Spring 2017 – CISC207

Sell Used College Books – Shopping/Everyone

Application Design Document

Tuulikki Laitinen

Date: 2/5/2017

Table of Contents

[1. INTRODUCTION 4](#_Toc481495817)

[1.1 Purpose 4](#_Toc481495818)

[1.2 Scope 4](#_Toc481495819)

[1.3 Requirements 4](#_Toc481495820)

[1.4 Design Overview 5](#_Toc481495821)

[2. SYSTEM ARCHITECTURE 5](#_Toc481495822)

[2.1 Architectural Design 6](#_Toc481495823)

# INTRODUCTION

## 1.1 Purpose

Sell Old College Books Application is a digital marketplace for Mesa College students to sell their used college books to other students, add images about the books, chat anonymously about the books, prices and agree about meeting location. Suggested meeting location will be in campus area. Application category is “Shopping”.

This application is needed, since currently the book selling method is the following: the students are leaving their ads on to the library wall with their information, where anybody can see them. This application offers easier and safer way to sell books.

## 1.2 Scope

The target audience of the app is Mesa College students. Audience can be later widened to cover the other San Diego Community College students too. The content rating of the application is “Everyone”.

## 1.3 Requirements

The application must have any five of the features described in Spring 2017 - CISC207 Blackboard site. These are the requirements:

* At least 3 different sounds, 3 images, and 1 animated button
* Menus. See: https://developer.android.com/guide/topics/ui/menus.html
* User Feedback screen (Activity) that basically has a big text box and a button to send you an email
* A random pop-up dialog at some point of your app to "Rate this App" where the user can select the number of stars (1 through 5)
* When the app starts up for the first time, display a Splash Screen with some image or logo on it (Between 1 and 10 seconds)
* A Settings menu option that takes you to a Settings page with at least two settings options that work (i.e. turn sound on/off)
* Internationalize your app (localization) with two additional language of your choice
* Create a Canvas and Bitmap, with touch events
* ListView of data
* Animated Sprite Sheet
* Animations using ViewAnimationUtils class
* In-game advertisements
* Google maps integration
* Camera or video capture capability
* Connecting to MySql Lite or to a mobile cloud service
* Background music
* Saving the state of the game, or any other document from your app to the phone device
* Ability to print the current screen of a page of your app, using PrintHelper
* Publish the app on Google Play
* Any additional feature of your choice (must be approved by me first)

## Design Overview

The main features of the application are:

1. Account and login information handling:

* Google/Facebook login is implemented with Firebase Google Sign-In and Facebook Login
* Email/Password authentication is implemented with Firebase Password Authentication
* To save account information and show settings is implemented with PreferenceFragment, and preference.xml is used with addPreferencesFromResource-method.

1. Set up book to be sold:

* Access to camera application, post book, set price, show Toast to user about posting, and set the book image and information to list of books, option to delete book from the list.

1. Negotiating about a book:

* Selecting book from list, checking individual book image and more information, chat to seller to negotiate about price and purchase location.

# 2. SYSTEM ARCHITECTURE

## 2.1 Architectural Design

Figure 2.1 - This deployment view depicts the interaction between subsystems. Each mobile client subsystem can access the server but not other mobile clients and there is only one central server subsystem.

The Mobile Client Query Application is broken down into two major subsystems, the mobile client and the server. The two subsystems will interact with each other by using a server/client model. The server will store all of the data, including but not limited to: the database of item information, login information, and error logging information. The client will send and receive data to and from the server to view or edit information about the items. Mobile clients and web clients will also send error logging information to the server, however, this information will only be visible via the web client with the right permissions.

The subsystems will use XML messaging to communicate. Each subsystem will be capable of converting requests into XML messages and parsing the XML messages back into usable data.

By using these design pattern, multiple mobile clients and web clients can access the data and each subsystem can be updated or switched out without affecting the other subsystems.

