Стандарт UML

Systems Analysis and Design

UML Overview

- What is UML?
  - is a family of graphical notations, that help in specifying and constructing the artifacts of systems.
  - Is a unification of:
    - Booch Method by booch
    - OMT Method by Rumbaugh
    - Objectory method by Jacobson

UML Overview (Cont.)

- Why Modeling?
  - A model helps the software team to simplify the system.
  - A model helps the software team to communicate.
  - Specify the structure and behavior of a system.
  - It serves as a road map for a developer.
  - Document the decisions you have made.

UML Overview (Cont.)

- Forward engineering:
  - Draws diagrams before write code
- Reverse Engineering:
  - Builds diagrams from existing code.
- Three Ways to Apply UML
  - UML as Sketch: incomplete diagrams, using paper or whiteboard.
  - UML as Blueprint: complete diagrams, using UML Tool.
  - UML as Programming Language: complete executable diagrams.

UML Overview (Cont.)

- UML Perspectives:
  - Conceptual Perspectives: diagrams in this perspective describe concepts of the domain (real-world)
  - Software perspective: diagrams in this perspective describe software abstractions
- The Meaning of "UML Class" in Different Perspectives
  - Class in the Domain Model (conceptual perspective) means conceptual class
  - Class in the Design Model (Software perspective) means Software class

UML Overview (Cont.)

- Modeling Types:
  - **Dynamic Modeling**: such as sequence diagram, help design the logic, the behavior of method bodies.
    - what messages to send, and to whom, and in what order.
  - **Static Modeling**: such as class diagram, help design the definition of the elements, and method signatures.

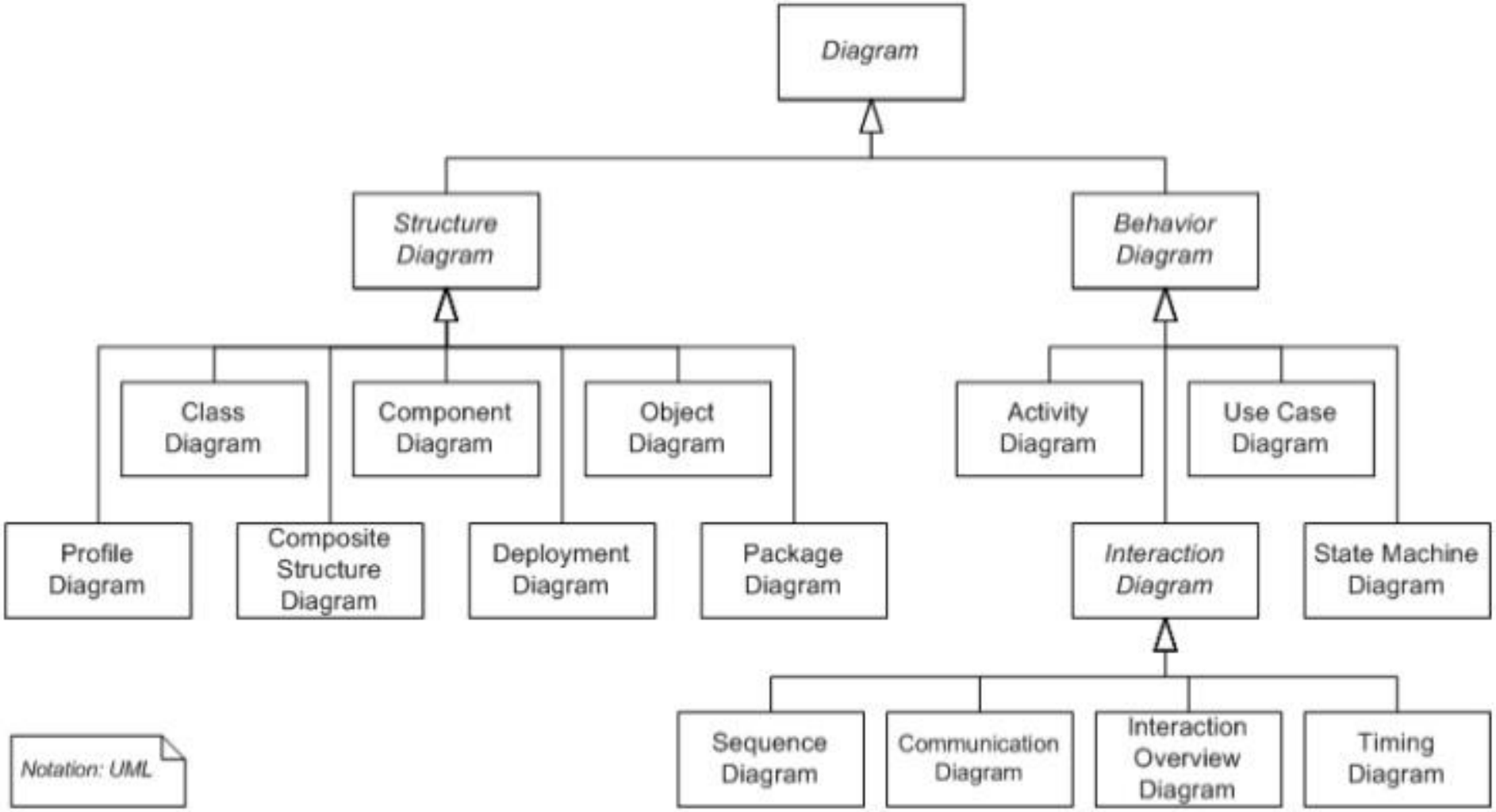


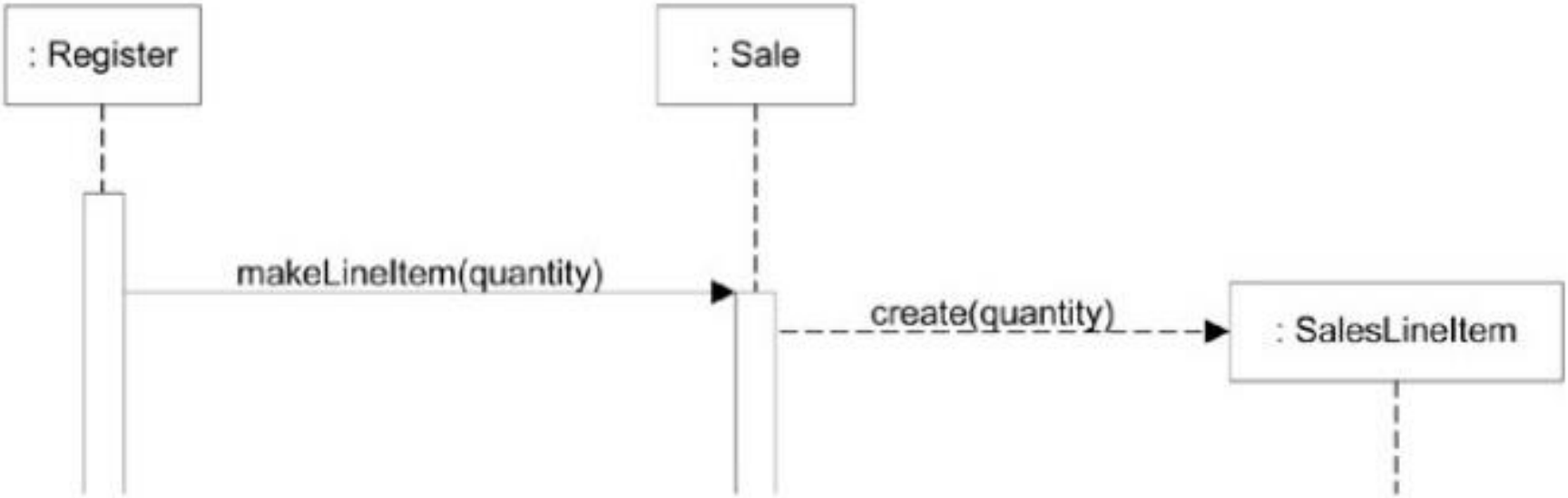
Diagram
Structure Diagram
Behavior Diagram
Class Diagram
Component Diagram
Object Diagram
Activity Diagram
Use Case Diagram
Profile Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Interaction Diagram
State Machine Diagram
Sequence Diagram
Communication Diagram
Interaction Overview Diagram
Timing Diagram
Notation: UML

UML Overview (Cont.)

- Dynamic Modeling is the most challenging, interesting, useful design work.
- Spend significant time doing interaction diagrams not just class diagrams.
- During dynamic modeling, we apply **responsibility-driven design**.

