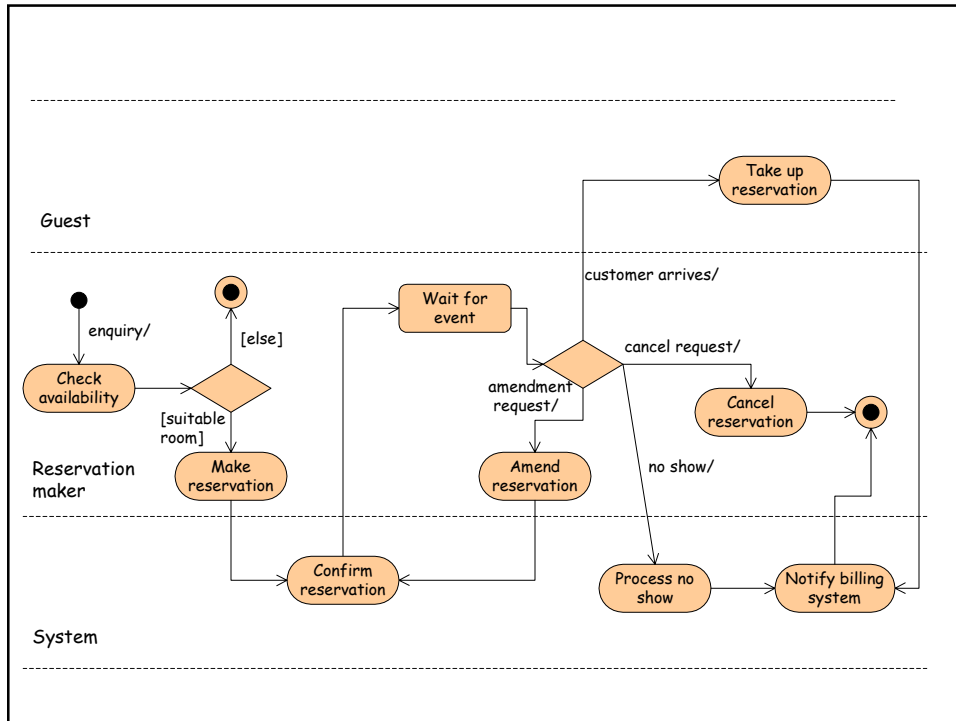


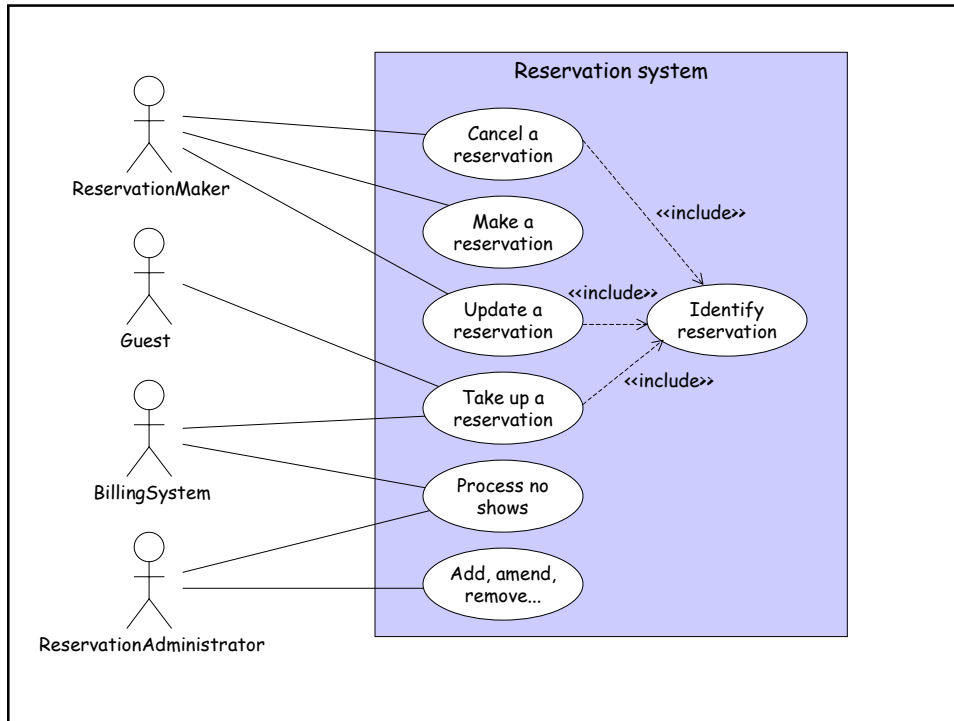
UML Components
(c) Addison Wesley 2000

Appendix A
Case Study Details

Business Process



Use cases



Name	Make a reservation
Initiator	Reservation Maker
Goal	Reserve room(s) at a hotel

Main success scenario

1. Reservation Maker asks to make a reservation
2. Reservation Maker selects in any order hotel, dates and room type
3. System provides price to Reservation Maker
4. Reservation Maker asks for reservation
5. Reservation Maker provides name and postcode (zip code)
6. Reservation Maker provides contact email address
7. System makes reservation and allocates tag to reservation
8. System reveals tag to Reservation Maker
9. System creates and sends confirmation by email

Extensions

3. Room not available
 - a) System offers alternatives
 - b) Reservation Maker selects from alternatives
- 3b) Reservation Maker rejects alternatives
 - a) Fail
4. Reservation Maker declines offer
 - a) Fail
6. Customer already on file (based on name and postcode)
 - a) Resume 7

Name	Take up reservation
Initiator	Guest
Goal	Claim a reservation and check in to the hotel

Main success scenario

1. Guest arrives at hotel and claims a reservation
2. Include Identify Reservation
3. Guest confirms details of stay duration, room type
4. System allocates room
5. System notifies billing system that a stay is starting

Extensions

3. Reservation not identified
 - a) Fail

Name	Identify reservation
Initiator	included only
Goal	Identify an existing reservation

