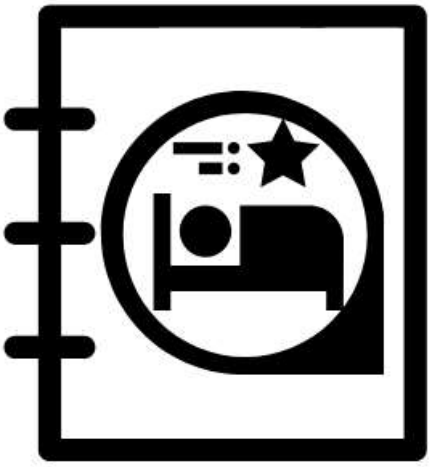
California State University Fullerton

CPSC 462

Object Oriented Software Design

Design Model

for the



Hotel Reservation System

Josh Ibad
Chief Software Architect
joshcibad@csu.fullerton.edu

Revision History:

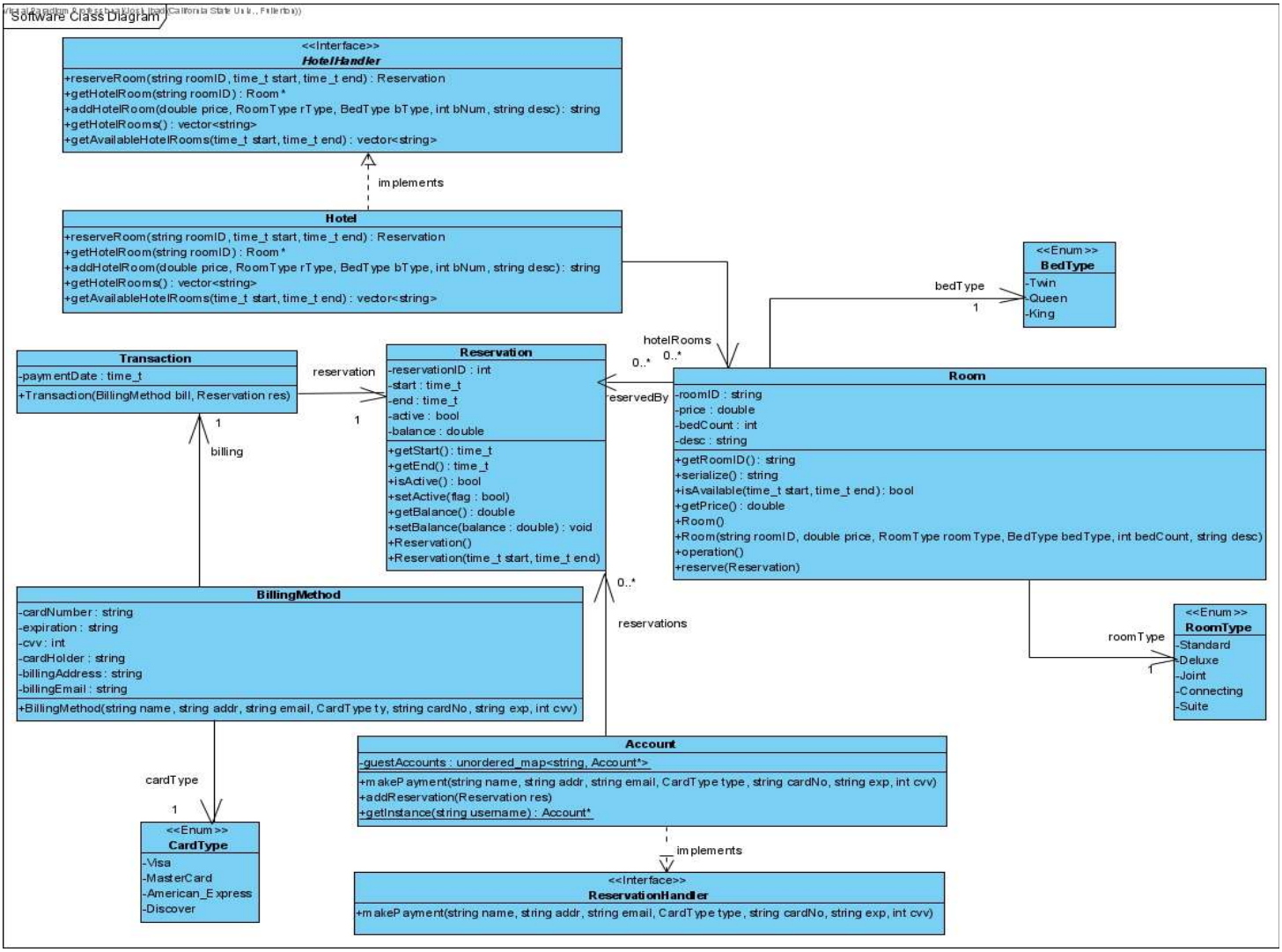
Version	Date	Summary of Changes	Author
1.0	2021-11-15	• Initial Release	Josh Ibad

