all problems within a 24 hr time window.
Schedule is shown in TRAC wiki on server: 12.23.19.109 or http://solidapps.com/project1234/wiki
MonkeyRunner will be used throughout project. JUnit should never be used for primary testing. JUnit Classes can be used as primary off campus after final release by technicians.

Risks and Contingencies

Natural Disasters are of large concern due to the recent tornado's and hurricanes. Redundant backups are done daily and off site servers will be put into affect in a worst case scenario. Due to budgeting concerns in the event of a Natural disaster employees are encouraged to work from home and telecommute.
Please review Disaster Plan found here: http://solidapps.com/documents/dplan.pdf

Approvals

In the event of a Natural disaster it will be assumed approved.

Team Lead 1: ______________________________

Team Lead 2: ______________________________

Team Lead 3: ______________________________