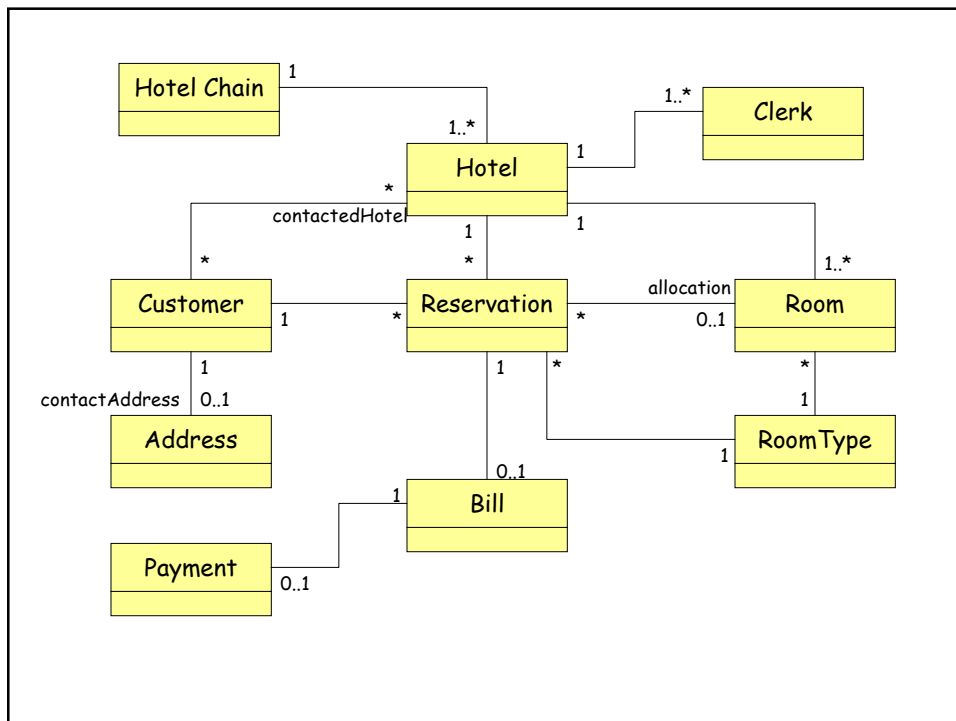
Main success scenario

1. Actor provides reservation tag
2. System locates reservation

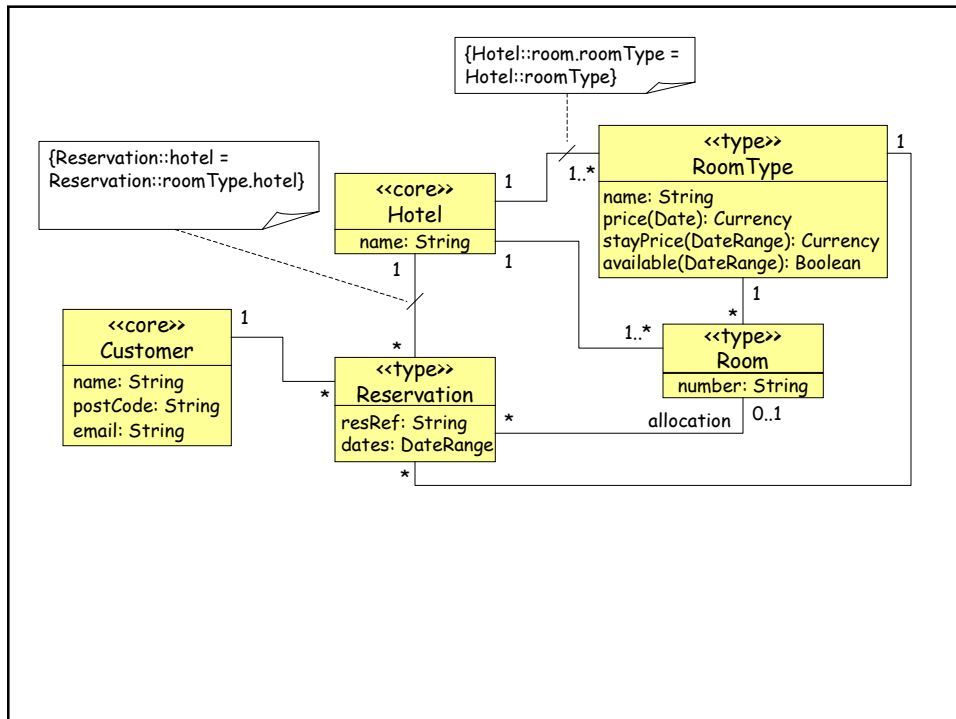
Extensions

2. System cannot find a reservation with the given tag
 - a) Actor provides name and post code
 - b) System displays active reservations for that customer
 - c) Actor selects the reservation
 - d) Stop
2. The reservation tag refers to a reservation at a different hotel
 - a2) Fail
- 2b) No active reservations at this hotel for this customer
 - a) Fail

Business Concept Model



Business Type Diagram



context RoomType

-- AVAILABILITY RULES

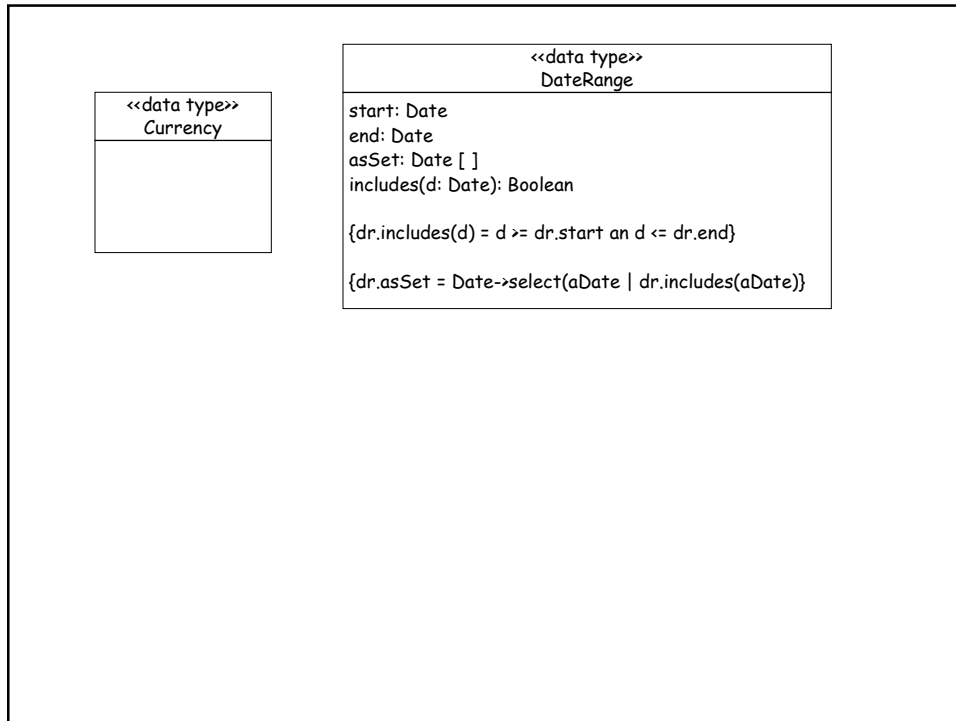
-- a room is available if the number of rooms reserved on all dates
-- in the range is less than the number of rooms
available(dr) = dr.asSet->collect(d | reservation->select(r | r.allocation->isEmpty and
r.dates.includes(d)->size)->max < room->size)

-- can never have more reservations for a date than rooms (no overbooking)
Date->forAll(d | reservation->select(r | not r.allocation->isEmpty and
r.dates.includes(d)->size) <= room->size

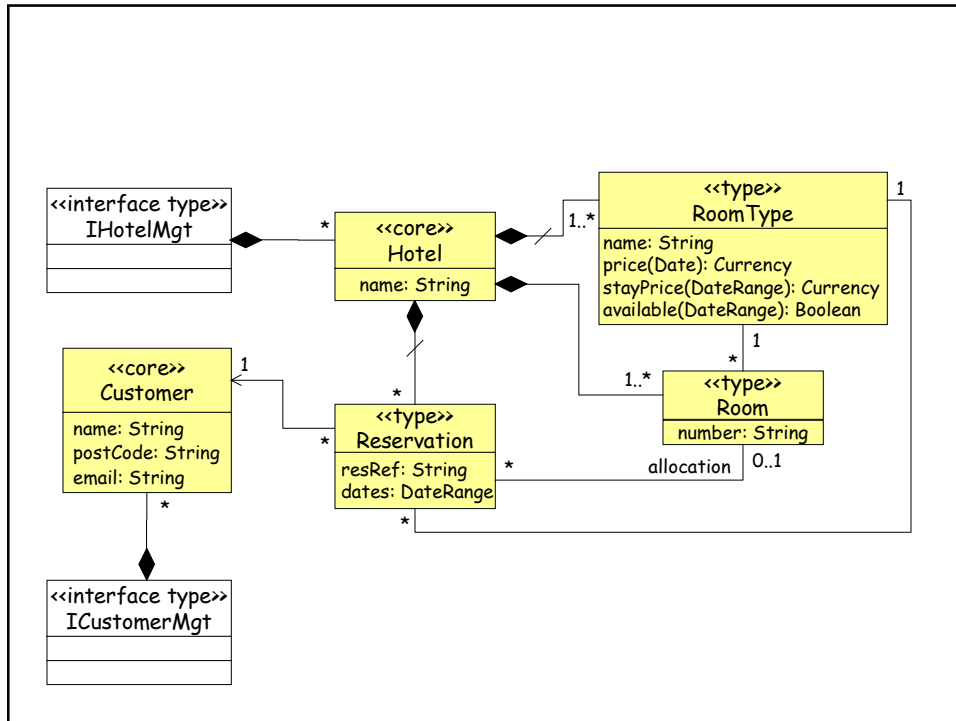
-- PRICING RULES

-- the price of a room for a stay is the sum of the prices for the days in the stay
stayPrice(dr) = dr.asSet->collect(d | price(d))->sum

