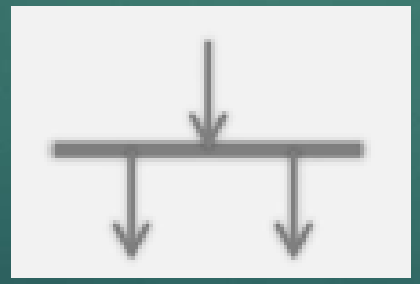
System Design & Analysis

LECTURE 09

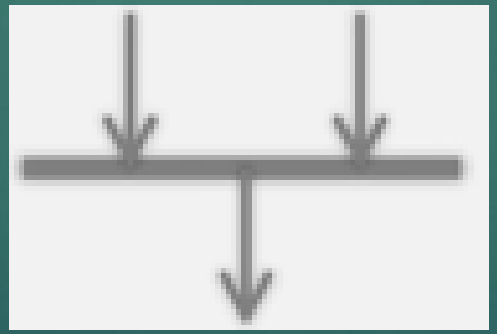
Fork

- Some actions can be performed parallely
- Such actions can be represented using fork
- Forks splits the incoming flow into multiple concurrent flows
- Fork is shown as a thick line segment with one incoming edge and multiple outgoing edges.

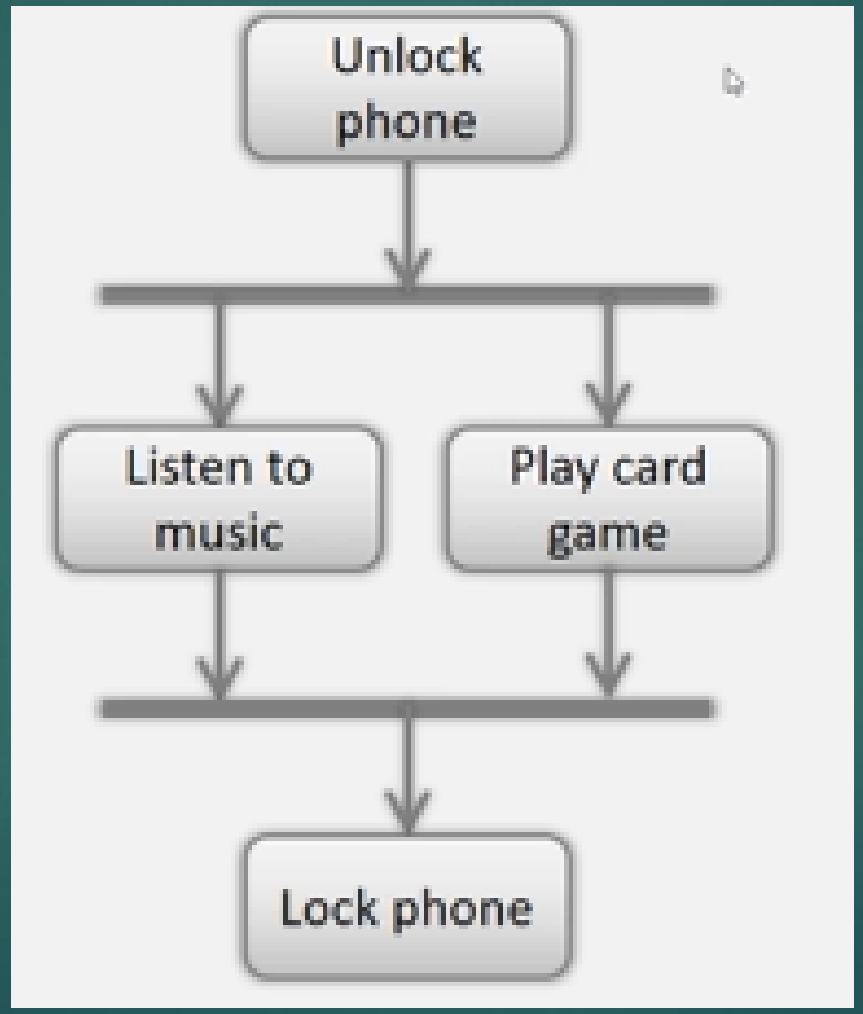


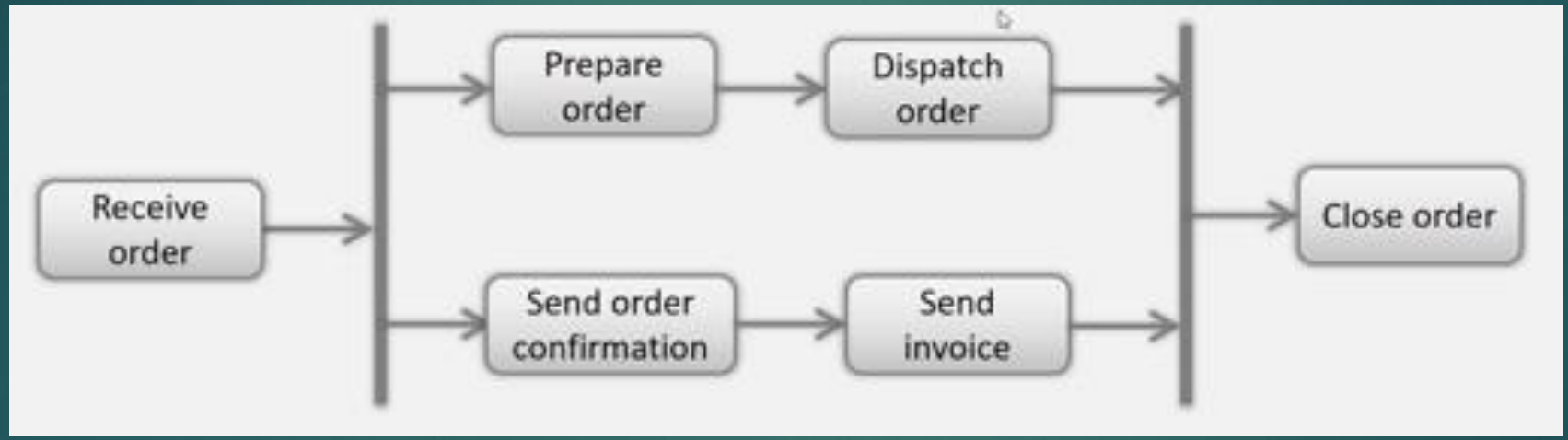
Join

