Exam 2015, questions and answers

System Analysis PG (Charles Sturt University)

Part A: Short Question and Answer

Answer the following questions

1. What are the models that describe use cases in more detail?

Fully developed use case description, activity diagrams, and system sequence diagrams.

2. What two UML diagrams are used to model domain classes?

Problem domain class diagram and state machine diagram.

3. Which part of a use case description can also be modeled by using an activity diagram?

The "flow of activities" section.

4. Explain the difference between a use case and a scenario. Give a specific example of a use case with a few possible scenarios.

A use case is the entire function or user goal or event. A scenario is one specific version or instance of that use case.

From RMO we have *Create customer account* as a use case. But we might have *Create online Customer account* and *Create instore customer account* and even *Create phone customer account* as different scenarios.

5. List the parts or compartments of a fully developed use case description.

- Use case name
- Scenario
- Triggering event
- Brief description
- Actors
- Related use cases
- Stakeholders
- Preconditions
- Postconditions
- Flow of activities
- Exception conditions

6. Compare/contrast precondition and postcondition.

A precondition describes the "states" of data and the system that must exist before the use case can begin. For example to add an item to a shopping cart, the item must exist in the database.

A postcondition describes the states of data and the system that must exist after the use case completes. For example after *Create a customer account* use case, a customer account object (record) must exist.

7. Compare/contrast postcondition and exception condition.

A post condition, as explained about describes states of the data and the system. An exception condition describes some non-normal situation in the processing, i.e. in the flow of activities, that must be handled in some way..

8. Compare/contrast business process and flow of activities for a use case. Explain how

an activity diagram can be used to model both.

A business process is larger than a use case. A business process might include various manual business procedures both before and after the "business event" that causes the use case to occur. However, an activity diagram is a powerful model to describe all types of sequences of tasks and activities. The various swimlanes can represent various users or user groups as well as system activities.

9. What is the purpose of an SSD? What symbols are used in an SSD?

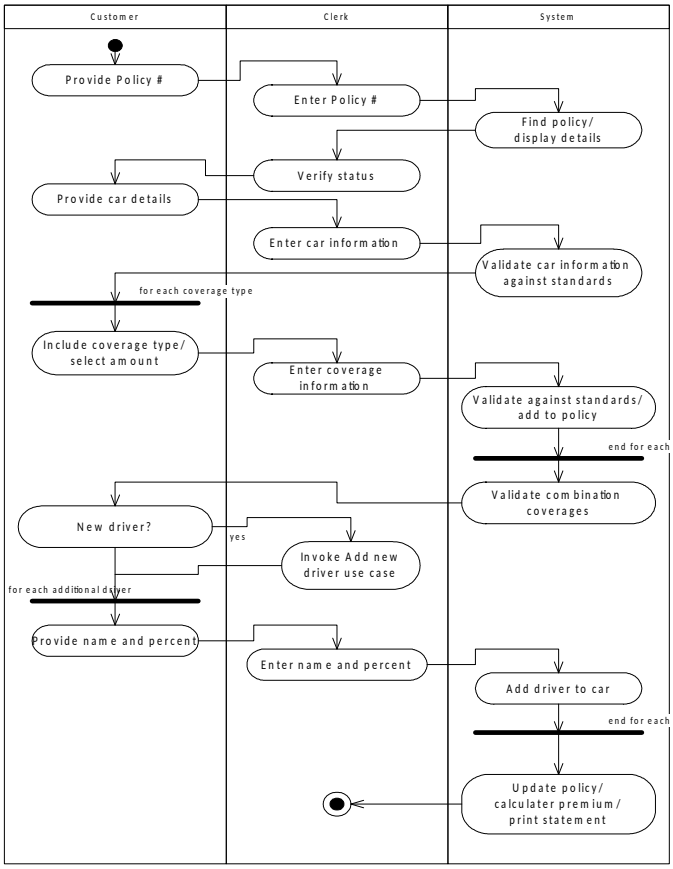
An SSD (system sequence diagram) is used to describe the messages that flow into and out of a system, i.e. between the system and the use case user. The symbols include:

- Stick figure for the actor
- Box with object name for the system object
- Vertical dashed lines for object lifelines
- Horizontal arrows for messages
- Horizontal dashed arrows for return data
- Comment box for comments

10. What are the steps required to develop an SSD?

1. Identify the input messages (from the activity diagram)
2. Describe the input message using the SSD message syntax
3. Add message conditions such as looping or true/false conditions
4. Add all output message data

1. Based on the following narrative, develop a fully developed description for the use case of *Add a new vehicle to an existing policy* in a car insurance system.



