6300-Summer-Team02

6300Sum14Project2

Software Architecture Document

Version <1.0>

Revision History

Date	Version	Description	Author
07/05/2014	1.0	Initial Draft	Traci Fairchild

Table of Contents

Software Architecture Document

1.Introduction

This document provides a description of and roadmap to the high level system design of the Coffee Cart System for CS6300-Summer-Team02.

1.1.Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system and will be used by the project leads and the development team.

1.2.Scope

The scope of this document involves the complete development of a Coffee Cart System which will be developed for the management of coffee and dessert sales at cart vendor establishments.

1.3.Definitions, Acronyms, and Abbreviations

See the Glossary section at the end of this document

1.4.References

Applicable references are:

1. Google Play, http://developer.android.com/distribute/googleplay/about.html
2. SQLite Android API, http://developer.android.com/training/basics/data-storage/databases.html
3. Android API Platforms, http://developer.android.com/tools/revisions/platforms.html
4. Oracle Technology Network, Java SE (JRE and JDK information), http://www.oracle.com/technetwork/java/javase/overview/index.html
5. Supplementary Specification Document, https://github.com/gt-ud-softeng/cs6300-summer-team02/tree/master/6300Sum14Project2/Deliverable2
6. Use-Case Document, https://github.com/gt-ud-softeng/cs6300-summer-team02/tree/master/6300Sum14Project2/Deliverable2
7. Android Application, UML Deployment Diagram Example, http://www.uml-diagrams.org/android-application-uml-deployment-diagram-example.html

1.5.Overview

This document attempts to represent the system from different viewpoints: architectural, user-based, design, process and deployment.

2.Architectural Representation

This document presents the architecture as a series of views; use case view, logical view, process view and deployment view. There is no separate implementation view described in this document. These are views on an underlying Unified Modeling Language (UML) model.

3.Architectural Goals and Constraints

There are some key requirements and objectives that have a significant impact on the architecture. They are:

1. The system shall be developed and compiled with the Android SDK for platform 4.4 Revision 2, API level 19. [3]
2. The system shall be compiled with JDK compliance level 1.7 [4]
3. The system will use the Android API, SQLite, for storing and retrieving data[2].
4. The system will be accessible for download from Google Play [1].
5. All Owner functionality must be available from any device, with or without internet connectivity
6. The system must ensure complete protection of data from unauthorized access. All remote accesses are subject to user identification and password control.
7. All performance and loading requirements, as stipulated in the Supplementary Specification [5], must be taken into consideration as the architecture is being developed.

4.Use-Case View

Below are the significant use-cases identified in the Coffee Cart System. They represent the central functionality of the final system. For a complete listing and full detail of the use-cases, refer to the References section. [6]

Use Case: Add Customer
Goal: The system will successfully add a new customer
Scope: Component

Use Case: Make Purchase
Goal: The customer's purchase is recorded in the system
Scope: Component

Use Case: Update VIP Points
Goal: The number of VIP points is incremented upon customer purchase
Scope: Component

Use Case: Pre Order Dessert
Goal: The customer successfully places a pre-order for a dessert
Scope: Component

Use Case: Calc Cust Status
Goal: The system successfully calculates and saves the customers VIP status
Scope: Component

Use Case: Cust Purchase Hist
Goal: The system successfully returns all the purchases made by a customer and the VIP points he or she earned in the
last 30 days and in total
Scope: Component

Use Case: Get Preorder
Goal: The system successfully returns all the preorders pending in the system
Scope: Component

Use Case: Daily Report
Goal: The owner will successfully produce two daily reports - a preorder report, and a daily purchases report
Scope: Component

4.1.Use-Case Realizations

Use Case: Add Customer
Primary Actor: Coffee Cart System
Brief: The system will add a new customer
Basic Flow:
1. The system presents the owner with a UI to collect all customer information, including a VIP card# that the owner obtains beforehand.
2. Upon submit, the system checks to see if the customer already exists in the db
3. If the customer exists, the user is informed and asked if it is the same customer
4. If so, the add request is denied
5. If not, the add request is processed
5. The system saves all customer information in the system database

Use Case: Make Purchase
Primary Actor: Owner, Customer
Brief: The customer makes a purchase and the details of the transaction are stored in the system.
Basic Flow:
1. The system provides an interface to the owner, allowing the owner to enter a customer purchase
2. The owner itemizes each purchase item
3. The system calculates the refill price, if applicable
4. The system calculates the order total
5. The system calculates the number of VIP points earned
6. The owner accepts payment and completes the order
7. The system saves the customer order in the database