Table of Contents

1 Static View

1.1 Hotel Reservation system - Room and Reservation Management

1.1.1 Software Class Diagram



1.1.2 Description

The Software Class diagram can be subdivided into two components. The first is the *Hotel* component which contains the interface HotelHandler, the classes Hotel and Room, and the enumerations RoomType and BedType. The HotelHandler serves as an interface for other components to interface with and to serve as a Facade or Controller for the Manage Hotel Rooms use case. System level messages sent to the interface are to be relayed to the implementing class Hotel. The Hotel stores a list of Rooms called hotelRooms, each of which have a BedType and RoomType. For the messages getHotelRoom, addHotelRoom, getHotelRooms, and getAvailableHotelRooms, these messages go to the HotelHandler interface, then to the Hotel, which then manages Rooms. For the reserveRoom message, the *Hotel* component stores Reservations from the *Reservation* component in a list in the corresponding instance of Room.

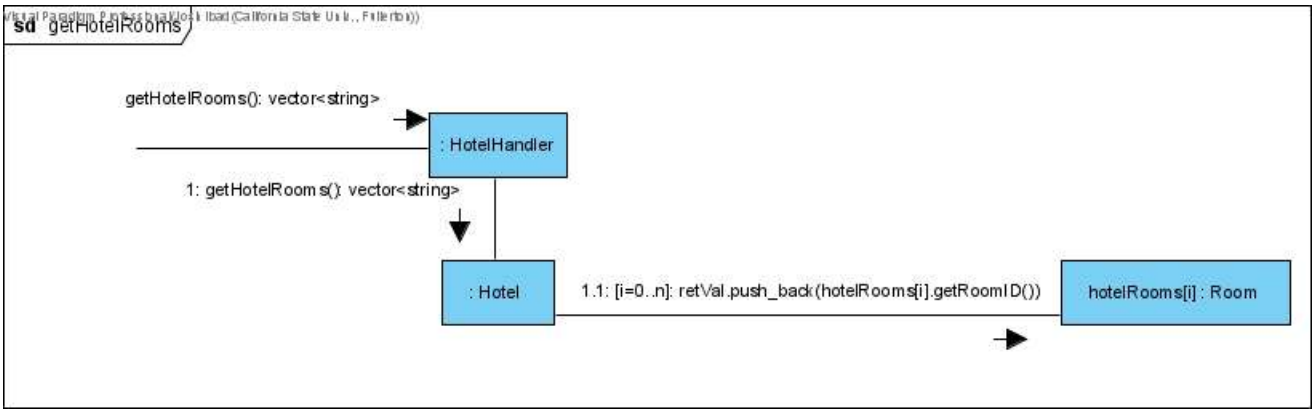
The second component is the *Reservation* component, containing the interface ReservationHandler, the classes Account, Reservation, Transaction, and BillingMethod, and the enumeration CardType. The ReservationHandler interface serves as an interface for System level messages such as makePayment. These messages are then

relayed to the implementing class Account, which follows a Multiton pattern. An Account stores a list of Reservations called reservations. A BillingMethod has a CardType called cardType, and a Transaction has a BillingMethod billing and an associated Reservation for which the transaction occurs for, called reservation. When the makePayment message is relayed from ReservationHandler to Account, the account looks through its list of Reservations to find the most recent unpaid reservation. A BillingMethod is then generated, then attached to a newly created Transaction for the unpaid Reservation.

