SOFTWARE DESIGN SPECIFICATION

HotRest

BY: S.M.Humraan (15k-2882)
Aleem Zulfiqar (15k-2847)
Azeem Khalid (15k-2845)

VERSION: REVISION – 1.3

6th November, 2017

REVISION CHART

.

Version	Primary Author(s)	Description of Version	Date Completed
Draft	S.M.Humraan Aleem Zulfiqar Azeem Khalid	Initial draft created for distribution and review comments	22/10/2017
Revision – 1.0	S.M.Humraan Aleem Zulfiqar Azeem Khalid	First Release	25/10/2017
Revision – 1.1	S.M.Humraan Aleem Zulfiqar Azeem Khalid	Second Release	29/10/2017
Revision – 1.2	S.M.Humraan Aleem Zulfiqar Azeem Khalid	Proof Reading	02/10/2017
Revision – 1.3	S.M.Humraan Aleem Zulfiqar Azeem Khalid	Final Release	05/10/2017

CONTENTS

1. INTRODUCTION

1.1 Purpose

HotRest is a front-desk hotel management software designed typically for billing, online food ordering and database management of guests checking in and out of the hotel and restaurants. It also includes options to display the features of the hotel and the restaurants and services offered by it.

1.2 Scope

This Design Specification is to be used by Software Engineering and Software Quality Engineering as a definition of the design to be used to implement the waiter-less Hotel and Restaurant Management.

1.3 Objective

Our aim to convert the cost of workforce of labors into the profits of the company by providing wireless technology to the customer so they can avail the functionalities of the hotel and restaurant online and getting rid of the hectic queues during the reservation and food ordering process.

2. SYSTEM OVERVIEW

.

2.1 Product Perspective

This product simulates a waiter-less website for the customer to provide them a better platform to complete their needs. It records the number of customers that are currently present in the hotel/restaurant. It also records the total preparation time of the food that will be required by both customer and the management. Depending on the values that are used by the simulator operator, it can be determined what is the optimal way to utilize a waiter-less management system to decrease the number waiting time for the customers.
.

2.1.1 Design Method

The design of this product utilizes an object-oriented approach.

2.1.1 User Interfaces

The user of this software product will be interfacing with the waiter-less system to help predict satisfy the customer according to their needs. The product allows the user to get familiar with the software without actually having the responsibility of standing in long queues.

2.1.2 Hardware Interfaces

Since the system is website, therefore it ca be executed on any PC/Smart Phones etc.

2.1.3 Software Interfaces

This system will execute on any platform running a proper browser.

.

2.1.4 Memory Constraints

Since it is an online website so no issues of the memory will be raised.

2.1.5 Operations

The operator will be required to fill the required field in order to further proceed.

2.1.6 Site Adaptation Requirements

This software is intended to execute on any platform with no modifications needed to

support different sites.

2.2 Product Functions

The system is decomposed into main three modules/components.

a. Hotel

b. Restaurant

c. User

Following part of the section briefly describe the above three listed modules.

a. <u>Hotel</u>

Room Reservation

- The system shall record reservations which would include customer's details, room type and dates of arrival and departure.

Check-in customers

- The system shall allow reservations to be modified without having to reenter all the customer information.

Checkout customers

- The system shall display the amount owed by the customer.
- To retrieve customer information the primary keys shall be used
- The system shall record that the room is empty.

Login (Admin Authentication)

- Admin will provide login ID and Password for signing in to account.

Payment records

- The system shall record the payment.
- The system shall record the payment type.

Check Notifications

- Admin will be able to check notifications after successfully login to the account.

b. **Restaurant**

Food Order

- The system will show all items available.
- The system shall track all meals purchased in the hotel.

Food Delivery

- The system should take proper information about the food delivery location.
- The system will provide estimated time of delivery.

Billing

- The system shall record payment done by customer.
- The system shall record mode of payment chosen by customer.

c. **User**

Room Reservation

- User will provide his details for the reservation of room which would include his personal information, room type, expected check-in and checkout dates

Table Reservation

- User can reserve table in a restaurant by providing required information to the system.

Order Food

- User can order his food online and can ask for the food delivery.

Pick N Drop Option

- Abroad Customers can avail pick n drop facilities from airport to the hotel.

View Bills

- Customer billing records are mailed to the customer on their email address.

Pay Bills

- Customer would pay bills at the time of checkout.

2.3 User Characteristics

The general characteristics of the intended users, include

- *educational level –bachelor of science*
- *experience- air traffic control knowledge*

2.4 Constraints

This website can only run on any system that has a web browser.

2.5 Assumptions and Dependencies

This system is a website then customers who use smartphone or PC (windows and ios) will be more benefited. Basic password authentication based security mechanism will be used to protect HotRest from unauthorized users. Database will be maintained with all the updates from the administrator to provide best services to the customers.

2.6 Apportioning of Requirements

Verification of customers through their email address will be done later.

3. DESIGN CONSIDERATIONS

3.1 Operating Environment

The HotRest waiter-less Hotel and Restaurant Management System can be operated in any operating environment with a working internet connection.

3.2 Fault Tolerant Design

Application errors will be handled by the system through different exception handling.

3.3 Design Conventions

The HotRest Management software design uses the Object Oriented methodology described in "Object Oriented Analysis and Design with applications" by Grady Booch.

3.4 Architectural Design

The software capabilities and requirements specified in the HotRest (waiter-less hotel and restaurant management system) Software Requirements Specification are transformed into programs that will execute on any platform with working internet connection. Software items are partitioned into classes and packages using Object Oriented methodology to maximize encapsulation and minimize interfaces. Packages are then built (compiled and linked) into executable programs.

3.5 User Interface:

The user or operator interfaces via a web page. The user is prompted for several values in order to perform the desired tasks.

3.5.1 Expected Input:

The user is prompted to enter values for the following variables:

Name, Email, and Address in string.
Phone Number in Integer.
And other required fields for authentication.

3.5.2 Output:

The user receives a report from the system indicating his order to him. That order can be room reserved or can be the food that the customer has ordered.

4. SYSTEM ARCHITECTURE

4.1 View of Product Classes (Figure 1)

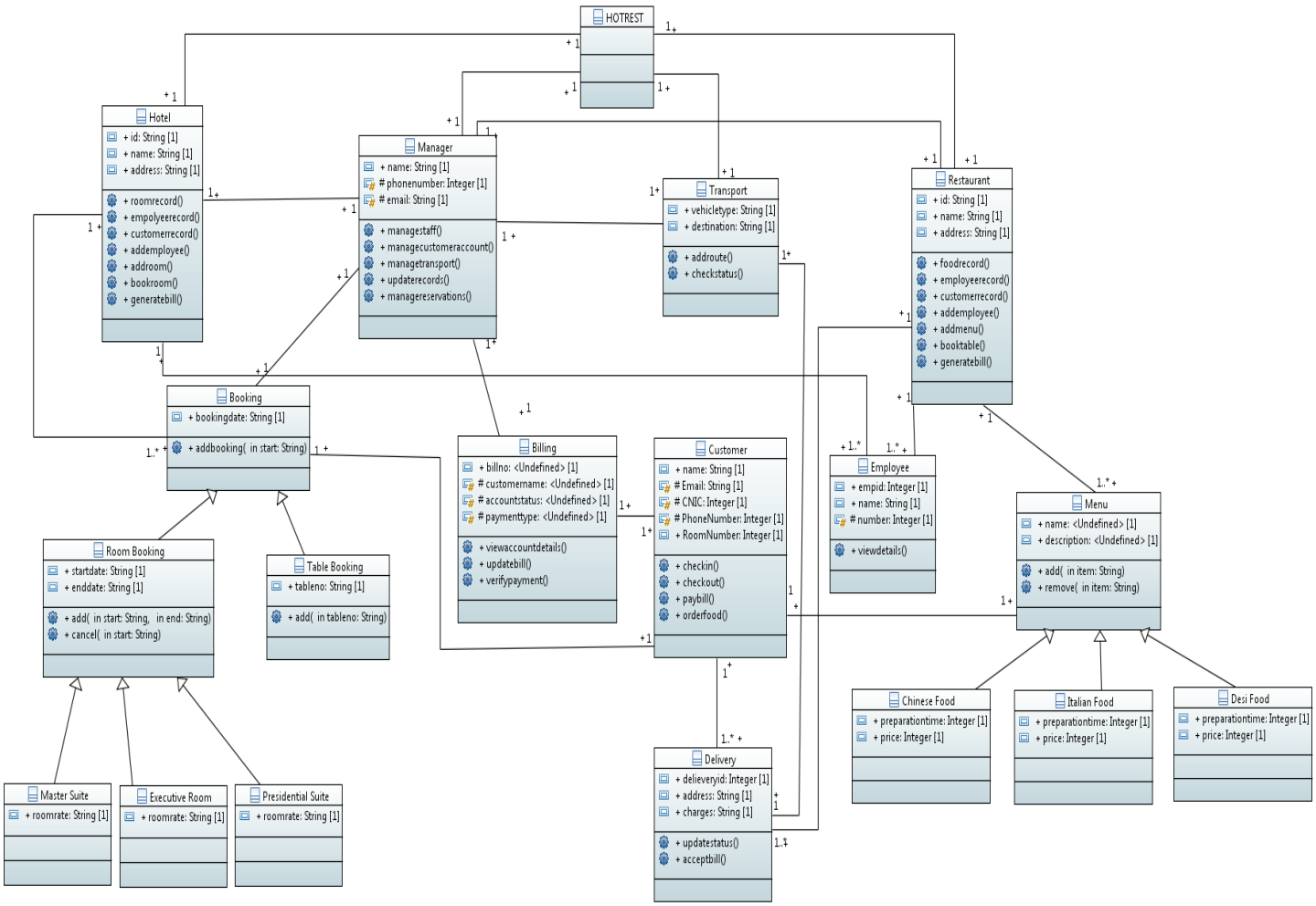

Figure 1- View of Participating Class Diagram

This figure shows the overall design of the system with outlining the individual classes and their relation to each other.