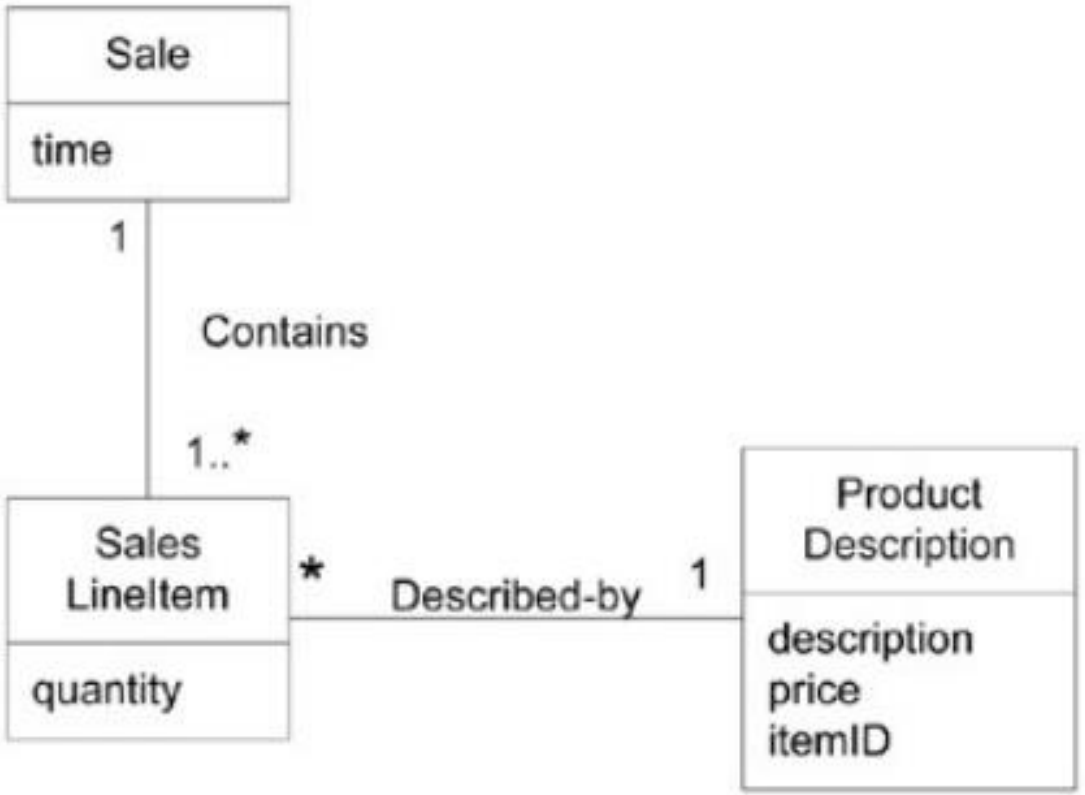
UML Overview (Cont.)

- Dynamic Modeling Example



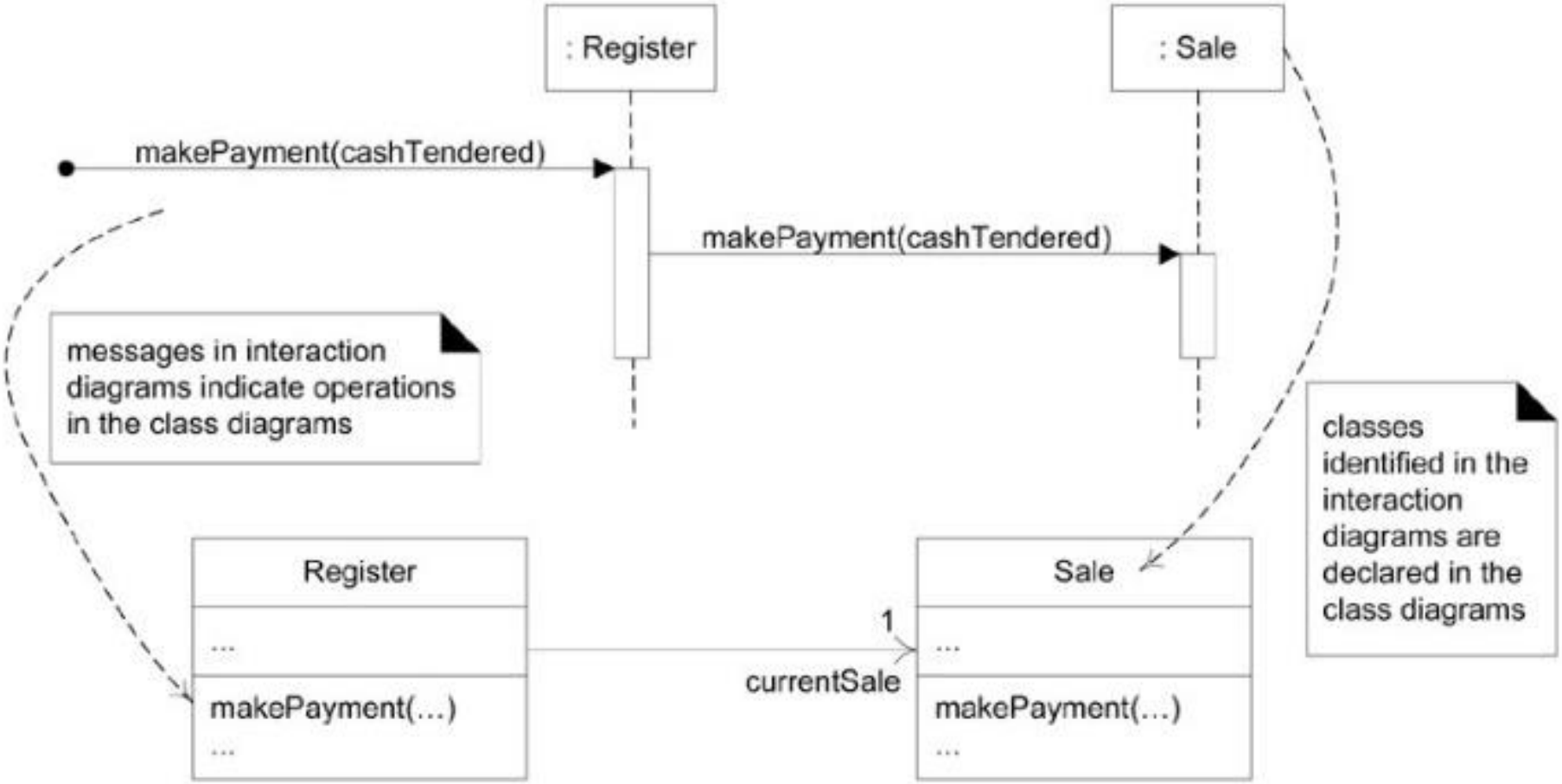
UML Overview (Cont.)

- Static Modeling Example



UML Overview (Cont.)

- The relationship between interaction diagram and class diagram



Requirements Overview (Cont.)

- Requirements Categorization
  - **Functionality**: features, capabilities
  - **Usability**: human factors, help, manual.
  - **Reliability**: frequency of failure, recoverability, predictability.
  - **Performance**: response times, throughput, availability.
  - **Supportability**: adaptability, maintainability.

Requirements Overview (Cont.)

- Requirements Analysis Artifacts (Outputs):
  - **Use-Case Model**: captures the functional requirements
  - **Supplementary Specification**: captures the non-functional requirements
  - **Glossary**: define noteworthy terms, validation rules, and acceptable values.

Use Case Overview (Cont.)

- Actor is something with behavior, such as a person (identified by role), computer system, or organization.
  - for example, a Student.

Use Case Overview (Cont.)

- **System behavior** is the result of the interactions between actors and the system (Clear Value).
  - Functionality Requirements is the set of system behaviors.
- For example:
  - In case of Auto Rental Company:
    - Reserve a Vehicle, Cancel a Reservation, View Customer Profile.
  - In case of Point of Sale System:
    - Process Sale, Handle Cash Payment, Handle Returns.
  - In case of Payroll System:
    - Create Employee, Maintain Timecard, Run Payroll.
  - In case of Course Registration System:
    - Register for Course, Maintain Student Information, Submit Grades.

Use Case Overview (Cont.)

- **Scenario:** is a sequence of actions and interactions between actors and the system.
- Different sequences of actions (scenarios) can happen, depending on the requests made and conditions surrounding the requests.

Use Case Overview (Cont.)

- **Use Case** is a set of scenarios that yields a system behavior.
- Use Cases are functional requirements that indicate what the system will do.
- Nothing object-oriented about use cases.

Use Case Overview (Cont.)

- Example:

Process Sale:

3. A customer arrives at a checkout with items to purchase.
4. The cashier starts a new sale.
5. The cashier enters item identifier.
6. The system records line item and presents a running total and line-item details.
7. Cashier repeats steps 3-4 until indicates done.
8. The customer enters payment information, which the system validates and records.
9. The system updates inventory.
10. The customer receives a receipt from the system and then leaves with the items.

Use Case Overview (Cont.)

- Use Case Flow of Events:
  - Has one normal basic flow of actions (happy scenario)
  - Several alternative flows of actions
- Each flow of actions called scenario
- Example:
- Basic Flow: Successfully purchasing items.
- Alternative Flow: Failing to purchase items because of a credit payment denial

Use Case Overview (Cont.)

- Why Use Cases:
  - Because they are simple, it possible for domain expert to participate in writing them.
  - Use cases can be a basis for writing the system user guide.
  - Use cases emphasize the user goals and perspective.
    - Who is using the system?
    - What are their goals?

Use Case Overview (Cont.)

- Use Case Model
  - is the set of all written use cases.
- Use Case Model may also include:
  - Use Case Diagram
  - Activity Diagrams
  - System Sequence Diagrams

Use Case Overview (Cont.)

