Interaction Modeling

Software Modeling and Design
Shehenaz Shaik

OO Models

- Class Model
  - Static structure of objects and relationships
  - Class diagram
- State Model
  - Changes over time or on events
  - State diagram
- Interaction Model
  - Interaction among objects
  - Use case diagram
  - Sequence diagram
  - Activity diagram

Interaction Model

- Describes how objects interact to produce useful results
- Holistic view of behavior across objects
  - State model – Reductionist view (each object)
- Various levels of abstraction
  - Use cases
  - Sequence diagrams
  - Activity diagrams

Interaction Model

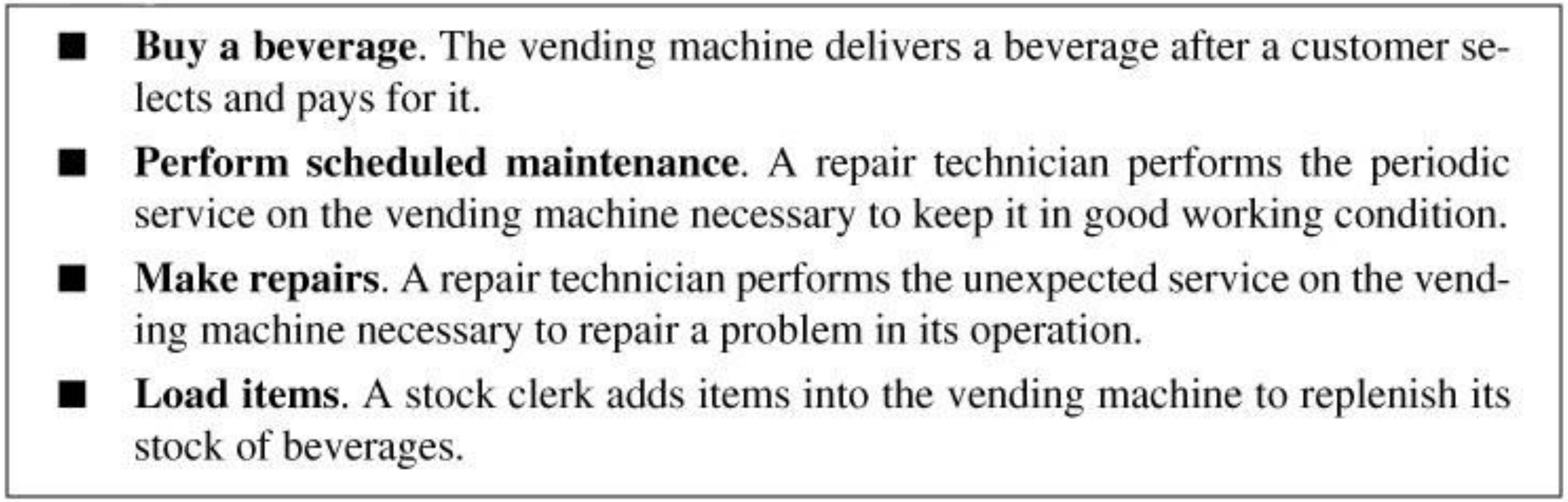
- Use cases
- Sequence diagrams
- Activity diagrams

Actor

- Direct external user of system
- Has specific behavior towards system
- Has a single well-defined purpose
  - Object with different facets of behavior → Multiple actors
  - Objects of different classes that interact similarly → Single actor

Use case

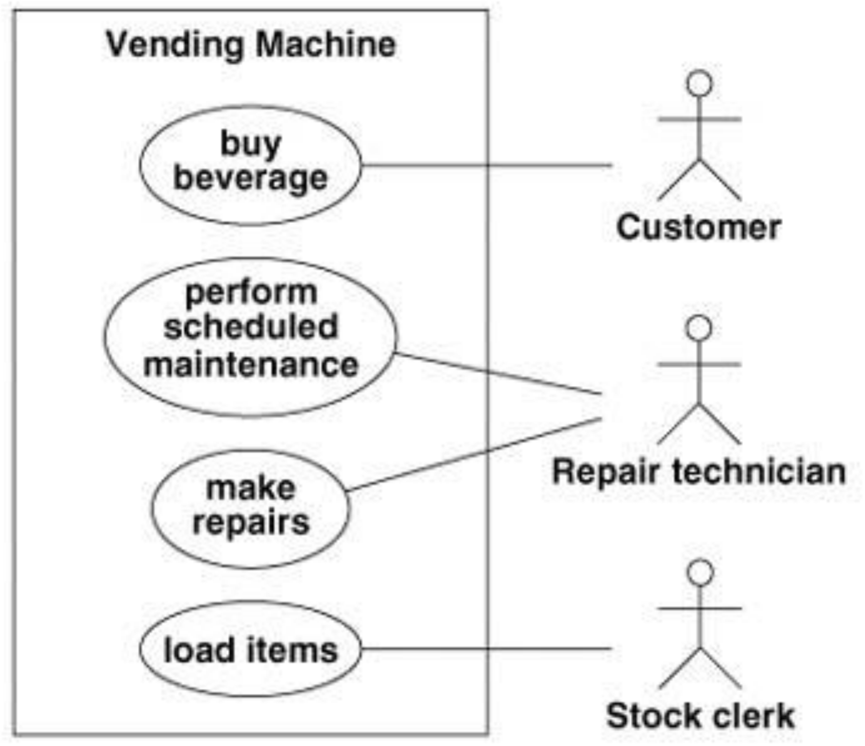
- Functionality provided by system
- Includes all relevant behavior
- Sequence of messages between system and actors
- Use Cases Vs. Traditional Requirements Lists (TRLs)