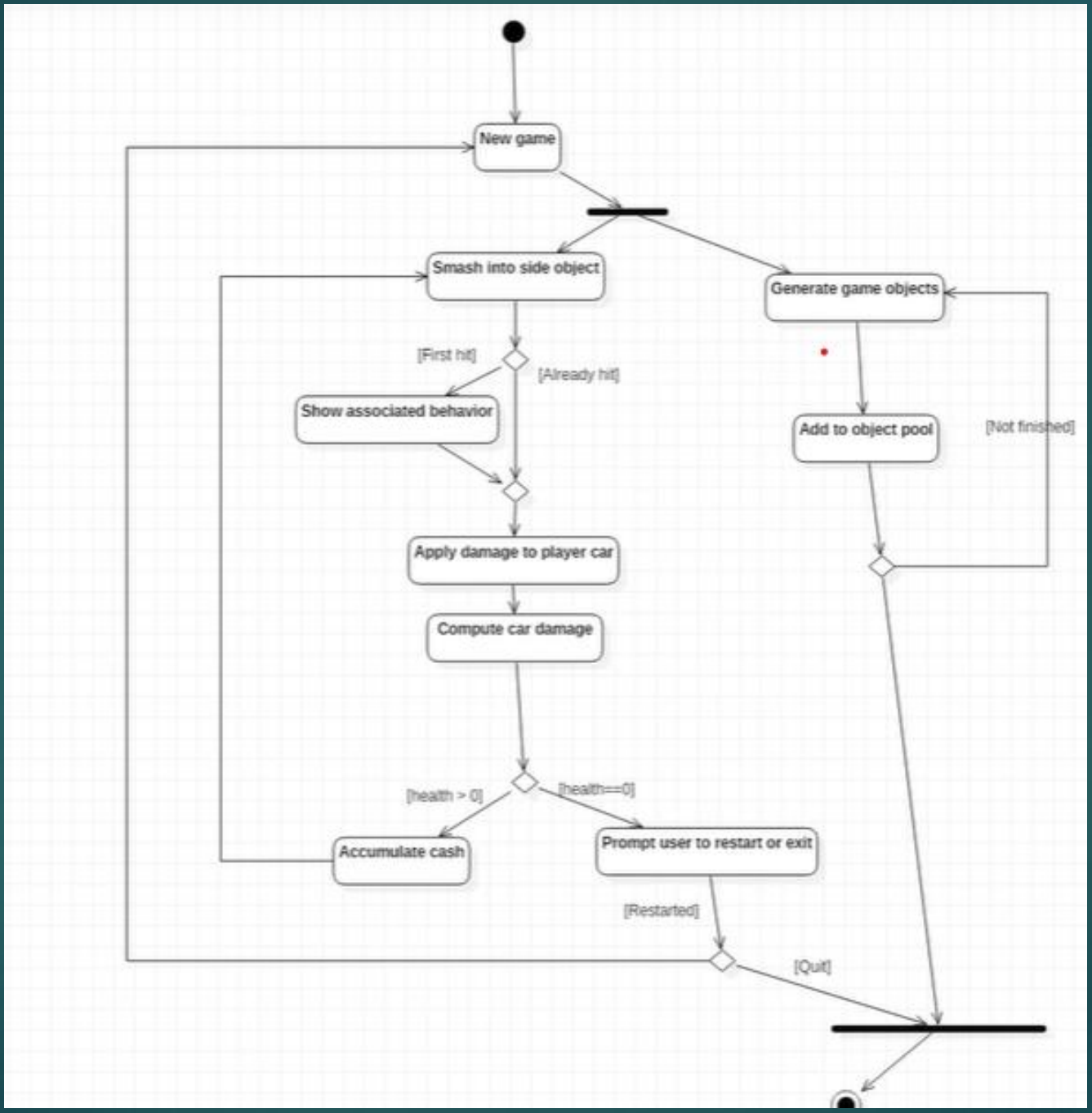
- Join synchronize multiple flows
- Has some notation as that of fork, but has multiple edges coming in and only one going out
- Edges coming into join are concurrent flows from an upstream fork