- Use Case Examples:
  - Example 1: Auto Rental System - Reserve a Vehicle
  2. This use case begins when a customer indicates he wishes to make a reservation for a rental car.
  3. The system prompts the customer for the pickup and return locations of the reservation, as well as the pickup and return dates and times. The customer indicates the desired locations and dates.
  4. The system prompts for the type of vehicle the customer desires. The customer indicates the vehicle type.

Use Case Overview (Cont.)

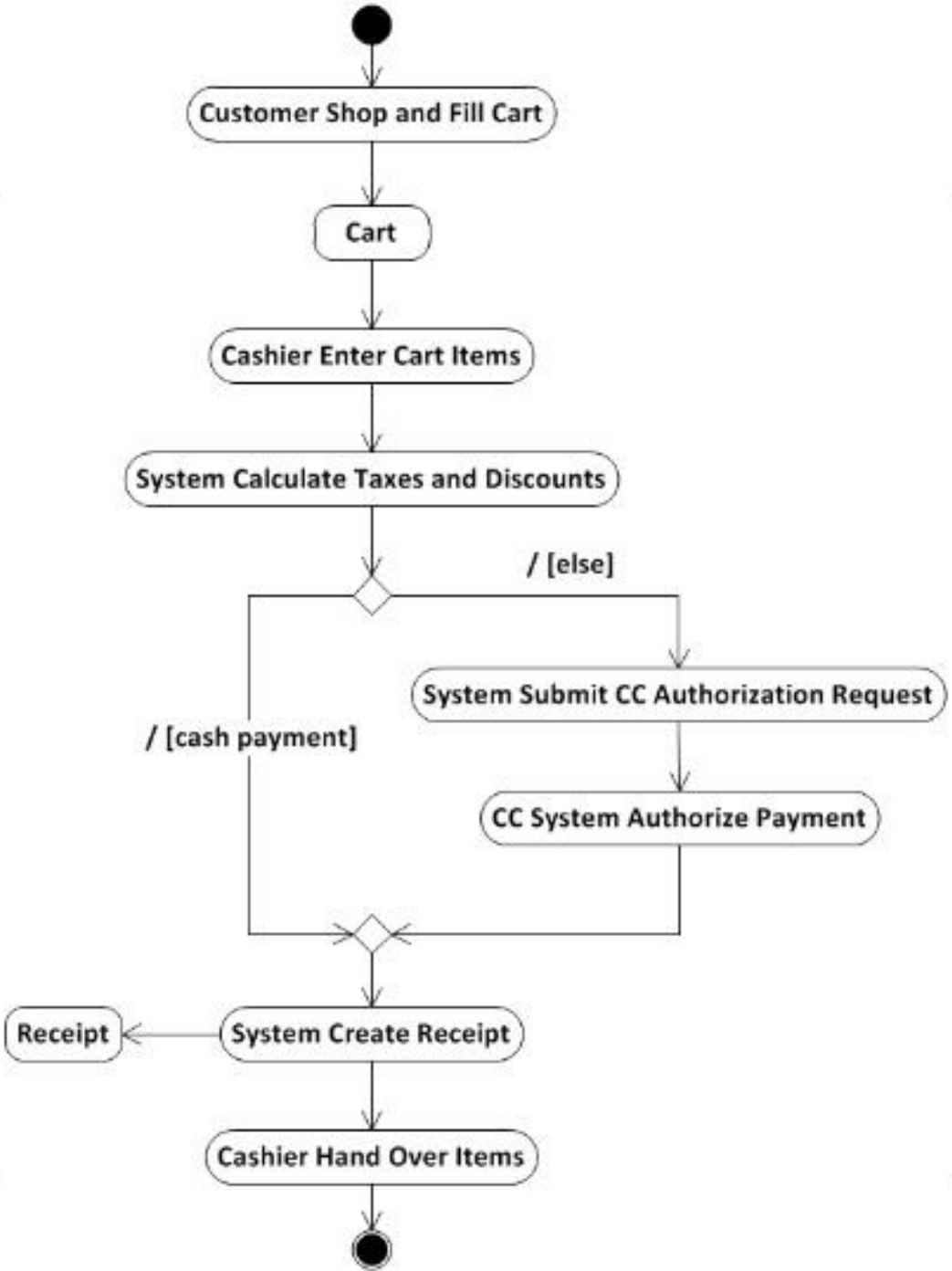
1. The system presents all matching vehicles available at the pickup location for the selected date and time. If the customer requests detail information on a particular vehicle, the system presents this information to the customer.
2. If the customer selects a vehicle for rental, the system prompts for information identifying the customer (full name, telephone number, email address for confirmation, etc.). The customer provides the required information.
3. The system presents information on protection products (such as damage waiver, personal accident insurance) and asks the customer to accept or decline each product. The customer indicates his choices.

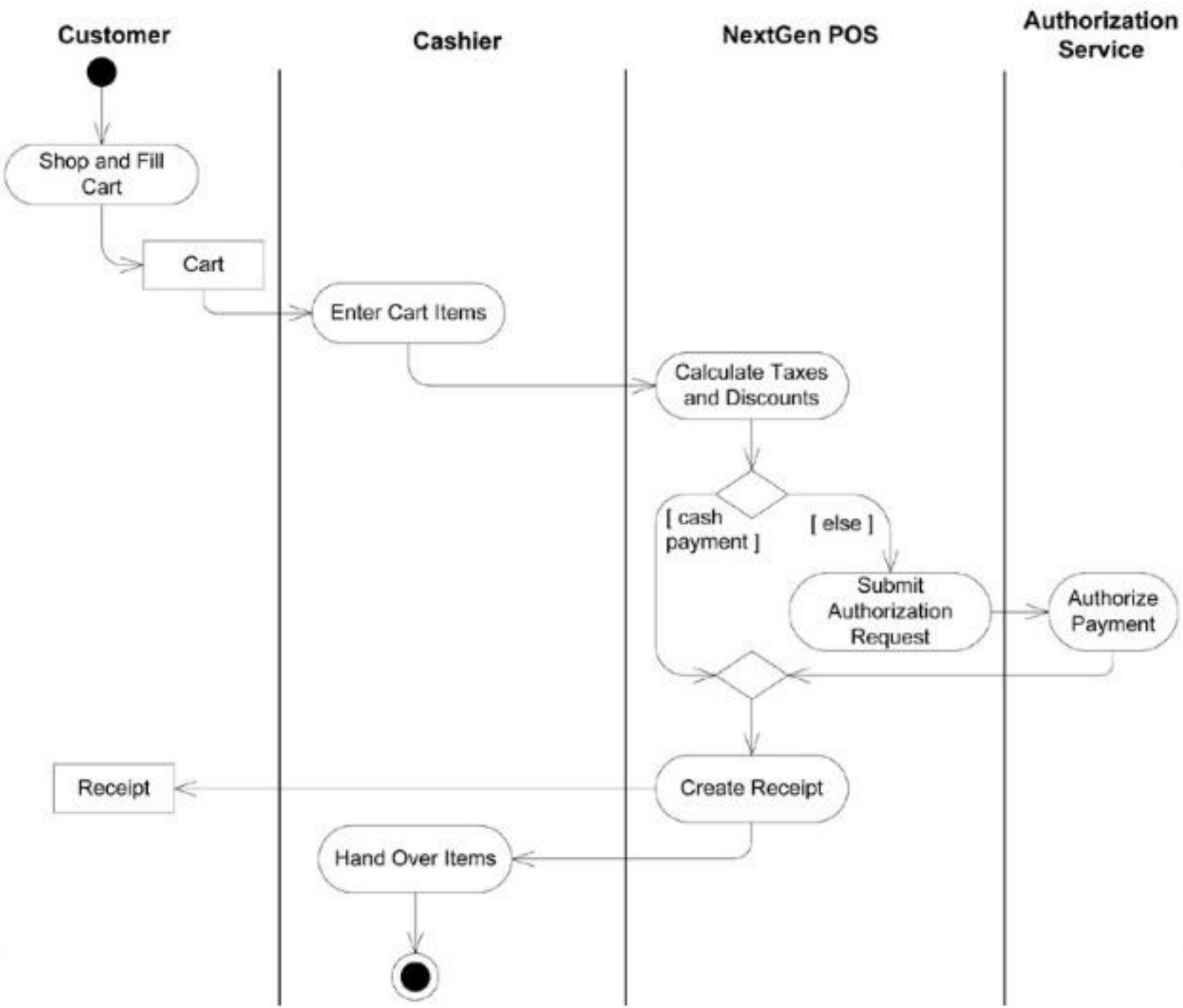
Use Case Overview (Cont.)

1. If the customer indicates "accept reservation," the system informs the customer that the reservation has been completed, and presents the customer a reservation confirmation.
2. This use case ends when the reservation confirmation has been presented to the customer.

Use Case Overview (Cont.)

- Applying UML:
  - Review: Use case is a set of **interactions** and **activities**.
  - For use cases that describe complex flow of actions, **Activity Diagram** can be applied to visualize the use case flow.
  - For use cases that describe complex interactions with the actor, **System Sequence Diagram** (SSD) can be applied to visualize the system operations.


