SWEN30006 Software Modelling and Design
Workshop 10: Architecture

School of Computing and Information Systems
University of Melbourne
Semester 2, 2020

> **Completion:** You *should* complete the exercises ***as a team***. To receive a workshop mark, you **need** to demonstrate your active participation during the workshop. Your tutor will observe your participation for you to be eligible to receive a mark for this workshop. The active participation include (but not limit to) present your solutions, help your teammates, and actively engage with the team. Attending the workshop with only passive participation will *not* be given a mark. See the Workshop Overview under Subject Information on the LMS for more details.

Requisite Knowledge and Tools

It is expected by this point that you are familiar with the following terms and concepts:

- Architectural Analysis
- Logical Architecture Refinement
- UML Architecture Diagram Notation
- Specifying Architectural Interfaces
- Functional and Non-functional Requirements

It is suggested that you have attended the lectures on Architectural Design and Architectural Analysis and read, at a minimum, chapters 13, 33, 34, and 38 in Applying UML and Patterns by Larman (the prescribed text).

Introduction

This week we will consider the high level architectural design for software for a mobile Point of Sale system (POS System). Note that this is a different POS System from the textbook case study (NextGen POS), though your understanding of the NextGen POS example will better prepare you to understand and model the mobile POS System outlined below. We will start by reviewing the system architecture challenges that we would face when implementing this POS system. We will be tackling this problem by looking at the architectural structure and control decomposition while modelling our solution first with a Block Diagram and then with UML Component Diagram.

Introductory Example

Imagine that you are working as a Software Engineer for an IT consulting company and you have just been informed that your company has struck a deal with a large hotel business. At a high level the hotel requires an online hotel booking application that is accessible via a website and that is able to support the following features:

- Client registration and authentication
- Clients can search for available hotel rooms
- Clients can create, update and cancel bookings

Even though your manager has not yet provided you with detailed requirements, he has requested that you produce a high-level component diagram so that he can understand the technical complexity of the project at hand. This will help him negotiate a fair contract.

Note: High level architecture diagrams are intended to be abstract and should be flexible to change. These diagrams serve to present all the high-level components and layers that define a system. This is completely different to a design class diagram that is highly focused on business logic.

Block Diagram

Before we draw the required component diagram, we will start with a block diagram. A block diagram is used to provide a high-level visual overview of a system. Such a diagram does not have a formal notation and can simply include named blocks connected by lines. Each block should define a feature or component of a system and each line should connect blocks that are associated in some way. Below is an example of a block diagram for the online hotel booking application.

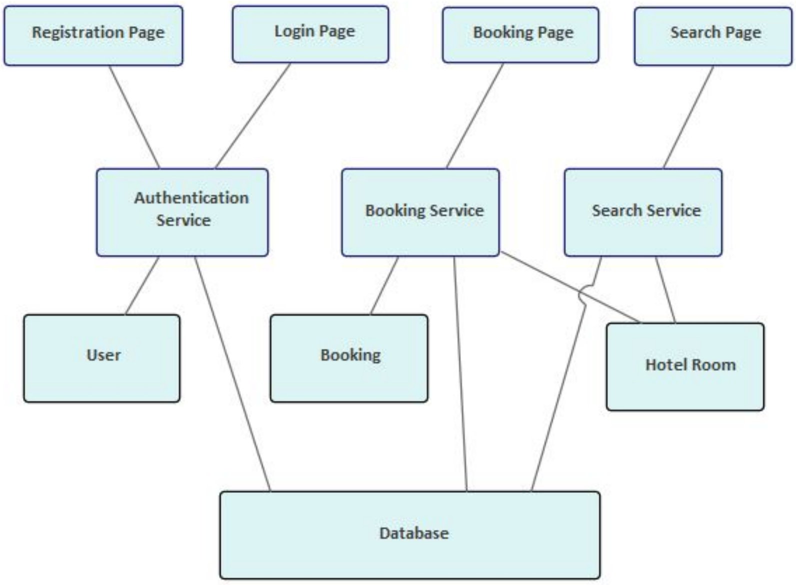

Figure 1: Block Diagram for Online Hotel Booking Application

Component Diagram

Just like the block diagram, a component diagram is also used to depict a high-level visual overview of a system. However, this diagram is slightly more detailed and formal with its notation. Each component is labelled with the component symbol in the top right corner and associated components are connected by a ball and socket. Consider the relationship between the Authentication Service and the Registration Page in which the Authentication Service is directly providing an interface to the Registration Page. The ball indicates that a component is providing a service or interface and the socket indicates that a component is using that service/interface. It is also possible for components to be embedded within other components. Below is a component diagram for the online hotel booking application.