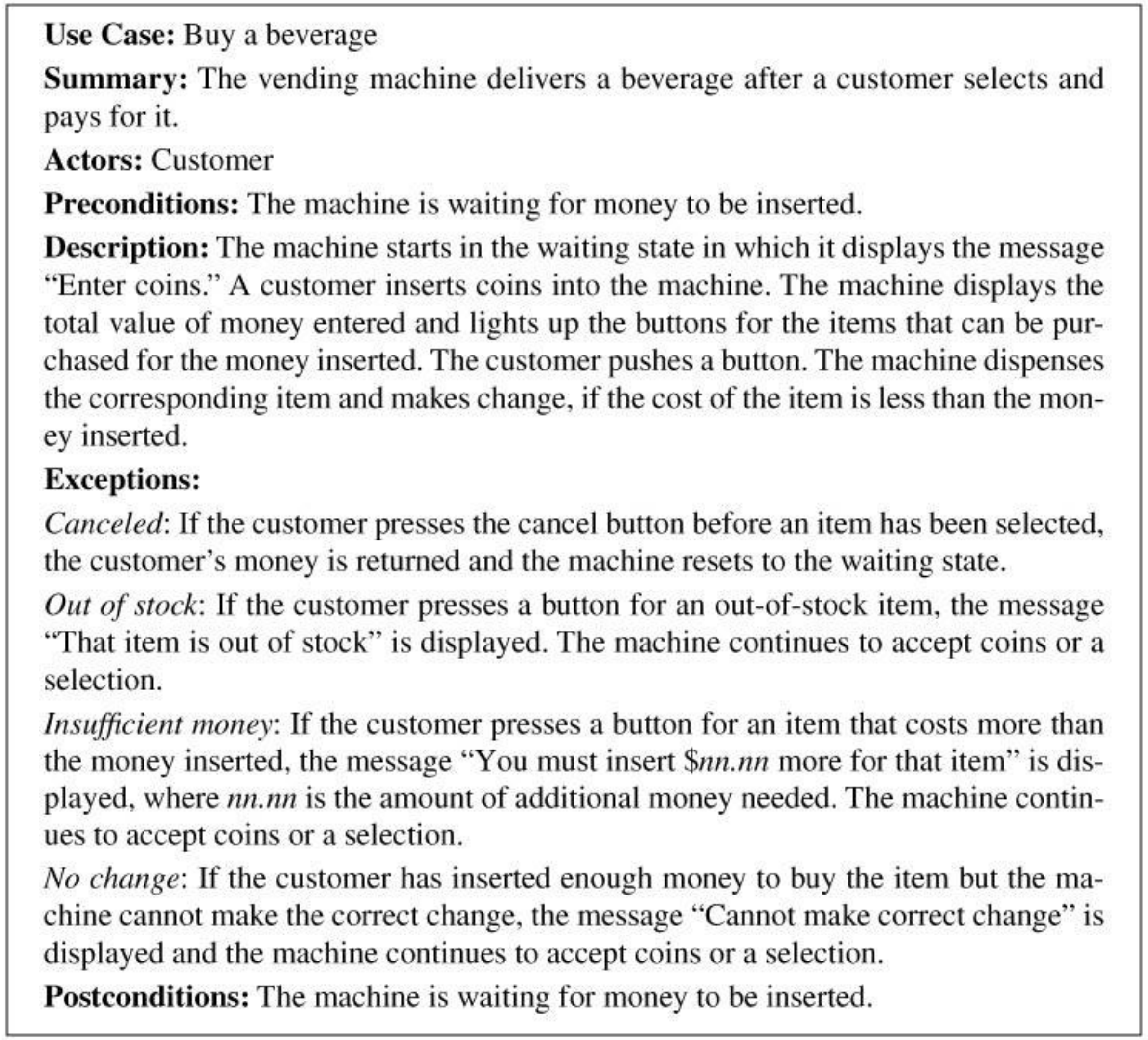
- **Buy a beverage**. The vending machine delivers a beverage after a customer selects and pays for it.
- **Perform scheduled maintenance**. A repair technician performs the periodic service on the vending machine necessary to keep it in good working condition.
- **Make repairs**. A repair technician performs the unexpected service on the vending machine necessary to repair a problem in its operation.
- **Load items**. A stock clerk adds items into the vending machine to replenish its stock of beverages.

Use Case Diagram: UML Notation

- System: Rectangle
- Use case: Ellipse
- Actor: Stick man
- Connections: Lines



Use Case Description (E.g.)



Use Case: Buy a beverage

Summary: The vending machine delivers a beverage after a customer selects and pays for it.

Actors: Customer

Preconditions: The machine is waiting for money to be inserted.

Description: The machine starts in the waiting state in which it displays the message "Enter coins." A customer inserts coins into the machine. The machine displays the total value of money entered and lights up the buttons for the items that can be purchased for the money inserted. The customer pushes a button. The machine dispenses the corresponding item and makes change, if the cost of the item is less than the money inserted.

Exceptions:

Canceled: If the customer presses the cancel button before an item has been selected, the customer's money is returned and the machine resets to the waiting state.

Out of stock: If the customer presses a button for an out-of-stock item, the message "That item is out of stock" is displayed. The machine continues to accept coins or a selection.

Insufficient money: If the customer presses a button for an item that costs more than the money inserted, the message "You must insert $*nn.nn* more for that item" is displayed, where *nn.nn* is the amount of additional money needed. The machine continues to accept coins or a selection.

No change: If the customer has inserted enough money to buy the item but the machine cannot make the correct change, the message "Cannot make correct change" is displayed and the machine continues to accept coins or a selection.

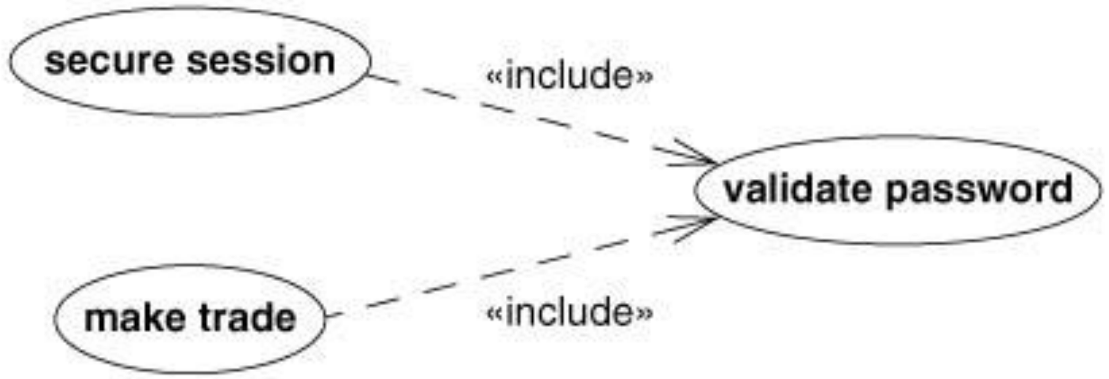
Postconditions: The machine is waiting for money to be inserted.