4.2 Individual Classes of System

4.2.1 HotRest Class

The HotRest class is the main acting class in which all the functions and attributes of the other classes will be called and executed.

4.2.2 Hotel Class

The Hotel class calls all the functions and attributed of hotel. Which includes the hotel name and address and functions like number of employees working and others.

4.2.2.1 Attributes of Hotel Class

- **Name** = Has the name of the hotel.
- **Address** = Has the address of the hotel.

4.2.2.2 Functions of Hotel Class

- **RoomRecord()** = Used to return record of room allocated and vacant.
- **EmployeeRecord()** = Used to return the record of all the employees working.
- **CustomerRecord()** = Used to return the record of all the customers entertained.
- **AddEmployee()** = Used to add the record of a new employee.
- **AddRoom()** = Used to add a new room in hotel.
- **BookRoom()** = Used to book a new room for customer.
- **GenerateBill()** = Used to return the total bill of availed facilities.

4.2.3 Manager Class

This class is used to handle all the activities done by the done or done through management.

4.2.3.1 Attributes of Manager Class

- **Name =** Has the name of the manager.
- **Email =** Has the email address of the manager.
- **PhoneNumber =** Has the phone number of the manager.

4.2.3.2 Functions of Manager Class

- **ManageStaff() =** Used to manage all the staff members.
- **ManageCustomerAccount() =** Used to manage the customer.
- **ManageTransport() =** Used to manage the transport provided by the hotel.
- **UpdateRecords() =** Used to update the record of customers.
- **ManageReservations() =** Used to manage the reservations made by customers.

4.2.4 Transport Class

This class is used to manage all the working of transport which provided to the customer by the hotel.

4.2.4.1 Attributes of Transport Class

- **VehicleType =** Used to enter the vehicle used.
- **Destination =** Used to enter the destination of customer.

4.2.4.2 Functions of Transport Class

- **AddRoute() =** Used to add route of the customers.
- **CheckStatus() =** Used to check status of the customers.

4.2.5 Restaurant Class

This class is used to manage all the activities of restaurant class.

4.2.5.1 Attribute of Restaurant Class

- **Id =** Used to store the ID of a restaurant.
- **Name =** Used to store the name of the restaurant.
- **Address =** Used to store the address of the restaurant.

4.2.5.2 Functions of Transport Class

- **FoodRecord() =** Used to get the food record.
- **EmployeeRecord() =** Used to get the record of employees.
- **CustomerRecord() =** Used to get the record of the customers.
- **AddEmployee() =** Used to add the employee in the restaurant.
- **AddFood() =** Used to add a new food item in the restaurant.
- **BookTable() =** Used to reserve table for customers in the restaurant.
- **GenerateBill() =** Used to generate bill for the customer availed facilities.

4.2.6 Booking Class

This class is used to handle all the activities of regarding booking, wheatear it is the booking of table in a restaurant or booking of a room in a hotel.

4.2.6.1 Attributes of Booking Class

- **BookingDate =** Used to store the date of booking.

4.2.6.2 Functions of Booking Class

- **AddBooking() =** Used to add the booking in the system made by the customer.

4.2.7 TableBooking Class

This class is used to handle all the booking of a table in a hotel or in a restaurant.

4.2.7.1 Attributes of TableBooking Class

- **TableNo =** Used to store the table number of booked table.

4.2.7.2 Functions of TableBooking Class

- **Add() =** Used to add a new table booked.
- **Cancel() =** Used to cancel a table booked.

4.2.8 RoomBooking Class

This class is used to handle all the booking of a room in a hotel.

4.2.8.1 Attributes of RoomBooking Class

- **RoomNo =** Used to store the room number.
- **StartDate =** Used to store the starting date of booking of room.
- **EndDate =** Used to store the end date of booking of room.

4.2.8.2 Functions of RoomBooking Class

- **Add() =** Used to add a new room booked.
- **Cancel() =** Used to cancel a booked room.

4.2.9 MasterSuite Class

This class is used to perform all the activities of a MasterSuite room type.

4.2.9.1 Attributes of MasterSuite Class

- **RoomRate =** Used to store the rate of MasterSuite room type.

4.2.10 ExecutiveRoom Class

This class is used to perform all the activities of a ExecutiveRoom room type.

4.2.10.1 Attributes of ExecutiveRoom Class

- **RoomRate =** Used to store the rate of ExecutiveRoom room type.

4.2.11 PresidentialSuite Class

This class is used to perform all the activities of a PresidentialSuite room type.

4.2.11.1 Attributes of PresidentialSuite Class

- **RoomRate =** Used to store the rate of PresidentialSuite room type.

4.2.12 Billing Class

This class is used to control all the billing activities of billing in hotel or in a restaurant.

4.2.12.1 Attributes of Billing Class

- **BillNo =** Used to store the bill number of customer.
- **CustomerName =** Used to store the customer name/id.
- **AccountStatus =** Used to determine the account status of a customer.
- **PaymentType =** Used to store the payment option of a customer.

4.2.12.2 Functions of Billing Class

- **ViewAccountDetails() =** Used to display the account details of customer's account.
- **UpdateBill() =** Used to update the bill of a customer.
- **VerifyPayment() =** Used to verify the payment of a customer.

4.2.13 Customer Class

This class is used to handle all the activities of a customer.

4.2.13.1 Attributes of Customer Class

- **Name =** Used to store the name of a customer.
- **CNIC =** Used to store the CNIC number of a customer.
- **PhoneNumber =** Used to store the phone number of a customer/
- **Email =** Used to store the email address of a customer.

4.2.13.2 Functions of Customer Class

- **CheckIn() =** Used to get the check-in date/time of customer in a hotel/restaurant.
- **CheckOut() =** Used to get the check-out date/time of customer in a hotel/restaurant.
- **PayBill() =** Used to get the bill payment by customer.
- **OrderFood() =** Used to get the food ordered by the customer.

4.2.14 Employee Class

This class is used to handle all the activities of employee in a hotel or in a restaurant.

4.2.14.1 Attribute of Employee Class

- **EmpID =** Used to store the employee id.
- **EmpName =** Used to store the employee name.
- **EmpNumber =** Used to store the employee phone number.

4.2.14.2 Functions of Employee Class

- **ViewDetails() =** Used to display all the details of a employee.

4.2.15 Menu Class

This class is used to handle all the activities regarding menu.

4.2.15.1 Attributes of Menu Class

- **Name =** Used to store the name of menu type.
- **Description =** Used to store the descriptions of a menu card.

4.2.15.2 Functions of Menu Class

- **Add()=**Used to add a new item in the menu card.
- **Remove() =** Used to remove an item from the menu card.

4.2.16 ChineseFood Class

This class is used to handle all the activities of a Chinese food.

4.2.16.1 Attributes of ChineseFood Class

- **PreparationTime =** Used to store the preparation time of a particular Chinese food.
- **Price =** Used to store the rpice of a particular Chinese food.

4.2.17 ItalianFood

This class is used to handle all the activities of Italian food.

4.2.17.1 Attributes of ItalianFood

- **PreparationTime =** Used to store the preparation time of a particular Italian food.
- **Price =** Used to store the price of a particular Italian food.

4.2.18 DesiFood

This class is used to handle all the activities of Desi food.

4.2.18.1 Attributes of DesiFood

- **PreparationTime =** Used to store the preparation time of a particular Desi food.
- **Price =** Used to store the rpice of a particular Desi food.