Use Case Name:	Add a new vehicle to an existing policy	
Scenario:	Telephone instance with customer and clerk	
Triggering Event:	Customer buys a new vehicle.	
Brief Description:	Customer provides car information, requests coverage with amounts, identifies drivers of the new car. System updates the policy.	
Actors:	Customer service clerk	
Stakeholders:	Customer Customer service department	
Preconditions:	Customer policy must exist and be up to date. StandardVehicle control tables for this vehicle type and year must exist. StandardCoverage tables exist.	
Postconditions:	New vehicle object created and connected to policy. Also connected to StandardVehicle. New coverage objects created and connected to vehicle. Also connected to StandardCoverage. New driver (InsuredPerson) (if necessary) created and added to policy. Existing drivers and percentages updated. Policy updated with new premiums.	
Flow of Activities:	Actor	System
	1. Clerk enters customer information. 2. Clerk verifies policy is current. 3. Clerk enters car identification information. 4. Clerk enters each type of coverage customer requests, including deductibles and coverage amount.	1.1 System finds policy and displays details. 3.1 System validates that car has known standard. 4.1 System validates coverage requests.

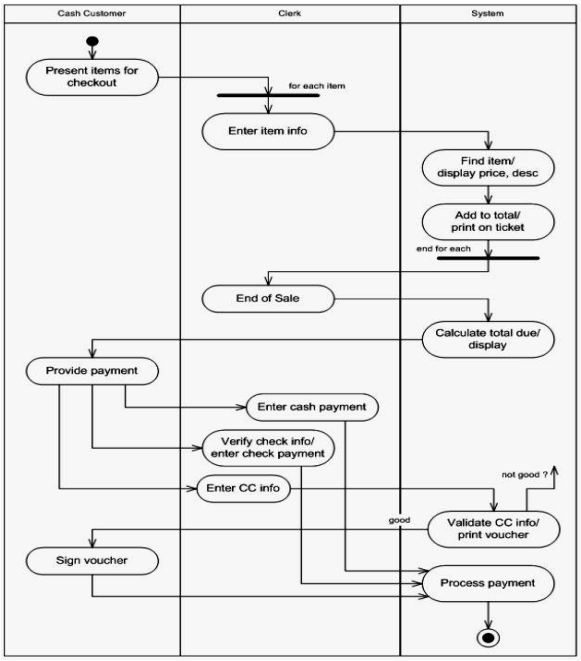
	5. Clerk indicates all coverages have been entered. 6. Clerk invokes *Add new person* use case if necessary. 7. Clerk changes driver percentages on this car and other cars. 8. Clerk indicates everything is complete.	5.1 System does combination validation on policy. 7.1 System updates driver information. 8.1 System updates policy, calculates new premium, prints new statement.
Exception Conditions:	2.1 If policy is not current, clerk requests payment or collects necessary information. 3.1 If car type is not in system, clerk refers customer to underwriting to handle this situation. 4.1 If coverage requests are out of range, clerk asks customer for changed amount. 5.1 If some combination is invalid, return to step 4.	

3. Given the following list of classes and associations for the previous car insurance system, list the preconditions and postconditions for the use case *Add a new vehicle* to an existing policy.

Preconditions:	Customer policy must exist and be up to date. StandardVehicle control tables for this vehicle type and year must exist. StandardCoverage tables exist.
Postconditions:	New vehicle object created and connected to policy. Also connected to StandardVehicle. New coverage objects created and connected to vehicle. Also connected to StandardCoverage. New driver (InsuredPerson) (if necessary) created and added to policy. Existing drivers and percentages updated. Policy updated with new premiums.

4. Develop an SSD based on the following narrative and activity diagram.

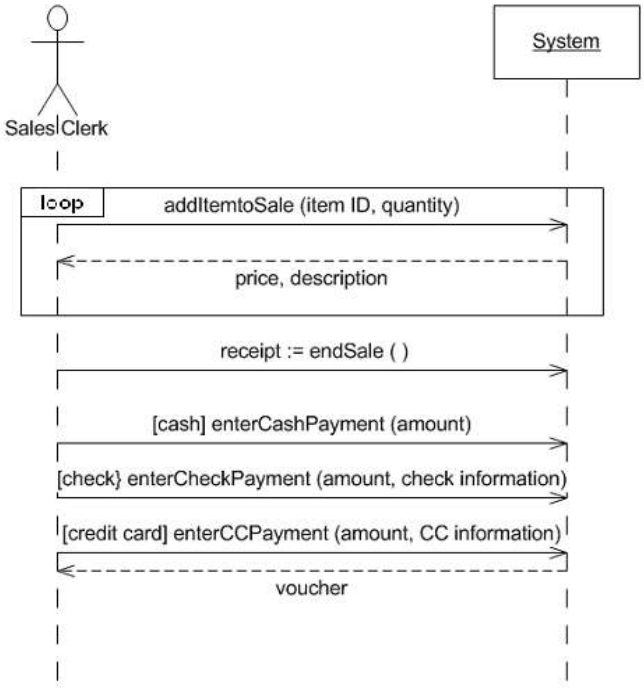
Sale to public activity diagram:



Sale to public fully developed use case description:

Use Case Name:	Create a new sale	
Scenario:	A new cash sale	
Triggering Event:	Cash customer wants to purchase items.	
Brief Description:	A cash customer wants to purchase items. The clerk enters the item ID, and the system creates a sales ticket. Customer pays with cash, check or credit card.	
Actors:	Sales clerk	
Stakeholders:	Sales clerk Accounting department Sales department	
Preconditions:	Inventory items must exist.	
Postconditions:	New sale is created. Sales line items are created and connected to the sale. Payment transaction is created.	
Flow of Activities:	Actor	System
	1. Clerk starts new cash sale. 2. Clerk enters each item. 3. Clerk indicates the end of the sale. 4. Clerk indicates type of payment and enters information.	 2.1 System finds item in inventory, finds price, displays information, adds to total. 3.1 System calculates total. 4.1 System processes payment and creates payment transaction.
Exception Conditions:	2.1 If system has information missing, sales clerk calls manager and manually enters information. 4.1 If customer credit card fails approval, require cash or cancel sale.	

Public sale:



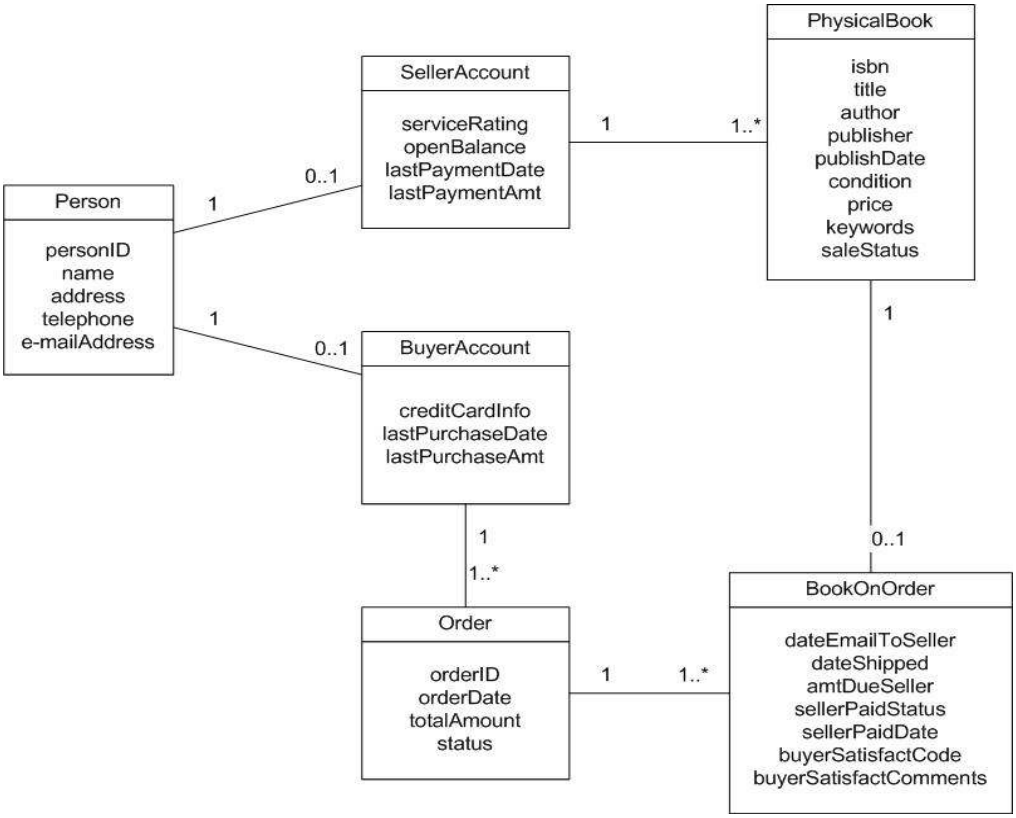
Solutions to End-of-Chapter Cases

Case Study: TheEyesHaveIt.com Book Exchange

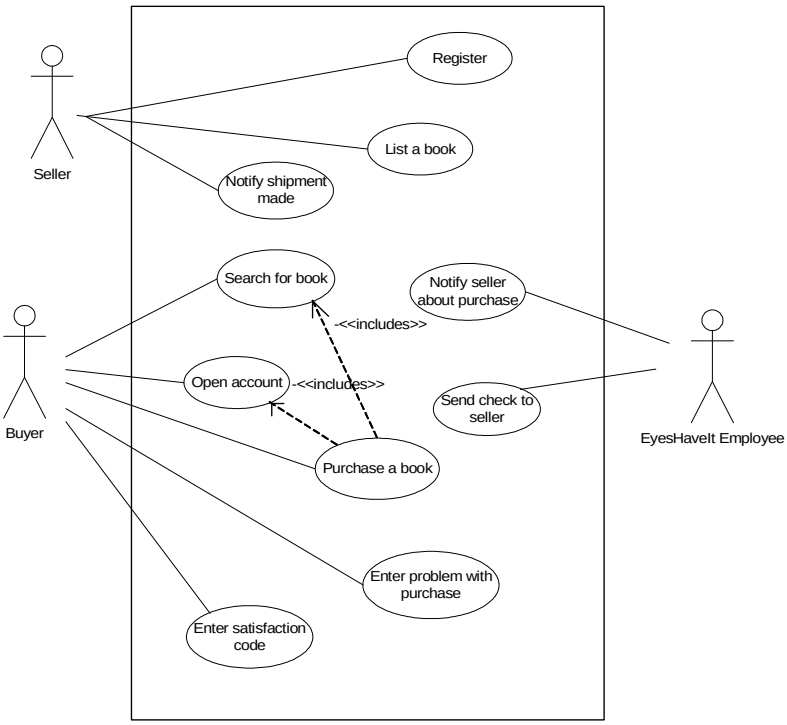
For this case, develop these diagrams:

1. **A domain model class diagram**
2. **A list of uses cases and a use case diagram**
3. **A fully developed description for two use cases: *Add a seller* and *Record a book order***
4. **An SSD for each of the two use cases in question 3**

1. A domain model class diagram



2. A list of uses cases and a use case diagram



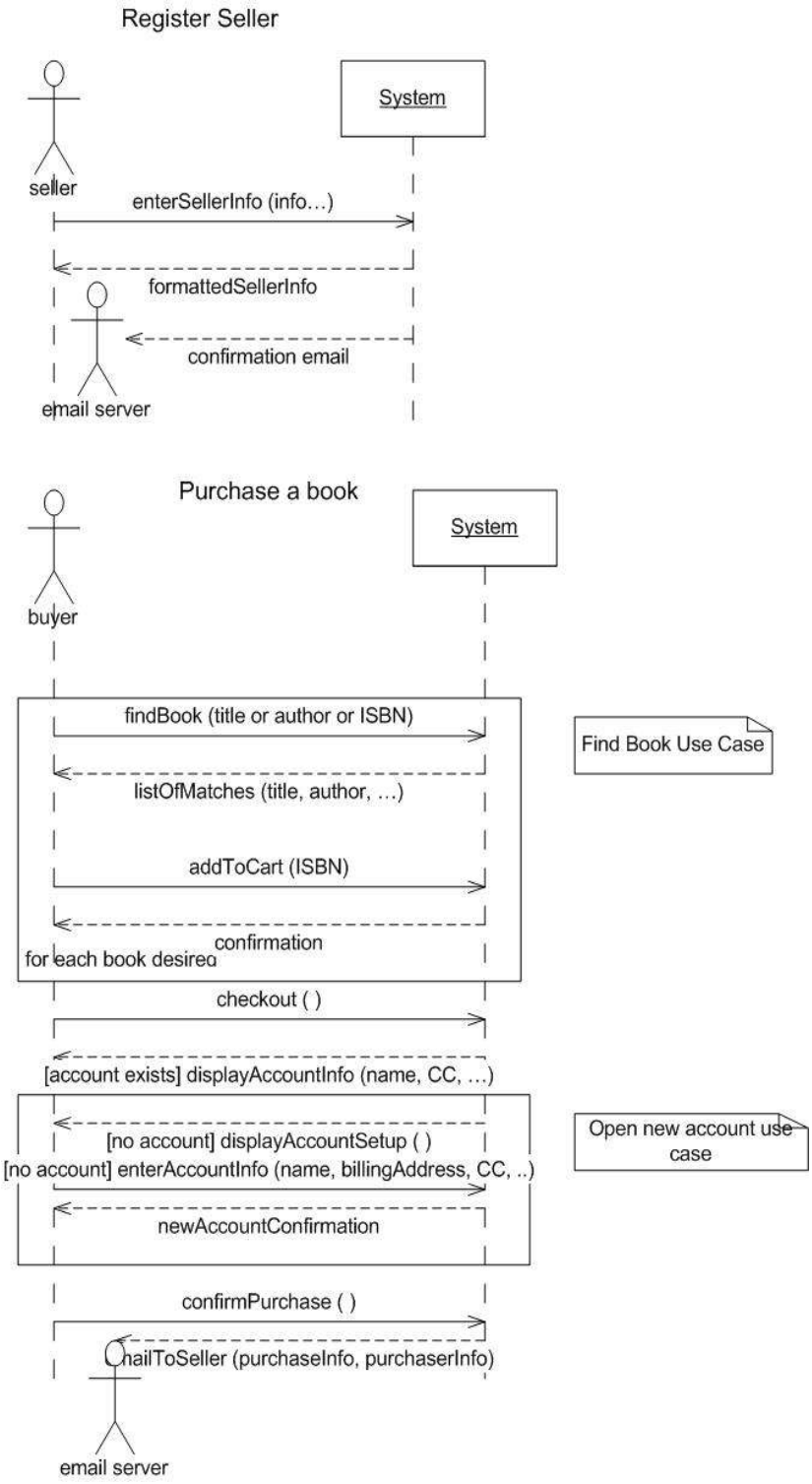
3. A fully developed description for two use cases: *Add a seller* and *Record a book order*