Use Case Relationships

- Facilitates building complex use cases
  - Include relationship
  - Extend relationship
  - Generalization relationship

Include Relationship

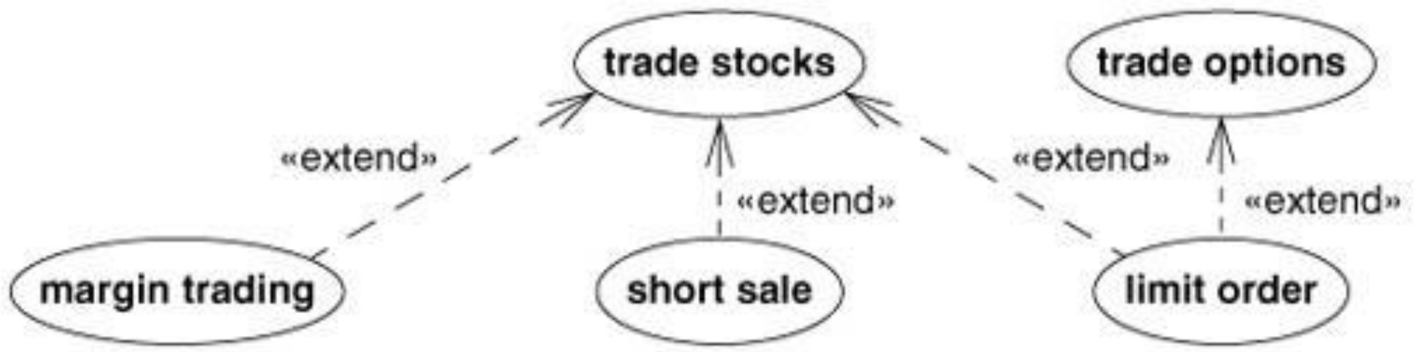
- Incorporates use case behavior within another
  - Inserted at a specific location within behavior sequence
  - Included use case may(/not) be usable on its own
- UML Notation:
  - Use case Description:
    - Include *use-case-name*
  - Use case Diagram:
    - Dashed arrow
    - <<include>> keyword



Extend Relationship

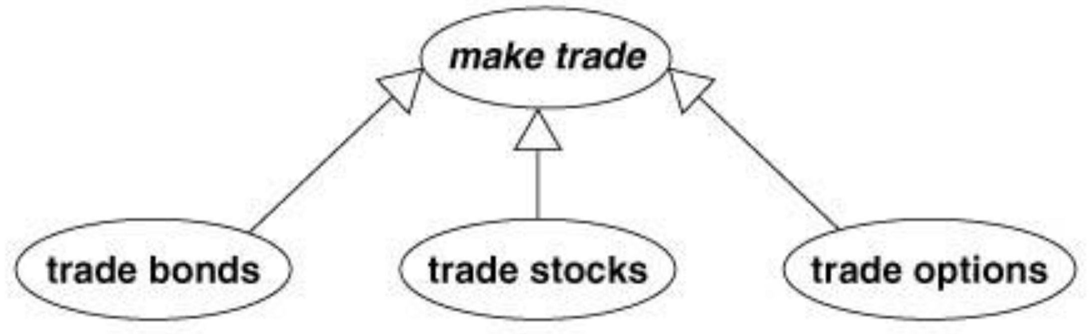
- Adds incremental behavior to use case
- Base / Extension use case