5. FIGURES

5.1 Use Cases (Figure 2)

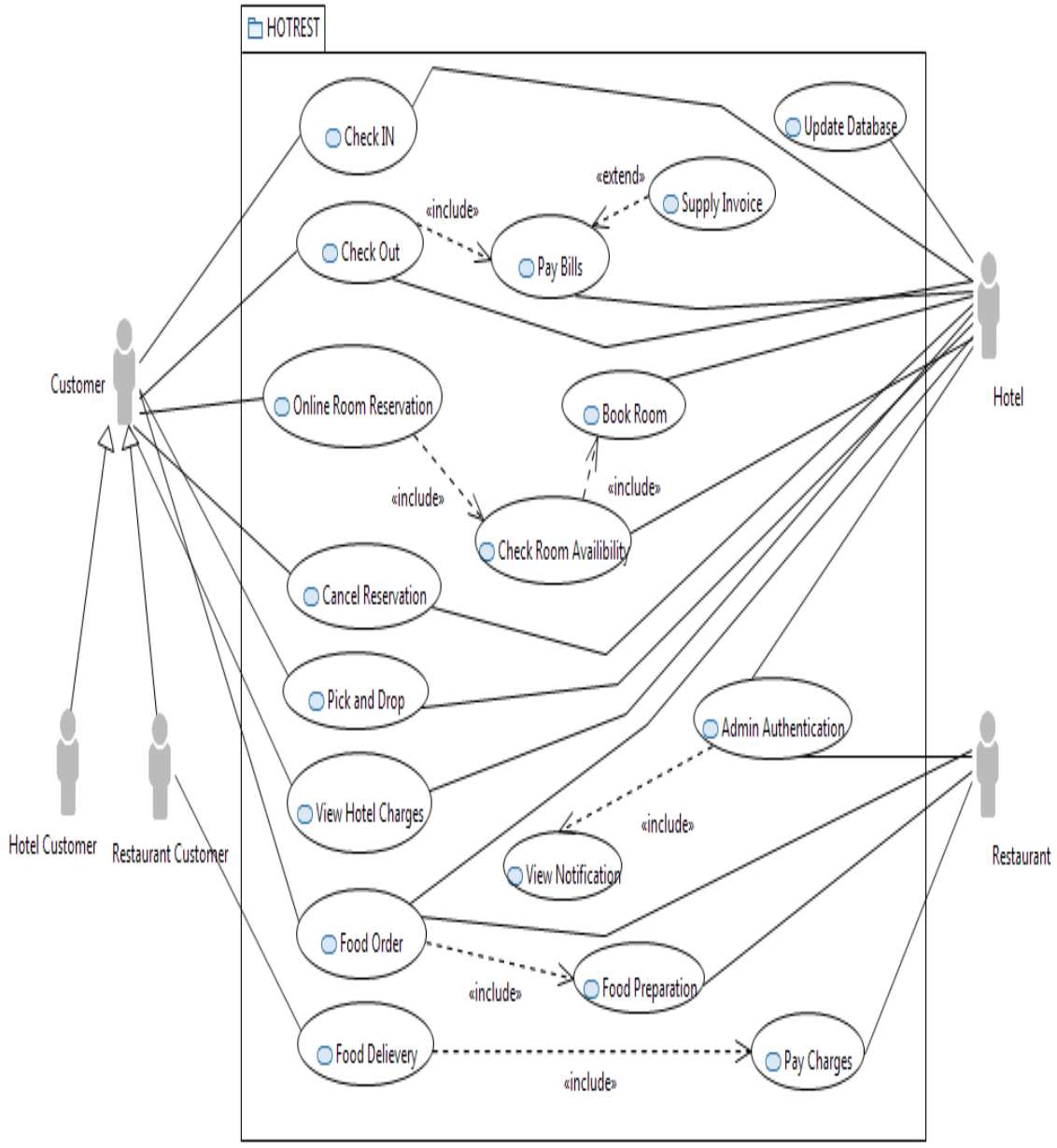

Figure 2- HotRest Use Case

5.2 Check-In

5.2.1 Check-In Sequence (Figure 3)

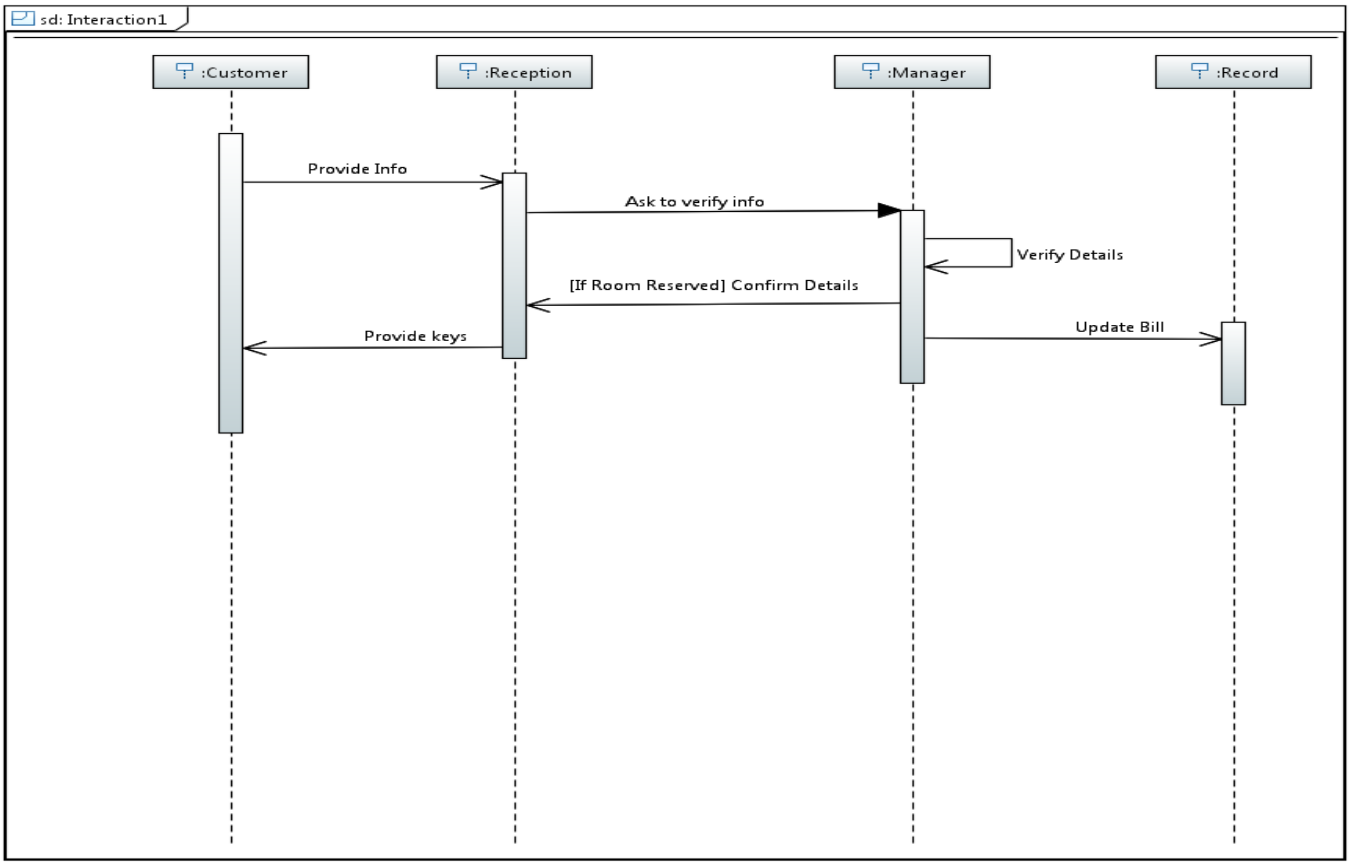

Figure 3- Check-In Sequence

5.2.2 Check-In Collaboration (Figure 4)

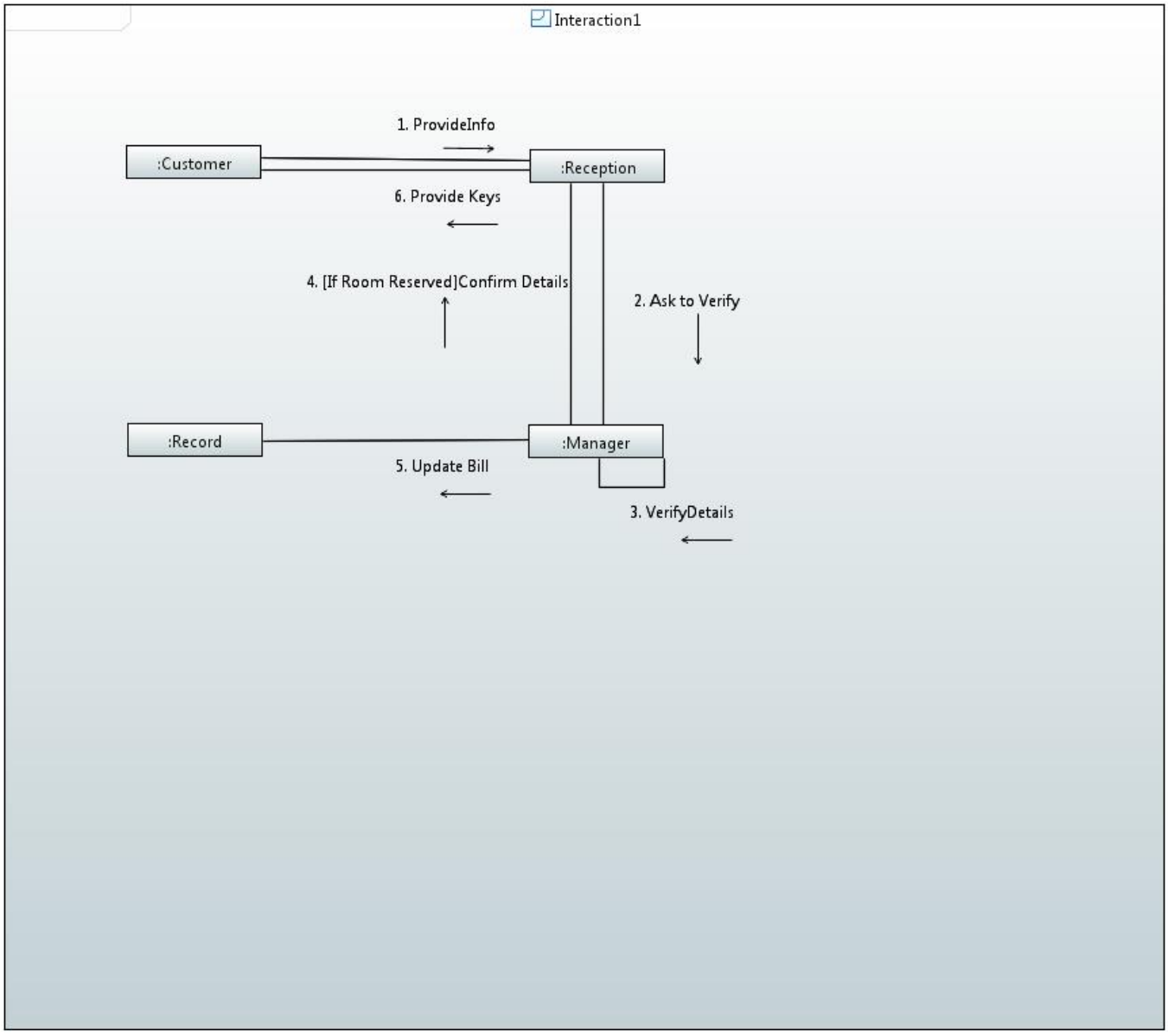

Figure 4- Check-In Collaboration

5.3 Reservation

5.3.1 Reservation Sequence (Figure 5)

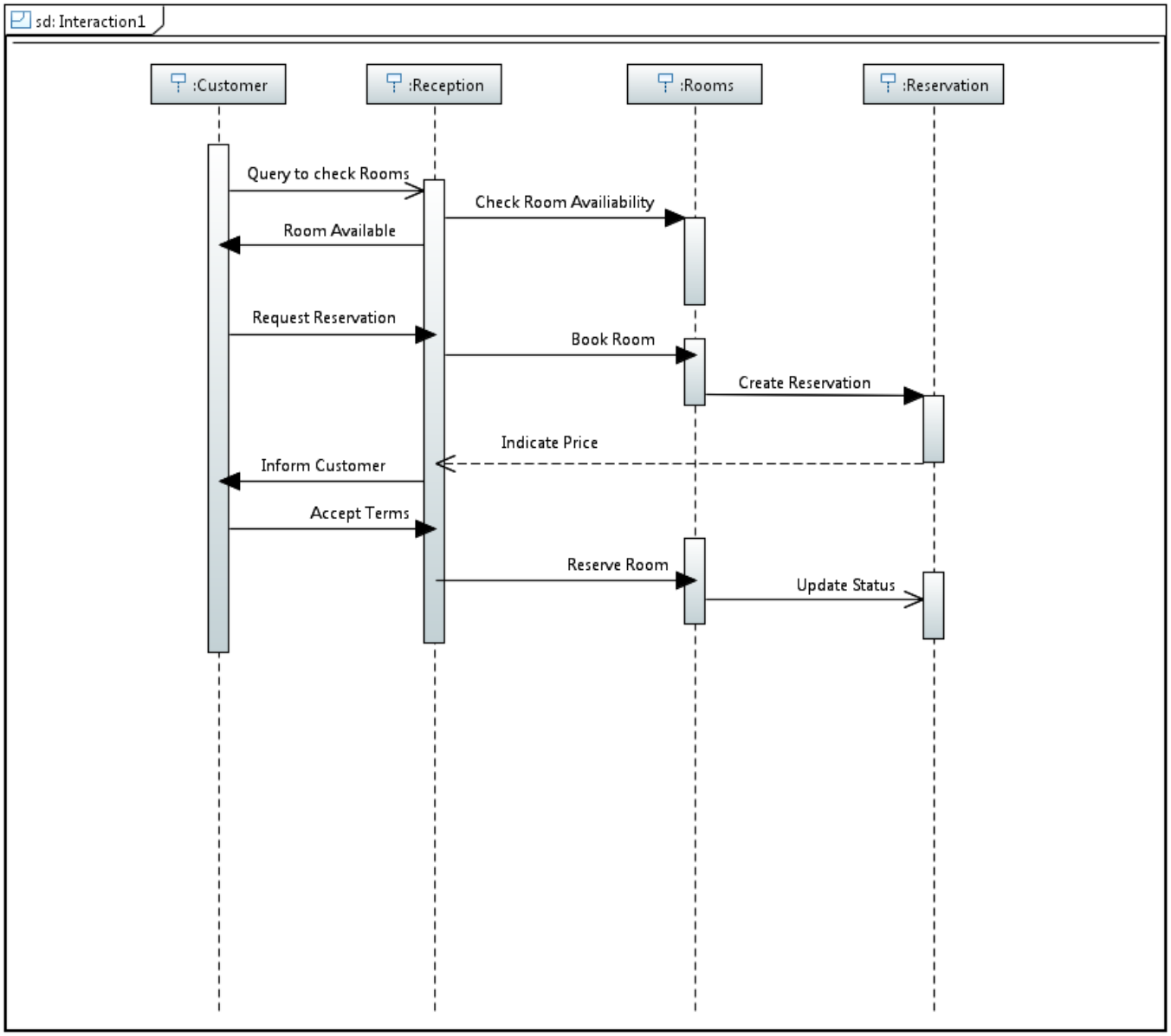

Figure 5- Reservation Sequence