Customer Shop and Fill Cart
Cart
Cashier Enter Cart Items
System Calculate Taxes and Discounts
/ [else]
System Submit CC Authorization Request
/ [cash payment]
CC System Authorize Payment
Receipt
System Create Receipt
Cashier Hand Over Items


Customer
Cashier
NextGen POS
Authorization Service
Shop and Fill Cart
Cart
Enter Cart Items
Calculate Taxes and Discounts
[cash payment]
[else]
Submit Authorization Request
Authorize Payment
Receipt
Create Receipt
Hand Over Items

Use Case Overview (Cont.)

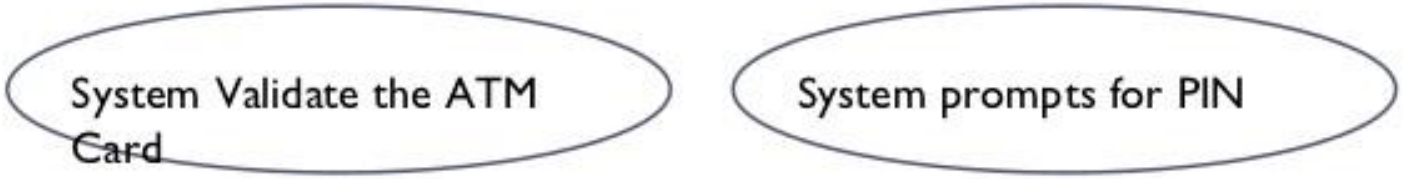
- Applying UML:
  - The following shows an example on how to build an **Activity Diagram** from Use Case (ATM Machine – Withdraw Cash Use Case)

Use Case Overview (Cont.)

- ATM Machine – Withdraw Cash Use Case – Main Scenario

2. The use case begins when the customer inserts his ATM card into the ATM machine.

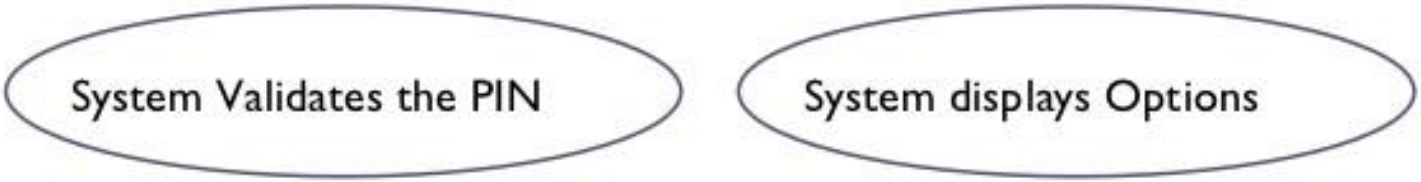
6. The system validates the ATM card and prompts the customer to enter PIN code.



Use Case Overview (Cont.)

1. The customer enters the PIN.

1. The system validates the PIN entered and displays the Options Menu Screen.

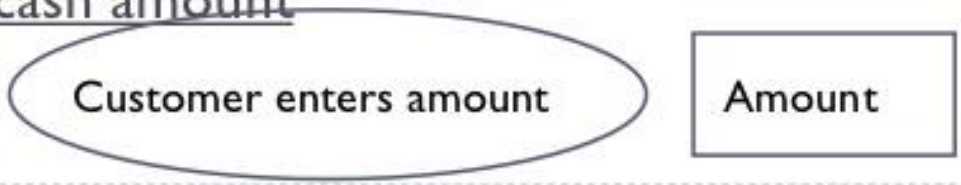


Use Case Overview (Cont.)

- The customer selects the 'Cash Withdrawal' option from the Options Menu.

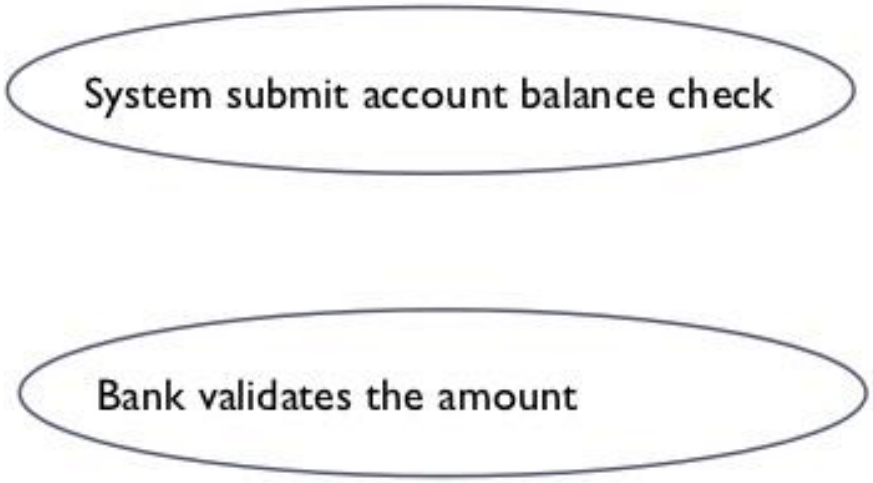
- The system prompts the customer to enter the amount of cash.

The customer enters a cash amount



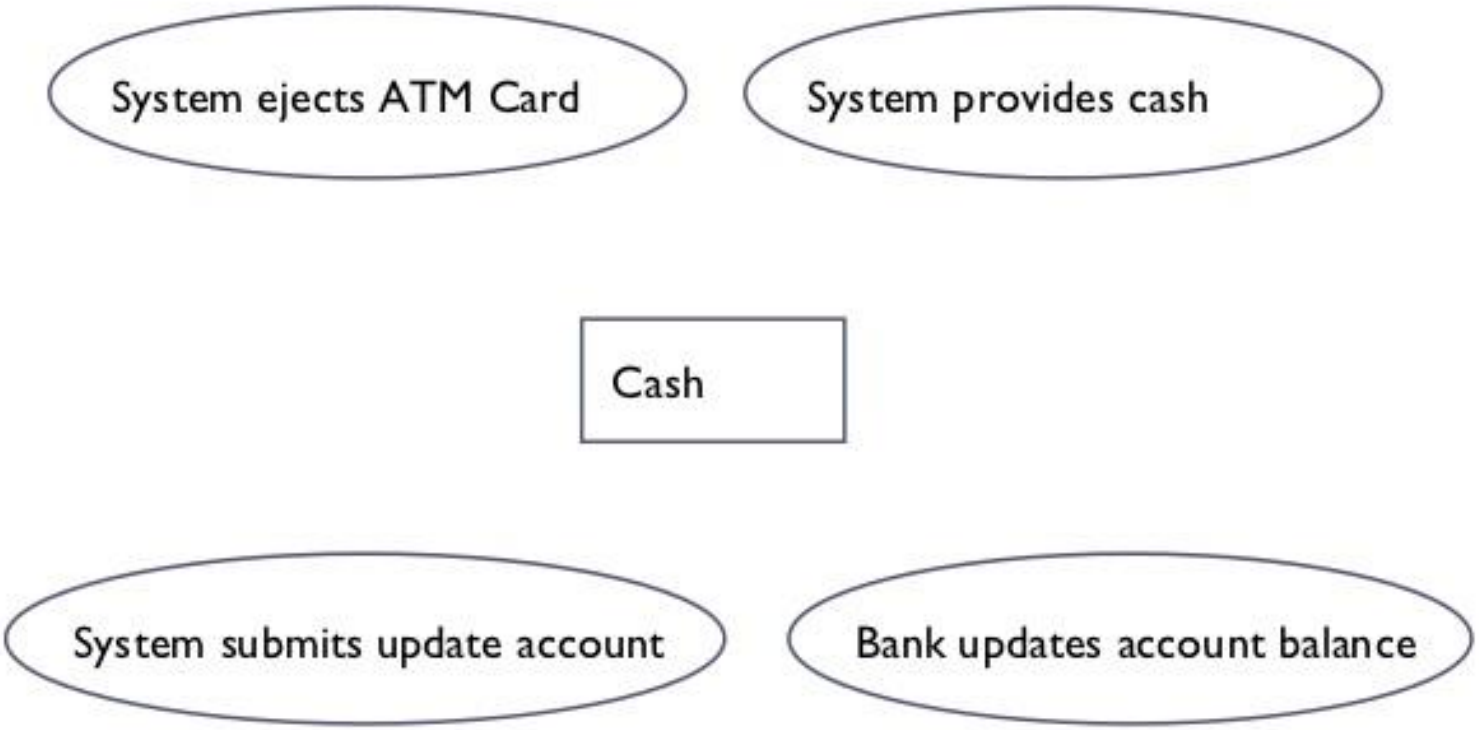
Use Case Overview (Cont.)