2 Dynamic View

2.1 getHotelRooms Sequence of Execution

2.1.1 Software Interaction Diagram



2.1.2 Description

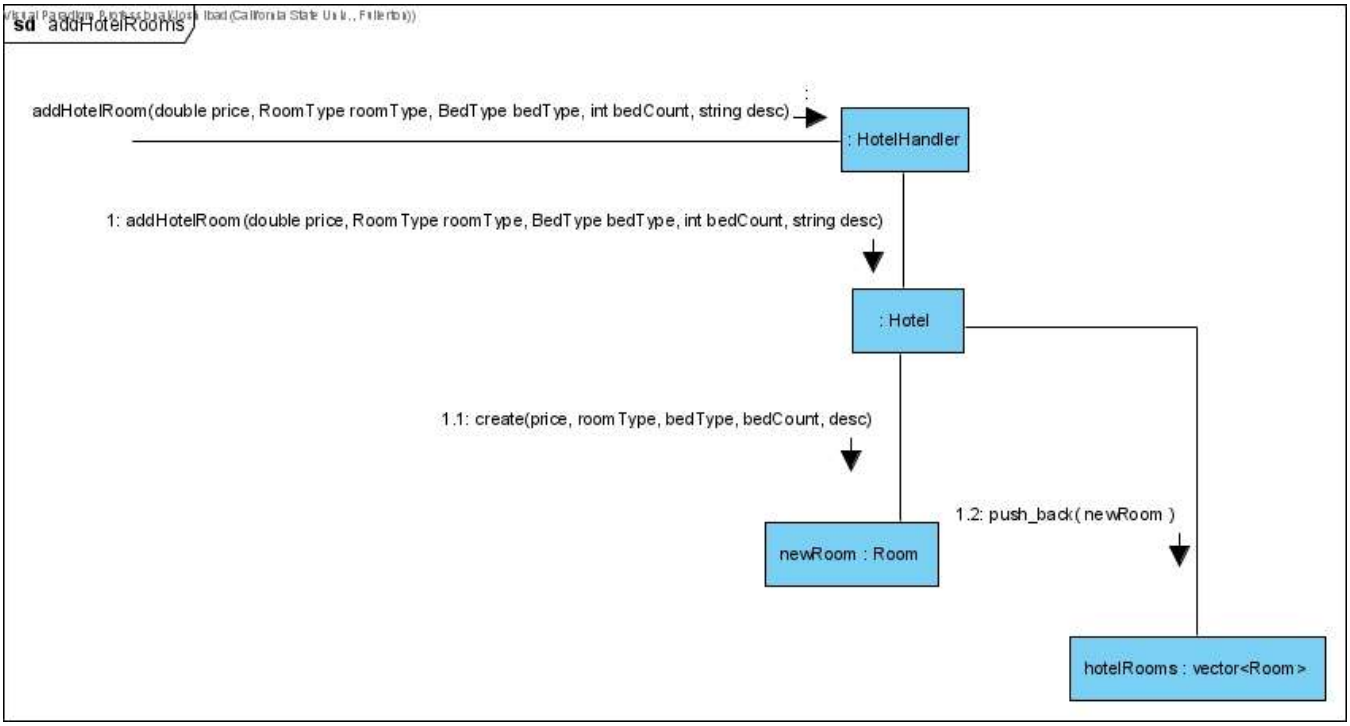
The system level message getHotelRooms is sent to the HotelHandler interface. The interface relays the message to an instance of the implementing Hotel class. The Hotel then iterates through its list of Rooms called hotelRooms, pushing the roomIDs of each room into a temporary vector called retVal. This temporary vector is then returned to the message caller.