Use Case Name:	Register	
Scenario:	Register/add a new seller	
Triggering Event:	A new seller wants to sell books.	
Brief Description:	Seller decides he/she would like to list a book(s). Seller registers and receives a confirmation e-mail.	
Actors:	Seller E-mail server	
Stakeholders:		
Preconditions:	Seller must not exist in the system. Seller must have all information necessary to register.	
Postconditions:	Seller has an account to list books.	
Flow of Activities:	Actor	System
	1. Seller connects to EyesHaveIt.com and fills out registration form. 2. Seller submits registration form.	 2.1 System notifies seller a confirmation e-mail will be sent. 2.2 System e-mails confirmation of registration to seller.
Exception Conditions:	1.1 If the seller already exists in the system, the system sends prior login and password to e-mail address. 2.1 If the seller was removed from the system for bad transactions/credit, the system sends the seller an e-mail notifying the seller of the situation and no account is created.	

Use Case Name:	Purchase a book
Scenario:	Purchase a book
Triggering Event:	A buyer decides to purchase a book from EyesHaveIt.com.
Brief Description:	Customer searches for a book(s) on EyesHaveIt.com. Customer selects from search results and adds a book(s) to the shopping cart. Customer then proceeds to checkout. If an account exists, the customer confirms purchase, and the system sends a confirmation e-mail to the customer. If an account doesn't exist, an account is created, the purchase is confirmed, and the confirmation e-mail is sent.
Actors:	Buyer E-mail server Seller
Stakeholders:	
Preconditions:	Books and book information must exist in the system.
Postconditions:	Customer account must exist. Order must be placed. E-mail must be sent to seller.

Flow of Activities:	Actor	System
	1. Buyer searches EyesHaveIt.com for a book(s).	1.1 Searches for all matches related to buyer's search criteria.
	2. Buyer selects a book(s) to purchase from search results.	2.1 Creates shopping cart. Adds selected item(s) to the shopping cart.
	3. Repeat steps 1 and 2 until all desired books are added to shopping cart.	
	4. Customer proceeds to checkout by selecting the Checkout button.	4.1 Displays list of shopping cart items for verification.
	4a. If customer wishes to remove an item, he/she selects the item to be removed and then selects the Delete button.	4a.1 Displays list of shopping cart items with deleted items removed.
	4b. If buyer wishes to add an item, he/she selects the Continue Shopping button and proceeds to steps 1 and 2.	
	5. Buyer verifies displayed information.	5. Displays buyer information.
	6. If information is incorrect or buyer account does not exist, buyer updates displayed information or enters new information into the registration form.	6.1 Updates new buyer information for existing account. 6.2 Creates account for new buyer and sends confirmation.
	7. Buyer confirms purchase.	7.1 Records order. 7.2 Sends e-mail to seller.
Exception Conditions:	1.1 If book is sold out, buyer cannot add book to shopping cart. 2.1 If buyer account does not exist, a new account must be created. If buyer is rejected for new account based on credit, system sends buyer notification. 3.1 If payment is rejected, system notifies buyer and seller.	

4. An SSD for each of the two use cases in question 3



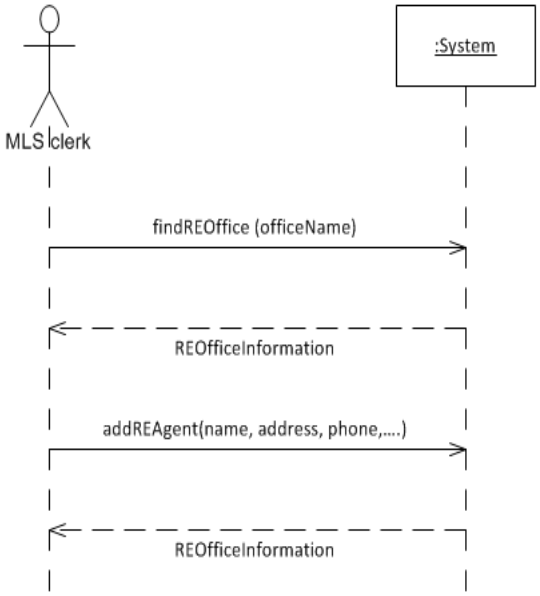
Running Cases: Community Board of Realtors

The Multiple Listing Service system has a number of use cases, which you identified in

Chapter 3, and three key domain classes, which you identified in Chapter 4: RealEstateOffice, Agent, and Listing.