- UML Notation
  - Dashed arrow
  - <<extend>> keyword

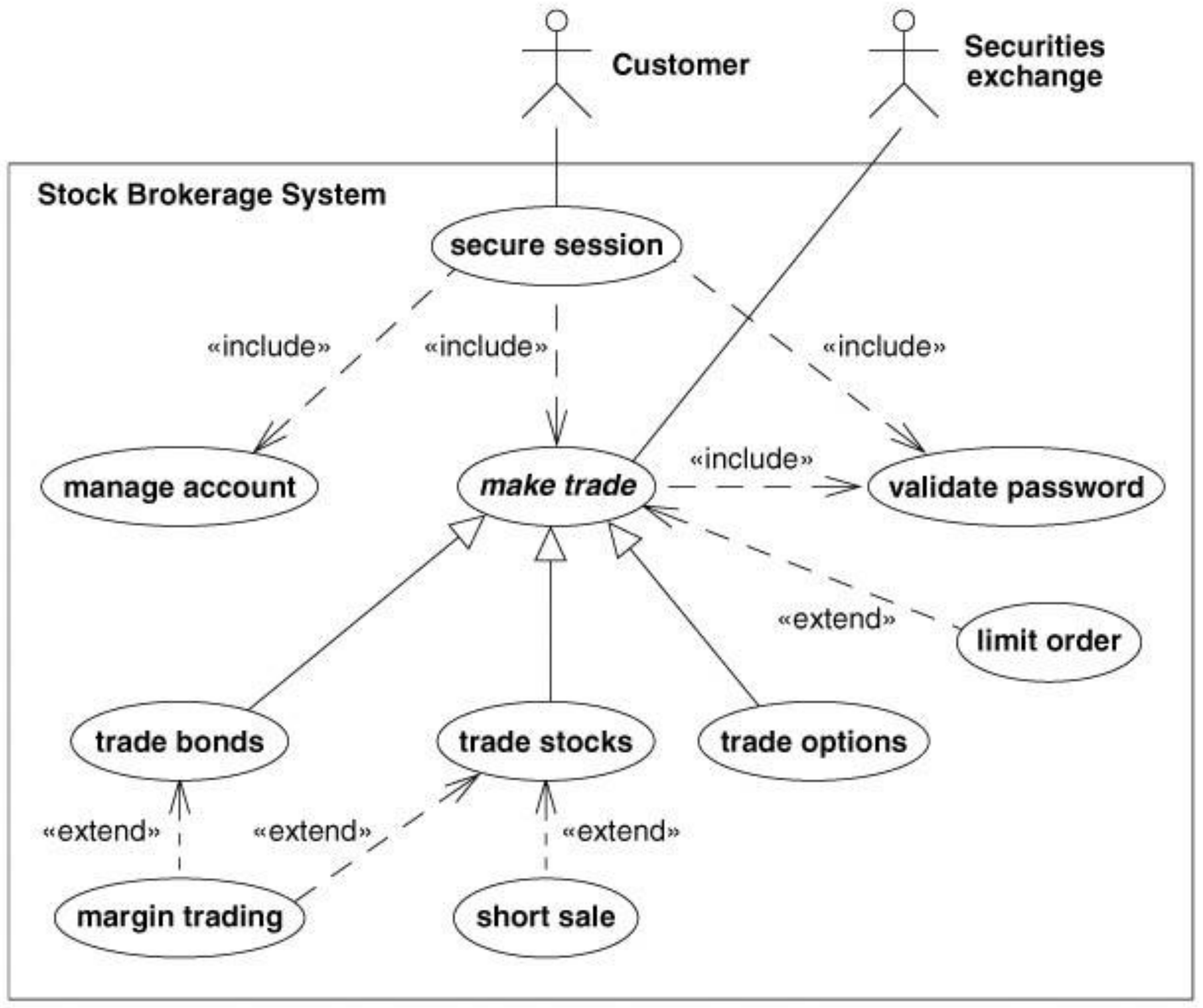


Generalization

- Shows specific variations on general use case
  - Parent use case / Child use case
- Use case Generalization Vs. Class Generalization
  - Parent may be abstract or concrete
  - Use case generalization
    - Order of subclass attributes not important
    - Multiple inheritance is not supported
- UML Notation
  - Arrow



Use Case Relationships: Combinations



Use Case Relationships: Guidelines

- Use case generalization
  - Model common behavior as abstract use case
- Use case inclusion
  - Well-defined behavior fragment
- Use case extension
  - Optional features
  - Facilitates testing without extensions

Use Case Model: Guidelines

- Determine system boundary
- Determine focused actors
- Ensure use cases provide value
- Relate use cases and actors

- Use cases may be distinct or structured
- Use cases are informal

Interaction Model

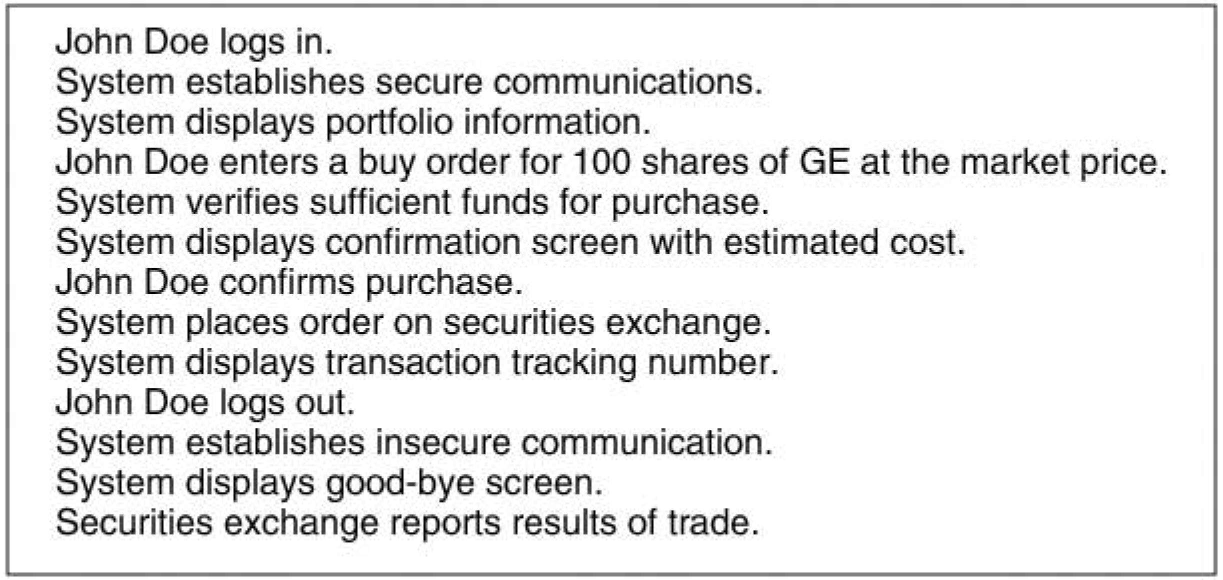
- Use cases
- Sequence diagrams
- Activity diagrams

Sequence Model

- Elaborates themes of use cases
- Two representations
  - Scenario
  - Sequence Diagram

Scenario

- Sequence of events during one execution of system
- Text description
  - Messages & Activities
- Steps of writing a scenario
  - Identify objects
  - Determine sender / receiver
  - Determine sequence of messages
  - Add activities for internal computations



