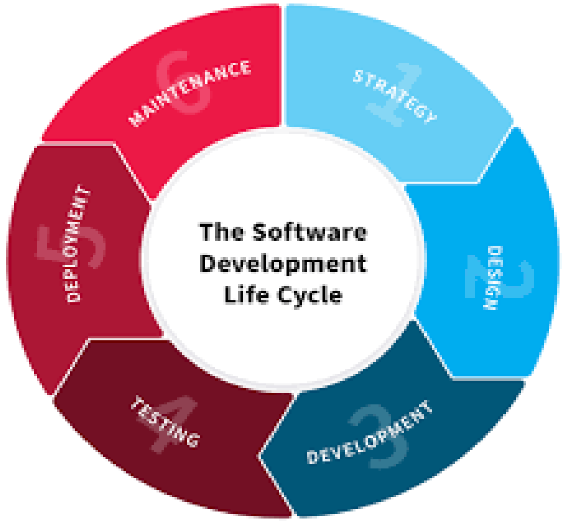
System Analysis Course

Week 04: UML (Activity diagram)

Ahmed Kord

Rana Khattab



Outline

- ❖UML
    - ❖Activity diagram
- ❖Practical Part – on Software Program

UML (Unified Modeling language)

UNIFIED
MODELING
LANGUAGE

Activity diagram

- UML **Activity Diagrams** are the object oriented equivalent of flow chart and data flow diagrams in function-oriented design approach.
- **Activity diagrams** represent the **dynamics** of the system.
- **Activity diagrams** can be *very useful to understand* the *complex processing* activities involving many components.
- ***They show:***
- – The flow of control from activity to activity in the system,
  – What activities can be done in parallel.
  – Alternate paths through the flow.

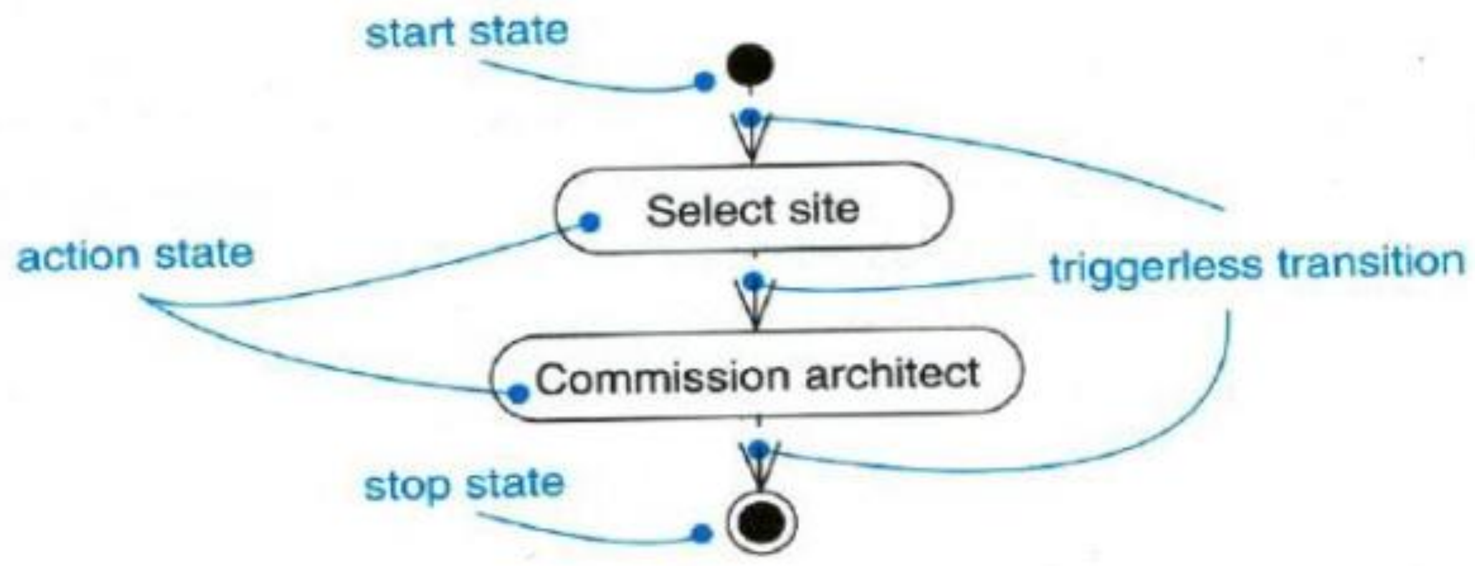
Activity diagram cont.