Use Case: Update VIP Points
Primary Actor: Customer
Brief: The customers VIP points are calculated based on the purchase total and incremented in the system
Basic Flow:
1. The customers purchase total is calculated
2. The number of points earned is the same as the purchase total, rounded up or down to the nearest integer
3. The database is updated/incremented with this number of points

Use Case: Pre Order Dessert
Primary Actor: Customer
Brief: The customer can pre-order deserts up to a month in advance, if there are slots available
Basic Flow:
1. The customer is presented with a UI to determine the dessert name of the pre-order and the date desired
2. The system will check the desert name to see if it is a best seller, and to determine if any preorder

slots are available
for the date desired
2a. If there are preorder slots available, then the system saves the preorder, using the customers VIP card #
2b. If there are no preorder slots available, then the system denies the preorder sale

Use Case: Calc Cust Status
Brief: The system will calclate the customers VIP status based on the total number of VIP points earned
Basic Flow:
1. The system receives a customer VIP card#
1a. The system looks up the current VIP status of the card#
1b. The system looks up the permanent indicator of the card#. If it is 1 (meaning, if the card# current status is a
permanent status) then return the status
2. (Otherwise) The system implements "cust purchase history" for this card#
3. The system calculates the 30 day total purchases, and the total purchases since the card became active (ie, the
customers "memberSince" data)
3a. The system calculates the VIP points earned for each time frame
5. If the total number of VIP points equals or exceeds 5000, change the VIP status on the card# to GOLD, update the
permanent indicator to 1 and
return the customer status, the 30 day total and the total VIP points calculated
6. If the current status is not GOLD and the 30 day total of VIP points exceeds 500 then upgrade the VIP status for the
card#
7. If the current status is GOLD and the 30 day total of VIP points is less than 500 then downgrade the VIP status for
the card#
8. Return the customer status, the total VIP points earned by the customer since becoming a member, and the 30-day
VIP points total

Use Case: Cust Purchase Hist
Primary Actor: Coffee Cart System
Brief: The system will return all purchases made by a particular VIP Card#
Basic Flow:
1. The system recieves a customer's VIP card#
2. The system pulls all saved purchase information for this customer VIP card#
3. The system runs the "calc cust status" process to obtain the customer status and vip points
3. The system returns all purchase information, the customer status, the total VIP points earned and the 30-day VIP
point total

Use Case: Get Preorder
Primary Actor: Coffee Cart System
Brief: The system will return all pending preorder information saved in the system
Basic Flow:
1. The system pulls all saved preorder information that is still pending
2. The system returns all preorder information

Use Case: Daily Report
Primary Actor: Owner

Brief: The system will produce two daily reports upon the owners request
Basic Flow:
1. The owner is presented with a screen asking which report to produce
2. If the report chosen is a preorder report, the system will request an input date
2a. The system will run the "Get Preorder" process to retrieve the report
3. If the report chosen is a daily purchase report, the system will run the "Cust Purchase Hist" report for all customers
4. The system will present the chosen report to the screen

5.Logical View

5.1.Overview

The logical view of the Coffee Cart system is comprised of the 3 main packages: The App menu and activity layout package, the Customer Package, and the Database Interface Package.

- The android app menu and activity layout package contains classes for each of the forms that the actors use to communicate with the System.
- The Customer Package contains classes for controlling and managing the customer base (adding, maintaining, deleting or reporting on the customers).
- The Database Package includes entity classes for the customer and boundary classes for the interface with the SQLite db system.

5.2.Architecturally Significant Design Packages

edu.gatech.coffeecartrewards

This package contains the main menu layout of the system, along with the forms that the actors use to communicate with the system. Significant classes within this package are

- MainActivity - Called when the app is first launched, it creates the main layout of the app and starts the applicable activity when selected.
- PreOrdersList - uses edu.gatech.coffeecartrewards.db to interface with the datasource and retrieves all preorders at any given time.
- PreorderViewByDate - returns a list of all preorders for a certain date
- SalesList - uses edu.gatech.coffeecartrewards.db to interface with the datasource and retrieves all sales for a given time period.