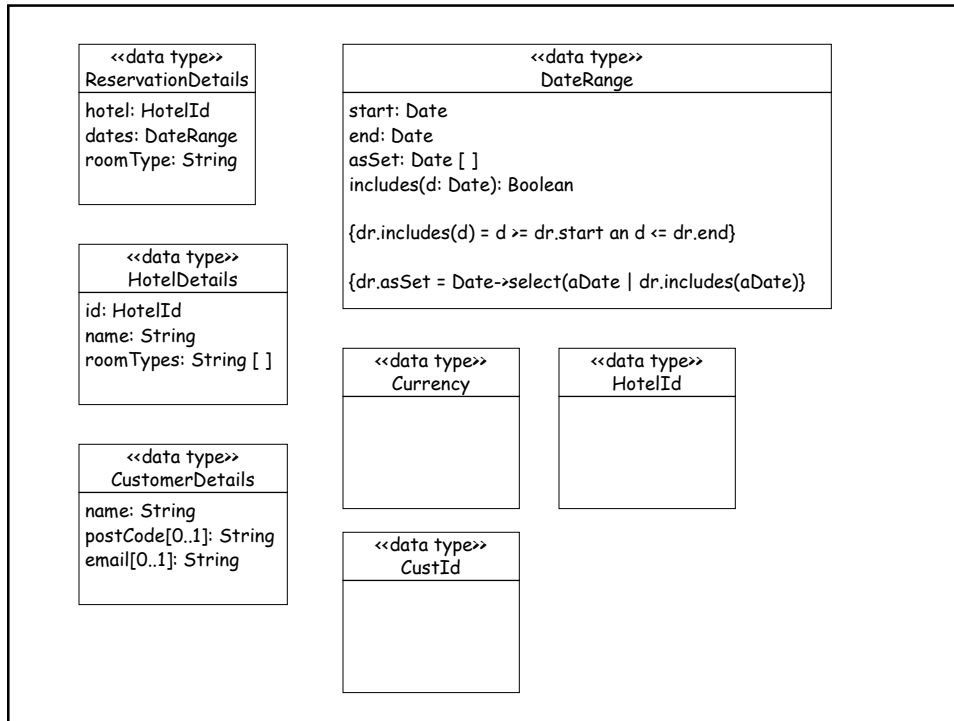
Structured Data Types (BTM)



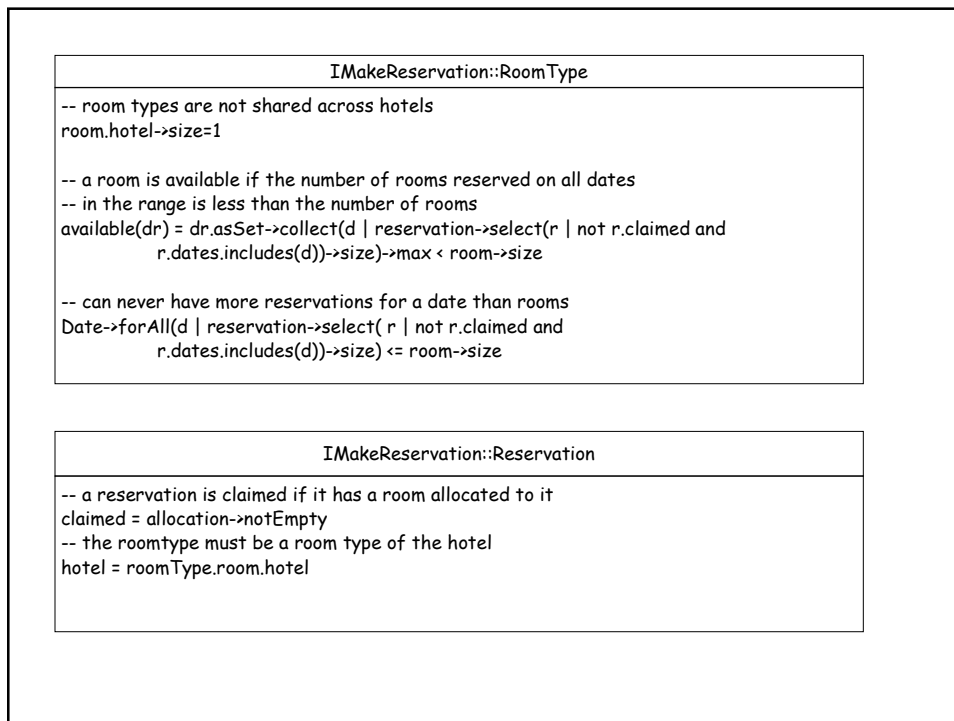
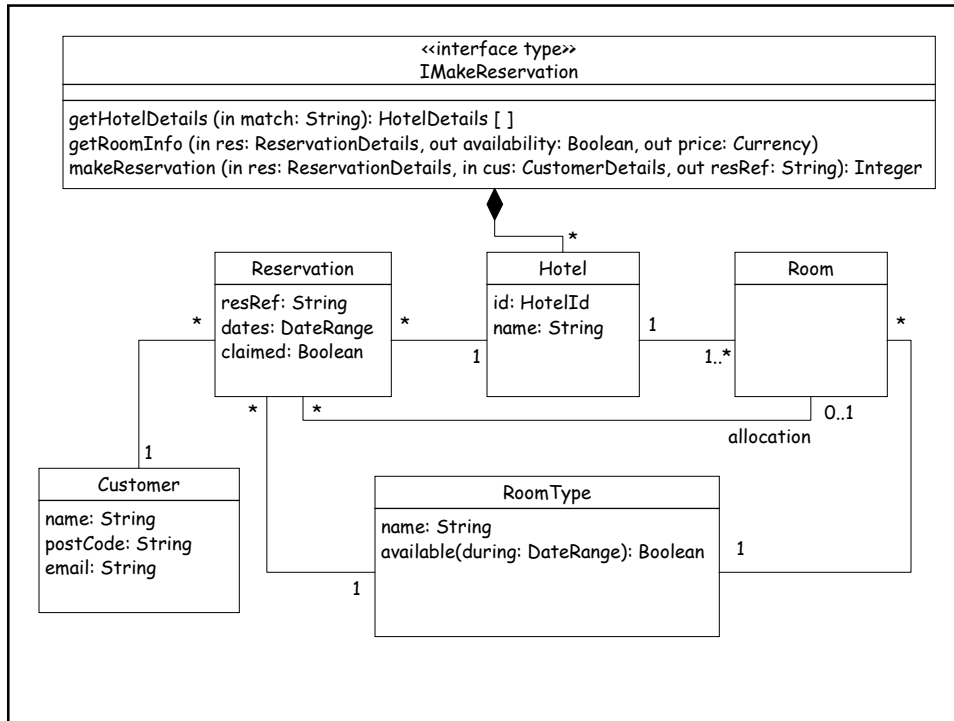
Interface Responsibility Diagram