5.3.2 Reservation Collaboration (Figure 6)

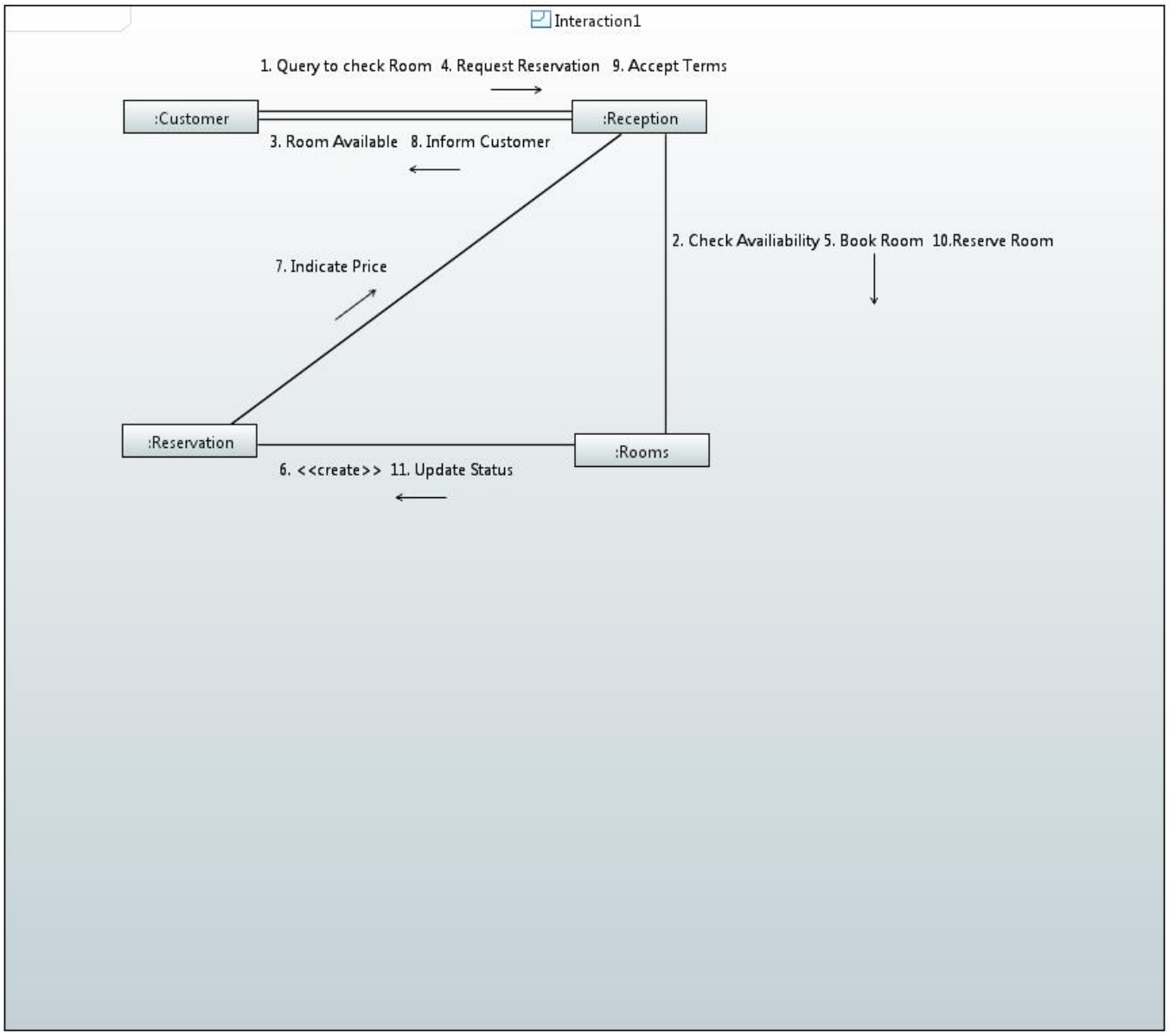

Figure 6 – Reservation Collaboration

5.4 Transportation

5.4.1 Transportation Sequence (Figure 7)

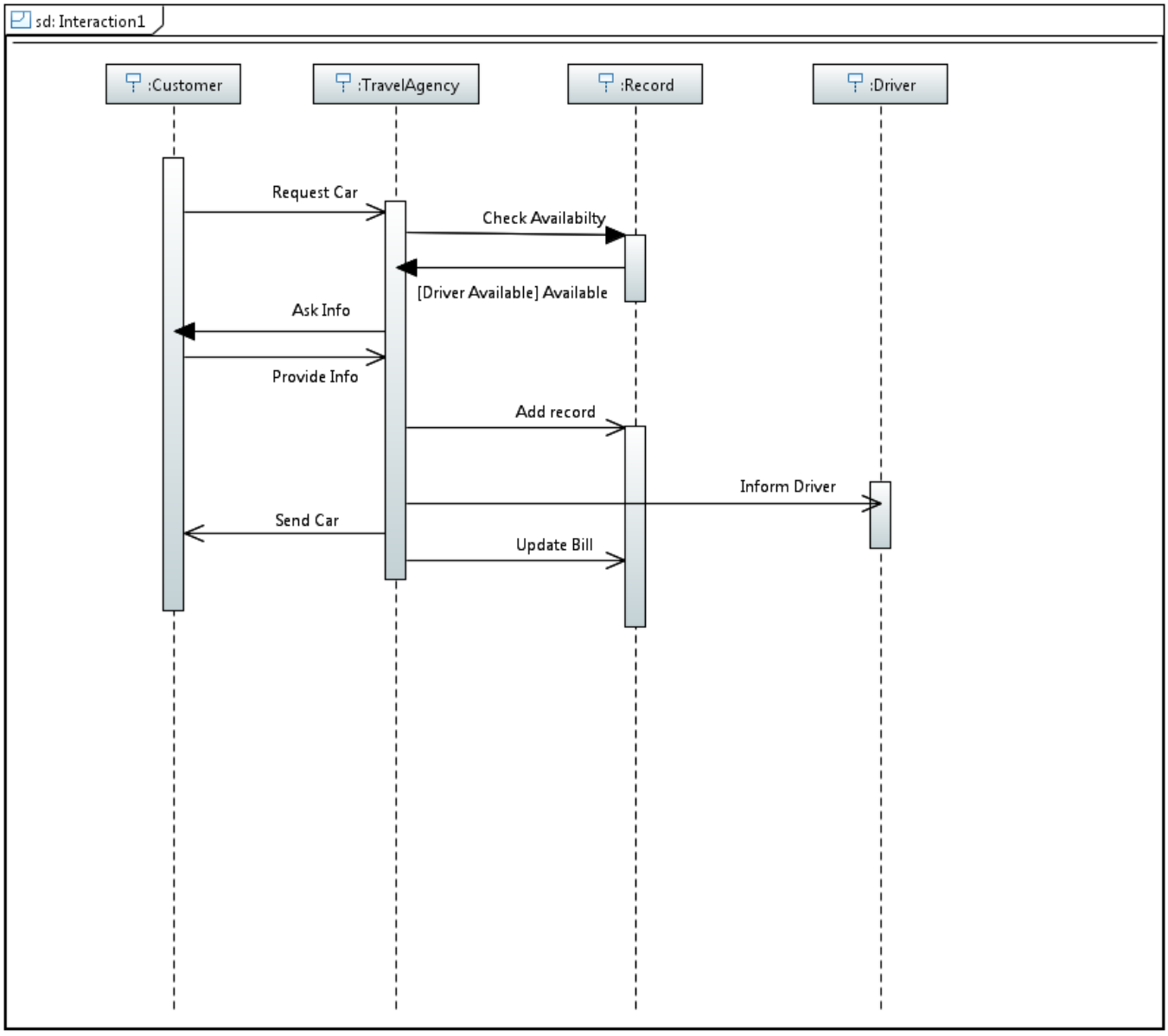

Figure 7 – Transportation Sequence

5.4.2 Transportation Collaboration (Figure 8)

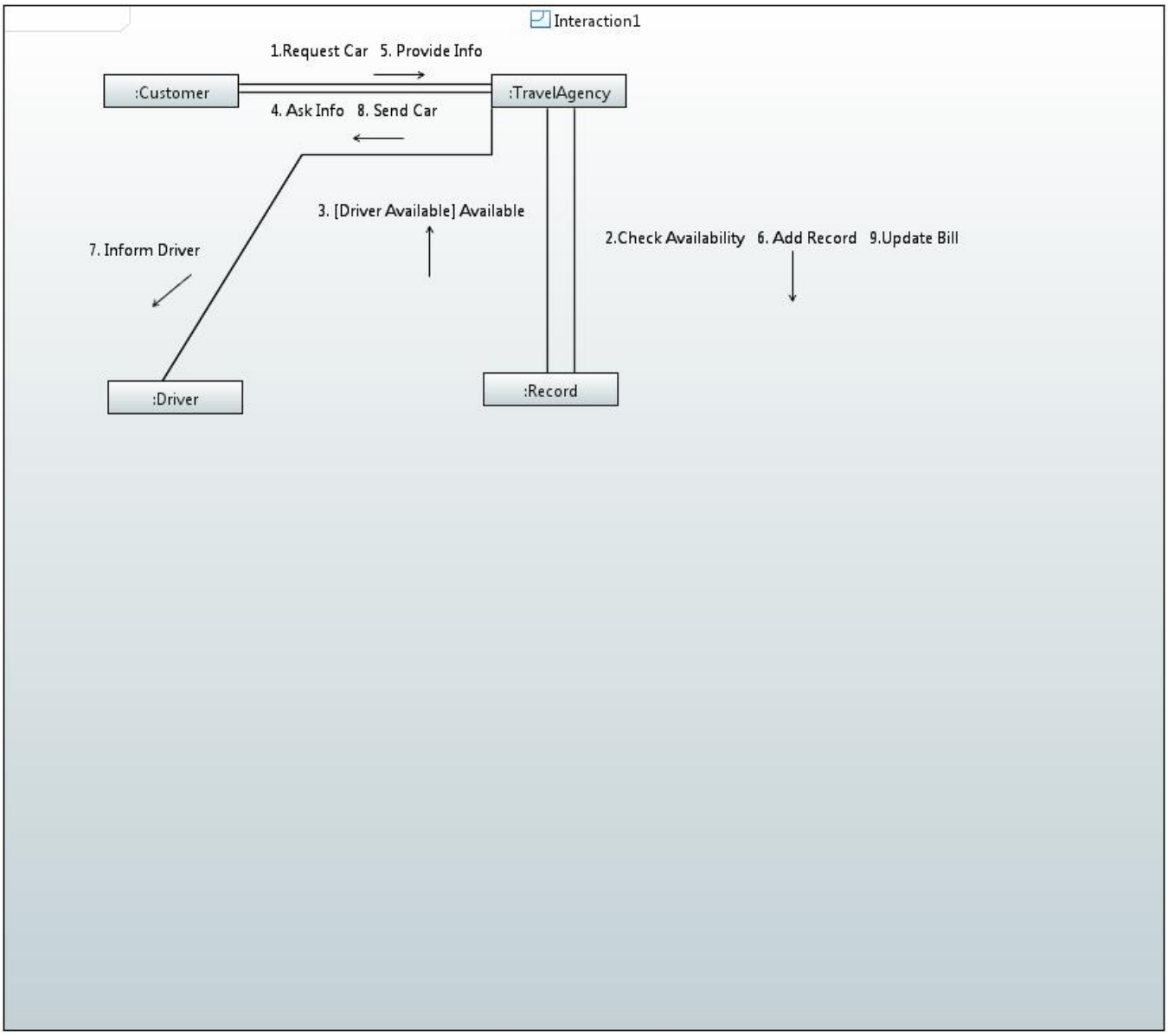

Figure 8- Transportation Collaboration

5.5 Admin Authentication

5.5.1 Admin Authentication Sequence (Figure 9)