Fork & Join

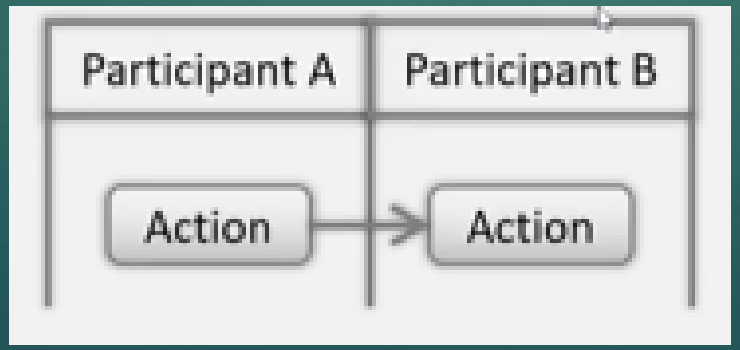


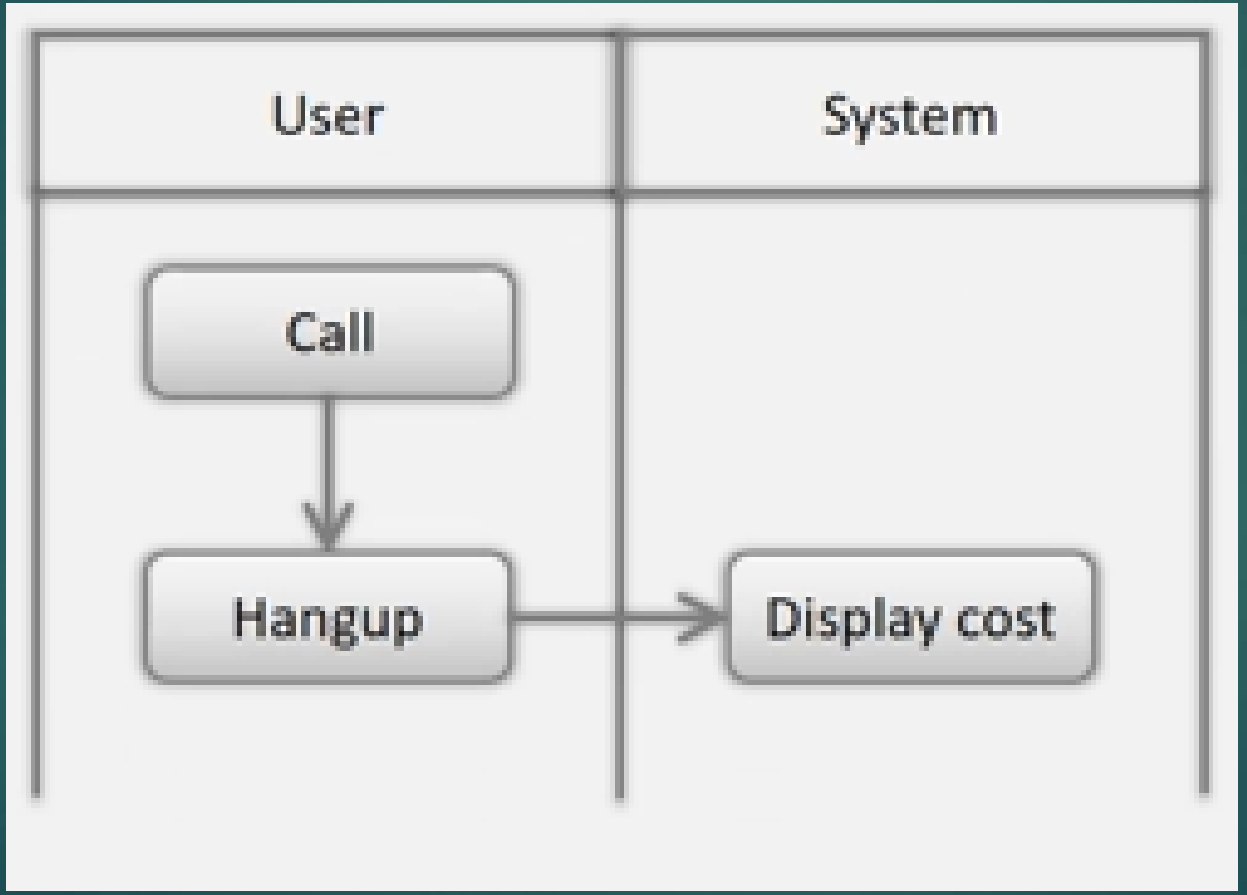

Receive order
Prepare order
Dispatch order
Send order confirmation
Send invoice
Close order