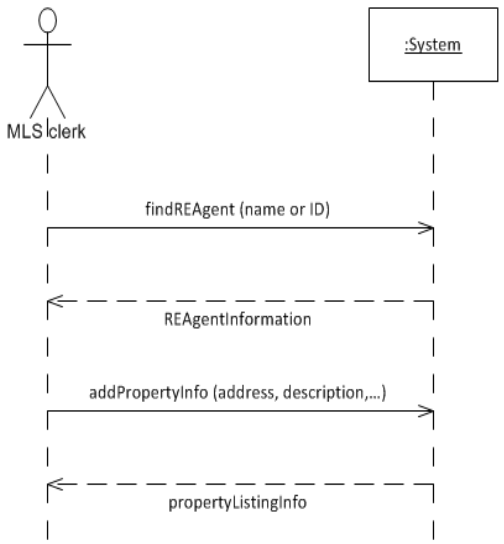
1. For the use case *Add agent to real estate office,* write a fully developed use case description and draw an SSD. Review the case materials in previous chapters and recall that the system will need to know which real estate office the agent works for before prompting for agent information.

Use case name:	Add agent to real estate office	
Scenario:	MLS clerk adding agent	
Triggering event:	New agent hired in a real estate office	
Brief description:	The correct real estate office is identified, and the new real estate agent information is entered into the system.	
Actors:	MLS clerk	
Related use cases:	Real estate office adds new agent (Web based version scenario)	
Stakeholders:	Real estate office, real estate agent	
Preconditions:	The real estate office must exist	
Postconditions:	Real estate agent is created and associated with real estate office	
Flow of activities	**Actor**	**System**
	1. Find correct real estate office 2. Enter new agent information	1.1 Display real estate office information 2.1 Create new agent record, including with relationship to real estate office
Exception conditions:	1.1 No real estate office found for requested id/name. Display not found message.	



2. For the use case *Create new listing*, write a fully developed use case description and draw an SSD. Recall that the system needs to know which agent made the listing before the system prompts for listing information.

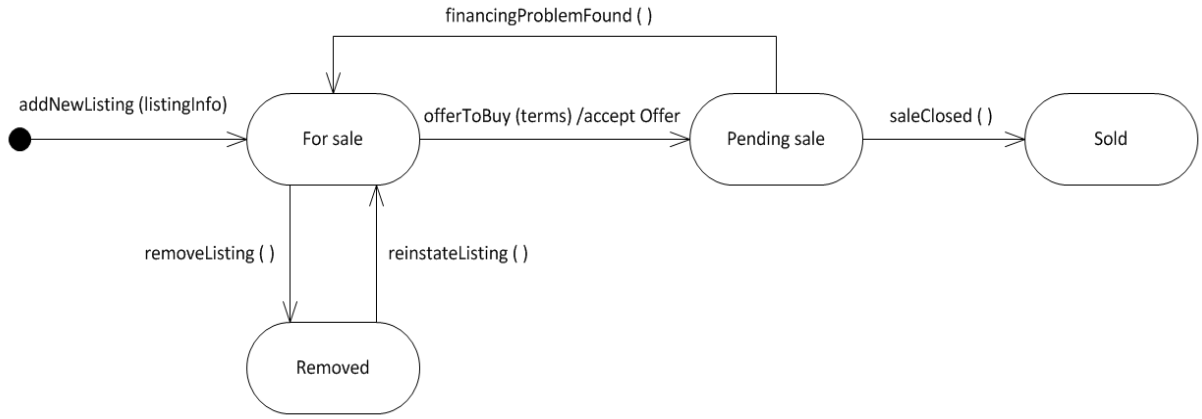
Use case name:	Create a new listing	
Scenario:	MLS clerk creates a new listing	
Triggering event:	New property is put up for sale	
Brief description:	The listing agent is identified and verified. The new property information is entered into the system, along with images etc.	
Actors:	MLS clerk	
Related use cases:	Real estate office/agent creates a new listing (Web version scenario)	
Stakeholders:	Real estate office, Real estate agent, Property owner	
Preconditions:	Real estate office must exist Real estate agent must exist	
Postconditions:	New listing must be created and associated with RE office and RE agent	
Flow of activities	**Actor**	**System**
	1. Find real estate agent 2. Enter new listing information	1.1 Display agent and office information 2. Create new property listing record, associated with agent. Display results.
Exception conditions:	1.1 Agent information not found. Display not found message.	



3. Draw a state machine diagram showing the states and transitions for a Listing object.

Note: Answers will vary because students will have to brainstorm valid states for a Listing object. In our solution we will use the following states and exit transitions. (The term “Sale” is used to mean either “Sale” or “Lease”)

State	**Exit transition**
For sale	Offer to buy
Sale pending	Close the sale
Sold	
Removed	Reinstate listing