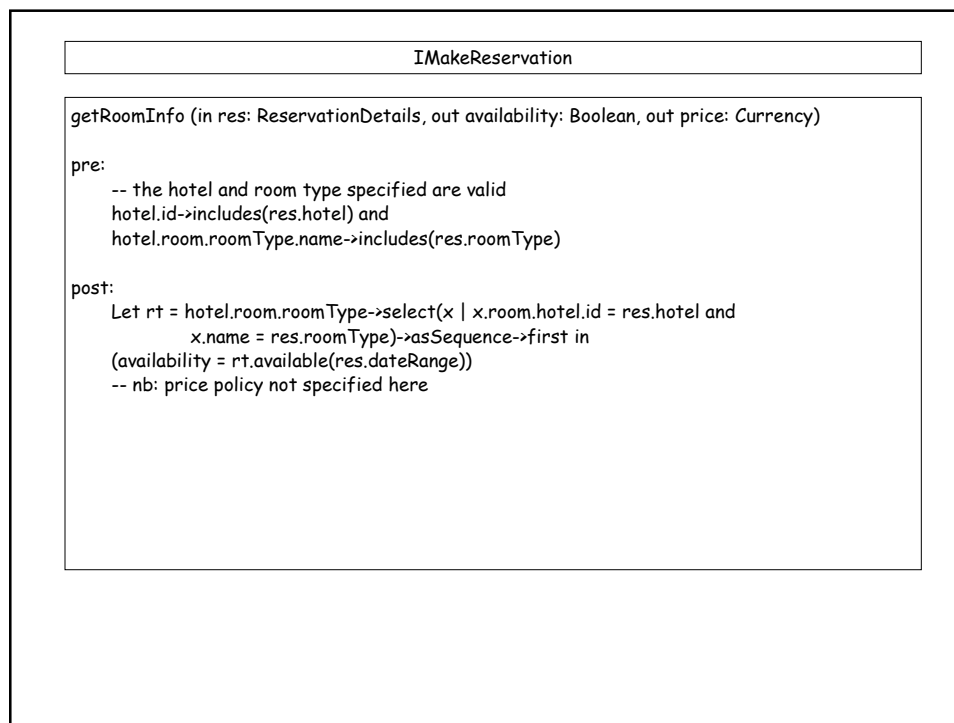


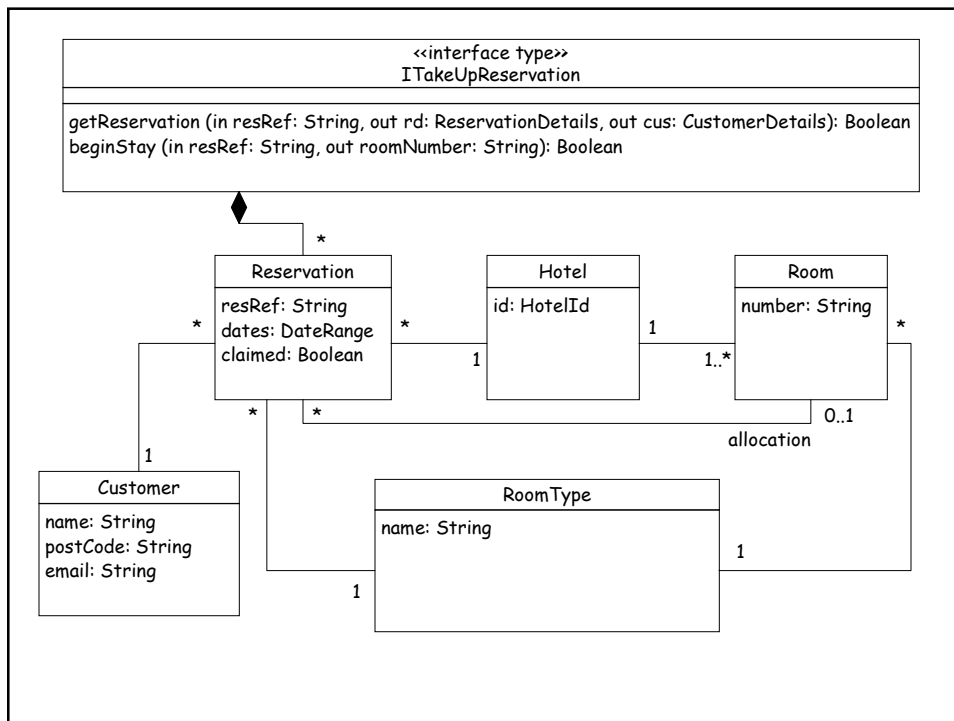
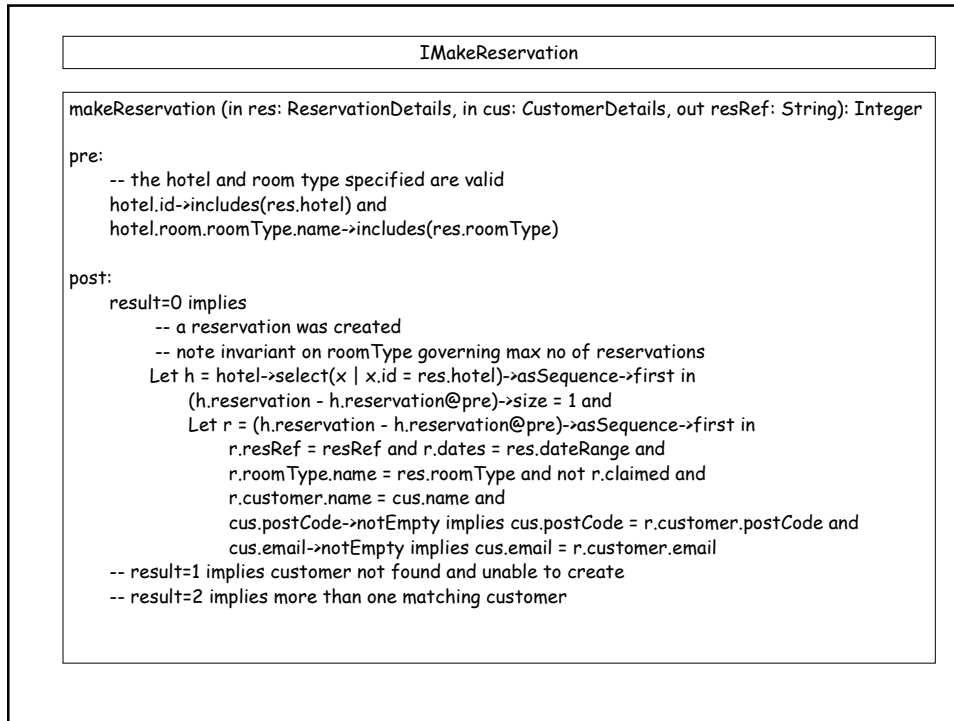
Structured Data Types (Interface specs)

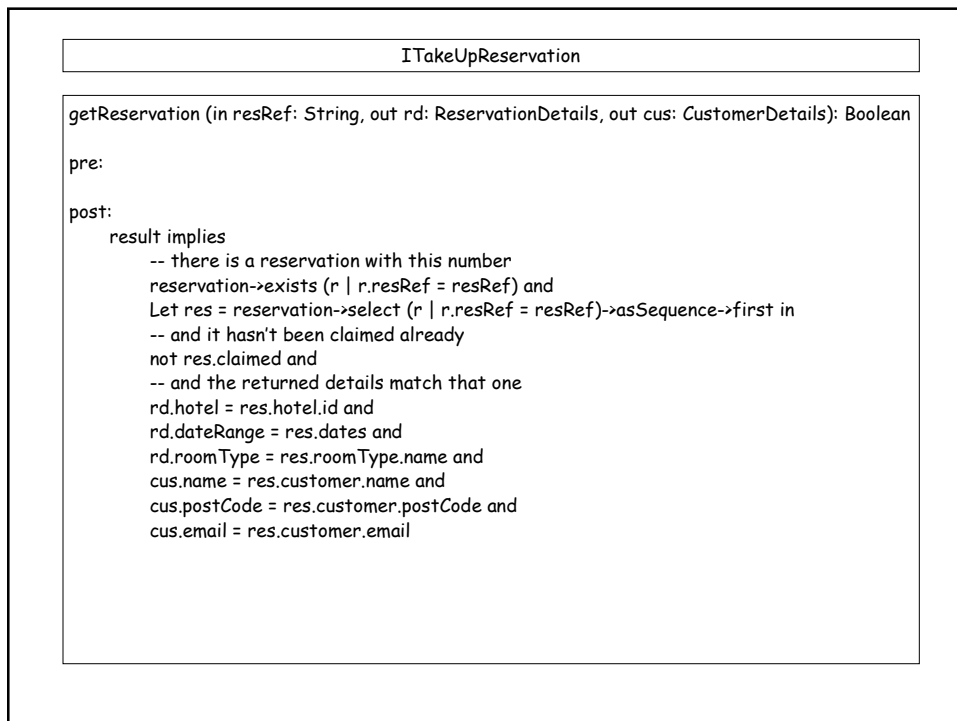
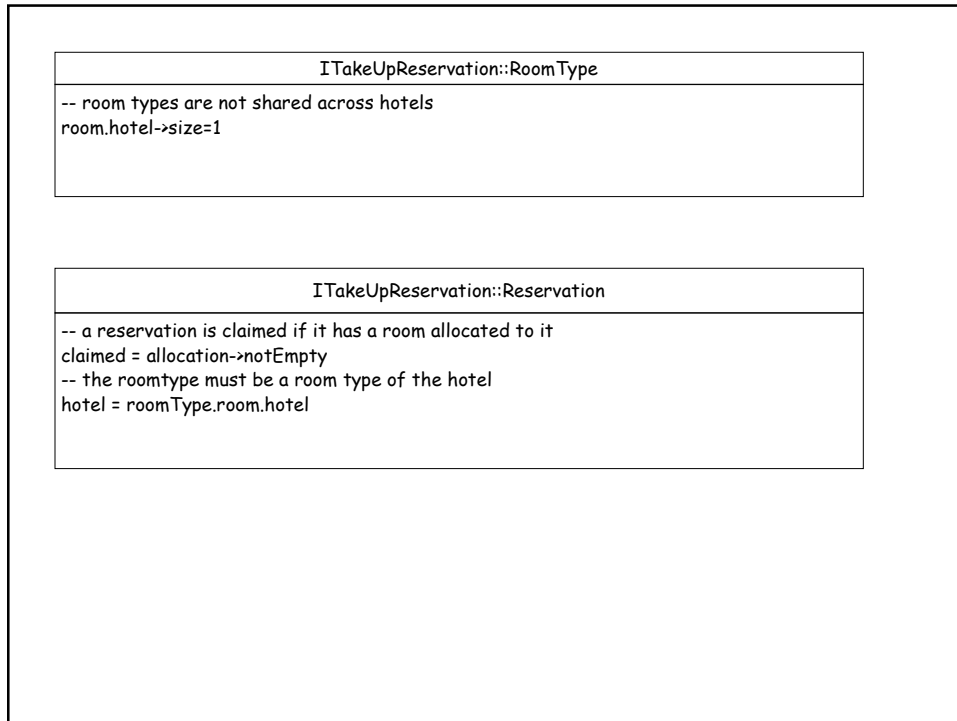