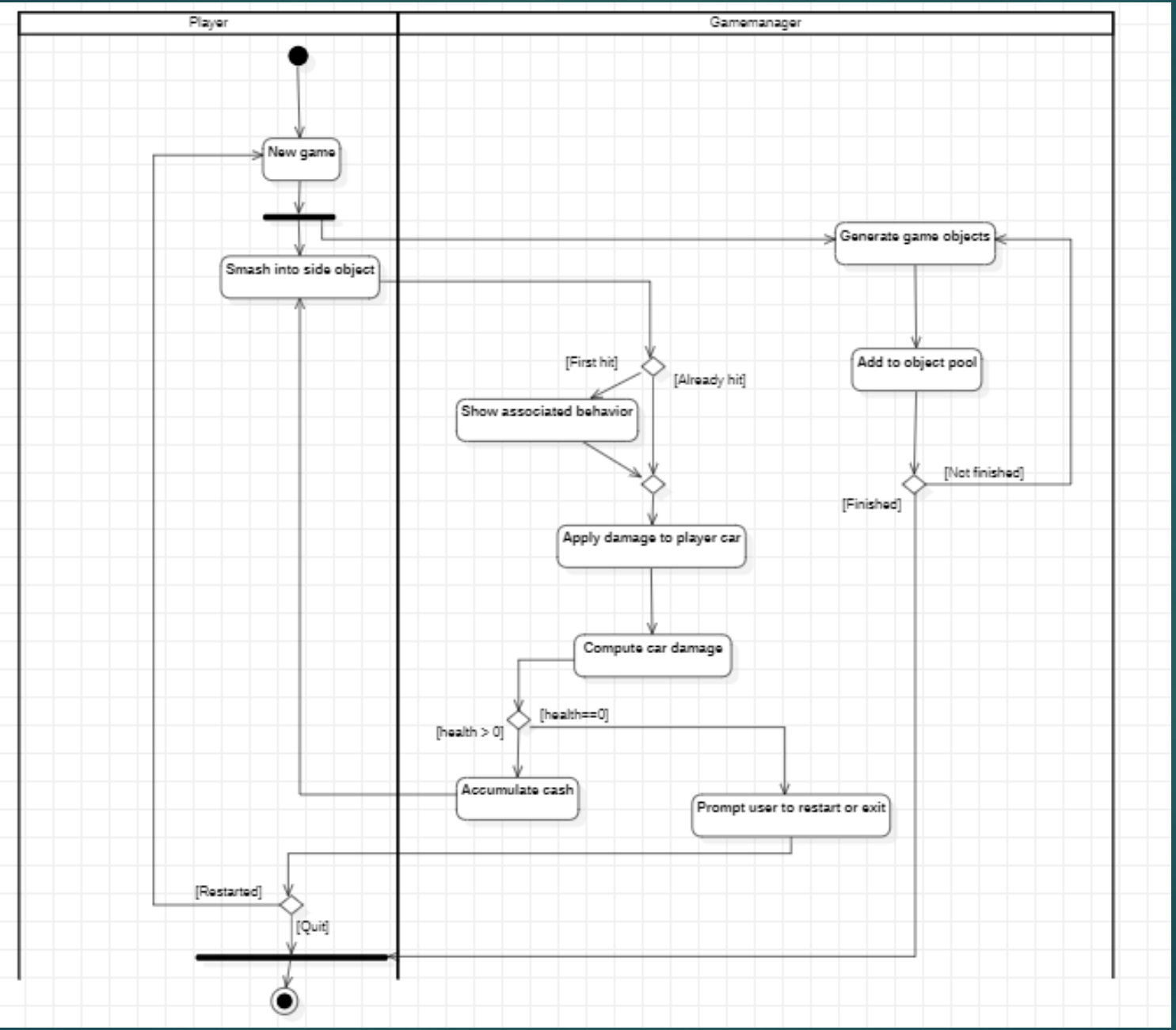

New game
Smash into side object
Generate game objects
[First hit]
[Already hit]
Show associated behavior
Add to object pool
[Not finished]
Apply damage to player car
Compute car damage
[health > 0]
[health==0]
Accumulate cash
Prompt user to restart or exit
[Restarted]
[Quit]

