Software Requirements Specification for the `PresDroid`

Authors:

Angel De Castro	Luis Retana
Nicholas Otto	Christopher Yip

Version 0.1

10/4/2013

Contents

1 Introduction

1.1 Purpose

This document describes the `PresDroid` product version 0.1. `PresDroid` aims to facilitate the creation of presentations that can be easily controlled through a simple Android UI. This SRS covers the entirety of the `PresDroid` project.

1.2 Intended Audience

1. Potential User (People wanting to give a presentation)

2. Dr. Pruski

3. Software Development Team

4. Software Testing and Validation Team

5. System Architect

This SRS contains the Product Scope, Overall Description, External Interface Requirements, System Features, Non-Functional Requirements, and Other Requirements of the system. The end of the SRS will cover the terms used (Glossary), the models of various aspects of the system (Analysis Models), and aspects of the system to still be determined (To-Be Determined List).

1.3 Product Scope

The system will have 2 main components that interact with the user. The 1st component is a desktop application that will enable the user to create and display presentations. The 2nd component is an Android application that will allow the presenter of the presentation (created via the 1st Component) to control the presentation. The presenter will have the following capabilities through the Android application:

- Scroll/flip through presentation slides

- Zoom in/out of the presentation

- Highlight text on the presentation

- Draw figures on the slides

- Point to specific content (like a laser pointer) through a simple Android UI and gestures

2 Overall Description

2.1 Product Perspective

The `PresDroid` is a new stand-alone system that does not depend on any other user systems. The desktop application portion will target the Windows and Linux operating system environments. The mobile application will be targeted for Android mobile phones, specifically Luis' and Angel's phone.
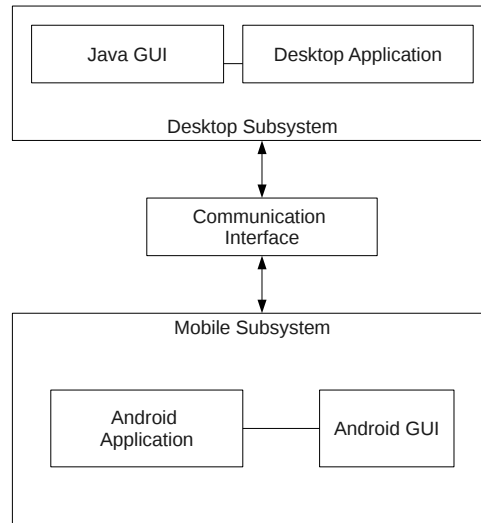
2.2 Product Functions

The desktop subsystem will perform the following functions:

- Build a new presentation
- Present previously built presentation
- Sync with mobile subsystem
- Interact with mobile subsystem

The mobile subsystem will provide the following functions:

- Sync with desktop subsystem
- Accept gesture input
- Interact with desktop subsystem

2.3 User Classes and Characteristics

The user groups will include students or professors giving simple, quick, and effective presentations.

2.4 Operating Environment

The hardware requirements for `PresDroid` are a Windows or Linux computer where a projector is compatible for the desktop portion and Android mobile phone. The computer and phone used must have networking (HTTP or Bluetooth, TBD) capabilities.

2.5 User Documentation

User manual.
Help section in desktop and mobile application.

2.6 Assumptions and Dependencies

We assume Bluetooth messaging will not be data-constrained and that the computers will have bluetooth messaging capabilities.

3 External Interface Requirements

3.1 User Interfaces

UI Characteristics for both Desktop and Mobile Applications:

- The applications will use the GUI Frameworks provided by the system they are built upon. For example, the Mobile application will make use of the native Android GUI Framework (Widgets, Spinners, etc.) while the Desktop Application will make use of the Java Swing Framework (JFrames, JButtons, etc.).

- The applications must provide a button within the UI which will present the user with a Help dialog box or screen.