- ***Activity diagrams*** model the flow of control from one activity to another. An activity diagram typically represents the invocation of an operation, a step in a business process, or an entire business process. It consists of activity states and transitions between them.
- The diagram shows flow of control and branches ***(small diamonds)*** can be used to specify alternative paths of transitions. ***Parallel flows*** of execution are represented by ***fork*** and ***join*** constructs (solid rectangles).
- ***Swimlanes*** can be used to separate independent areas.

SYMBOLS –STATES

- ***Start state:***
  - The filled circle is the starting point of the diagram
- ***Stop state:***
  - The filled circle with a boarder is the ending point. An activity diagram can have zero or more activity final state.
- Action states are atomic and cannot be decomposed.

SYMBOLS –STATES cont.

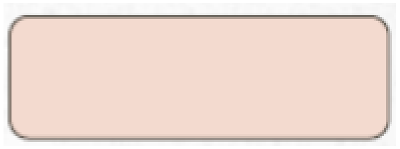


SYMBOLS -TRANSITIONS

- ***Transitions*** indicate the completion of an action or sub activity and show the sequence of actions or sub activities.
- A ***transition*** can be split into multiple transitions that can reach ***multiple action states.***
- Two or more ***transitions*** can be combined together using a ***merge***.

SYMBOLS -TRANSITIONS cont.

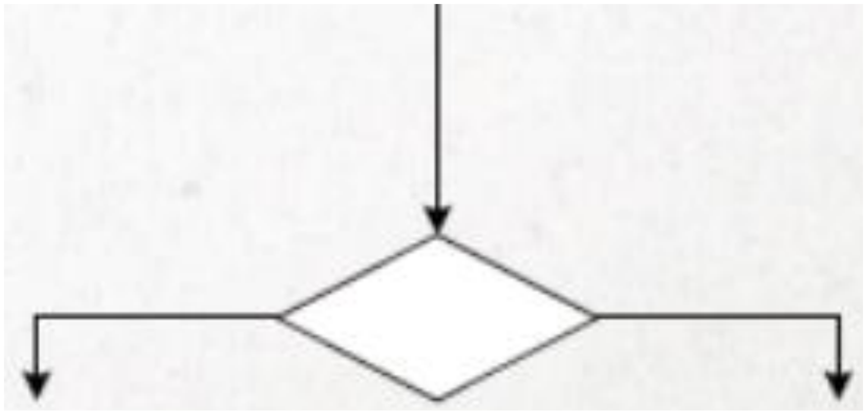
- An activity is ***some task*** which needs to be done.
- Each activity can be followed by another activity (sequencing).
- An activity is a specification of behavior.
- The ***rounded circle*** represents activities that occur.