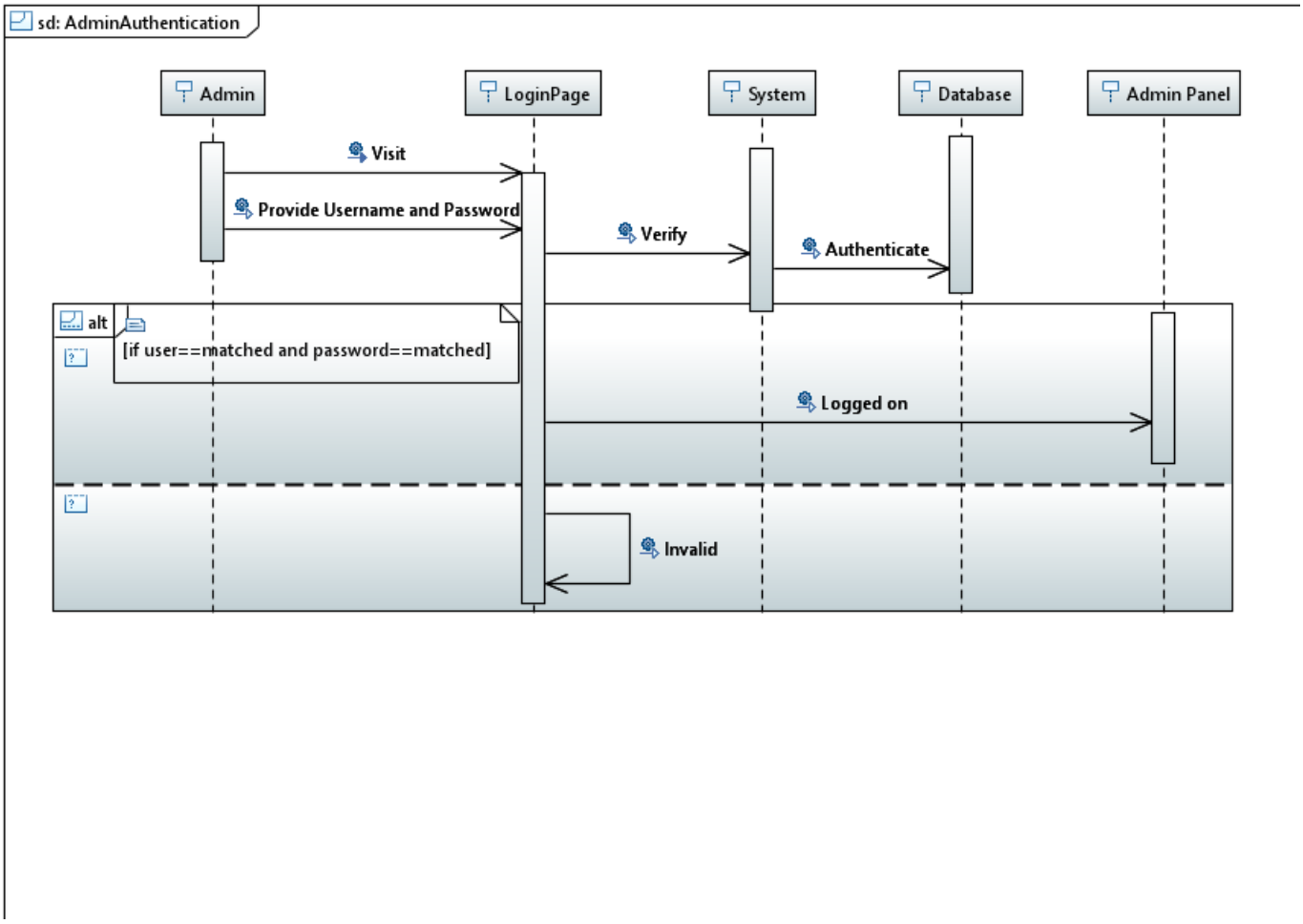

Figure 9 – Admin Authentication Sequence

5.5.2 Admin Authentication Collaboration (Figure 10)

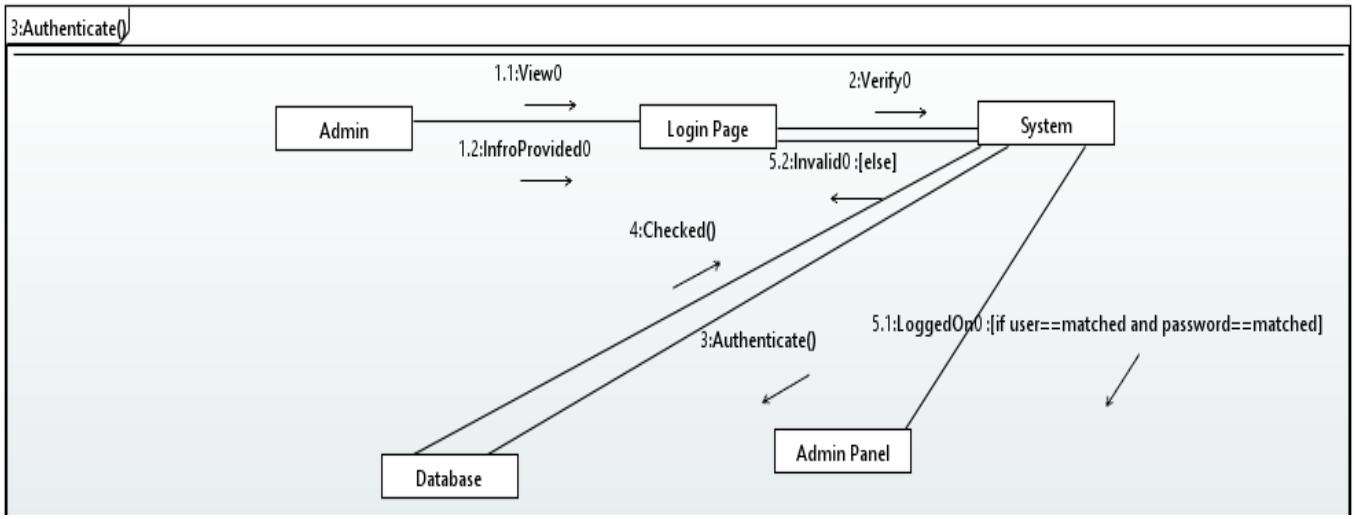

Figure 10 – Admin Authentication Collaboration

5.6 Payment

5.6.1 Payment Sequence (Figure 11)

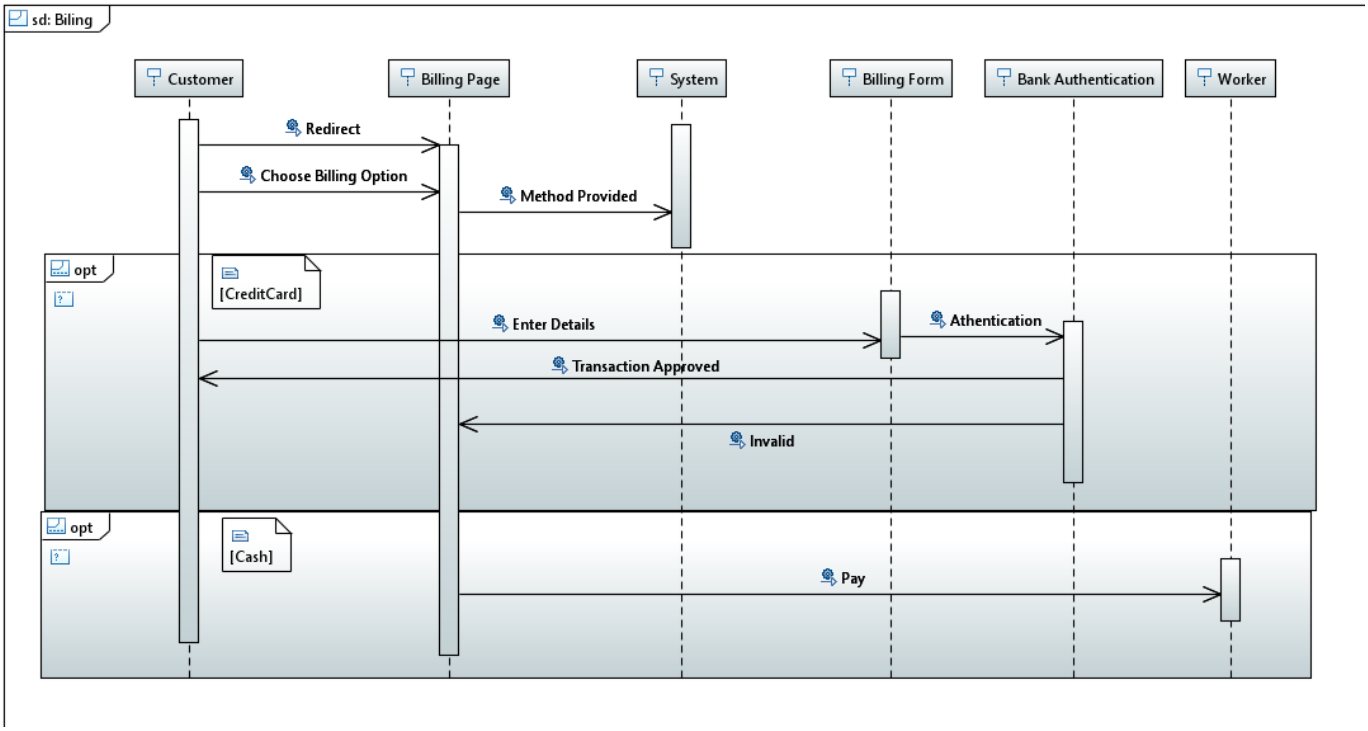

Figure 11 - Payment Sequence

5.6.2 Payment Sequence (Figure 12)

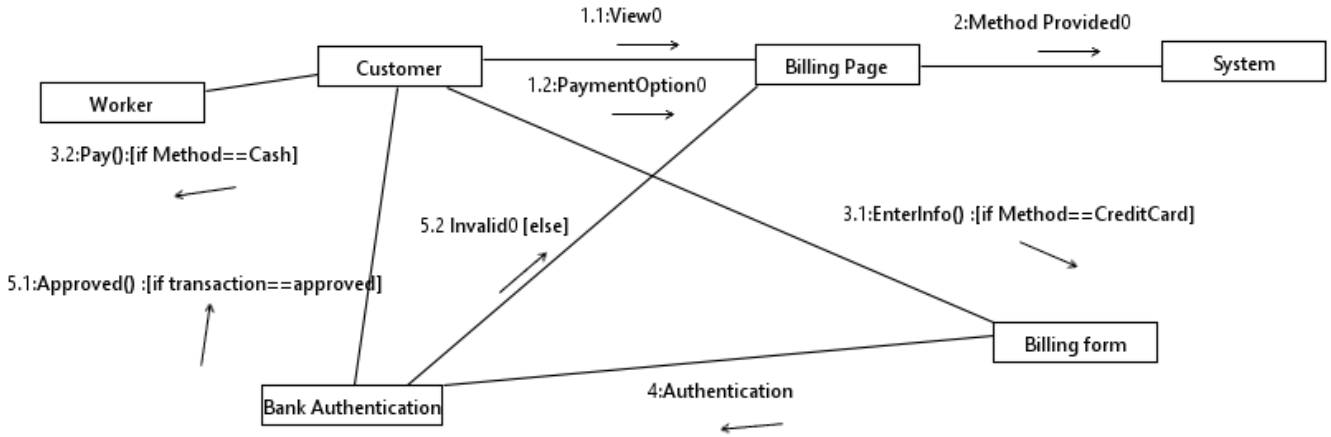

Figure 12 - Payment Collaboration

5.7 Update