- The applications will present errors in two ways:
    - 1) If the error is critical (such as the connection has been lost between the desktop application and mobile application), then a Message/Dialog Box will appear that the user must acknowledge before proceeding.
    - 2) If the error is NOT critical (such as invalid input), then text should appear towards the bottom of the UI in red text to notify the user of a non-critical error occurring.

- The layout of UI elements should adapt to suit to the system that application is installed on. For example, the Android Application UI should be able to adapt to different types of mobile devices (Phone and Tablet) so that elements in the UI dont appear in unexpected places.

3.2 Hardware Interfaces

The interfaces between both applications and the underlying hardware will be provided through the frameworks used in development such as the Android SDK/Framework and the Java Framework. As mentioned earlier, the supported devices will include Android devices for the Mobile Application and Windows and Linux machines for the desktop application.

3.3 Software Interfaces

Both the Mobile and Desktop subsystems will be built upon Java and Android frameworks, respectively. These frameworks will provide abstract interfaces to operating system features such as file management and multi-threading. Both subsystems (mobile and desktop) will need to create and process a variety of external messages, including, but not limited to:

- Pair up with mobile or desktop application request

- Zoom presentation in/out message

- Move pointer on presentation screen to a specific point message

- Kill connection between desktop and mobile applications message

3.4 Communications Interfaces

The two major subsystems (as depicted in the Product Perspective section above) will communicate using a Communication Interface. This Interface will store all messages or requests to be processed and will handle communication between the two subsystems. It will provide a simple and consistent interface where a component can easily check if it has any requests or messages to process. An example of a request is an incoming request for the desktop application (from the Mobile application) to pair up with the mobile device so that the two applications can be in sync and can communicate. This Communication Interface will need to rely on some form of networking (HTTP, Bluetooth, etc.) to be provided by the host machine. The Communication Interface will need to run on a separate thread so that the program will not stall while waiting for incoming messages. The Communication Interface must make sure to protect against corruption of its message pool through the use of Semaphores and/or Mutexes, etc.

4 System Features

Our system needs to communicate two devices, a computer running either Linux or windows, with a portable Android device. The communication is needed so we can control the computer application from the Android device. The computer application will show a simple interface for presentations. The Android device will have to control the change of slides, zoom in/out, writing, moving mouse, go to page and any other function supported by a simple presentation program. If possible we intend to see the pc screen on the Android device.

4.1 Communications

4.1.1 Description and Priority

The Android device needs to send and receive data from the computer application. To achieve this goal a communications protocol must be used such as Bluetooth or HTTP. The feature has the highest priority of a 9 since the whole system relies on the Android device communicating with the computer.

4.1.2 Stimulus/Response Sequences

This feature must be present from the start, right after the user turns both devices on. The moment the devices are switched on they must be able to communicate with each other so the normal use of the application can continue

4.1.3 Functional Requirements

The communications must be robust, the state of the link must be constantly checked so there is no down time. In case of loss of communications the system must be able to create a new channel of communication. If this fails the user must be prompted of the error.

4.2 Synchronization

4.2.1 Description and Priority

Before any other function becomes available we need to synchronize the Android device with the computer. This must be done by selecting in both machines the sync button. Once both devices find each other they can pair up. The feature has the highest priority of a 9 since the whole system relies on the Android device maintaining a link with computer

4.2.2 Stimulus/Response Sequences

This feature must be present from the start, right after the user turns both devices on. Once on, the first task to complete must be a successful synchronization, so the normal use of the application can continue. To accomplish this link both devices must be searching for a pair, once found they can create the link.

4.2.3 Functional Requirements

The link must be robust, the state of the link must be constantly checked so there is no down time. For these checks any package sent must receive a confirmation package. If data fails to reach its destination the system must resend the package. In case of complete loss of the link, the system must be able search for the same device to try and reconnect automatically. If this fails the user must be prompted of the failure.