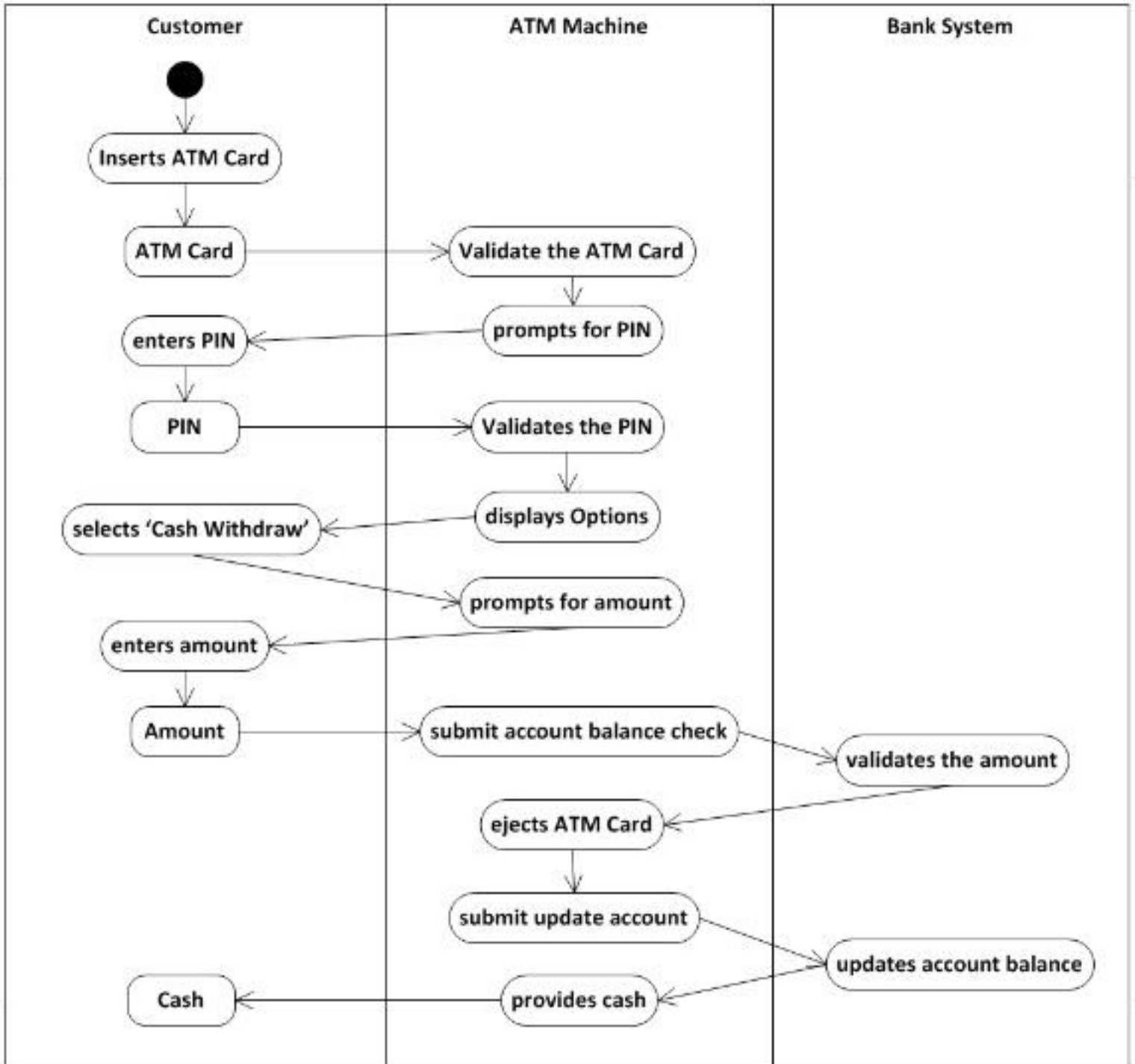
1. The system validates the amount entered; asks the bank system to check the account balance.



Use Case Overview (Cont.)

1. The system ejects the ATM card, provides the cash and requests to update the account balance of the customer in the bank system.




Customer
ATM Machine
Bank System
Inserts ATM Card
ATM Card
Validate the ATM Card
prompts for PIN
enters PIN
PIN
Validates the PIN
displays Options
selects 'Cash Withdraw'
prompts for amount
enters amount
Amount
submit account balance check
validates the amount
ejects ATM Card
submit update account
updates account balance
Cash
provides cash

Use Case Overview (Cont.)

- Exercise:
  - The previous activity diagram visualizes the main path of the use case; update the activity diagram to visualize the alternative paths.

Use Case Overview (Cont.)

- System Sequence Diagram, for a particular use case, shows:
  1. The external actors that interact with the system.
  2. The system as a black box.
  3. The system events that the actors generate (Interaction).

Use Case Overview (Cont.)

Cash only Process Sale Use case:

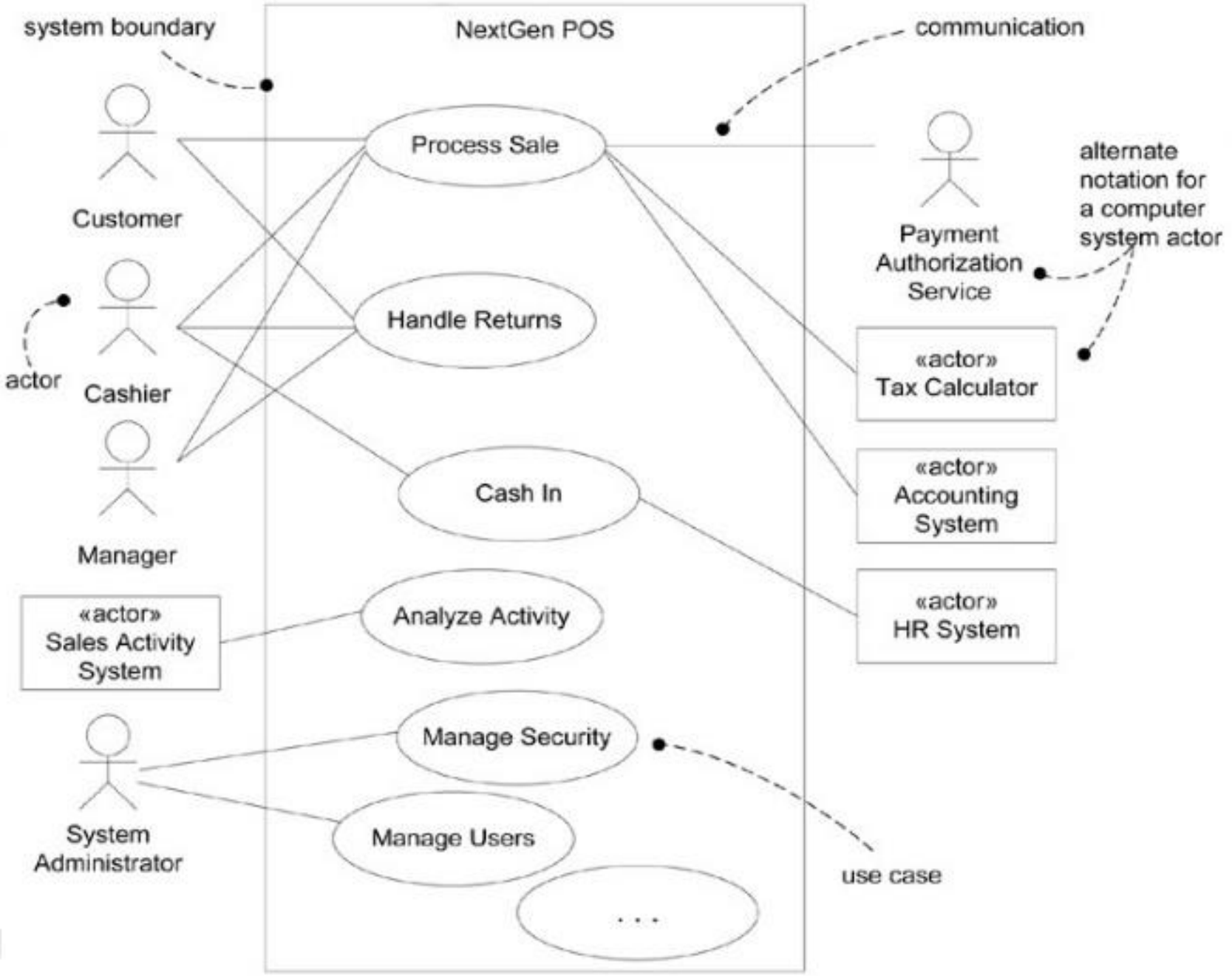
1. A customer arrives at a checkout with items to purchase.
2. The cashier starts a new sale.
3. The cashier enters item identifier.
4. The system records line item and presents a running total and line-item details.
5. Cashier repeats steps 3-4 until indicates done.
6. The customer enters payment information, which the system validates and records.
7. The system updates inventory.
8. The customer receives change due and receipt from the system and then leaves with the items.

Use Case Overview (Cont.)