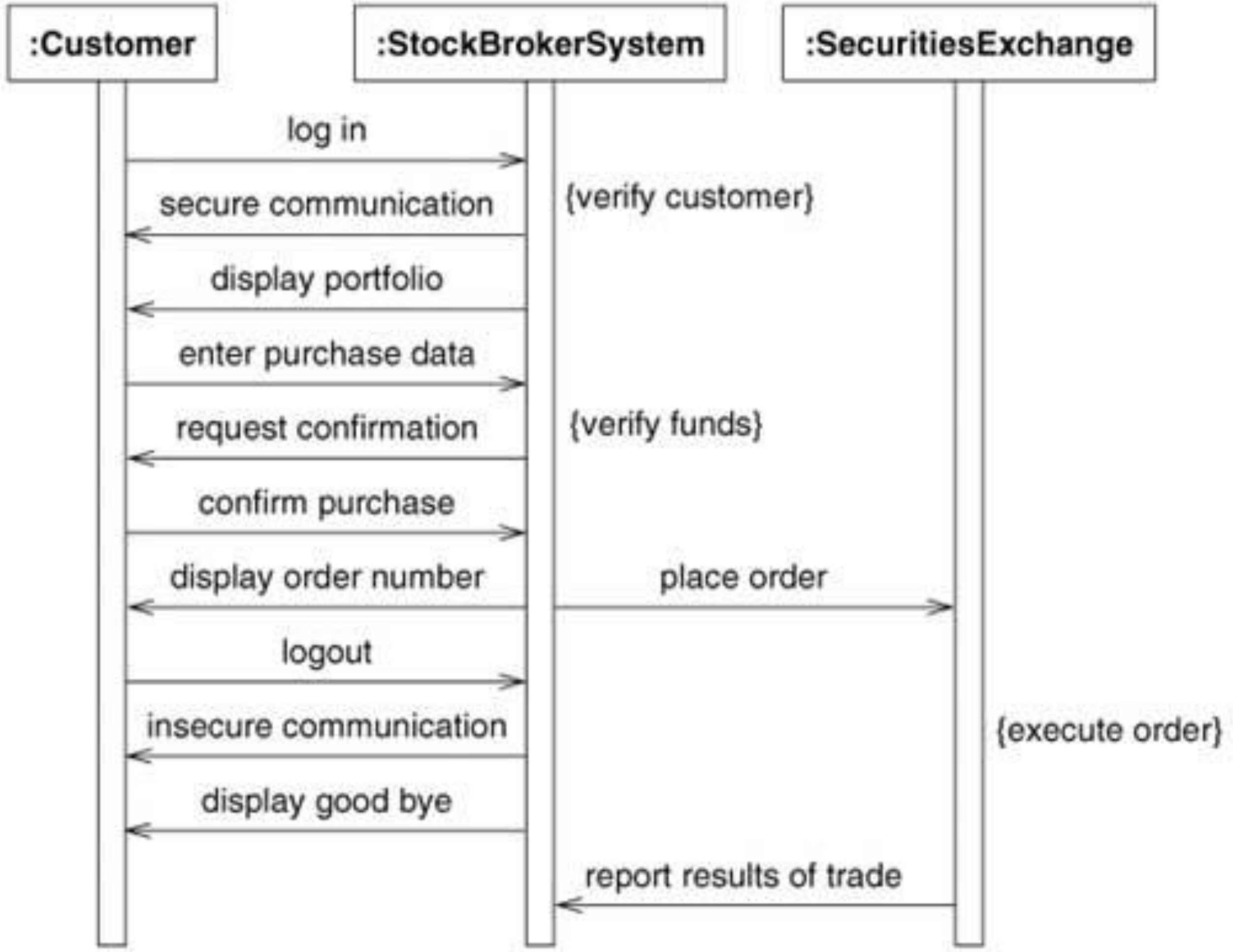
John Doe logs in.
System establishes secure communications.
System displays portfolio information.
John Doe enters a buy order for 100 shares of GE at the market price.
System verifies sufficient funds for purchase.
System displays confirmation screen with estimated cost.
John Doe confirms purchase.
System places order on securities exchange.
System displays transaction tracking number.
John Doe logs out.
System establishes insecure communication.
System displays good-bye screen.
Securities exchange reports results of trade.

Sequence Diagram

- Participants & Sequence of messages
- Particular behavior sequence of use case
- Use case: Multiple sequence diagrams

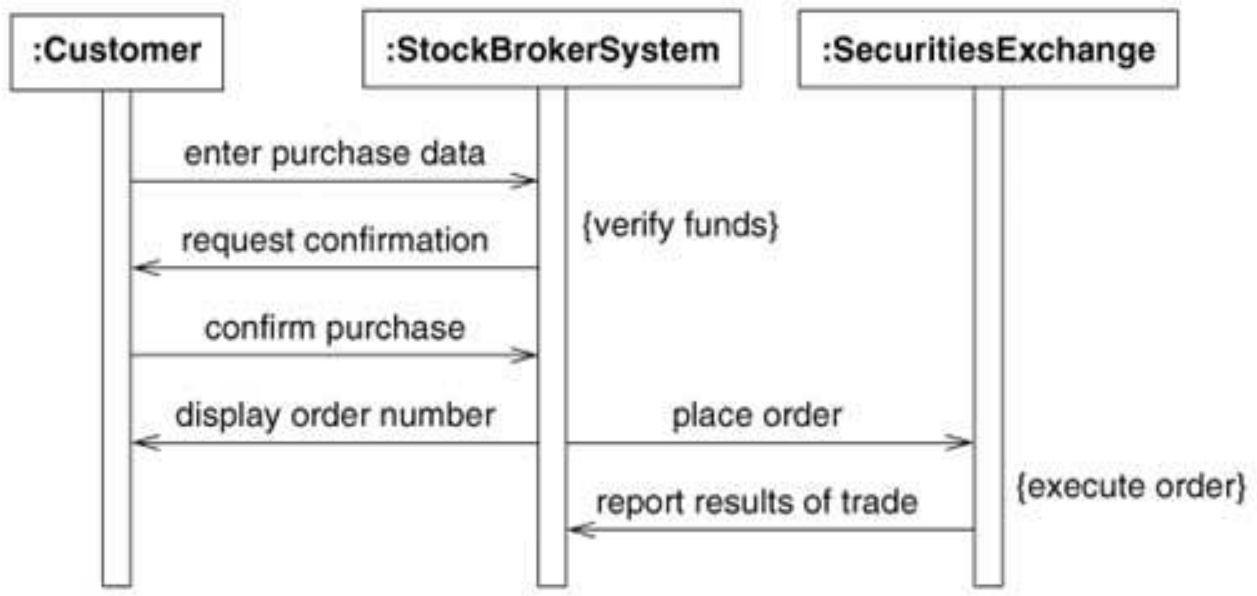
- UML Notation
  - Actor / System: Vertical line (Lifeline)
  - Message: Arrow

Sequence Diagram (E.g. Online stockbroker Session)