2.2 Decomposition Description

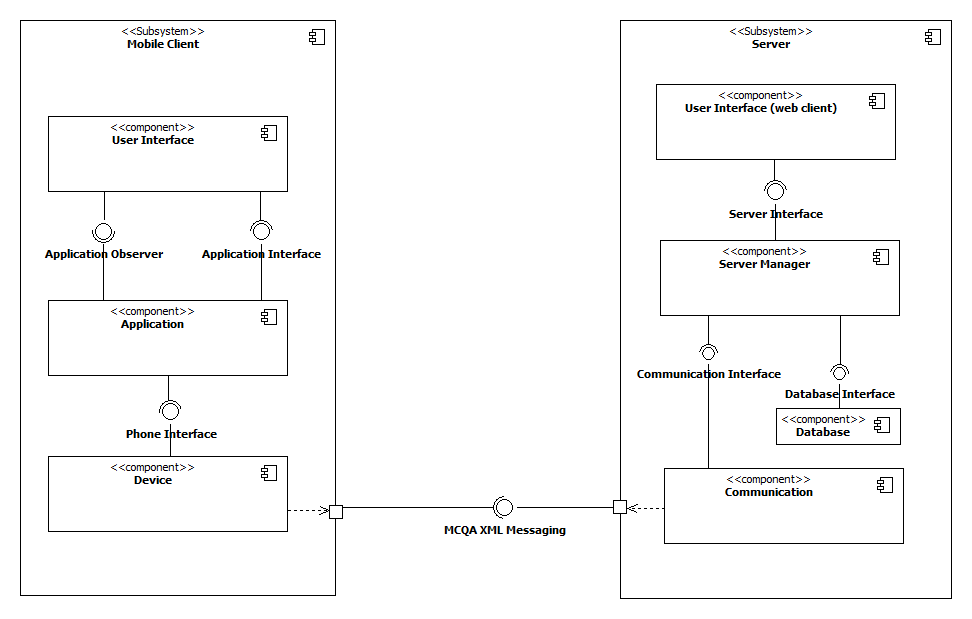


Figure 2.2 - This logical view depicts the decomposition of each subsystem into components and how the components interact with each other.

The mobile client subsystem is divided up into a three layered architecture, it has a user interface, application, and device layer. Each layer has its own interface that other layers can use to interact with it. The user interface layer contains an observer object and updates its data, using data from the observable application layer, via the observer pattern. The application layer handles threads, logs, and converts messages from the user interface layer into XML messages and sends them to the device layer. The device layer handles the interactions with the hardware, all the features of the phone necessary for the application, including but not limited to: wifi, gps, and ports to send and receive data to and from the server. Each layer is also a component that can be individually updated or replaced as long as the interface remains the same.

The server subsystem contains a web client component, server manager component, database component, and communication component. The web client component is a simple desktop application that connects to the server and follows the same rules for communication as the mobile client, it sends and receives XML messages. The server manager component handles the threads, message parsing, and database queries. The database component stores all of the data for the system. The communication component handles the ports and the sending and receiving of messages.

2.3 Design Rationale

The server/client model was chosen due to its ability to store all of the data in a central place and allow access to multiple system simultaneously. This design also allows supports the use of multiple types of clients, which is especially important as the system already contains two different clients.

This design has a few issues and trade-offs associated with it. Since all of the data is stored in a central location, if something happens to that server then all of the clients will lose connection and permanent data lose can occur. However, by having all of the data in a central location, it is cheaper and simpler to manage.

The design also hinges on network access. If there is no network access available, the client systems will not be able to send or receive any data. This is acceptable as the projected size of the database will exceed the space limits on almost all of the clients. In addition, syncing databases between large numbers of clients is unfeasible.

The layered and component architectures were chosen as they both support principles of maintainability, portability, flexibility, testability, reusability, and interoperability. The system can be ported to different platforms by changing the device component to match the hardware the system will be ported to. This swapping feature also allows for easier maintenance and updates to the system, replacing a single component rather than re-downloading the entire application. This architecture enhances interoperability by using interfaces to communicate between components, thus increasing cohesion.