- Exercise:
  - Select one of the following use cases and visualize its interactions with the actors using System Sequence Diagram (SSD);
    - Auto Rental System - Reserve a Vehicle
    - Payroll System - Run Payroll
    - Flight Information System – Request Available Flights
    - ATM Machine – Withdraw Cash

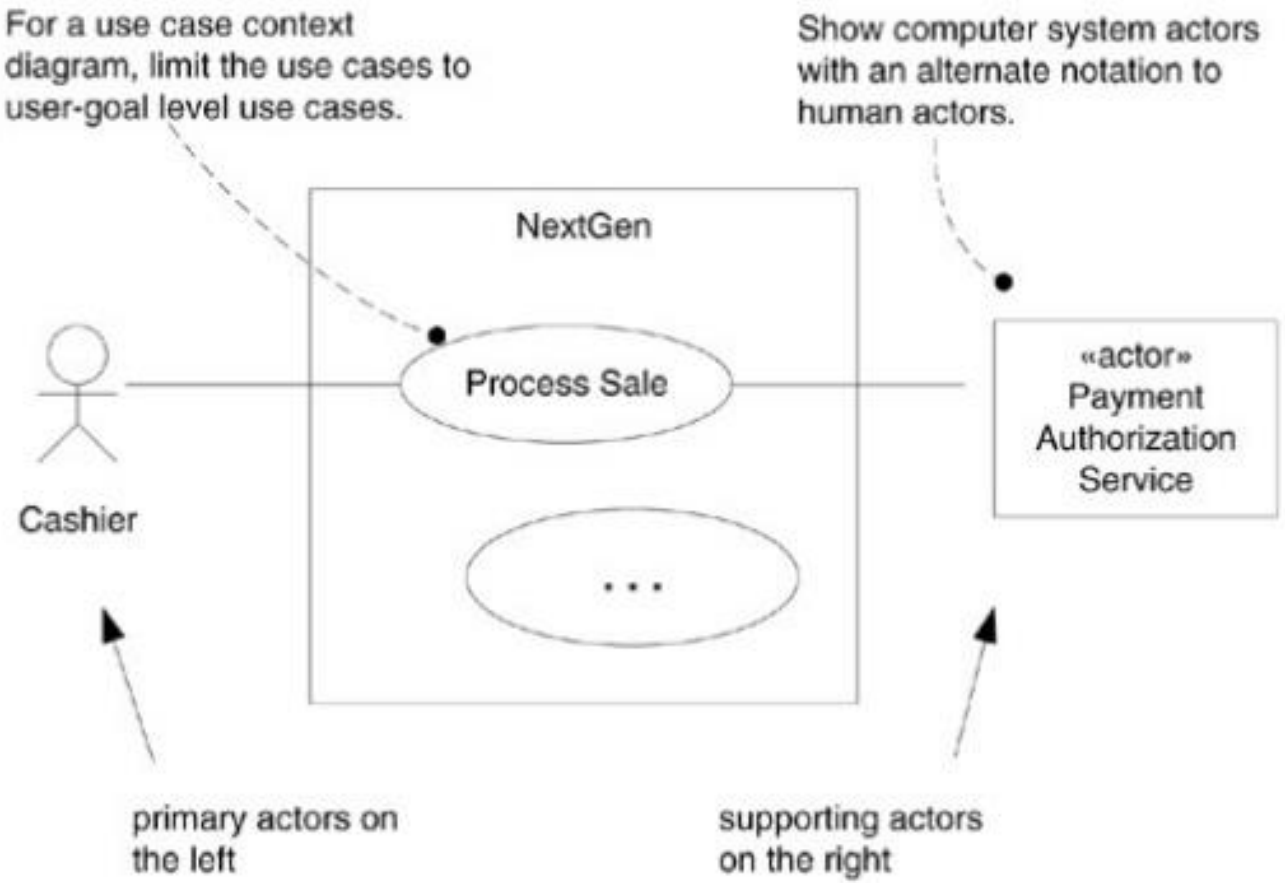
Use Case Overview (Cont.)

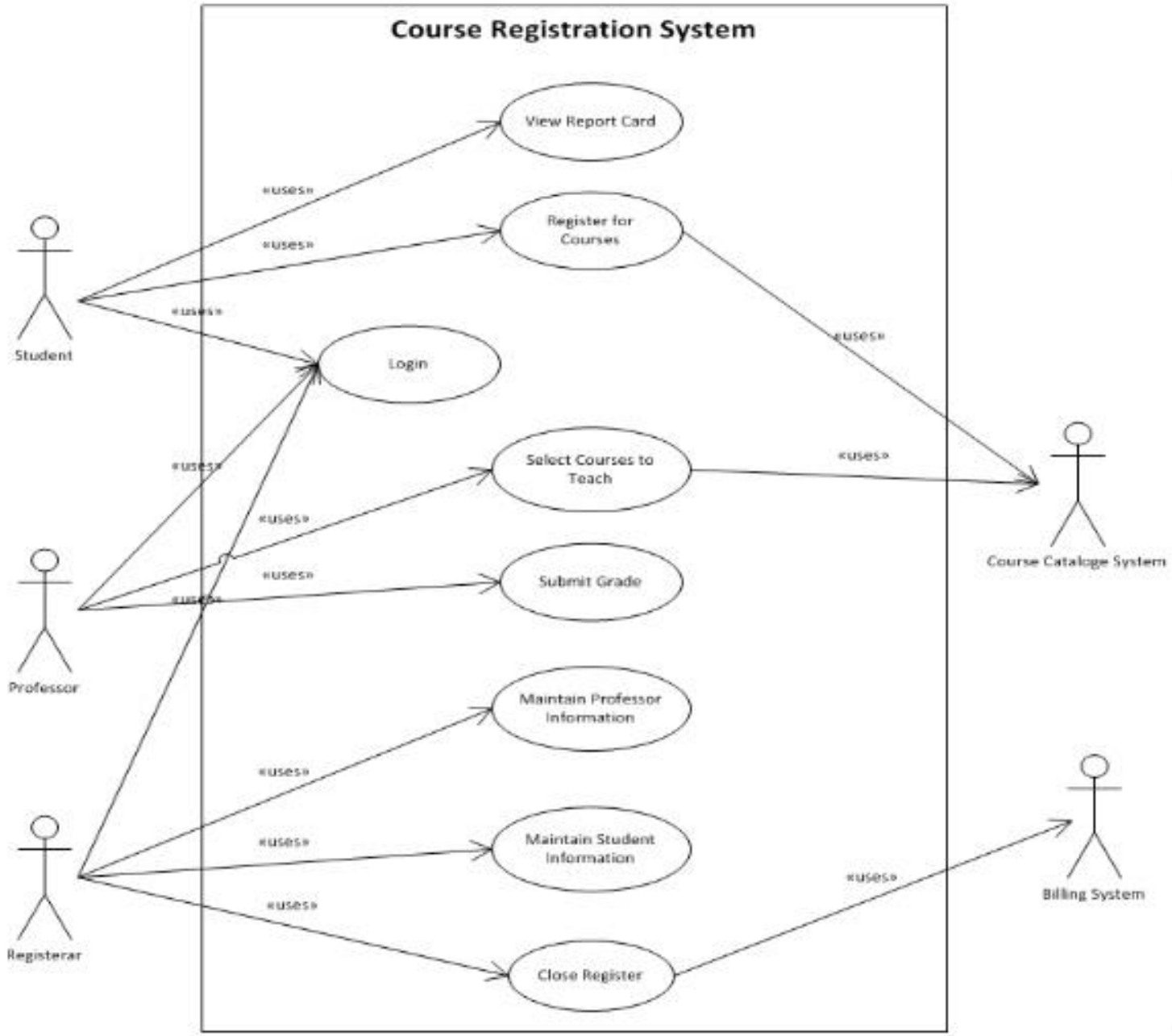
- Use Case Diagram:
  - The UML provides use case diagram notation to illustrate
    1. The system boundaries.
    2. The names of use cases.
    3. The name of actors.
    4. The relationships between the use cases and actors.


system boundary
NextGen POS
communication
Process Sale
Customer
Payment Authorization Service
alternate notation for a computer system actor
Handle Returns
actor
Cashier
«actor» Tax Calculator
Cash In
«actor» Accounting System
Manager
«actor» HR System
Analyze Activity
«actor» Sales Activity System
Manage Security
System Administrator
Manage Users
use case
. . .