2.1.3 SSD Traceability

In the SSD 1 called 'Manage Hotel Rooms - 1. Add Room', getHotelRooms is the second message. This SSD can be found in document 03.1 - Use Case Model Annex 1, in section 2.1.2. The interface HotelHandler is located in the Domain Layer's Interface Diagram, inside the Hotel component. This can be found in document 07 - SW Architecture Document, in section 3.1.2.1. Hotel.

2.2 addHotelRooms Sequence of Execution

2.2.1 Software Interaction Diagram



2.2.2 Description

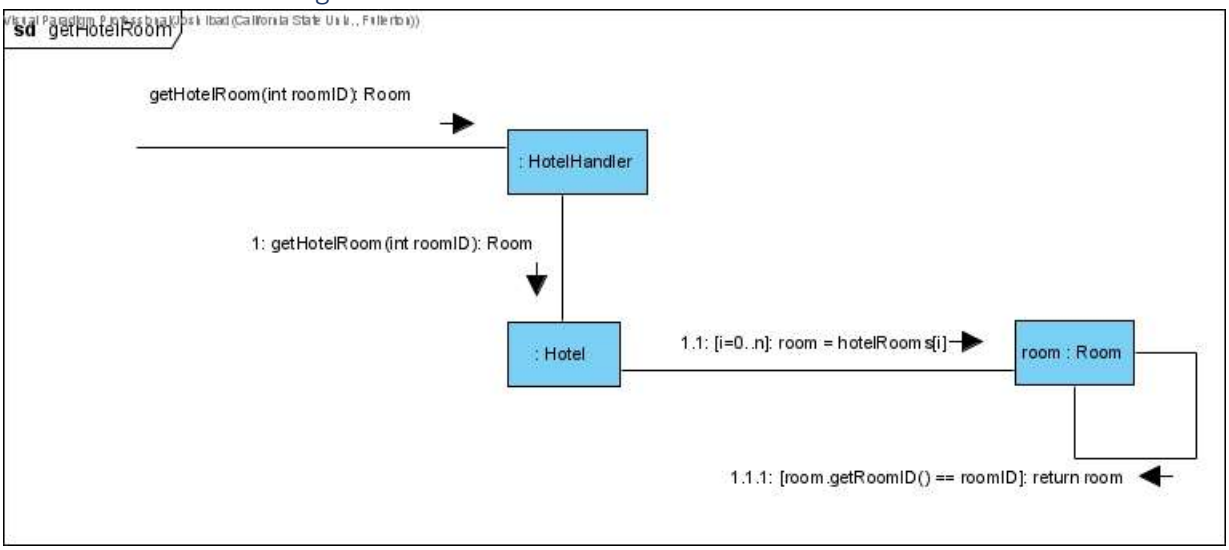
The system level message addHotelRoom is sent to the HotelHandler interface. The interface relays the message to an instance of the implementing Hotel class. The Hotel then creates a new Room called newRoom using the parameters passed in by the message caller. Specifically, a Room is created with the given price, roomType, bedType, bedCount, and description. The new room is then pushed back into the list of Rooms stored by the Hotel object, called hotelRooms.

2.2.3 SSD Traceability

In the SSD 1 called 'Manage Hotel Rooms - 1. Add Room', addHotelRooms is the third message. This SSD can be found in document 03.1 - Use Case Model Annex 1, in section 2.1.2. The interface HotelHandler is located in the Domain Layer's Interface Diagram, inside the Hotel component. This can be found in document 07 - SW Architecture Document, in section 3.1.2.1. Hotel.