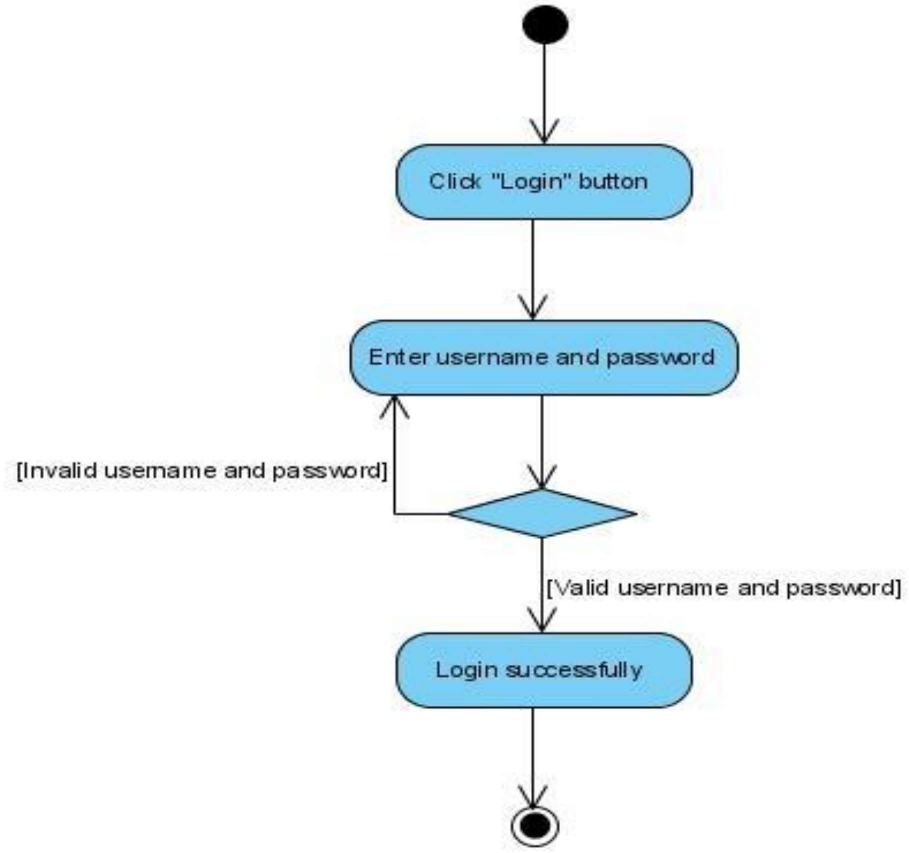
- Difference between an activity and an action:
  - ***Activity:*** A *sequence of actions that take finite time* and can be interrupted.
  - ***Action:*** An *atomic task that cannot be interrupted* (at least from user's perspective).
  - An ***action*** can invoke an activity to describe its action more finely.

SYMBOLS -DECISION

- A ***decision represents*** a specific location where the workflow may branch based upon guard conditions.
- There may be more than two outgoing transitions with different guard conditions, but for the most part, a decision will ***have only two outgoing transitions determined by a Boolean expression***.

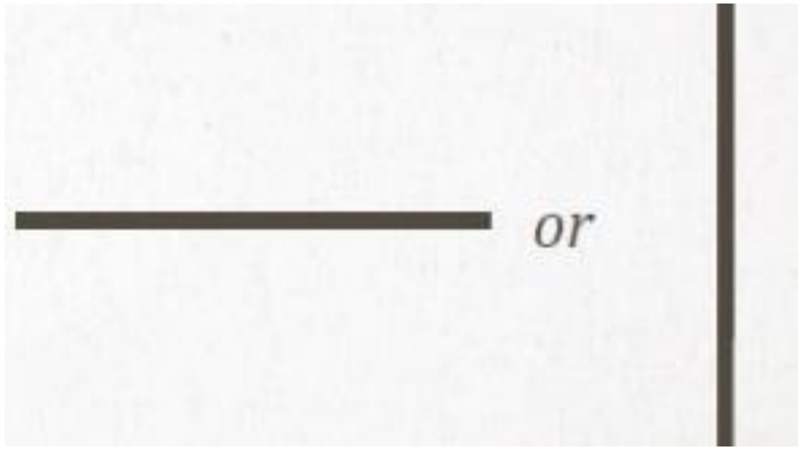


Login Activity Example 1


Click "Login" button
Enter username and password
[Invalid username and password]
[Valid username and password]
Login successfully

SYMBOLS -SYNCHRONIZATION

- ❖ Synchronizations enable you to see a simultaneous workflow.
- ❖ Synchronizations visually define forks and joins representing parallel workflow
- ❖ The next synchronization bar closes the concurrency.



Fork

- A black bar ***(horizontal/vertical)*** with *one* flow going into it and *several leaving it*.
- Denotes the beginning of ***parallel activities.***
- A ***fork*** may have one incoming transitions and two or more outgoing transitions.
- each transition represents an independent flow of control.
- conceptually, the activities of each of outgoing transitions are concurrent.

Join

- ❖ A ***black bar*** with *several flows entering* it and *one leaving* it. This denotes the end of parallel activities.
- ❖ A ***join*** may have ***two or more incoming transitions and one outgoing transition.***
- ❖ above the ***join***, the activities associated with each of these paths continues in parallel.
- ❖ At the join, the concurrent flows synchronize
  - ❖ each waits until all incoming flows have reached the join, at which point one flow of control continues on below the join.