Partitions/Swimlanes

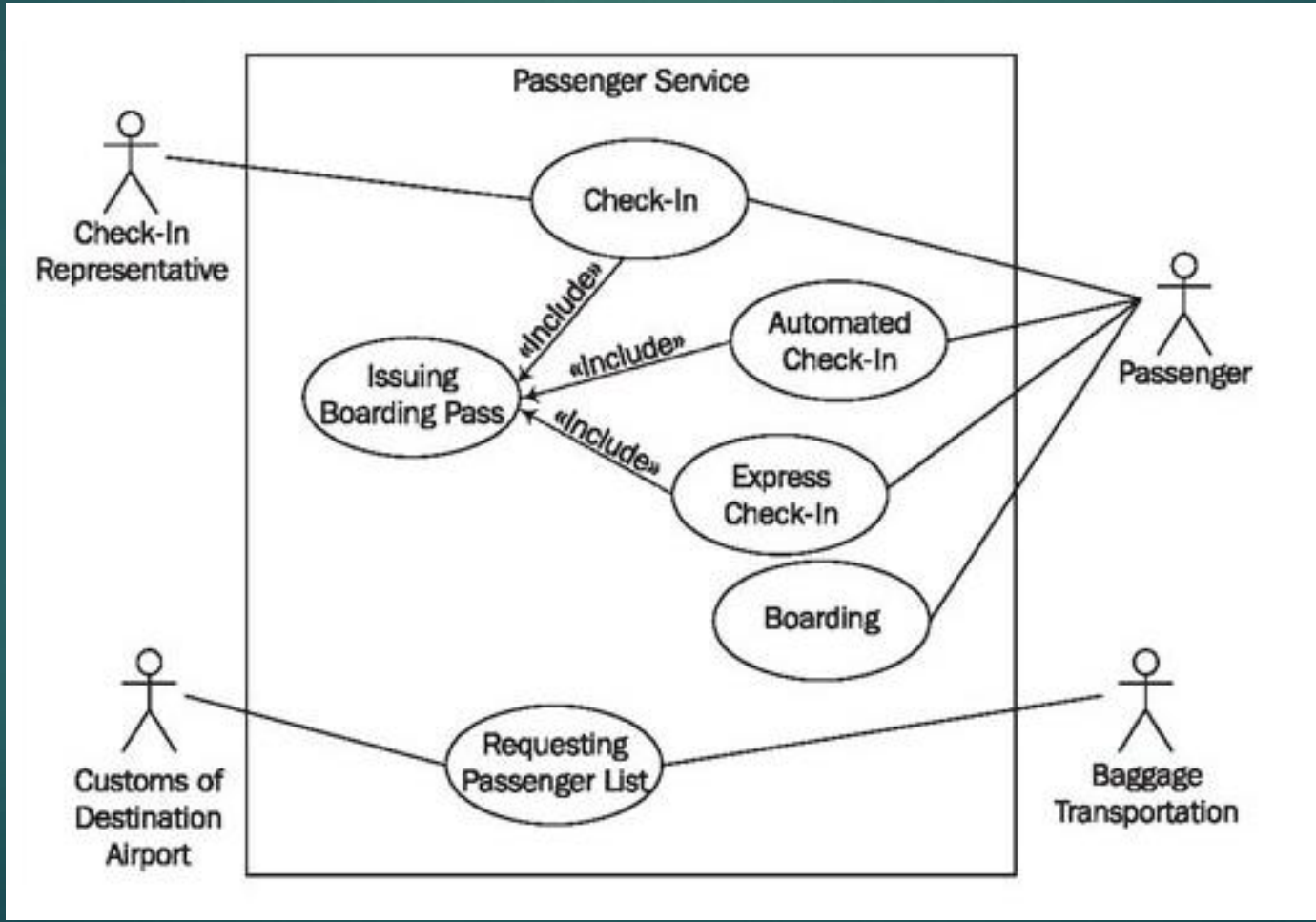
- Most activities involve different participants to complete the activity
- It becomes helpful to display which participant is responsible foe a particular action
- Shown through two parallel lines (vertically or horizontally)
  - Has label at one end that represents the participant
  - Activity nodes between these lines are considered to be part of that partition (& executed by that participant)