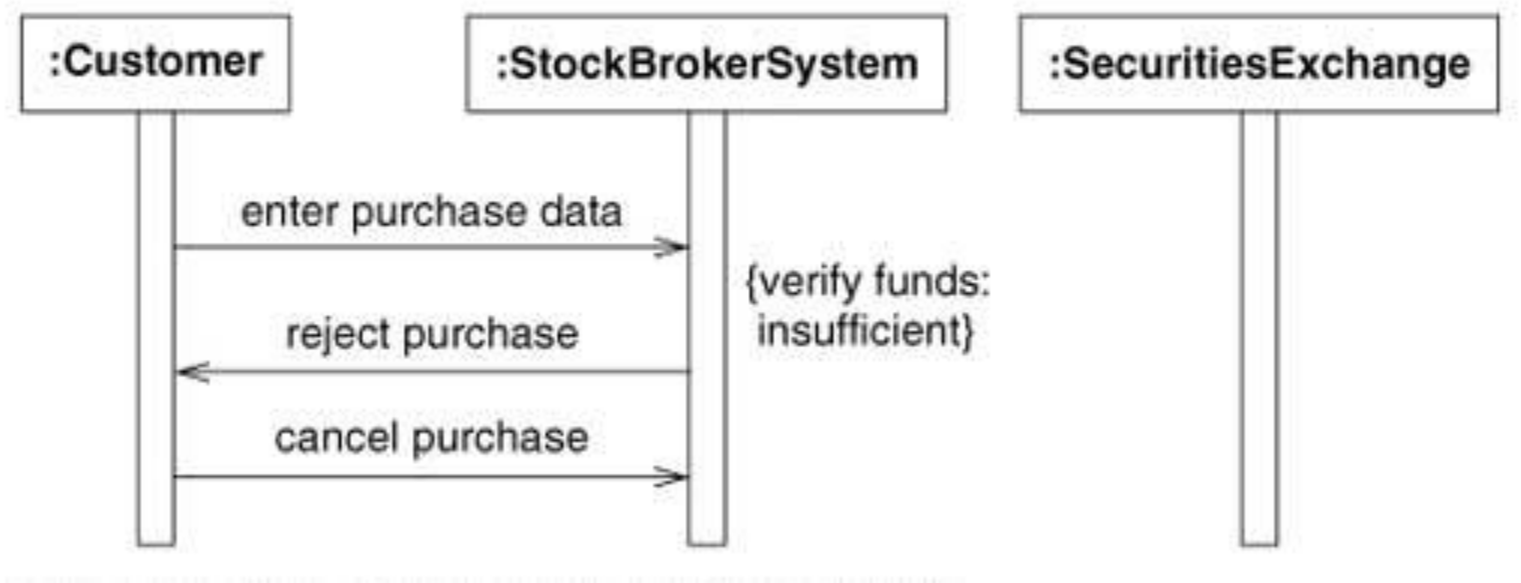
Sequence Diagram (E.g. Stock purchase)

- Sequence diagram for each task



Sequence Diagram (E.g. Stock purchase that fails)

- Sequence diagram for each exception



Sequence Model: Guidelines

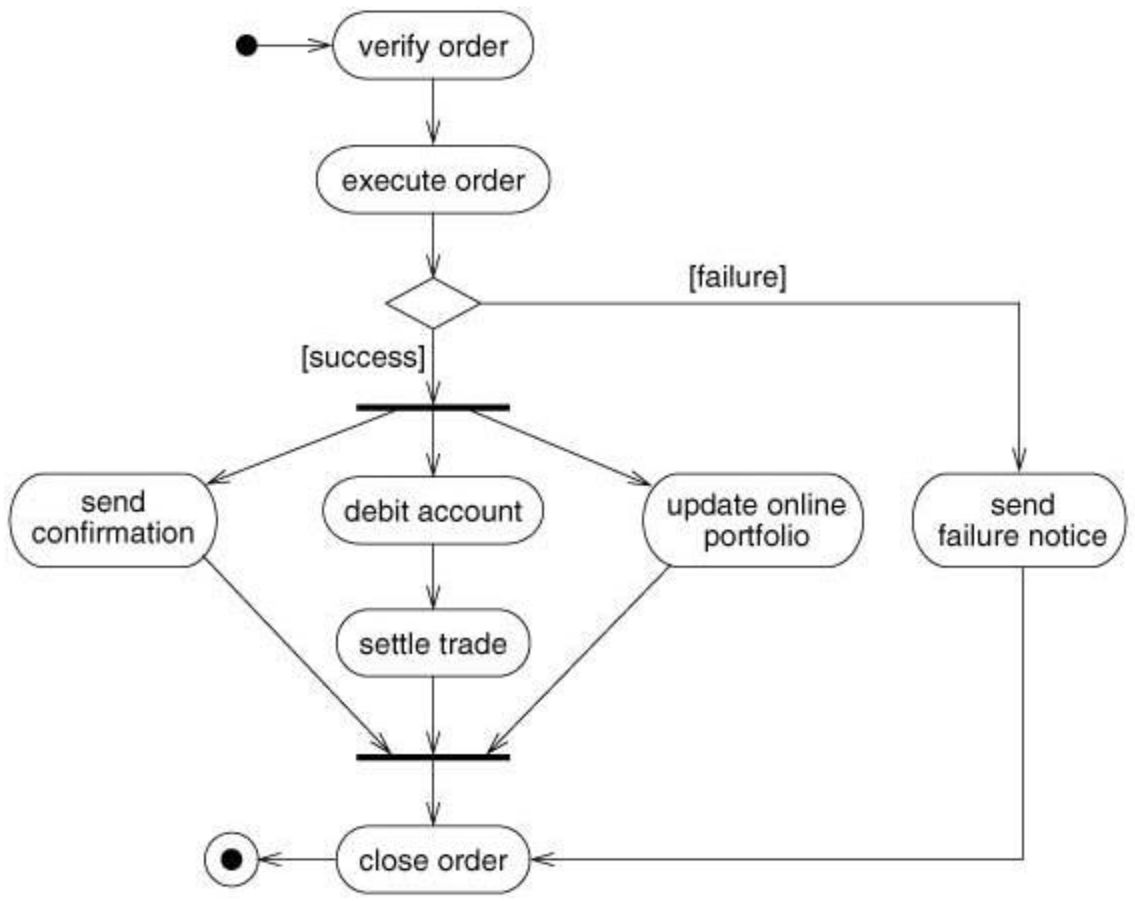
- At least one scenario per use case
- Abstract scenarios into sequence diagrams
- Divide complex interactions into tasks
- Sequence diagram for each error condition

Interaction Model

- Use cases
- Sequence diagrams
- Activity diagrams

Activity Model

- Activity Diagram
  - Shows sequence of steps
- Activity
  - Each step is an operation
  - Same as 'Activity' from state model
- Completion of activity
  - Completion event