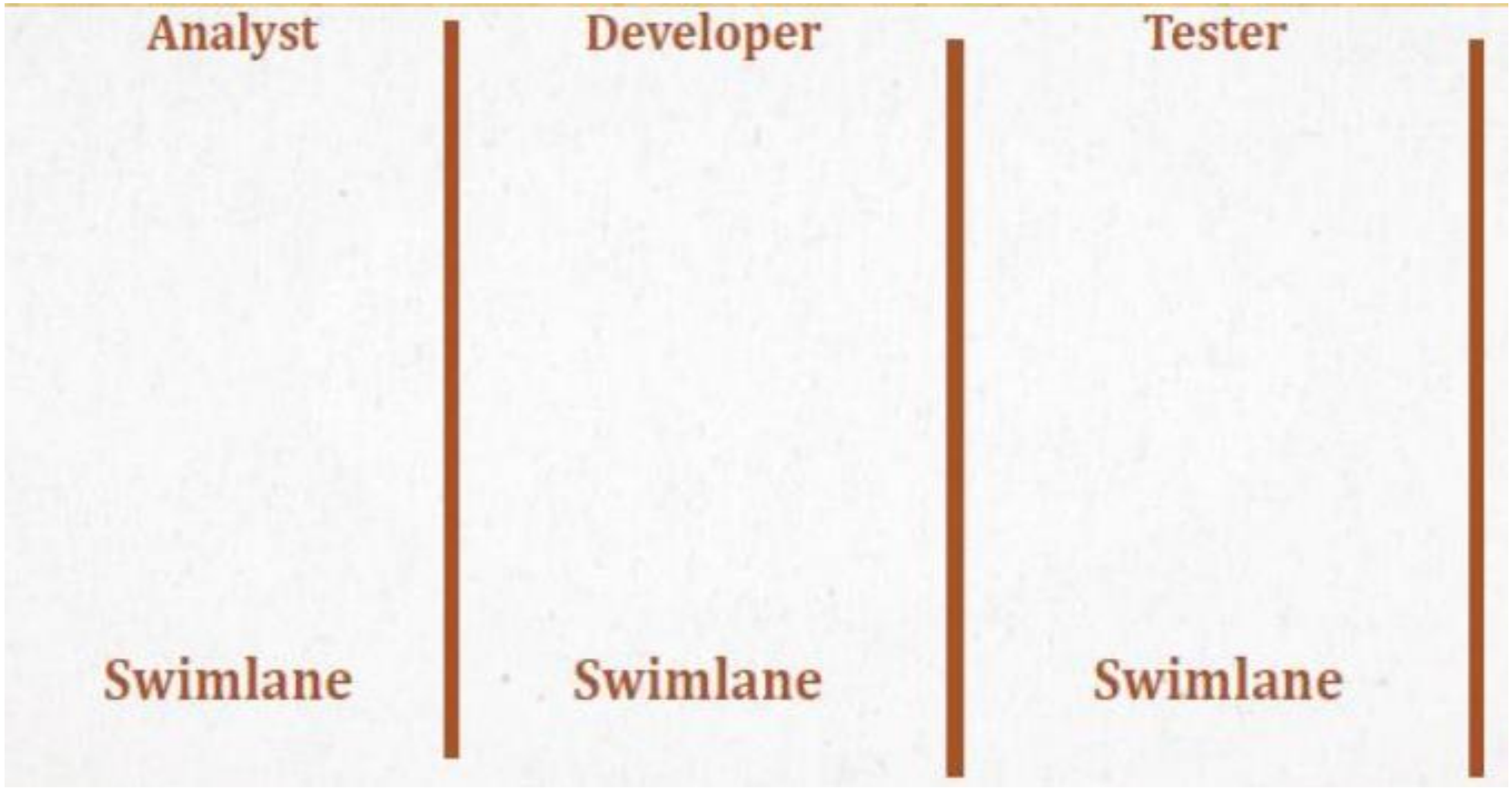
(Merge and Join)

- ***Merge:*** A diamond with several flows entering and one leaving. The implication is that all incoming flow to reach this point until processing continues.
- ***Difference between Join and Merge:***
  - A *join* is different from a merge in that the join ***synchronizes two inflows and produces a single outflow***. The outflow from a join cannot execute until all inflows have been received.
  - A ***merge*** passes any control flows straight through it. If two or more inflows are received by a merge symbol, the action pointed to by its outflow is executed ***two or more times***.