edu.gatech.coffeecartrewards.customers

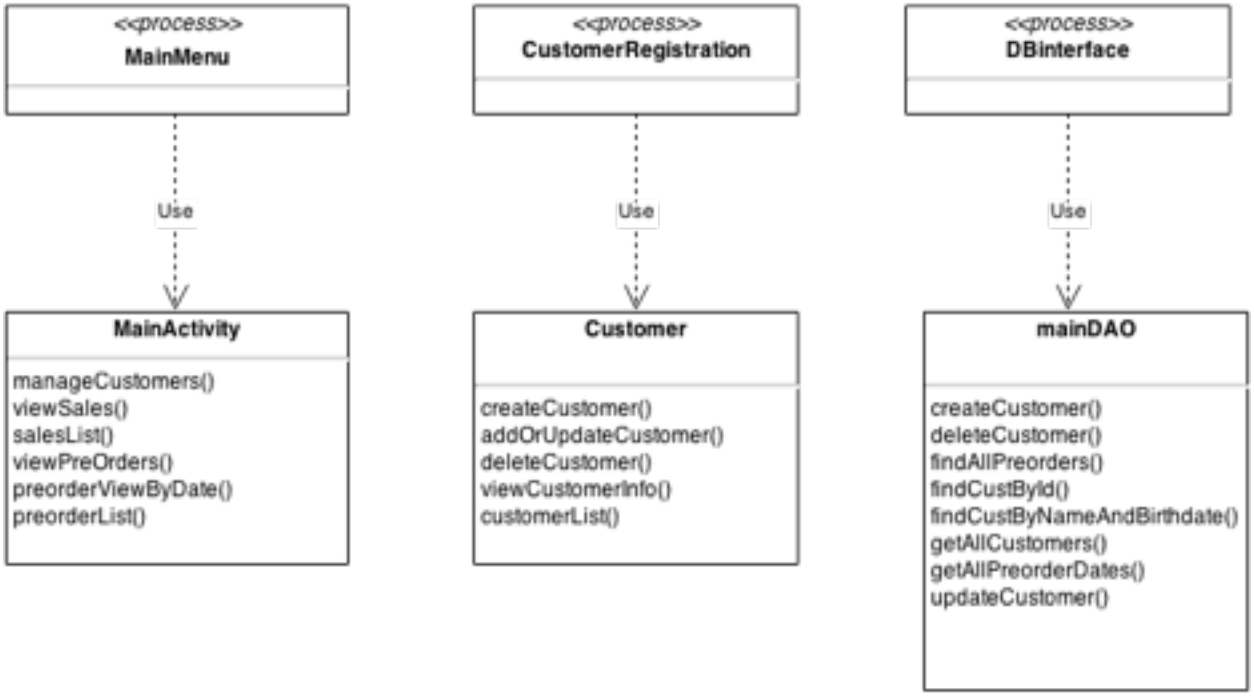
This package contains the main classes necessary to maintain the systems customer base. Significant classes within this package are

- CustomersAdd - this class will pull the information from the input fields, open up a connection to the datasource and attempt to add the customer, if not already present.
- CustomersAddOrUpdate - attempts to retrieve an existing customer from the datasource. If successful, the object is updated with the input fields, if not successful, a new customer record is input.
- CustomersList - this class displays all the customers in a list when the activity is selected.