**3. COMPONENT DESIGN**

3.1 Component Description

3.1.1 Mobile Client Subsystem

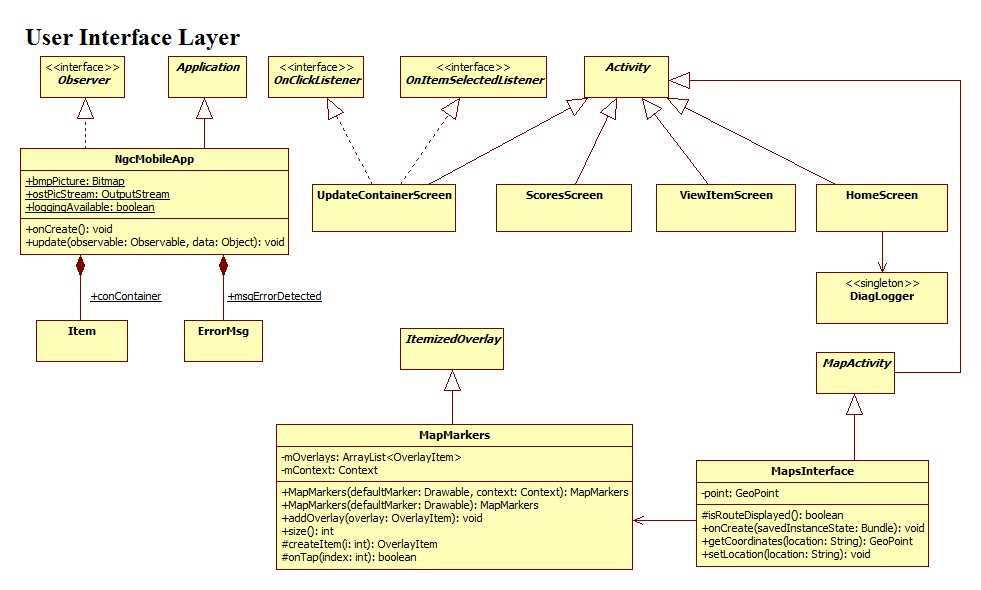


Figure 3.1 - The detailed design of the User Interface component.

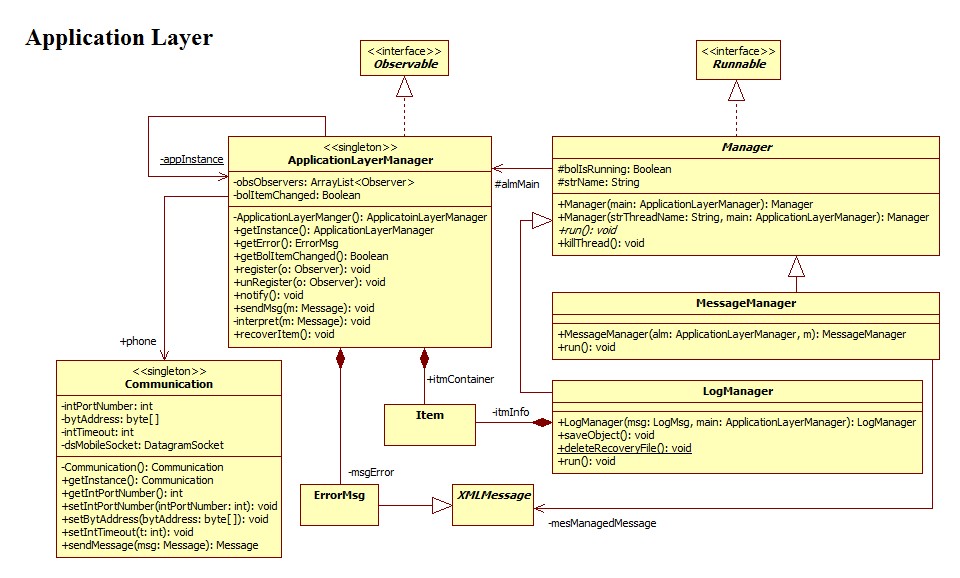


Figure 3.2 - The detailed design of the Application component.

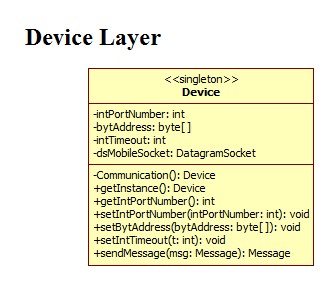


Figure 3.3 - The detailed design of the Device component.