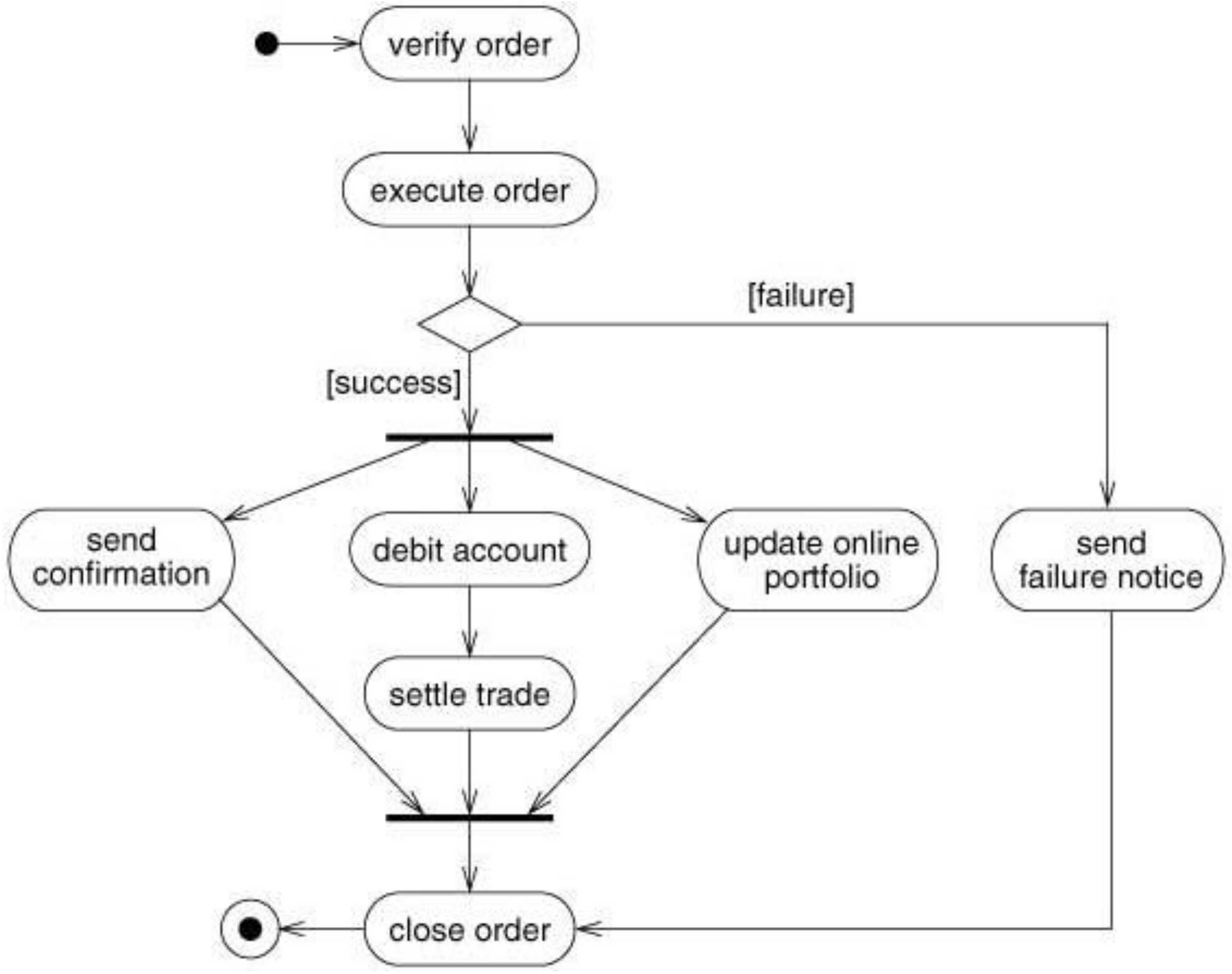


Activity Diagram: Branches

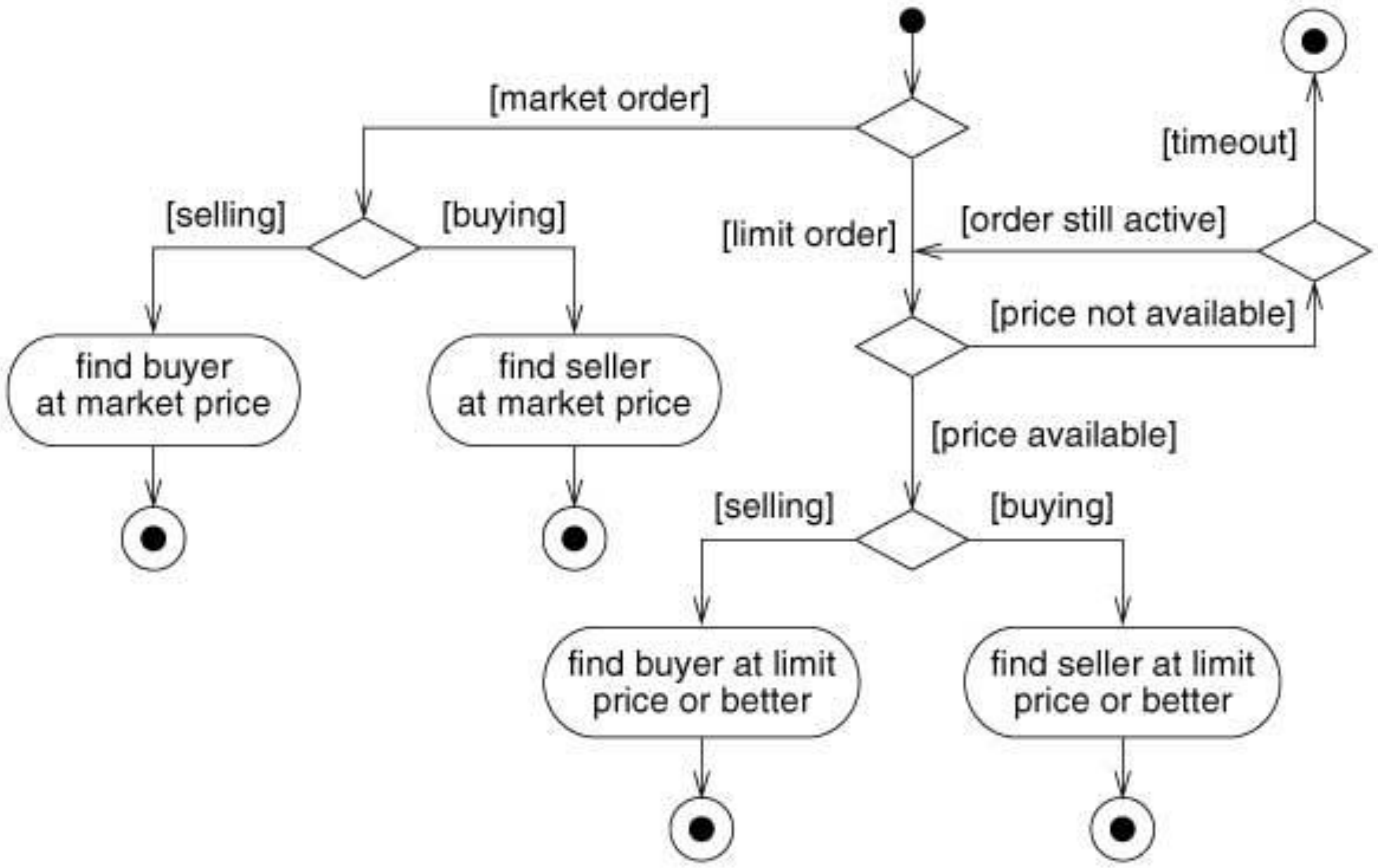
- If more than one successor to activity,
  - then label with condition