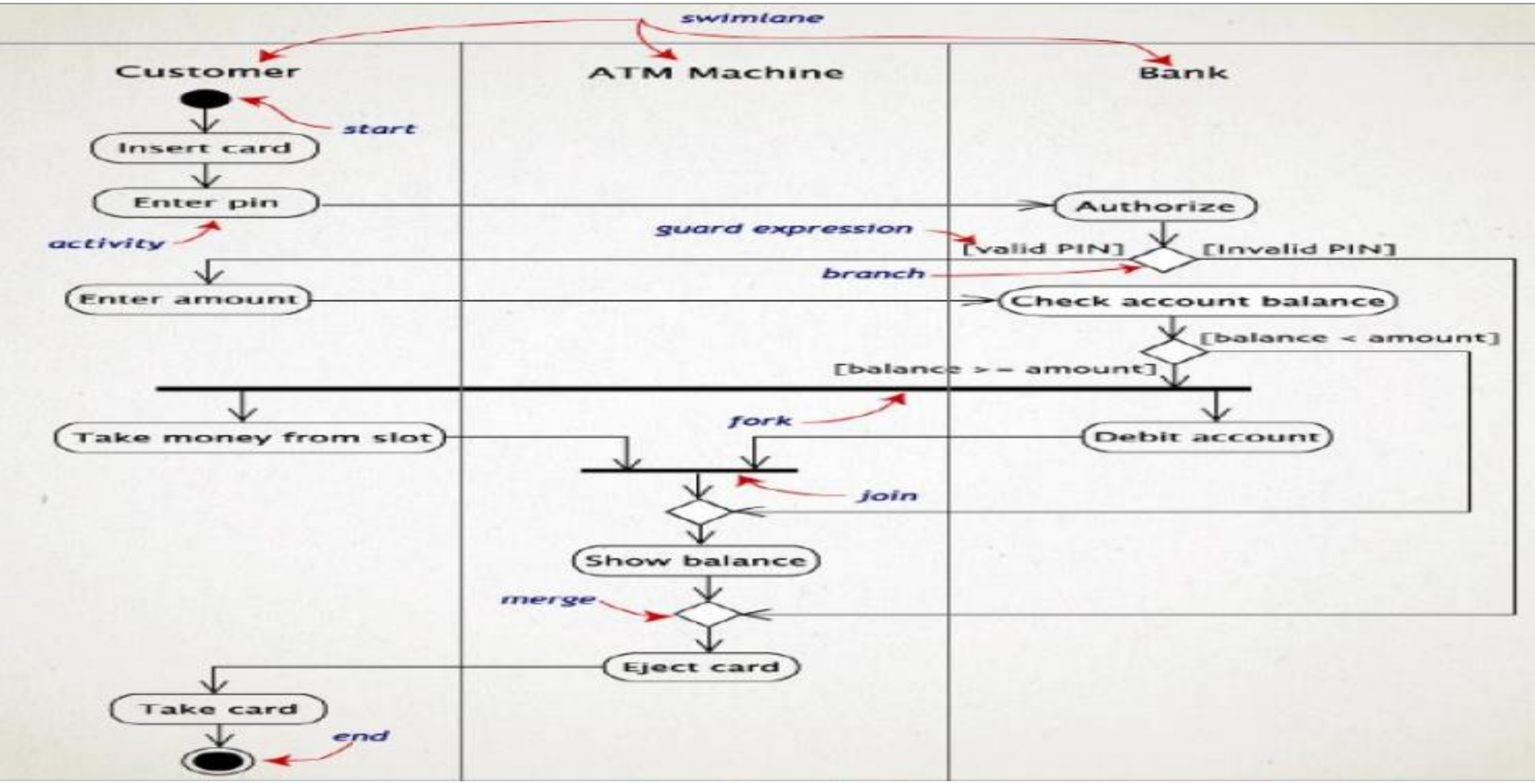
SYMBOLS –SWIMLANES Cont.

- Each ***swimlane*** has a name unique within its diagram.
- Each ***swimlane*** may represent some real-world entity.
- Each ***swimlane*** may be implemented by one or more classes.
- Every activity belongs to exactly one ***swimlane***, but transitions may cross lanes.