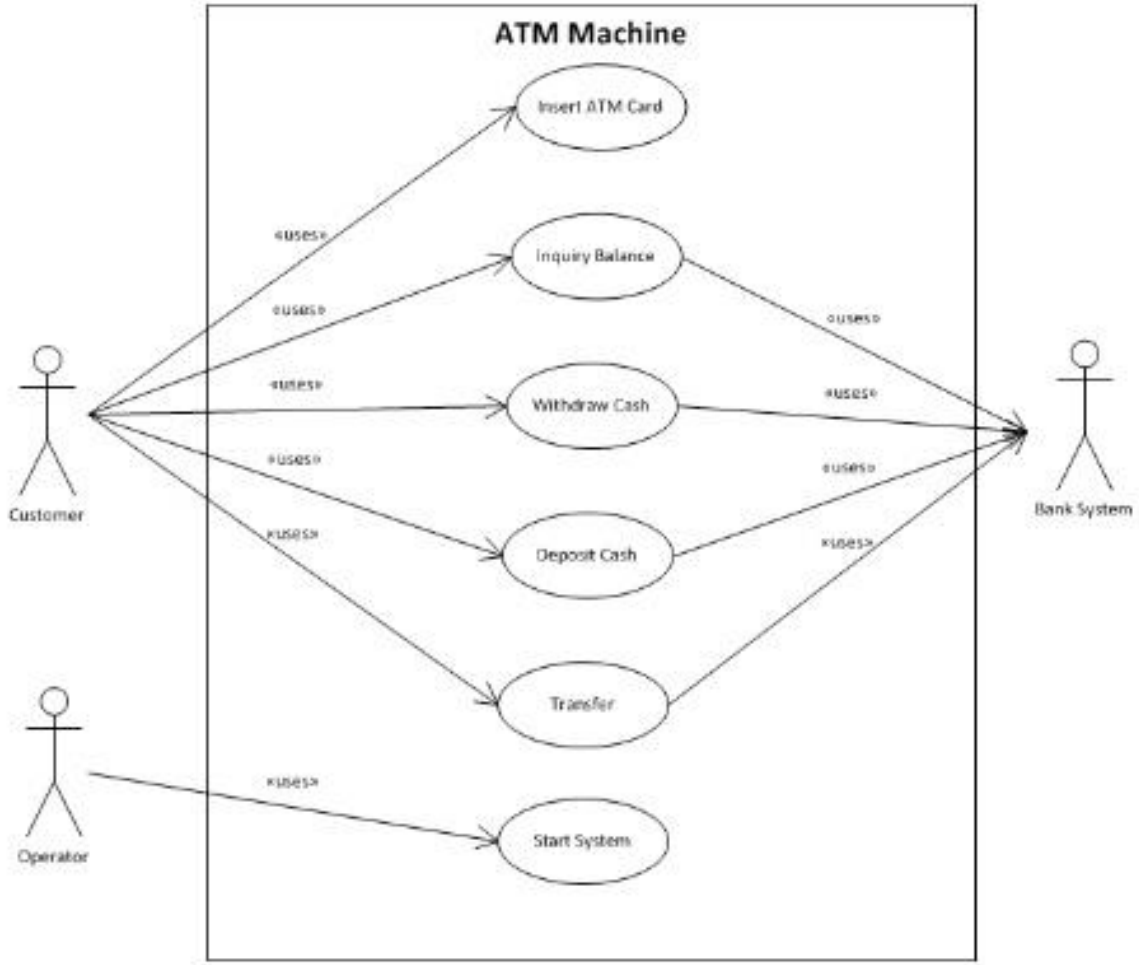
Use Case Overview (Cont.)

- Use Case Diagram Guidelines:

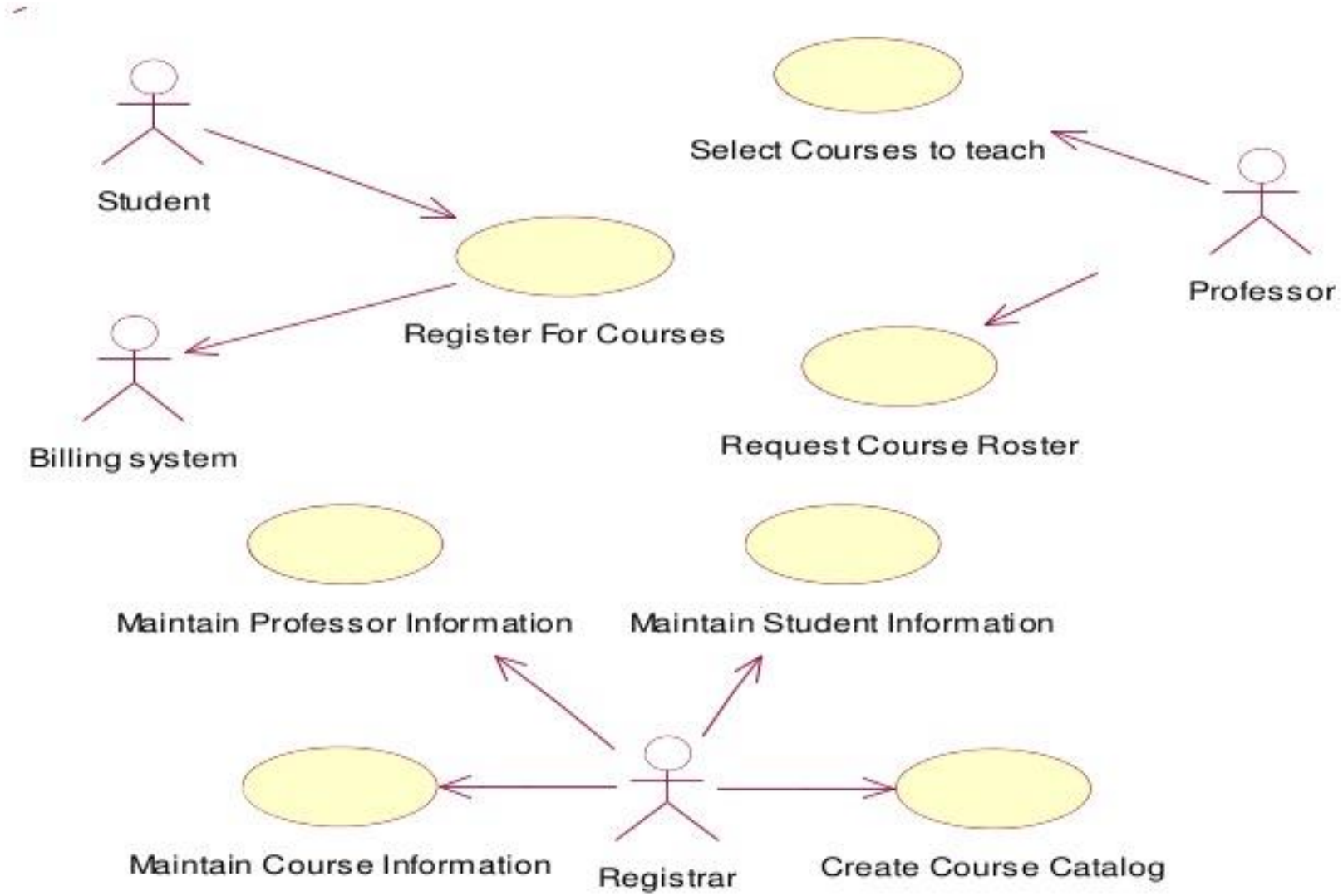



Course Registration System
View Report Card
Register for Courses
Login
Select Courses to Teach
Submit Grade
Maintain Professor Information
Maintain Student Information
Close Register
Student
Professor
Registerar
Course Catalogs System
Billing System
«uses»

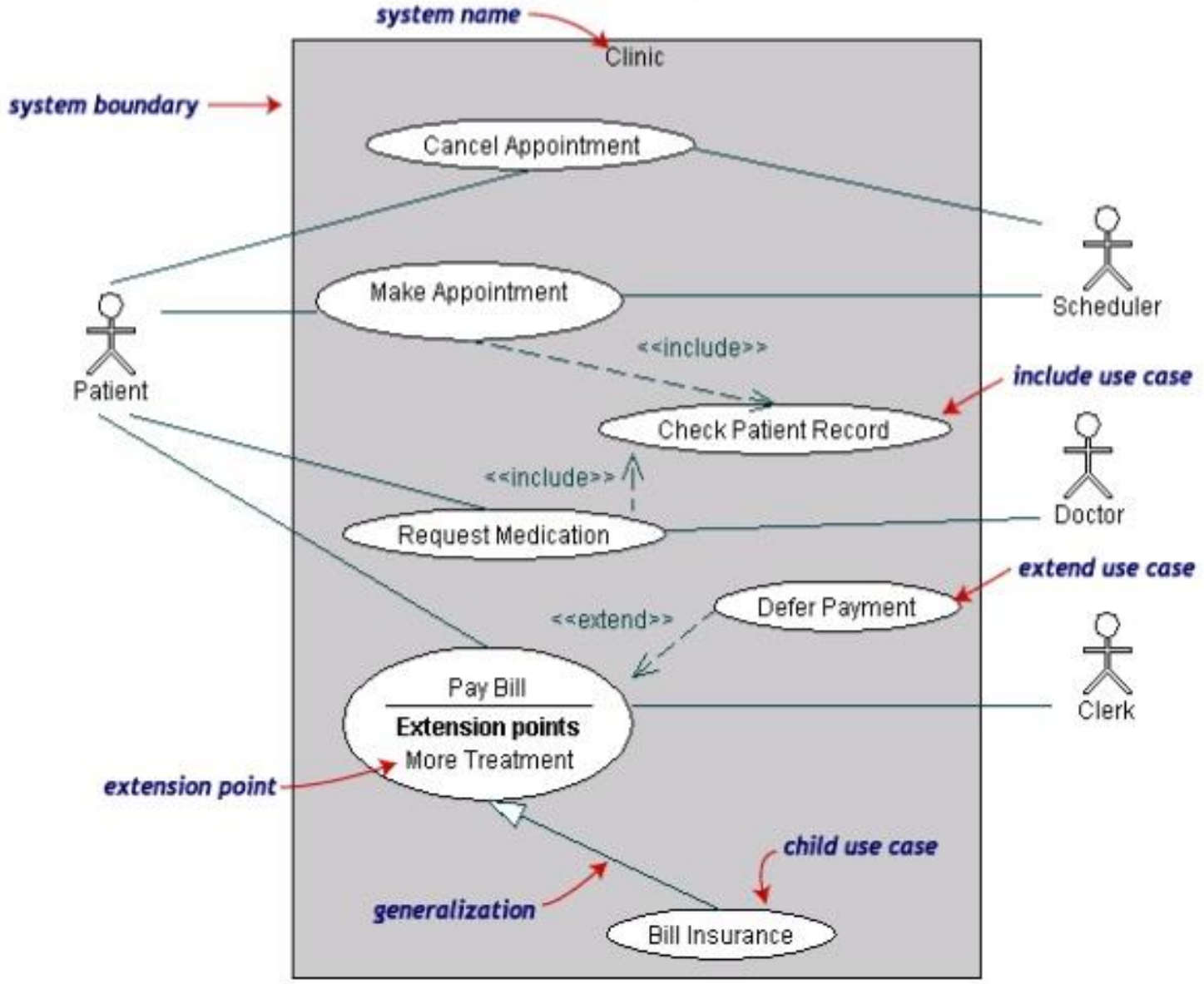
Use Case Overview (Cont.)