  - If one condition is satisfied
  - If none satisfied
  - If multiple conditions are satisfied

Activity Diagram (E.g. Stock trade processing)

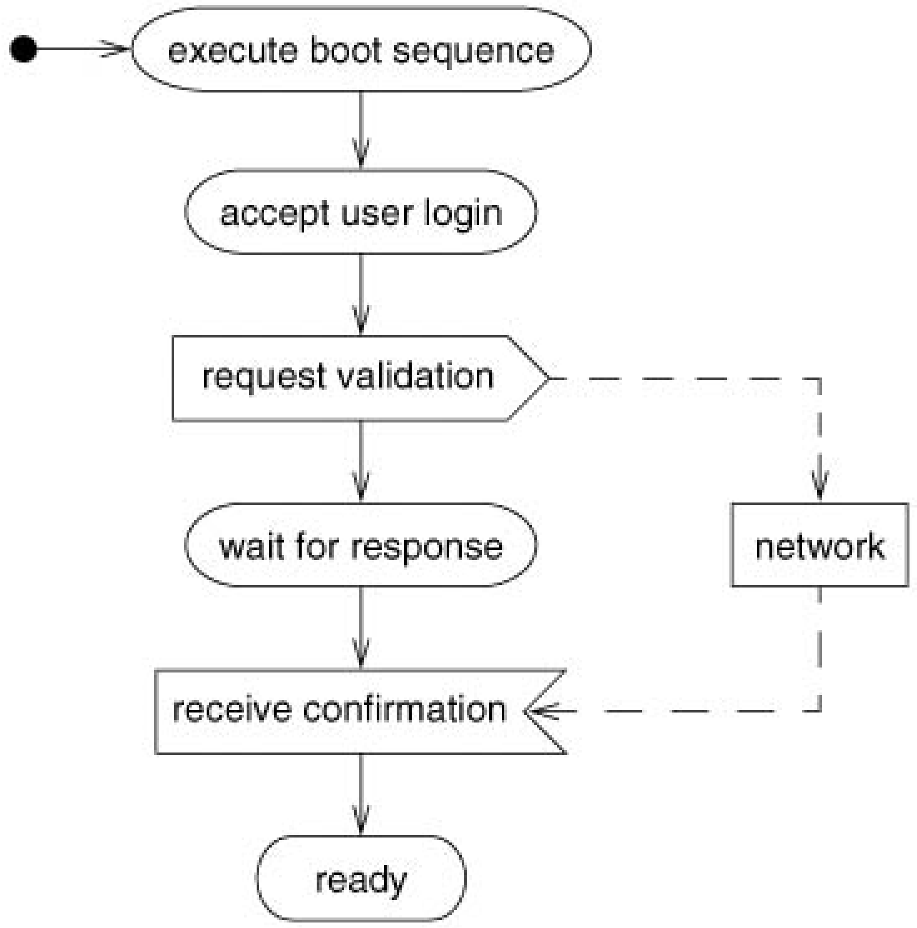


Activity Diagram
(E.g. Execute order)



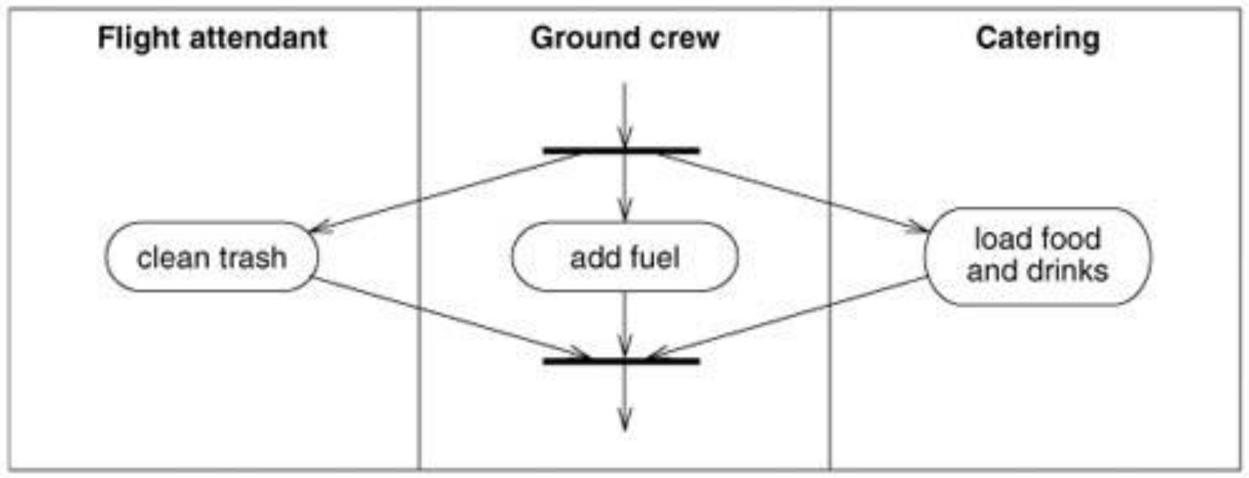
Activity Models: Sending and Receiving Signals

- UML Notation
  - Signal Send
    - Convex pentagon
  - Signal Receive:
    - Concave pentagon



Activity Models: Swimlanes

- Partition activities among organizations
- Each column → Swimlane
- Crossing lines → interactions among organizations



Activity Model: Guidelines

- Don’t misuse activity diagrams
- Consistent level of detail
- Be careful with branches and conditions
- Be careful with concurrent activities