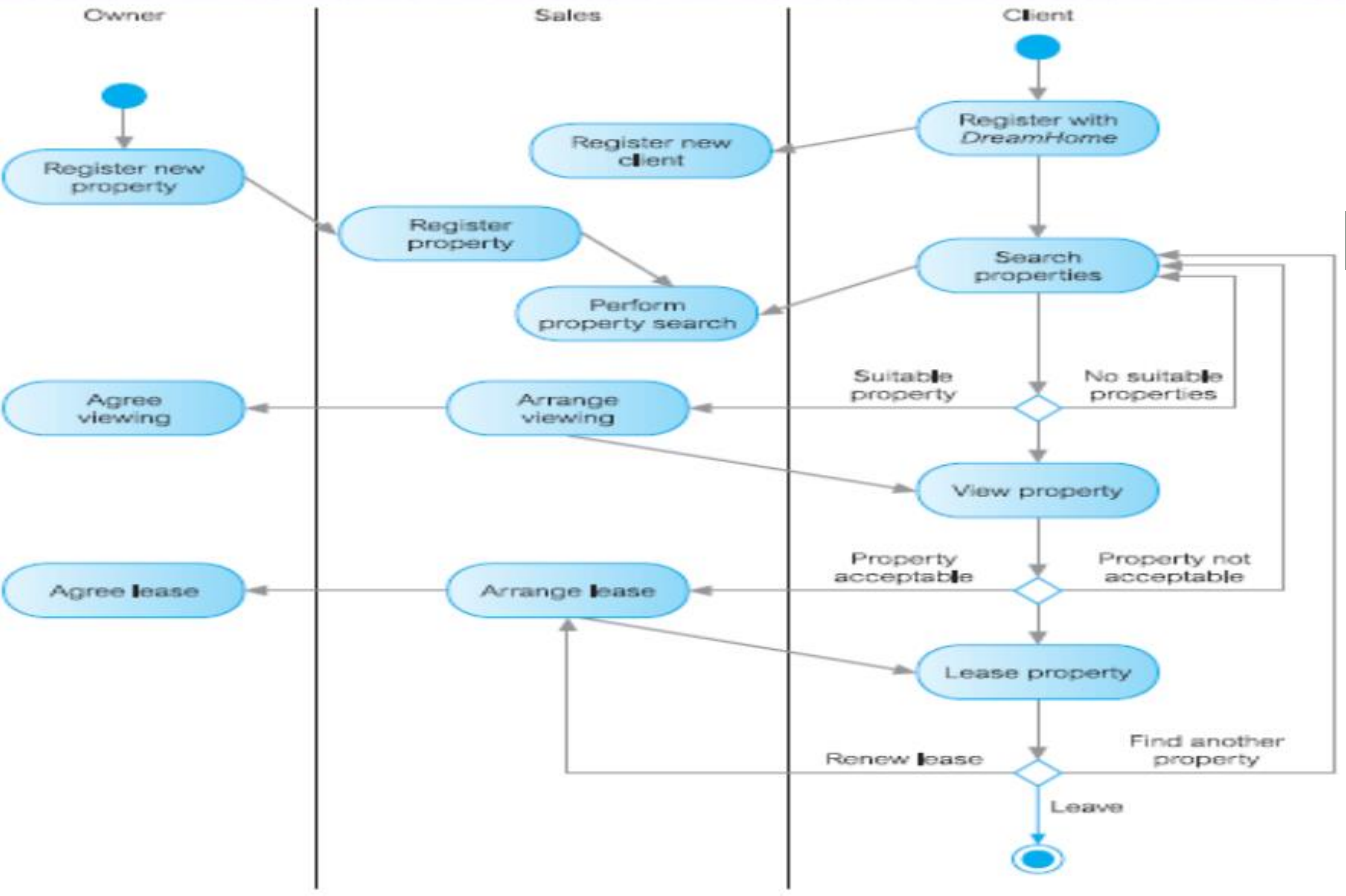
SYMBOLS -SWIMLANES



Activity DIAGRAM Example 2



Activity DIAGRAM Example 3



Outline

- UML
  - Activity diagram
- Practical Part – on Software Program

Thank You