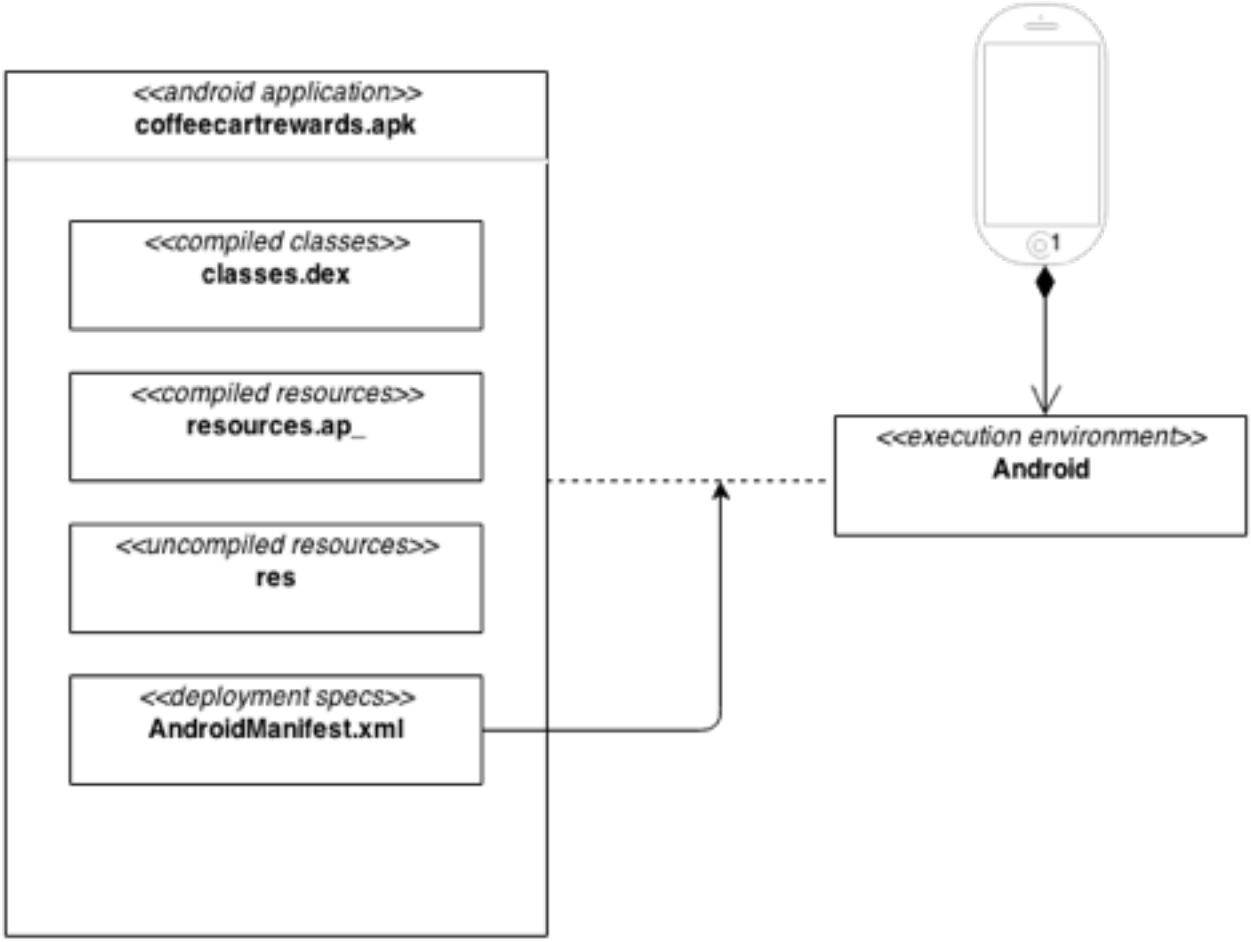
edu.gatech.coffeecartrewards.db

- Customer - the main class that identifies and contains the attributes for a customer object.
- Item - the main class for an item that can be purchased
- Preorder - the main class for a preorder
- Sale - the class implemented in a purchase
- CustomersDataSource - the main class that allows the customer object to interact with the datasource, and allows the application to retrieve existing data from the datasource.
- MainDAO - this class is the main class that allows data access to and from the datasource, such as inding, getting or deleting customer data.
- SQLiteHelper - this class creates the SQLite tables used within the application

6.Process View



7.Deployment View



Android is a software stack for mobile devices that includes an operating system, middleware and key applications. Android relies on Linux OS for core system services such as security, memory management, process management, network stack, and driver model. The Linux kernel also acts as an abstraction layer between the hardware and the rest of the software stack.
Android applications are written in Java. Android SDK tools compile and package the code along with any required data and resource files into Android application **archive file** having .apk suffix. The .apk file represents one Android application to be deployed to the Android-enabled mobile devices. [7]

8.Data View

The persistent data, such as customer information, order information, etc. is stored in a SQLite database on the device.

9. Size and Performance

The chosen software architecture supports the key sizing and timing requirements, as stipulated in the Supplementary Specification [5]:

The system should be able to manage and store upwards of 5000 customers, their customer information as well as their ongoing customer purchases.
The system should be able to manage and store upwards of 50 dessert items, including the relevant item and preorder information.
The system should be able to store the history of customer information and purchases indefinitely.
The app size should be no larger than 4GB, with an optimum size of 50MB or less

10. Quality

The chosen software architecture supports the key quality requirements, as stipulated in the Supplementary Specification [5]:

The system application shall be accessible to every Coffee Cart Owner within the enterprise, 24/7, except during scheduled maintenance periods.
The system application shall be available for download and installation at any time a Coffee Cart Owner joins the enterprise, or needs to add to or update the devices with which the software will run on.