5.7.1 Update Sequence (Figure 13)

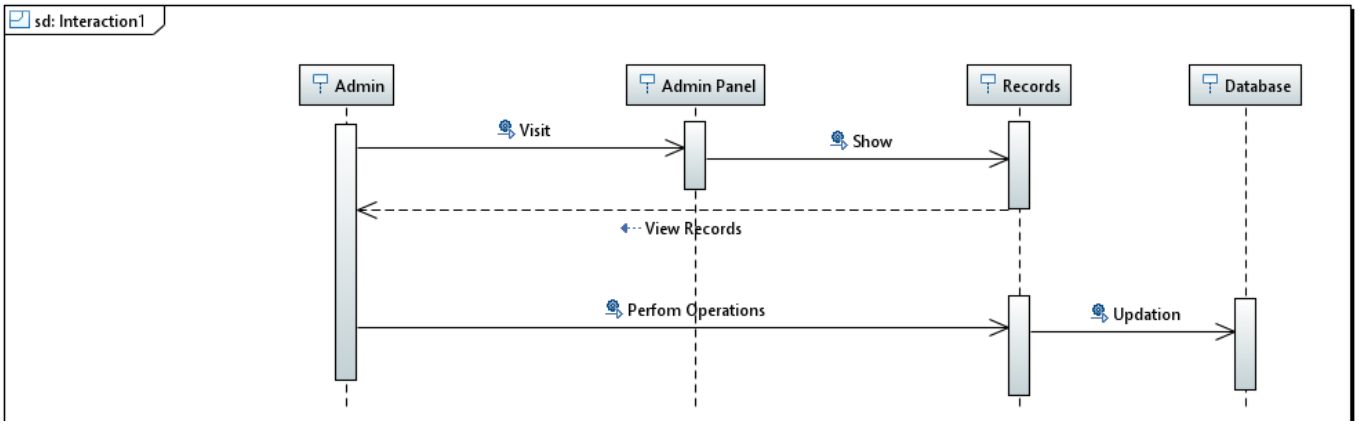

Figure 13 - Update Sequence

5.7.2 Update Collaboration (Figure 14)

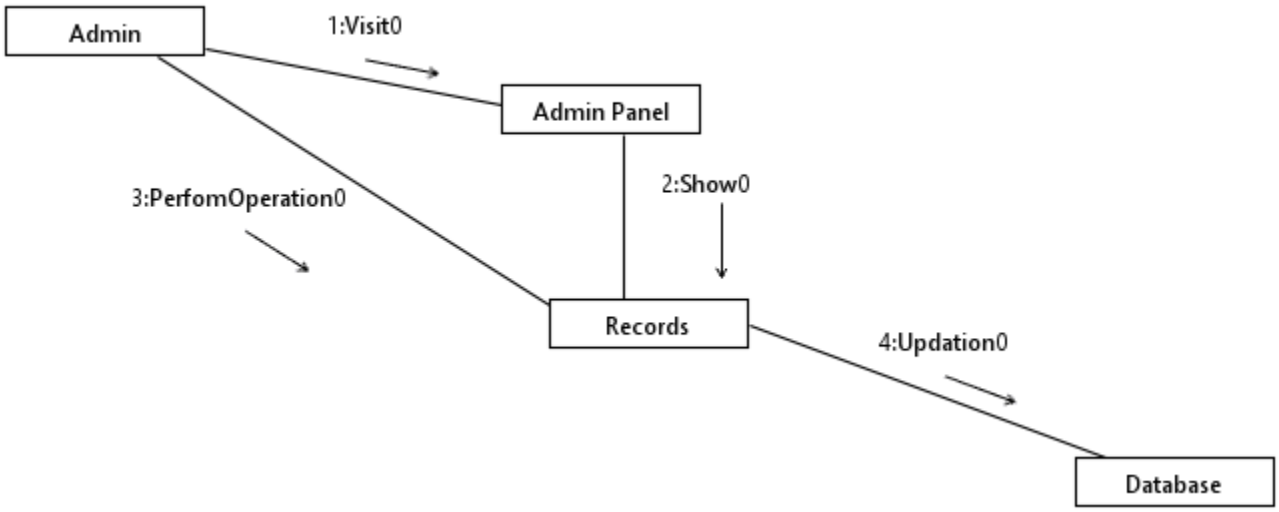

Figure 14 - Update Collaboration

5.8 Billing

5.8.1 Billing Sequence (Figure 15)

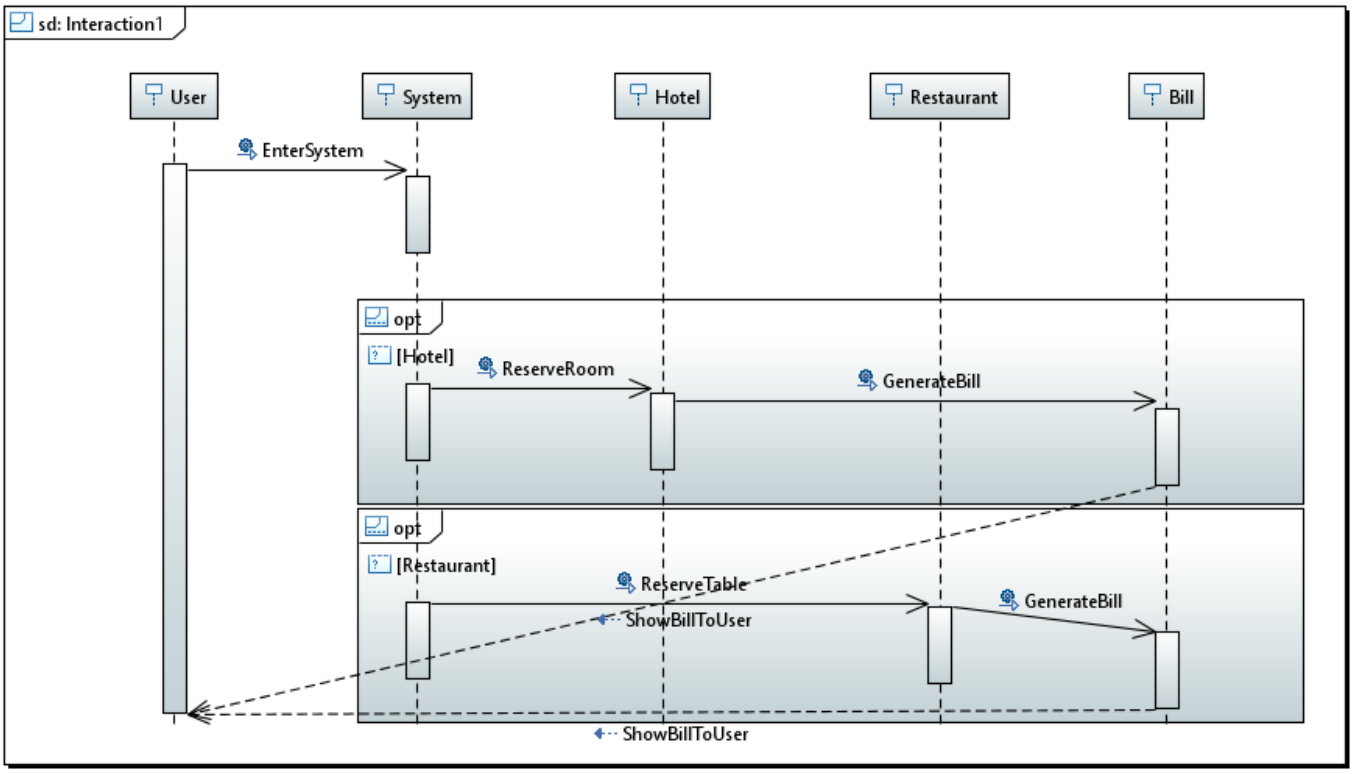

Figure 15 - Billing Sequence

5.8.2 Billing Collaboration (Figure 16)

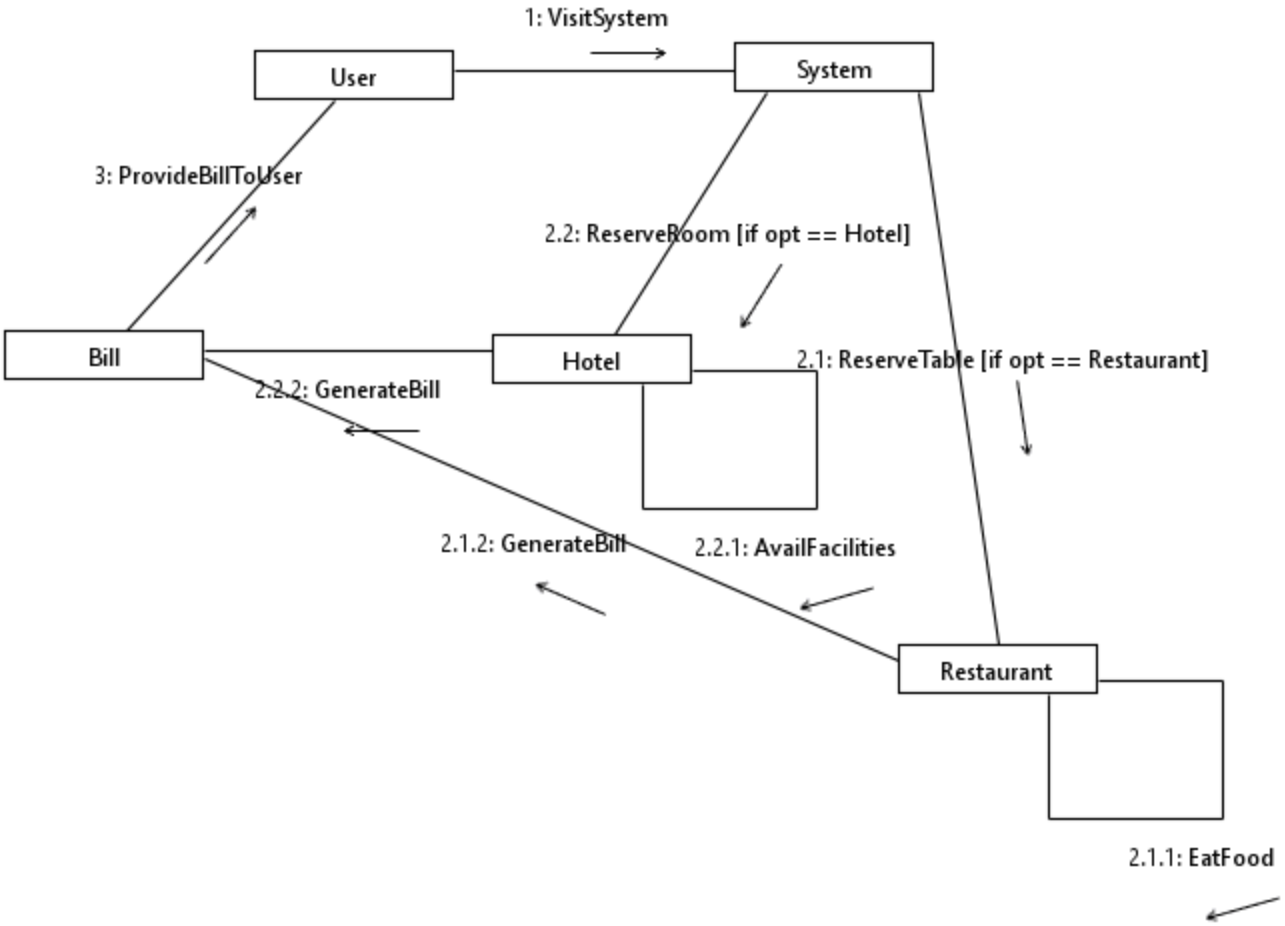

Figure 16 – Billing Collaboration

5.9 Food Order