2.3 getHotelRoom Sequence of Execution

2.3.1 Software Interaction Diagram



2.3.2 Description

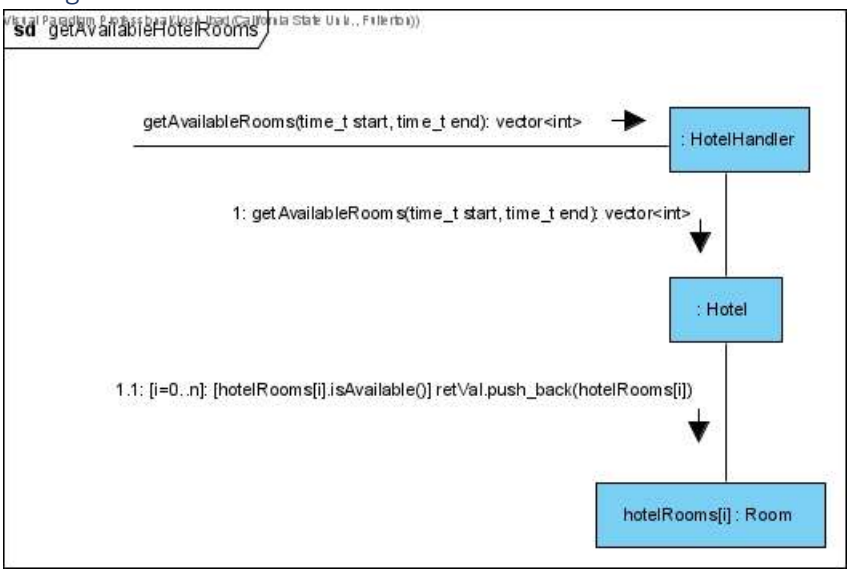
The system level message getHotelRoom is sent to the HotelHandler interface. The interface relays the message to an instance of the implementing Hotel class. The Hotel then iterates through its list of Rooms in hotelRooms. For each Room it iterates through called room, it get's the roomID using getRoomID, and tests it against the roomID specfied by the getHotelRoom message caller. Once a matching Room is found, this Room is returned up the chain to the message caller.

2.3.3 SSD Traceability

In the SSD 1 called 'Manage Hotel Rooms - 1. Add Room', getHotelRoom is the fourth message. This SSD can be found in document 03.1 - Use Case Model Annex 1, in section 2.1.2. The interface HotelHandler is located in the Domain Layer's Interface Diagram, inside the Hotel component. This can be found in document 07 - SW Architecture Document, in section 3.1.2.1. Hotel.