4.3 Android Gestures

4.3.1 Description and Priority

The Android Application must be able to accept input from the user by hand gestures. Using the Androids touch screen the user will send commands to the Android application. These commands must be then sent to the computer were they will be processed and a task will be executed. Such tasks will include the passing of slides, zoom in/out, moving the mouse, etc. This is feature is of the highest priority 9, since its the core function of the system.

4.3.2 Stimulus/Response Sequences

After both applications have synced this feature must be made available, a drawn gesture on the Android application must be directly followed by an action in the computer application.

4.3.3 Functional Requirements

This feature must be robust and responsive with no lag. Any input into the Android device must be immediately show on the computer screen. This way the user gets a feeling of complete control and responsiveness. In case of an error the user will be prompted with a message to repeat the gesture.

4.4 Simple Presentation Interface on Computer

4.4.1 Description and Priority

The task of the computer application is to function as a presentation GUI, it must be able to open and create presentation showing them on screen. The priority of this feature is the

highest 9, since all future features rely on it, needing it as a visual interface.

4.4.2 Stimulus/Response Sequences

The presentation interface must appear once both applications have been synchronized, the screen must reflect the changes from any input the user sends through the Android device screen.

4.4.3 Functional Requirements

Should be a robust and reliable program, it's controlled by gestures through the Android device so the inputs will be largely varied and vague.

4.5 Computer Screen on Android Device

4.5.1 Description and Priority

Instead of having a specific Android interface the computer screen could be shown on the Android device. This way the user can select or underline exactly were intended without the need of a mouse cursor. This also makes it easier for the user to change slides or zoom in/out at specific points. This feature has a priority of 5, it's not essential for the system it simply makes it more interactive.

4.5.2 Stimulus/Response Sequences

The presentation must appear on the Android device once both applications have been synchronized, the screen must reflect the changes from any input the user sends through the Android device screen.

4.5.3 Functional Requirements

Should be robust and reliable, it's controlled by gestures through the Android device so the inputs will be largely varied and vague.

5 Other Nonfunctional Requirements

5.1 Performance Requirements

5.1.1 Mobile to Desktop Communication

The delay between user interaction on mobile device and registration of user action on the desktop environment should be minimized.

1. Requirement Completely Met: Mobile to desktop communication appears instantaneous.

2. Acceptable: The mobile to desktop communication contains predictable lag that does not distract user performance.

3. Not Met: The user is unable to compensate for inconsistencies or poor mobile to desktop communication performance.

5.1.2 Syncing of Mobile to Desktop

The time it takes to sync the mobile device to the desktop suite should be less than 4 seconds.

5.1.3 Desktop Program Start-Up

The time it takes to start the desktop suite should be less than 8 seconds.

5.1.4 Desktop Program Response

The presentation creation program should have negligible response times accepting and displaying user input.

5.2 Safety Requirements

There are no known physical safety requirements at this time. Caution should be exercised when using any unfamiliar device, however.

The desktop suite shall not modify any user files other than the files currently opened by the user. Furthermore, the desktop suite shall perform an "auto-save" while the user modifying a presentation. These auto-saves will occur once every 5 minutes.

5.3 Security Requirements

- Each presentation will only allow syncing to one mobile device.

- A mobile device will not disconnect another mobile device from its connection to the desktop suite.

- A desktop will not automatically sync with a mobile device.

- Requests made outside of standard mobile to desktop interface will return a standard error message.

No requirement is given that communication needs to be encrypted.

5.4   Software Quality Attributes

The source code will adhere to the following attributes:

- The source code will be reasonably documented. All non-trivial functions or methods will have header documentation describing the following:

  1. Name
  2. General description, including pre-conditions
  3. List and description of input parameters
  4. List and description of returned values
  5. Information about state change during method execution (post-condition)

- The source code will adhere to a single programming style, namely the Stroustrup variant of K&R style.

- All code blocks will be enclosed in curly brackets, even if not explicitly required.

Above all else, the system will strive to be usable. We want flexibility, but if rigid assumptions must be made to release the product on time, then so be it.

5.5   Business Rules

This product will ship by the end of the fall semester.