User
System
Call
Hangup
Display cost


Player
Gamemanager
New game
Smash into side object
Generate game objects
Add to object pool
[Not finished]
[Finished]
[First hit]
[Already hit]
Show associated behavior
Apply damage to player car
Compute car damage
[health==0]
[health > 0]
Accumulate cash
Prompt user to restart or exit
[Restarted]
[Quit]

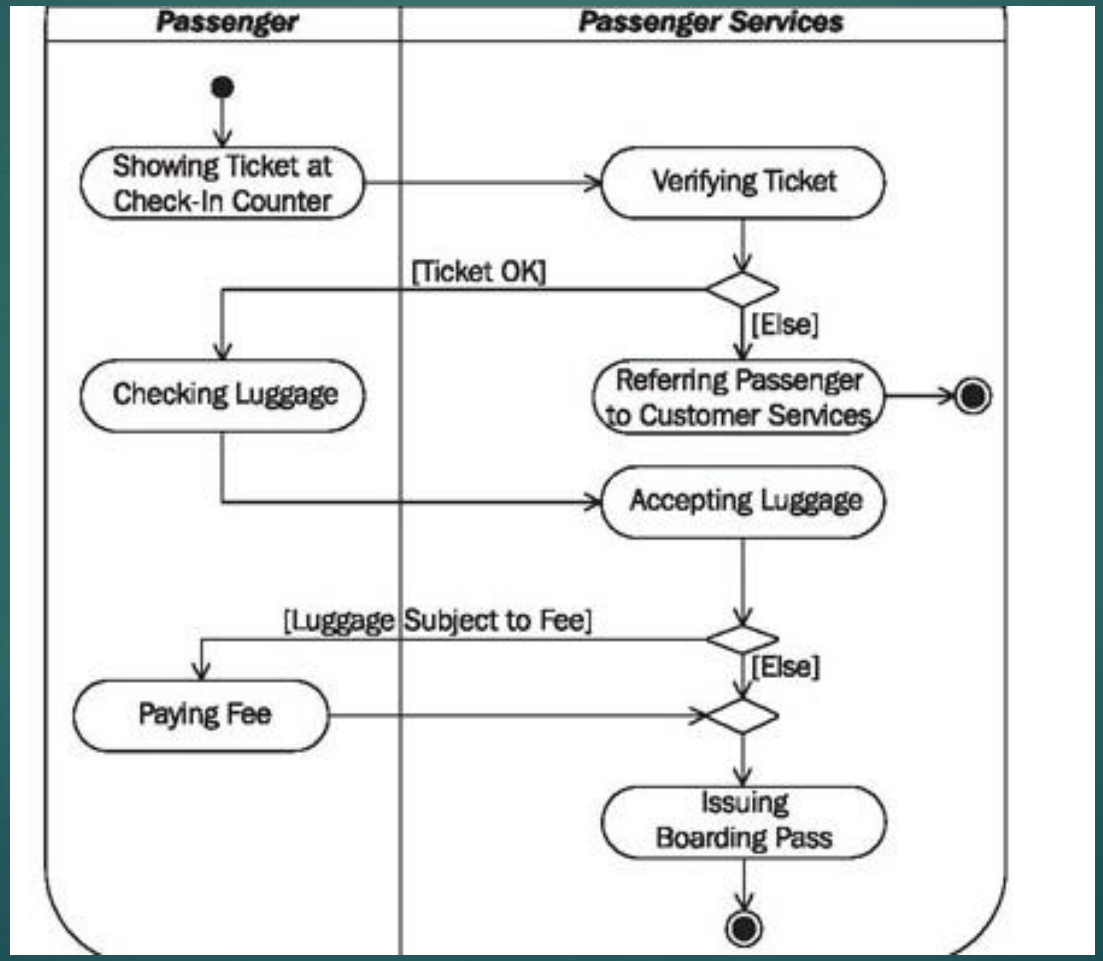
Example-> Passenger Service



Example -> Passengers Check-in

Identify use cases

Activity diagram



Example-> donating to charity