5.9.1 Food Order Sequence (Figure 17)

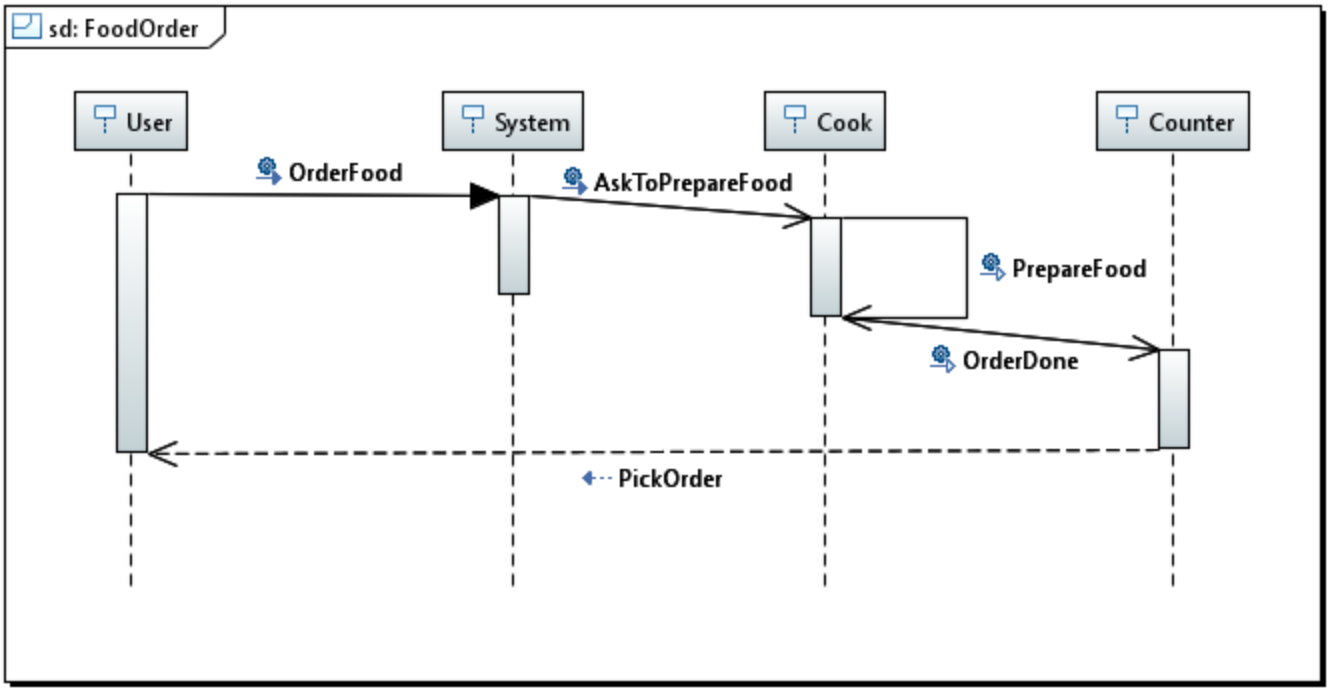

Figure 17 – Food Order Sequence

5.9.2 Food Order Sequence (Figure 18)

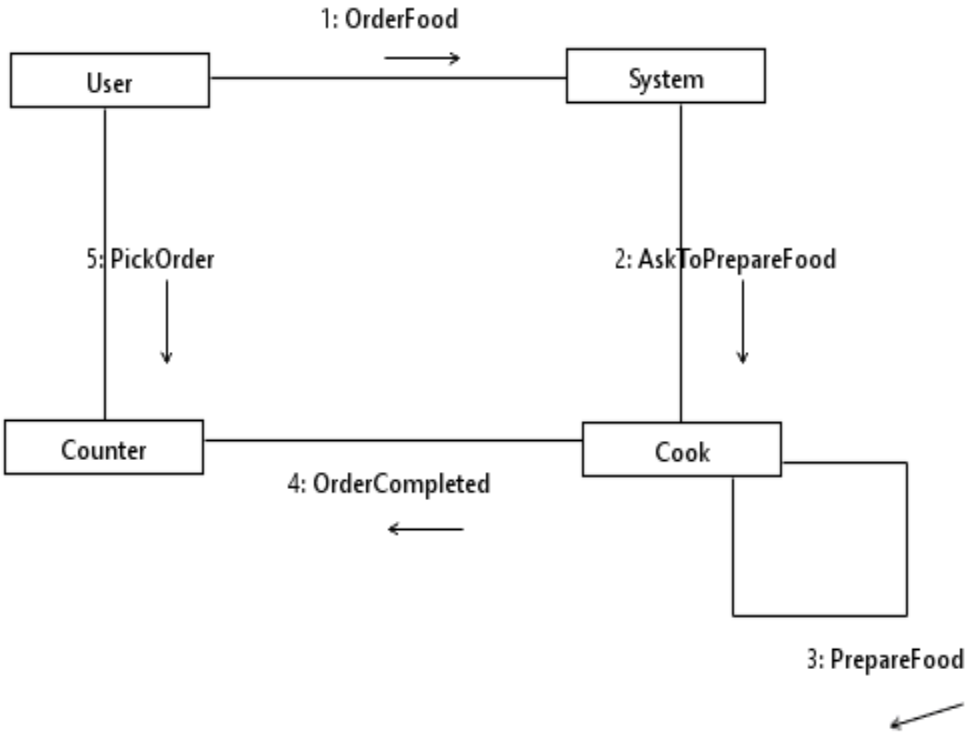

Figure 17 – Food Order Sequence

5.10 Food Delivery

5.10.1 Food Delivery Sequence (Figure 19)

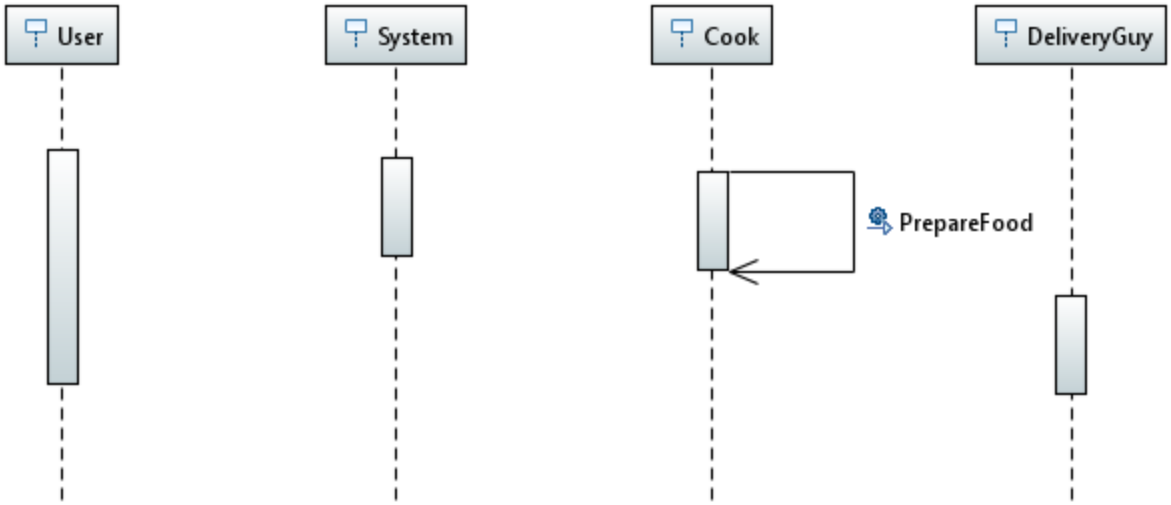

Figure 19 – Food Delivery Sequence

5.10.2 Food Delivery Collaboration (Figure 20)

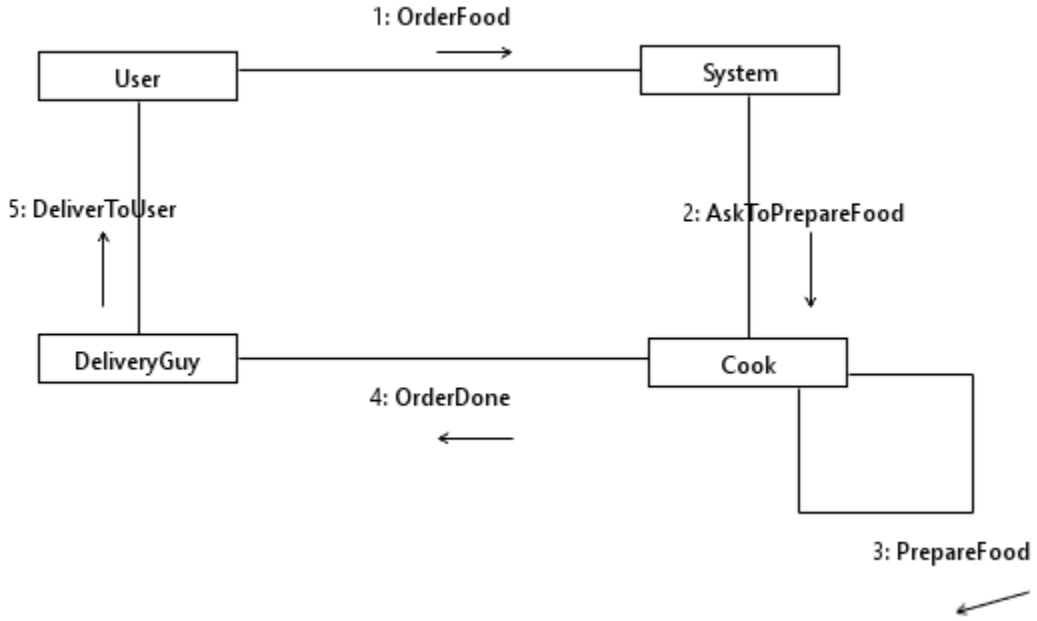

Figure 20 – Food Delivery Collaboration

6. REFERENCES

[1] Robin Nixon, “Learning PHP, MySQL & JavaScript: With JQuery, CSS & HTML5, O'Reilly Media, December 2014

[2] Luke Welling, "PHP and MySQL Web Development, 4th Edition", Laura Thomson, 2009

[3] Rasmus Lerdorf, "Programming PHP", 2002

[4] Brett McLaughlin, "PHP & MySQL: The Missing Manual", November 2011

[5] https://www.lynda.com/PHP-training-tutorials/282-0.html

[6] Object Oriented Analysis and Design with applications 6th Edition by Grady Booch.