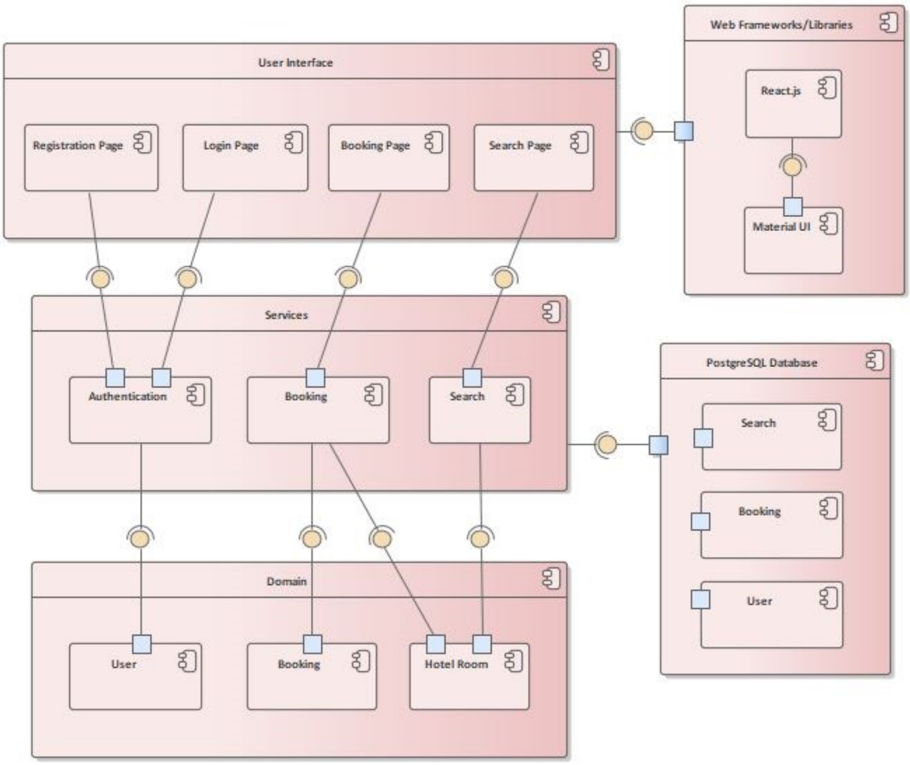

Figure 2: Component Diagram for Online Hotel Booking Application

The Point of Sale (POS) System

A timber and hardware store sells different kinds of timber, paints and painting products, lights and household electrical products, plumbing and bathroom products, and gardening products.

A project is underway with the aim of equipping staff with mobile service terminals that they can use to advise customers, get the latest prices if not displayed and inform customers about the latest discounts and sales in the store. Once sales are made the aim is to automatically adjust the level of stock through a stock monitoring system and to release the items sold so that they can pass through the exits monitored by a security system without raising an alarm.

Your task is to design a Mobile Service Terminal (MST) and inventory monitoring but not the security subsystem. You need to assume that you have an interface to the security subsystem that registers the barcodes of released items.