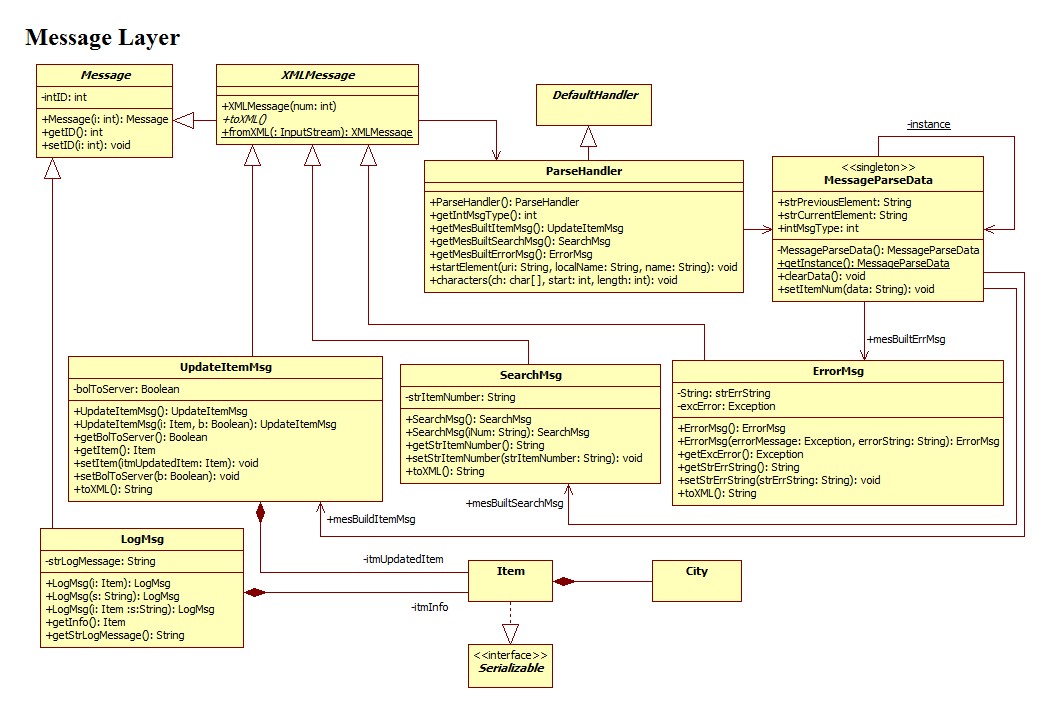


Figure 3.4 - The detailed design of the Messaging component.

3.2 Data Dictionary

The information domain of our system consists of the different types of information about commercial cargo items. Our contact described a list of data types which has been added to a central database. The data stored in our central database is queried by the server which receives messages from the web-app and the Android app. The data is organized by tables in a relational database.

* Additional Notes - lets users add any additional information about the item to the database.
* Analytic Scores - up to 5 analytic scores that let users.
* Client Number - ALPHA-NUMERIC item name.
* Consignee - Company having item shipped.
* Current Destination - Current item location.
* Fused Score - Averaged analytic score.
* Hazmat Code - Codes to relate to any unstable or concerning contents.
* Manifest Description - What is in the item.
* Password - Security code
* Previous Port - Shows the previous location of the item.
* Shipper - Information on who was shipping the item.
* State of Client - Set the item to normal or anomalous.
* TransPort - Up to the last 5 previous locations.
* User Name - Save usernames to allow access.