2.4 getAvailableHotelRooms Sequence of Execution

2.4.1 Software Interaction Diagram



2.4.2 Description

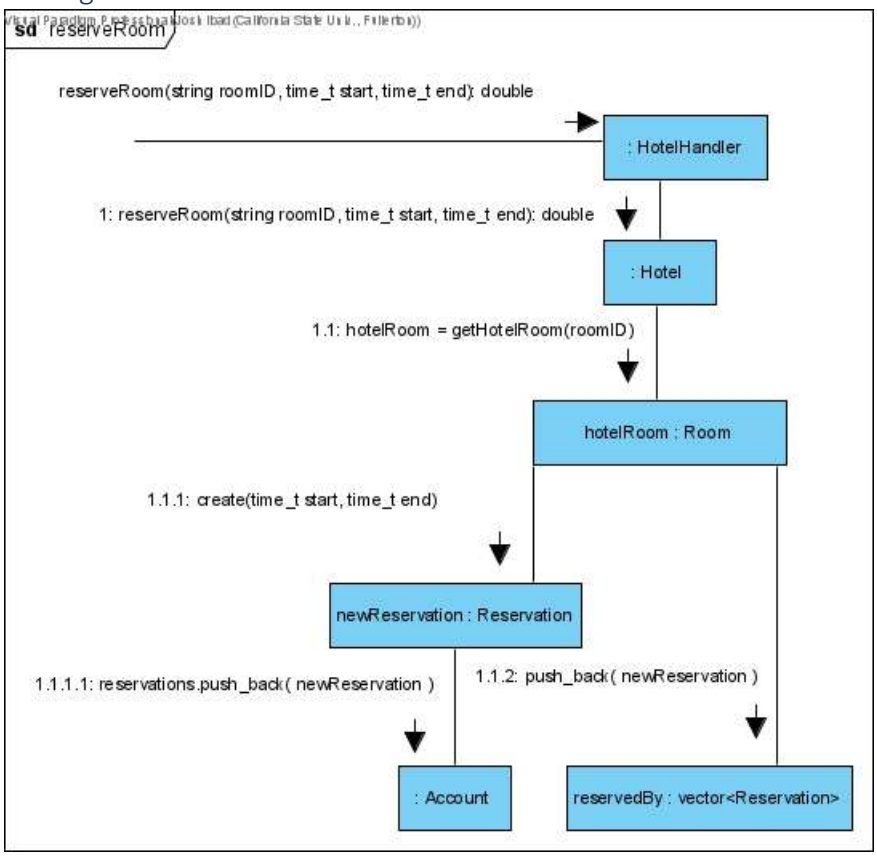
The system level message getAvailableHotelRoom is sent to the HotelHandler interface. The interface relays the message to an instance of the implementing Hotel class. The Hotel then iterates through its list of Rooms in hotelRooms. For each Room it iterates through called hotelRooms[i], it checks if the Room is available, calling the isAvailable function. If the Room is available, then it is pushed into the temporary vector called retVal. After iterating through every Room, the Hotel then returns the temporary vector up the chain to the message caller.

2.4.3 SSD Traceability

In the SSD 3 called 'Manage Reservations - 1. Create Reservation', getAvailableHotelRooms is the second message. This SSD can be found in document 03.1 - Use Case Model Annex 2, in section 2.1.2. The interface HotelHandler is located in the Domain Layer's Interface Diagram, inside the Hotel component. This can be found in document 07 - SW Architecture Document, in section 3.1.2.1. Hotel.