Rules of POS Operation

1. Store staff can walk around the store with an MST. A staff member can complete sales of items without the need to go to a register or sales station provided that customers have credit cards or store cards.
2. To complete a sale with a *credit card* the following sequence of steps must occur:
   i. The item's barcode is scanned.
   ii. The credit card is inserted into the machine and the card is validated against the credit agency for the card.
   iii. If the card is valid then the price of the item is charged to the card if enough credit remains in the card. If not the sale will be *declined* and no further action is possible for this sale.
   iv. If the amount is accepted then the item is 'released' from the store by sending its barcode to the central monitoring system. Released items do not raise alarms when they pass through an exit monitored by the security system.
3. Store cards are discounted debit cards, that is, the card acts as a debit card but applies a 10% discount to all non-sale and non-discounted items. The following sequence of steps must occur for a sale with a store card:
   i. The item's barcode is scanned.
   ii. The store card is inserted into the machine and the card is validated against the customer's debit balance.
   iii. If the card balance is sufficient for the item then the sale is made and the customer's balance is adjusted according to the price of the item. If the customer's balance is not sufficient then the sale will be declined and they can be directed to a cash register for combined cash/card sales.
   iv. If the sale proceeds then item is 'released' from the store.
4. The staff can also handle stock arrivals using their MSTs. They do this by scanning barcodes or stock numbers of new items. Once an item is scanned the inventory is updated automatically.
5. If no barcode or stock number exists on the item then they can look up the item's stock number by searching based on a description of the item. The staff member must select the correct item in this case and the selected item's stock number is used as in 4. above.
6. Supervisors can use the mobile data terminals to check sales for a specific sales consultant, or to view the available margin in an item's price.
7. Card sales must be done reliably, that is, that there is a 99.9999% chance that the sale goes through without a failure.

Use Case for the POS System

Below we present a single use case for the point of sale system.

Use Case POS-1 Credit Card Sale

Use Case	POS-1 Credit Card Sale
Actors	Customer, Staff Member, Credit Card Agency
External Trigger	Staff member initiates a sale using the Mobile Service Terminal (MST).
Internal Trigger	N/A
Brief Description	The use-case describes how the customer, staff member, credit card agency and MST system interact to achieve a credit card sale.

Flow of Events

1. The staff member holding the MST initiates a sale.
2. The staff member scans the barcodes of all the items in the sale.
3. The MST calculates a total sale price applying any customer discounts or sale policies that are in place.
4. The credit card details are read by the MST .
5. The MST system validates the card details against the credit agency. The validation procedure is as follows:
   i. The card details are read by the MST.
   ii. The customer types in their PIN.
   iii. The card number, the sale amount and the pin are sent to the credit agency.
   iv. The credit agency sends a message saying a card is valid and that the sale is approved.
6. **If** the card is valid **and** enough credit remains in the account linked to the card then the total price of the items is charged to the card account.
7. The MST system records the transaction and stores the transaction for accounting and later data analysis.
8. All of the purchased items are *released* from the store by sending a list of barcodes to the security system:
   - Released items do not raise alarms when they pass through an exit monitored by the security system.

Alternative Flows

- **6a)** If the card is not valid or the account does not have enough credit for the item then the sale will be declined and no further action is possible for this sale. The use-case ends.

Exercise One

The aim for your team is to develop a software architecture natural to the implementation of the POS system. Let's start by thinking about the problem! You can refer to lecture/textbook coverage of Architectural Analysis, including architectural factors and technical memos, as a guide.

- Can you group the requirements into cohesive subsets?
- What are the non-functional requirements in the problem?
- What requirements - *functional or non-functional* - will have the biggest influence on the architecture?

Now think about possible architectures.

- What is a natural decomposition for the POS system?
- What is a natural control pattern for the POS system?

Using a simple block diagram, sketch your architecture and assign responsibilities to the modules in your architecture. Consider any frameworks or libraries which you could use to make implementing your architecture easier. Do you need to make any adjustments or changes to use these frameworks or libraries?

When considering control, you are thinking about what is driving the system bahaviour. Is everything driven from the user interface? Are there services which are continuously waiting for requests? Is processing taking place concurrently?

Exercise Two

Specify your Software Architecture using Component Diagrams.

Now lets turn your simple block diagrams into UML component diagrams (you may wish to look here for help with the notation). Your task is to specify the POS system using a UML component diagram.

Here are some questions to help you understand your architecture better. The term "messages" below refers to concrete messages, that is messages that need to be transmitted in some form as in cases where communicating components run on different physical platforms.

- You will need to think carefully about the interfaces between components, the internal structure of components - for example, are there subsystems existing inside your components?
- Consider carefully the information provided and the objects which would need to be represented in a domain model. Are there any objects or other data elements that need to be shared between components?
- Do you have operations on the interfaces between components that send or receive the shared data?
- Which of your components communicate using messages and which communicate using method calls?
- Can you specify the format of messages that must be sent between components?