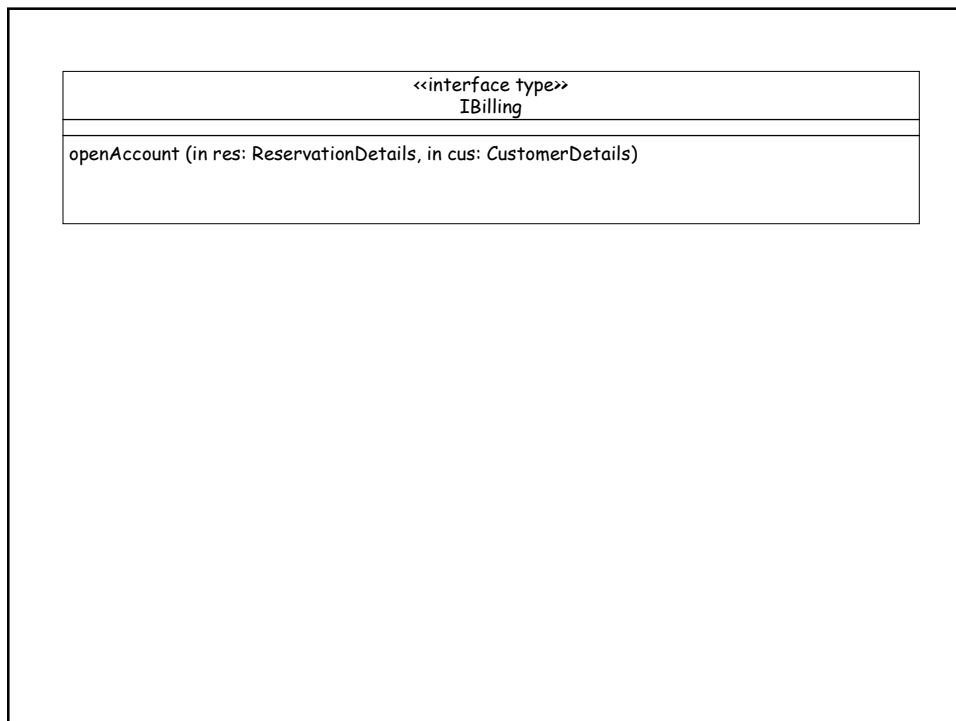
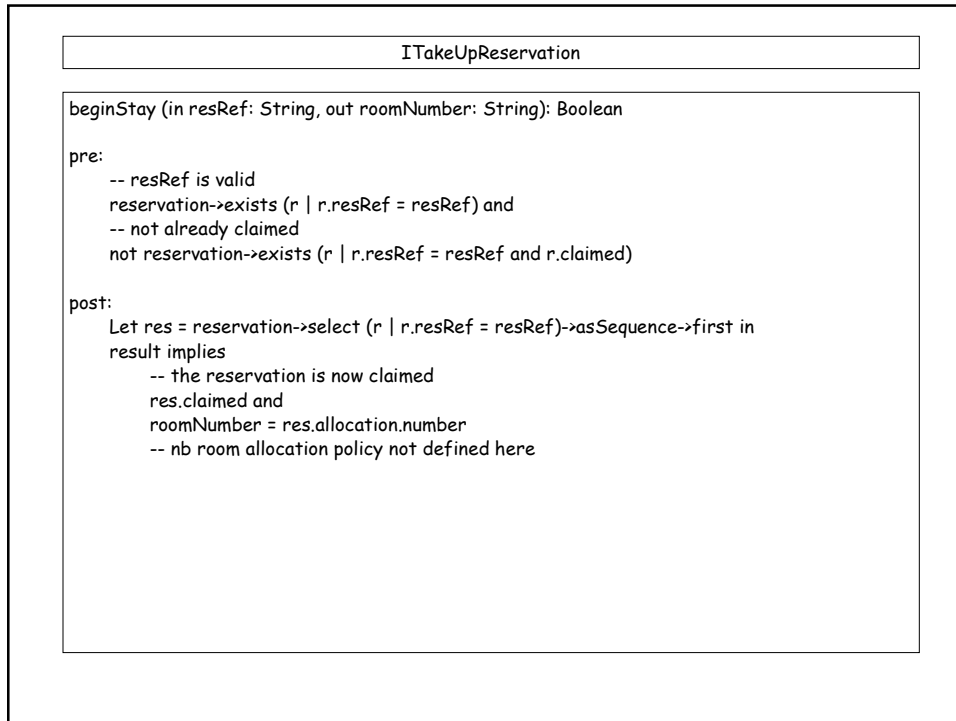
System interfaces