Running Cases: The Spring Breaks 'R' Us Travel Service

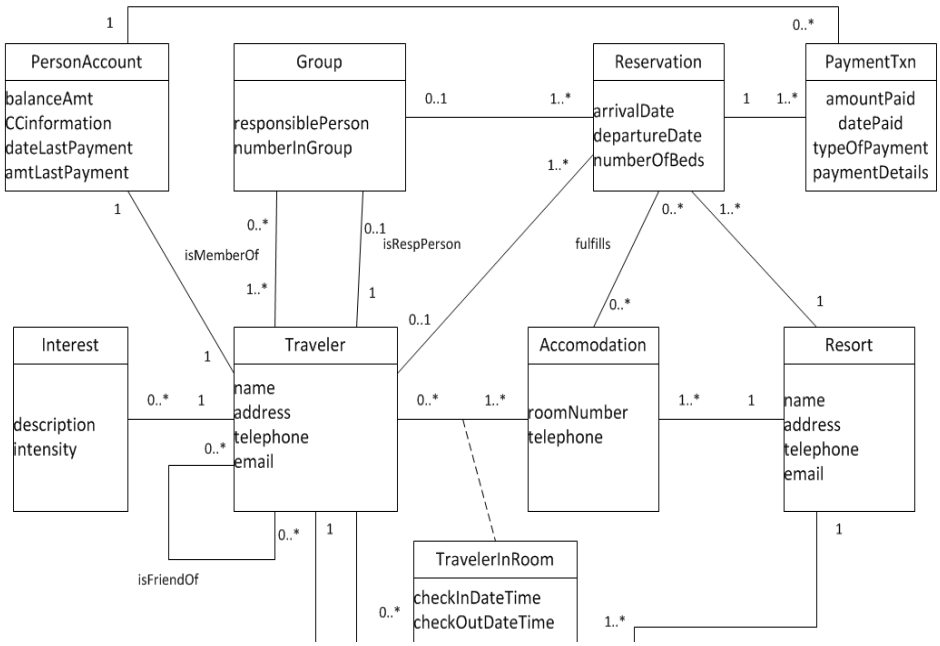
The Spring Breaks ‘R’ Us Travel Service system has many use cases and domain classes, which you identified in Chapters 3 and 4. Review the domain model class diagram to get a feel for the complexity of some of the use cases.

1. For the use case *Book a reservation*, write a fully developed use case description and draw an SSD. Review the classes that are associated with a reservation in the domain model to understand the flow of activities and repetition involved.

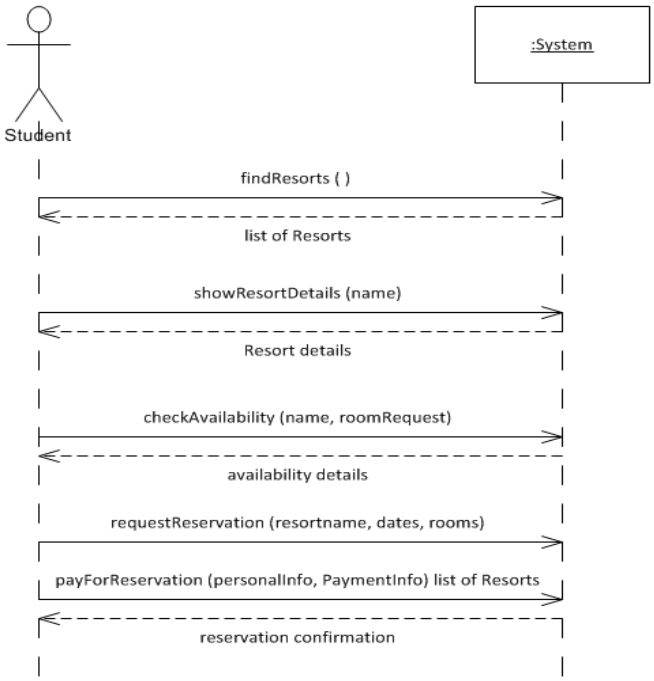
Use case name:	Book a reservation
Scenario:	Book a reservation online
Triggering event:	Student wants to make a reservation and initiates booking
Brief description:	Student searches or browses the resorts. He/she checks accommodations and availability. Then he/she makes a reservation for either a single person or a group. (Allow both individual and group reservations.)
Actors:	Student
Related use cases:	Create individual account (includes Traveler) Create group account Add person to group (new use case previously undefined)
Stakeholders:	Student, Resort

Preconditions:	Traveler and Individual account must exist Group must exist (for group reservation) Resort must exist	
Postconditions:	Reservation must be created and associated with Resort and Group/Traveler Payment must be created and associated with IndividualAccount	
Flow of activities	**Actor**	**System**
	1. Find a resort (search or browse) 2. Check availability of accommodations 3. Choose reservation type 4. Enter reservation details 5. Enter reservation payment information	1.1 Display resort and accommodation information 2.1 Display accommodation availability information 4.1 Make reservation 5.1 Verify individualInfo and paymentInfo Create PaymentTransaction for Reservation Display confirmation Send email confirmation
Exception conditions:	5.1 Payment transaction fails	

Note: Based on the requirements of this use case, and the previously defined use cases, the partial domain model class diagram created in Chapter 4 needs to be enhanced with several more classes: Group, Reservation, PaymentTxn, PersonalAccount. The class diagram for Chapter 4 only focused on the social networking part. See the partial class diagram below.