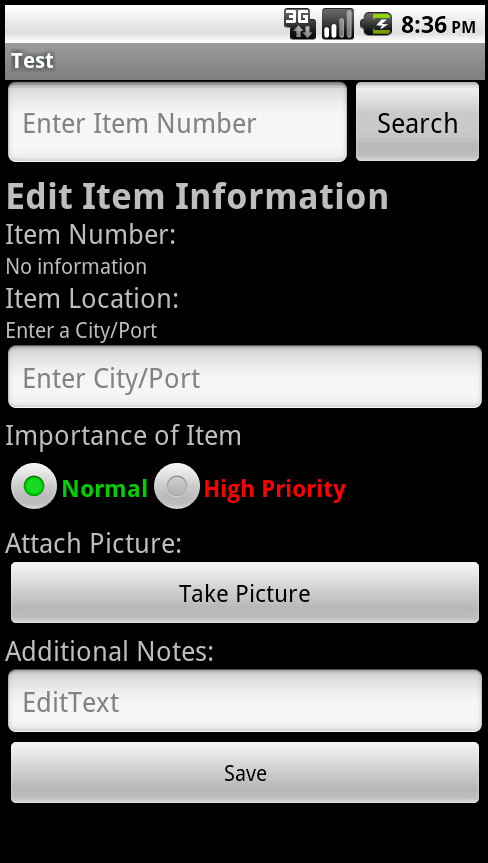
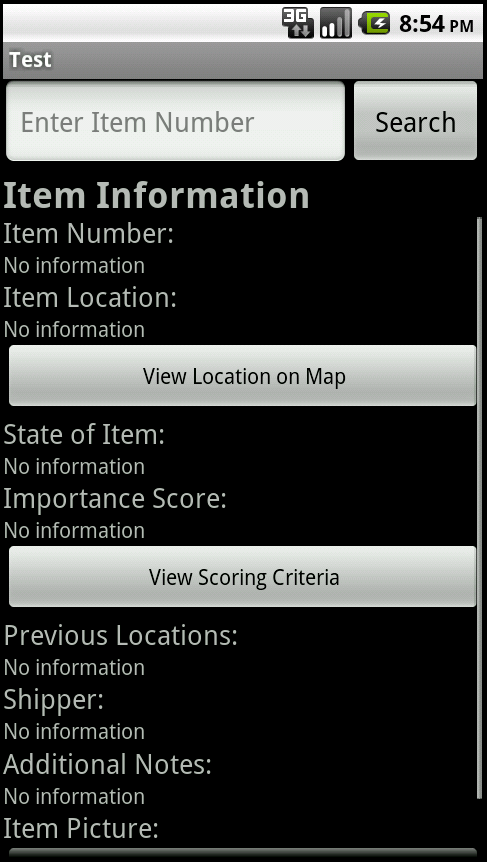
**4. HUMAN INTERFACE DESIGN**

4.1 Overview of User Interface

The user interface has been designed to be similar to the standard android interface, a black background with white text was chosen as our standard because the ease of visibility and ability to read. The flow of the interface has been designed around the idea of searching for items, updating information, and displaying the returned. After displaying the information returned an option will be available at the bottom of the list to edit information about that item and save that new information. After selecting save information from the update information screen it will take you back to the item information screen and display the new information that was changed about that item.

4.2 Screen Images

4.2.1 Mobile Client UI



Start Up Screen

Client Information

Update Information

Analytic Scores

4.3 Screen Objects and Actions

* Buttons - Clickable boxes that allow data manipulation and screen changes.
* Text Boxes - Allows user to input any ALPHA-NUMERIC characters in order to search for items and change information.
* Radio Buttons - Circular buttons that allow a single item in a group to be selected.
* Spinner - Generated drop down list of defined items that allow one to be selected.

**5. REQUIREMENTS MATRICIES**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Mobile Client Requirements | | | | | | |
|  | MCR-01 | MCR-02 | MCR-03 | MCR-04 | MCR-05 | MCR-06 | MCR-07 |
| User Interface | X |  | X | X | X | X | X |
| Application | X |  |  |  | X | X |  |
| Device |  | X |  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Server Requirements  (Web Client Requirements) | | | |
|  | WCR-01 | WCR-02 | WCR-03 | WCR-04 |
| User Interface (web client) | X | X |  | X |
| Server Manager |  |  |  |  |
| Database |  |  |  |  |
| Communication |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Server Requirements  (Database Requirements) | | | | | | |
|  | SDR-01 | SDR-02 | SDR-03 | SDR-04 | SDR-05 | SDR-06 | SDR-07 |
| User Interface  (web client) |  |  |  |  |  |  |  |
| Server Manager |  |  |  |  |  |  |  |
| Database | X | X | X | X | X | X |  |
| Communication |  |  |  |  |  |  |  |

The current builds design supports the alerts features. However, the current implementation does not support that feature due to time constraints.