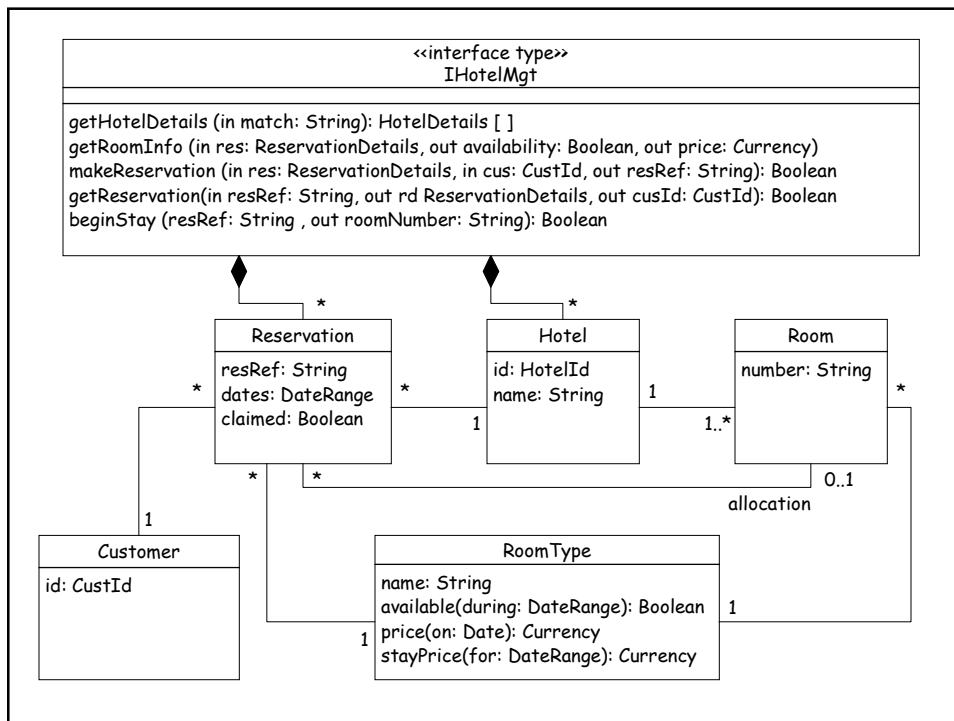


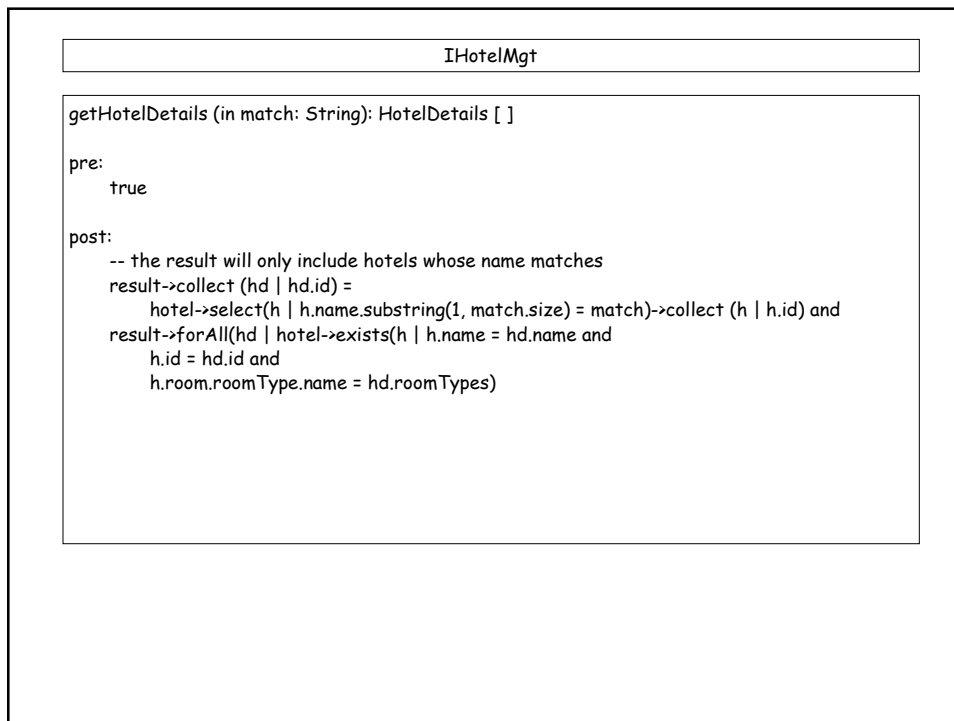
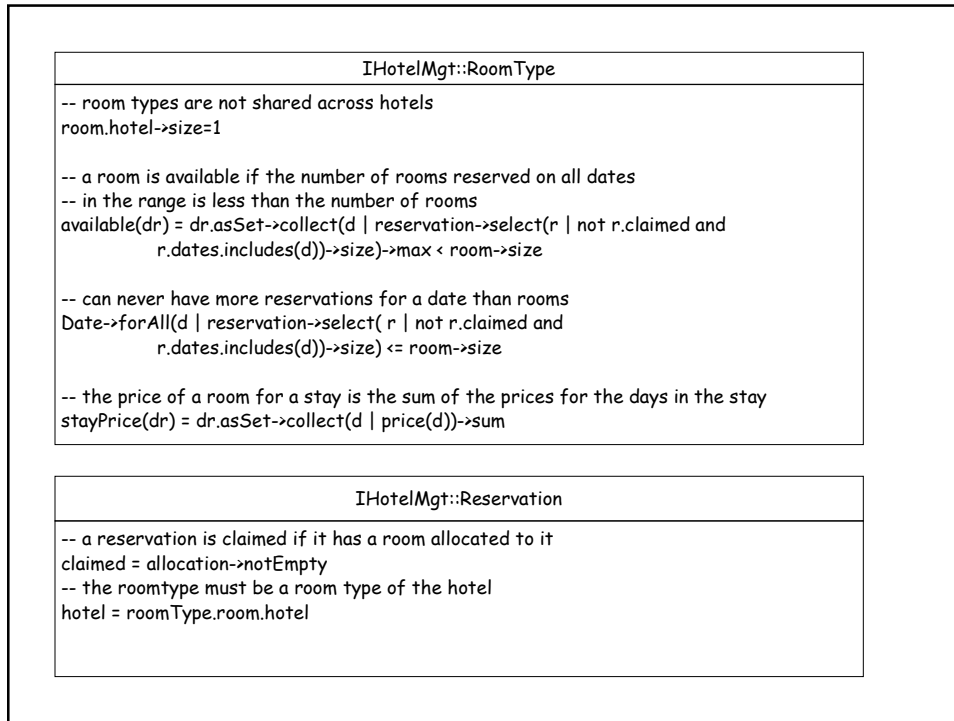


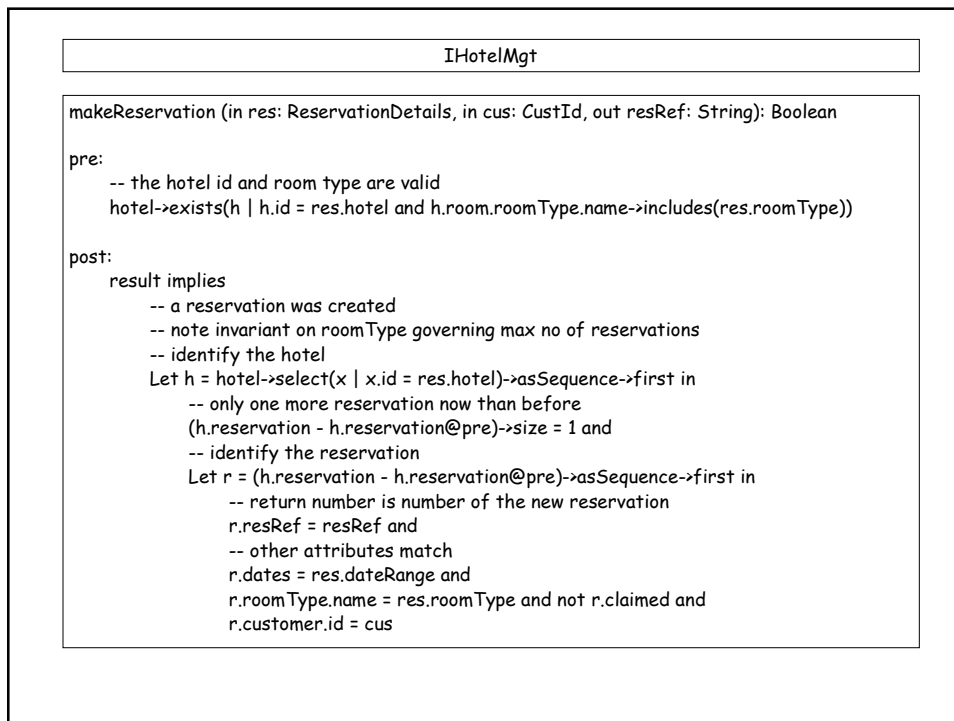
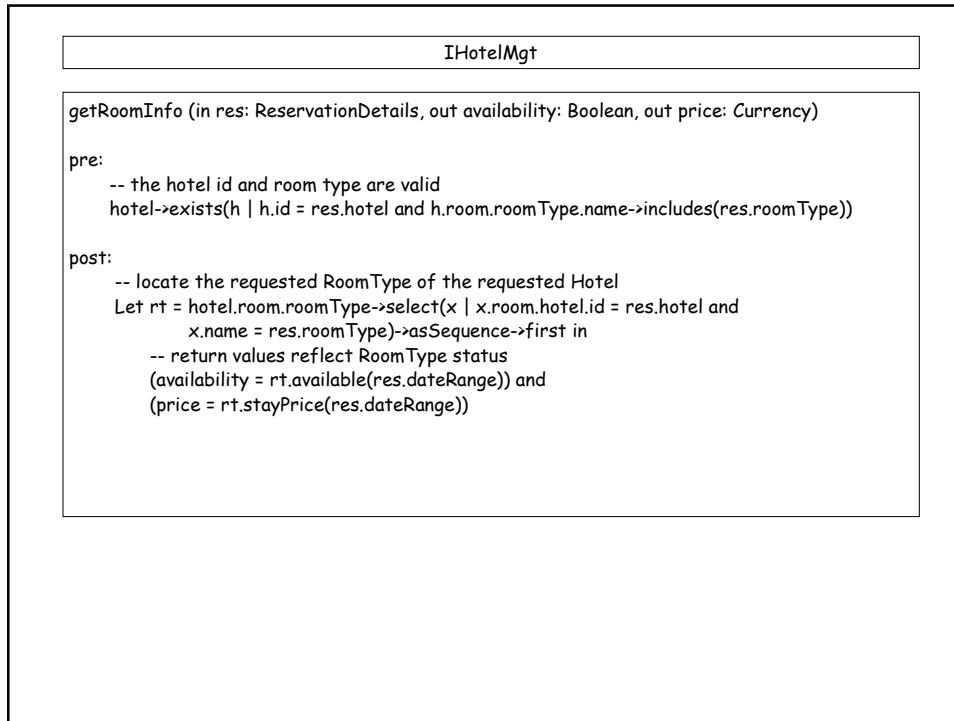