==========

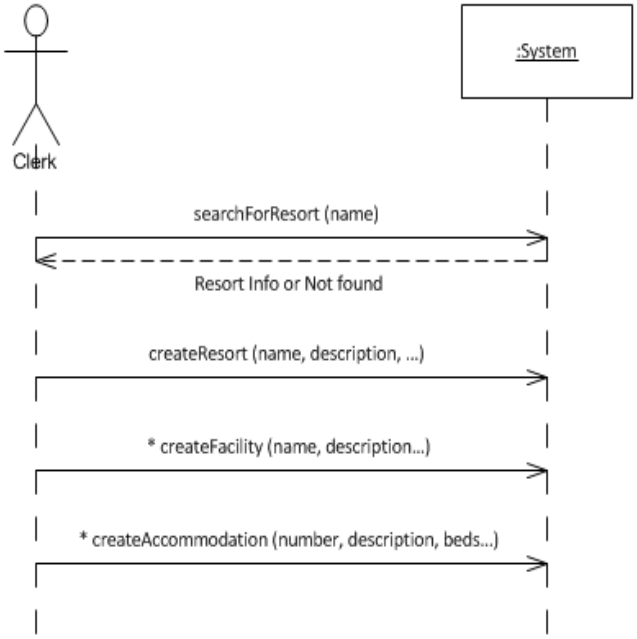


=======

2. For the use case *Add new resort*, write a fully developed use case description and draw an SSD. Review the classes that are associated with a resort in the domain model to understand the flow of activities and repetition involved.

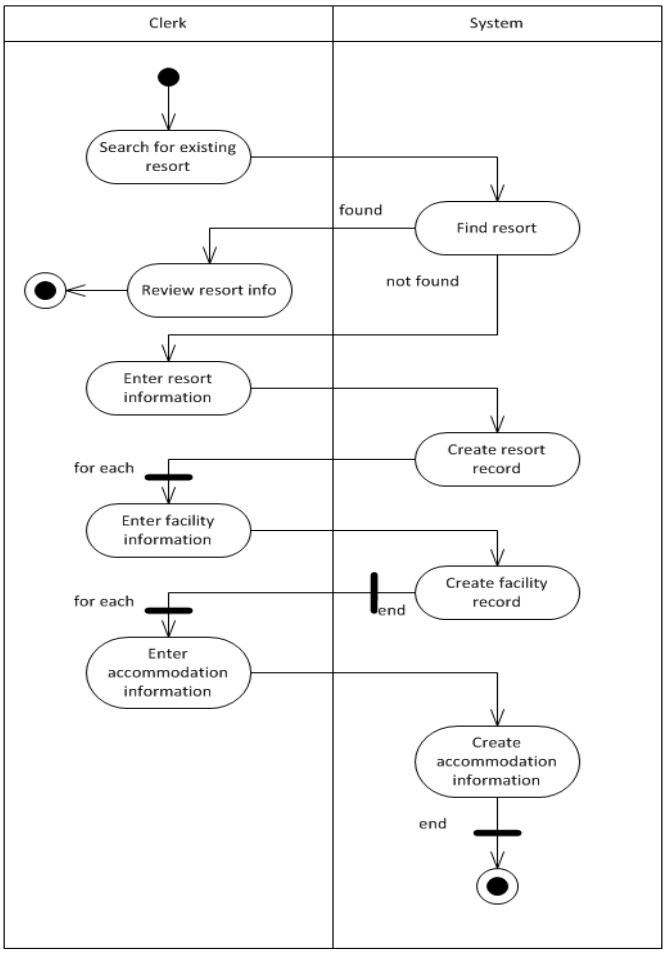
Use case name:	Add a new resort	
Scenario:	Add a new resort	
Triggering event:	A new resort contracts with SBRU to participate in the vacation program	
Brief description:	A new resort is added with descriptive information. Information about the accommodations available to this program are entered. Information about the facilities available for activities in this program are entered	
Actors:	SBRU clerk, Resort employee	
Related use cases:		
Stakeholders:	SBRU management, Resort management	
Preconditions:	Resort must not already exist	
Postconditions:	Resort is created Facilities are created and associated with the resort Accommodations are created for this resort	
Flow of activities	**Actor**	**System**
	1. Verify that the resort does not exist 2. Enter resort description	1.1 Check database for resort information 2.1 Create resort record

	3. (loop) Enter facilities information 4. (loop) Enter accommodations information	3.1 Create facilities record 4.1. Create accommodations record
Exception conditions:	1.1 Resort already exists	



3. Draw an activity diagram to show the flow of activities for the use case *Add a new resort*.

Note: This activity diagram is based on the updated problem domain model.



4. Draw a state machine diagram showing the state and transitions for a Reservation object.

State	Exit transition
Open	Fulfill reservation, cancel reservation
Fulfilled	
Canceled