2.5 reserveRoom Sequence of Execution

2.5.1 Software Interaction Diagram



2.5.2 Description

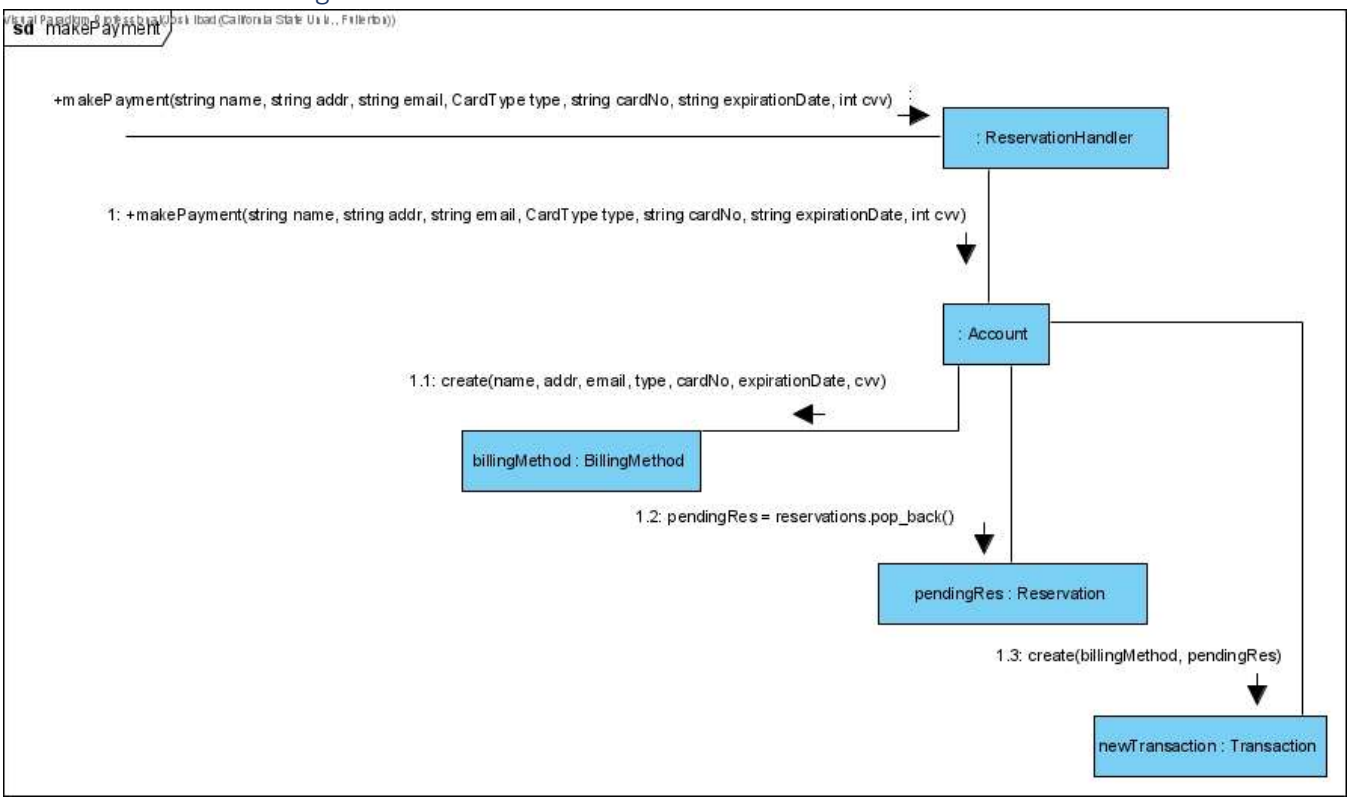
The system level message reserveRoom is sent to the HotelHandler interface. The interface relays the message to an instance of the implementing Hotel class. The Hotel then calls the getHotelRoom function described earlier in 2.1, and obtains the Room specified by the parameter roomID, storing it in the hotelRoom variable. The Room then creates a Reservation called newReservation, with the specfied start and end time parameters. The Reservation then pushes itself into the Reservations list of the calling user's Account. The newly created Reservation is then used by the Room and pushed into the Room's list of Reservations called reservedBy. The price of the Reservation is then sent up the chain from Room, to the Hotel, to the HotelHandler, to the message caller.

2.5.3 SSD Traceability

In the SSD 3 called 'Manage Reservations - 1. Create Reservation', reserveRoom is the fourth message. This SSD can be found in document 03.1 - Use Case Model Annex 2, in section 2.1.2. The interface HotelHandler is located in the Domain Layer's Interface Diagram, inside the Hotel component. This can be found in document 07 - SW Architecture Document, in section 3.1.2.1. Hotel.

2.6 makePayment Sequence of Execution

2.6.1 Software Interaction Diagram



2.6.2 Description

The system level message getHotelRoom is sent to the ReservationHandler interface. The interface relays the message to an instance of the implementing Account class. The Account then creates a new BillingMethod called billingMethod using the passed in parameters of name, address, email, card type, card number, expiration date, and cvv code. The newly created billingMethod is held by the Account while the Account pops the latest Reservation in its list of reservations, and stores it in pendingRes. The Account then creates a new Transaction using the held BillingMethod and Reservation called billingMethod and pendingRes respectively.

2.6.3 SSD Traceability

In the SSD 3 called 'Manage Reservations - 1. Create Reservation', makePayment is the fifth message. This SSD can be found in document 03.1 - Use Case Model Annex 2, in section 2.1.2. The interface ReservationHandler is located in the Domain Layer's Interface Diagram, inside the Reservation component. This can be found in document 07 - SW Architecture Document, in section 3.1.2.2. Reservation.