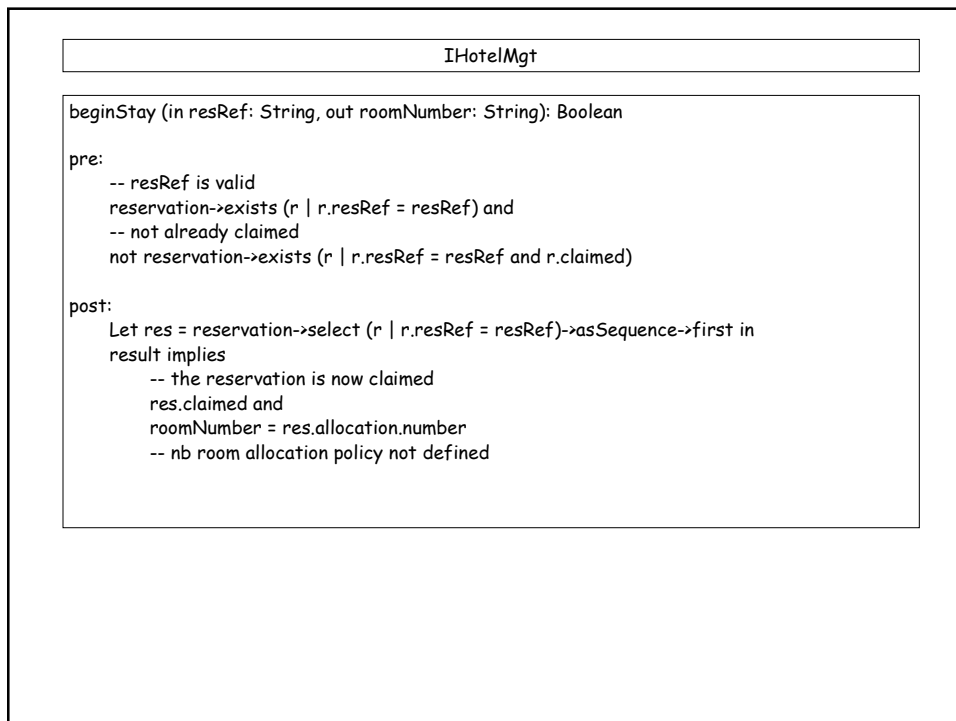
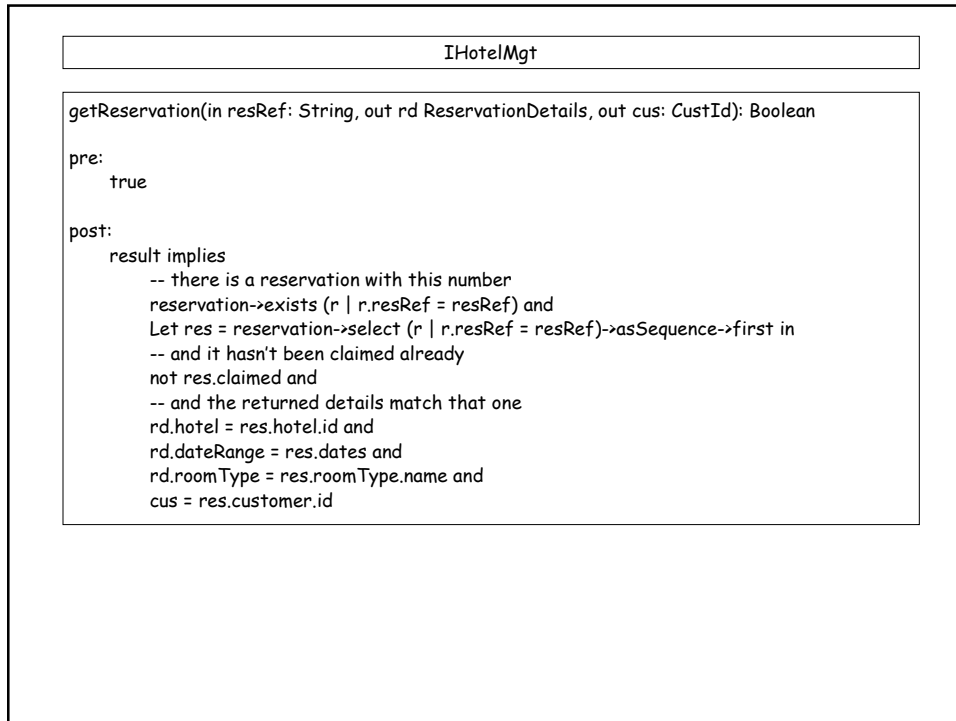


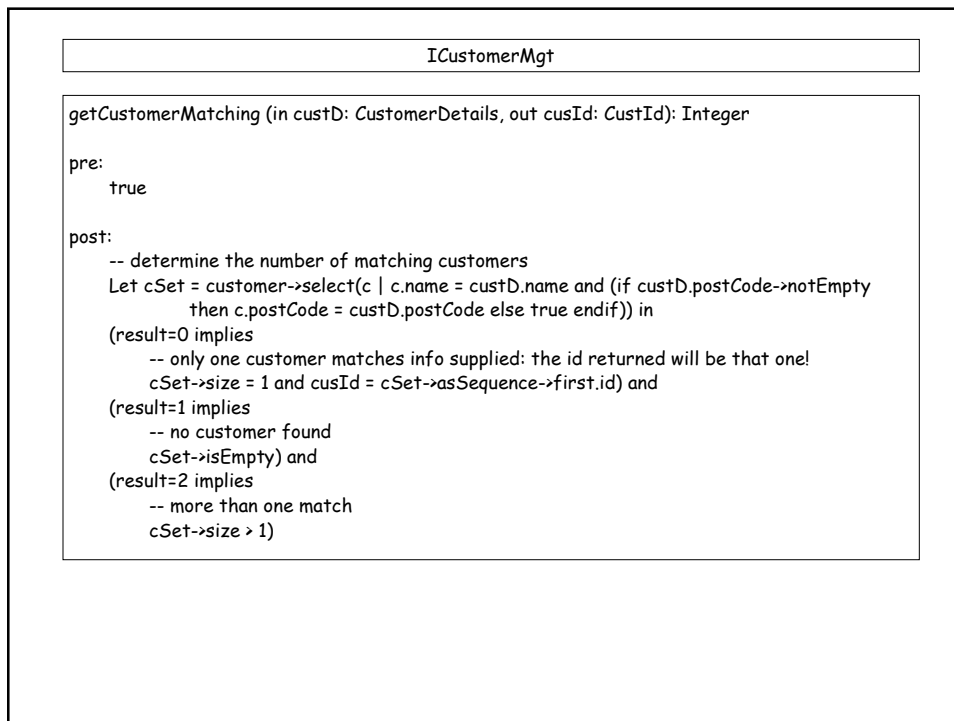
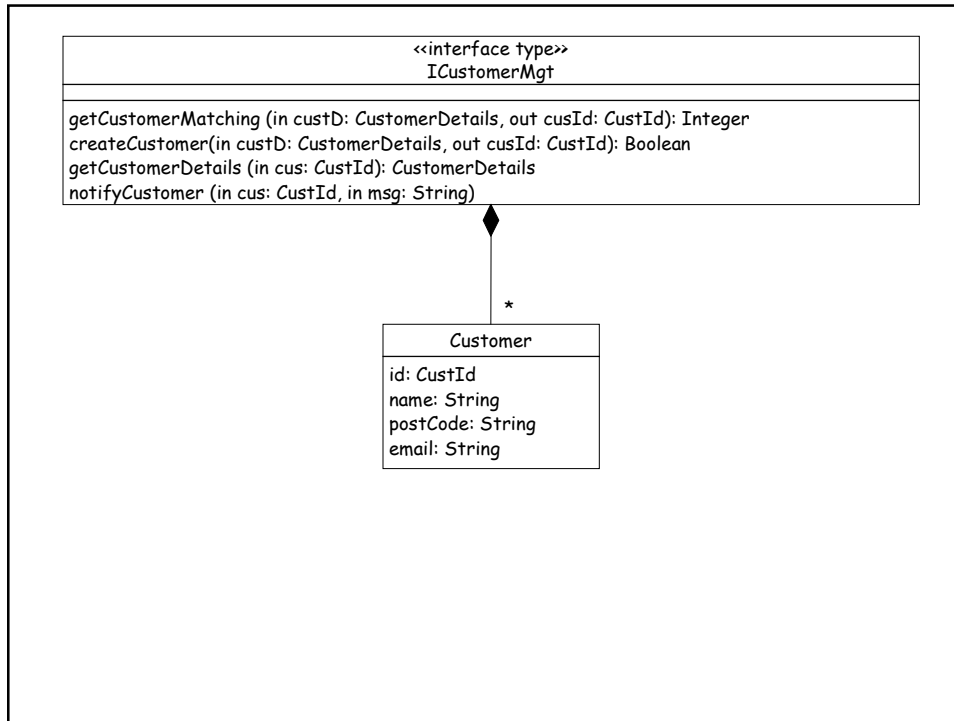
Business type interfaces