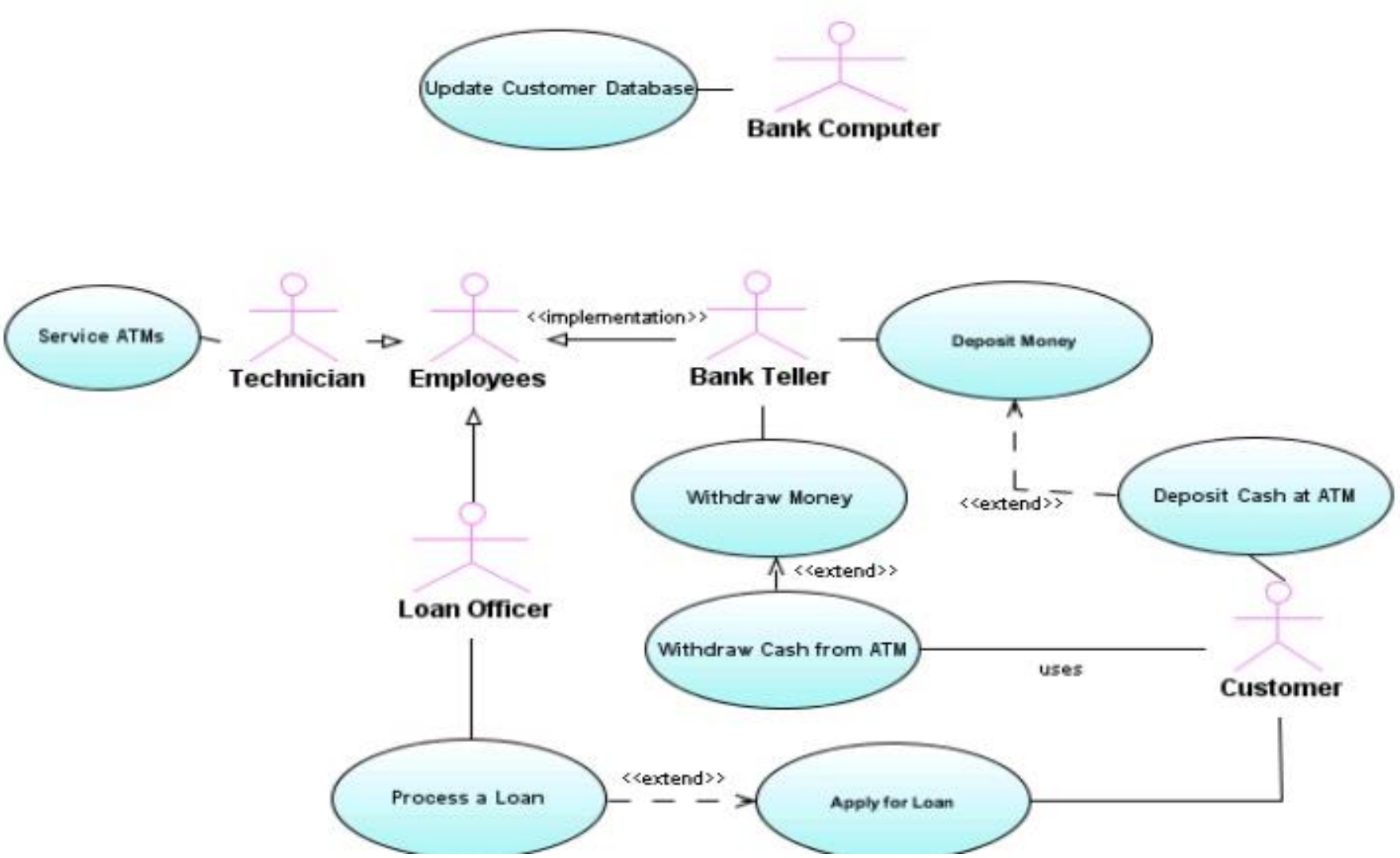
Use Case Diagram in the ESU Course Registration System



Use Case: Clinic System Example

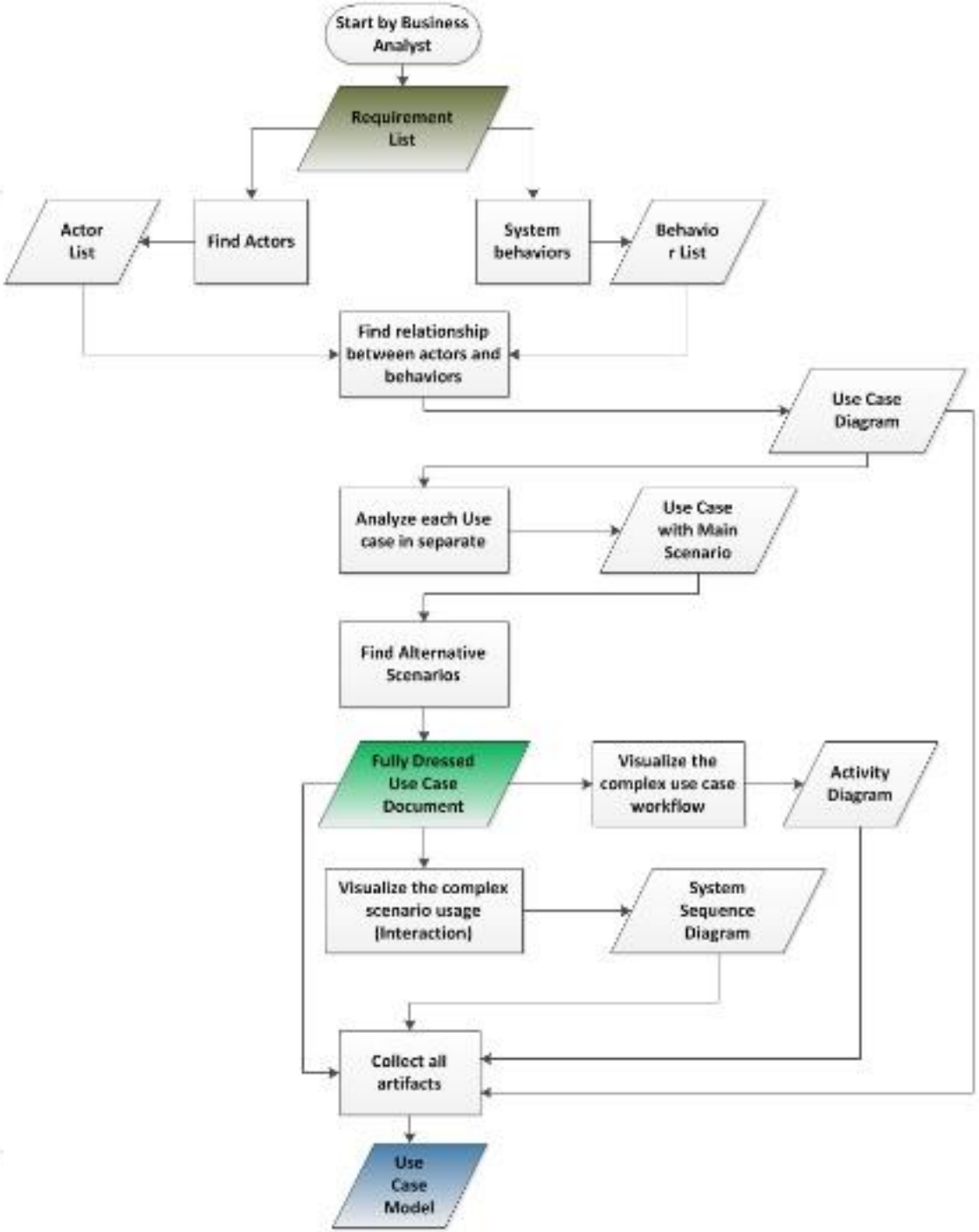


Use Case: Bank System Example



Use Case Overview (Cont.)

- Use Case Modeling:
  - g Starts by requirement list (Input).
  - e Find system behaviors list
  - v Find actors
  - v Draw **use case diagram**
  - r For each behavior **write a use case**
  - A For complex use cases draw at least one **activity diagram**
  - E For complex use cases draw at least one **SSD**
  - c Then collect (Use case diagram, use cases, activity diagrams and SSDs) in one **Use-Case Model** (Output).


Start by Business Analyst
Requirement List
Actor List
Find Actors
System behaviors
Behavio r List
Find relationship between actors and behaviors
Use Case Diagram
Analyze each Use case in separate
Use Case with Main Scenario
Find Alternative Scenarios
Fully Dressed Use Case Document
Visualize the complex use case workflow
Activity Diagram
Visualize the complex scenario usage (Interaction)
System Sequence Diagram
Collect all artifacts
Use Case Model