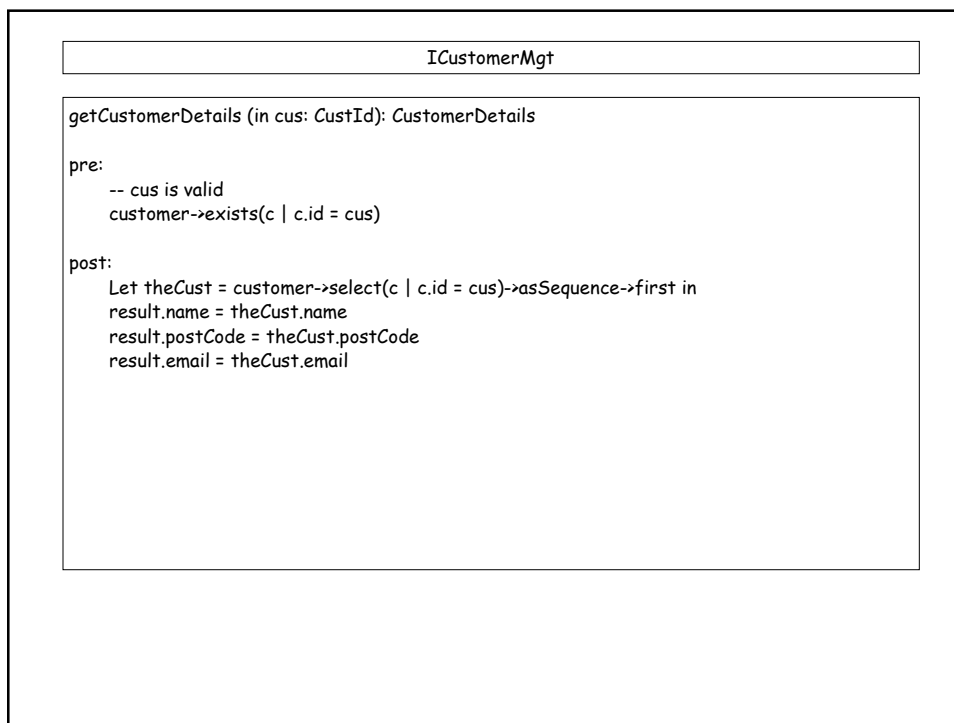
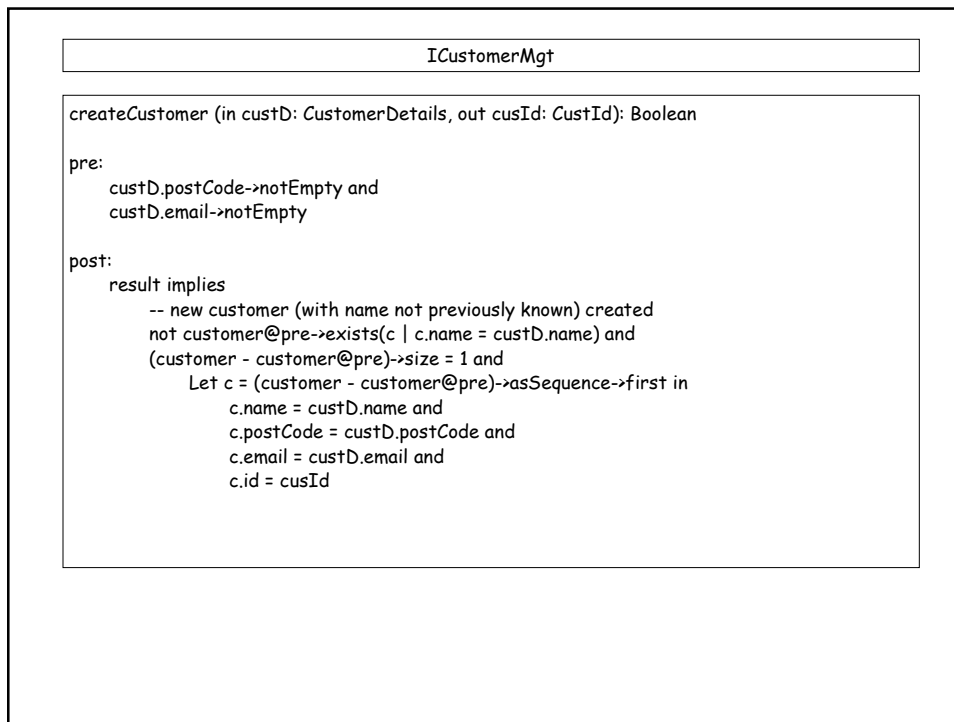


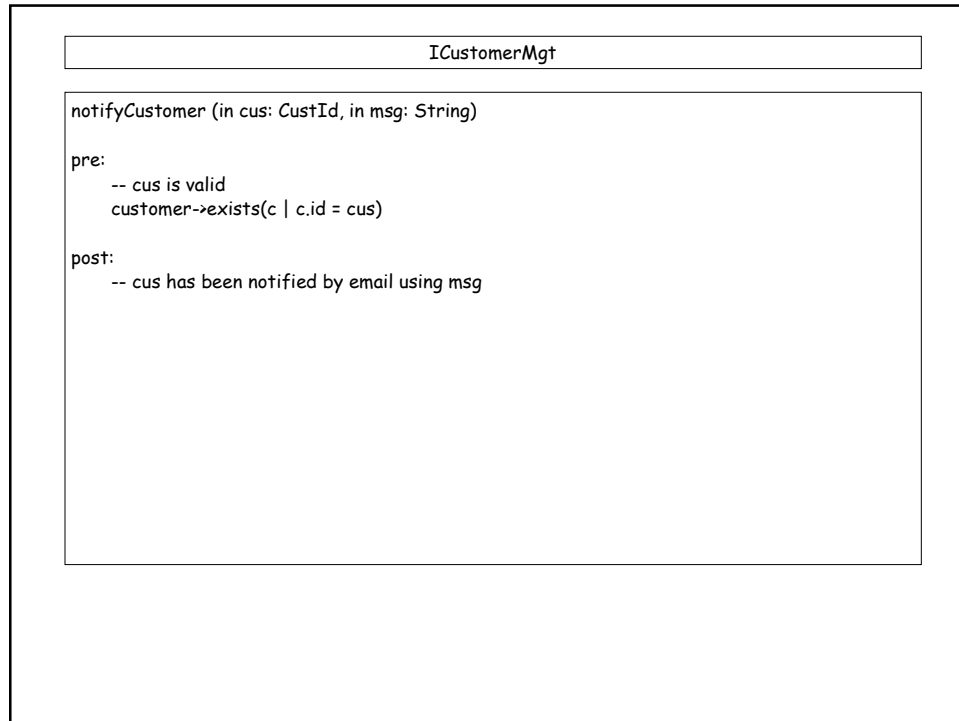




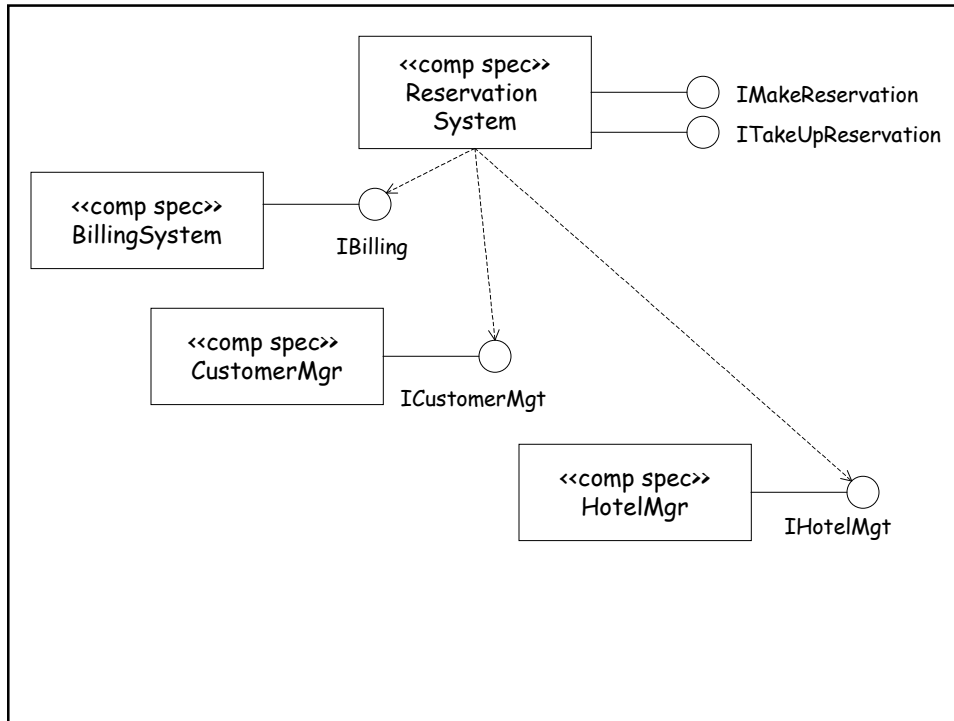








Component Architecture



Component Specs

