Detailing Use Cases Scenarios and Workflows

UML based modeling and analysis

Dániel Varró

Scenarios

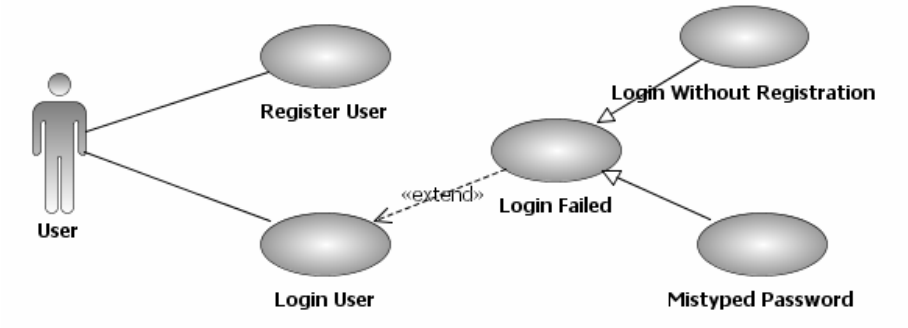
Content of a UC: Scenarios

- Main success scenario (MSS): sequence of numbered steps
  - A step: an interaction between an actor and the system
    - Simple statement stating who carries out the step
    - Intent of the actor and not how it is done on the GUI
- Extensions: Other scenarios (successes or failures)
  - A condition: when should we deter from the MSS?
  - Differences: what should be different?
  - Questions to ask:
    - How could MSS go differently?
    - What could MSS go wrong?

Scenarios (cont.)

- Precondition: what should be ensured before the system allows the UC to begin
- Guarantee: what the system will ensure at the end of the UC
- Trigger: specifies the event that forces to UC started

Example: User Management (Last Lecture)



Example Scenario: Register User

- Main success scenario:
  1. User types a user name of his or her choice
  2. User types a password
  3. User retypes the password

  Submit
  4. System checks if the user name is not already in use
  5. System checks if the two passwords are identical
  6. System registers the new player with the given parameters (user name, password)

Example Scenario: Register User

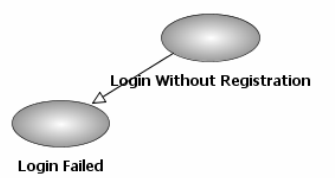
- Extensions:
  4a. User name is already in use
    .1 User is requested to select another user name and password
  5a. The two passwords are different
    .1 User is requested to retype (twice) his/her password
- Trigger: User selects the "Register User" link
- Precondition: User is not logged in
- Guarantee: User becomes a registered player

Example Scenario: Login User

- Trigger: User selects the "Login" link
- Precondition: The user is not yet logged in
- Guarantee: The user can access the main menu of the championship manager system in a new session
- Main success scenario:
  1. User types his/her user name
  2. User types his/her password
  3. System checks if the given login parameters are valid
  4. System creates a new session for the user
- Extensions:
  3a User provides invalid login parameters
  (see Login Failed)

Example Scenario: Login Failed

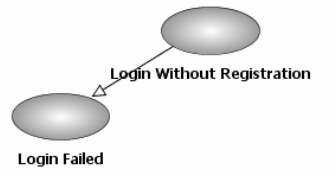
- Login Failed:
  - Precondition: The User provides invalid login parameters
    - Precondition should be identical with the condition of the extension point
  - Main success scenario:
    1. System redirects the User to the Login page
- Login without Registration:
  - Precondition: The User typed a non-registered user name
    - Precondition should refine the precondition of Login Failed
  - Main success scenario:
    1. System redirects the User to the Login page
    2. System informs the User that he/she typed a non-registered user name



Question:
How could one capture scenarios more precisely?

Example Scenario: Login Failed

- Login Failed:
  - Precondition: The User provides invalid login parameters
    - Precondition should be identical with the condition of the extension point
  - Main success scenario:
    1. System redirects the User to the Login page
- Login without Registration:
  - Precondition: The User typed a non-registered user name
    - Precondition should refine the precondition of Login Failed
  - Main success scenario:
    1. System redirects the User to the Login page
    2. System informs the User that he/she typed a non-registered user name



Solution:
Use UML Activity / Sequence diagrams in a semi-formal way

Organization of Use Case Models

- Actors Package
- Use Cases Package
  - User Management Package
    - Login User Use Case
      - Login User Activity
        - » Login User Activity Diagram
    - Register User Use Case
      - Register User Activity
        - » Login User Activity Diagram
  - Game Management Package
  - Championship Management Package

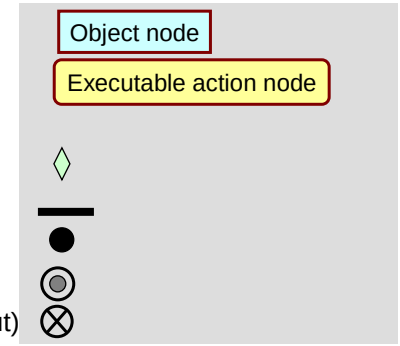
UML 2.0 Activity Diagrams

UML : Activity diagram

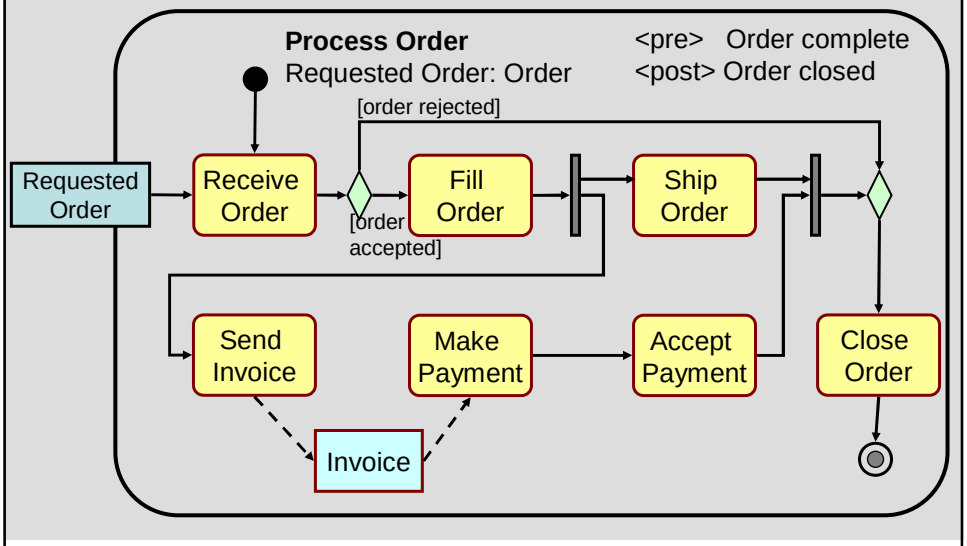
- Aim:
  - **Description of business workflows** (as in HW)
  - High-level description of UC scenarios
  - Detailed description of dynamic behavior
    - Methods
    - Actions of statecharts
- Two levels of abstraction
  - Action (akció): atomic operation
  - Activity (aktivitás): high-level grouping of actions
- Combined control and data flow model
  - semantics ≈ dataflow networks

Activity diagram

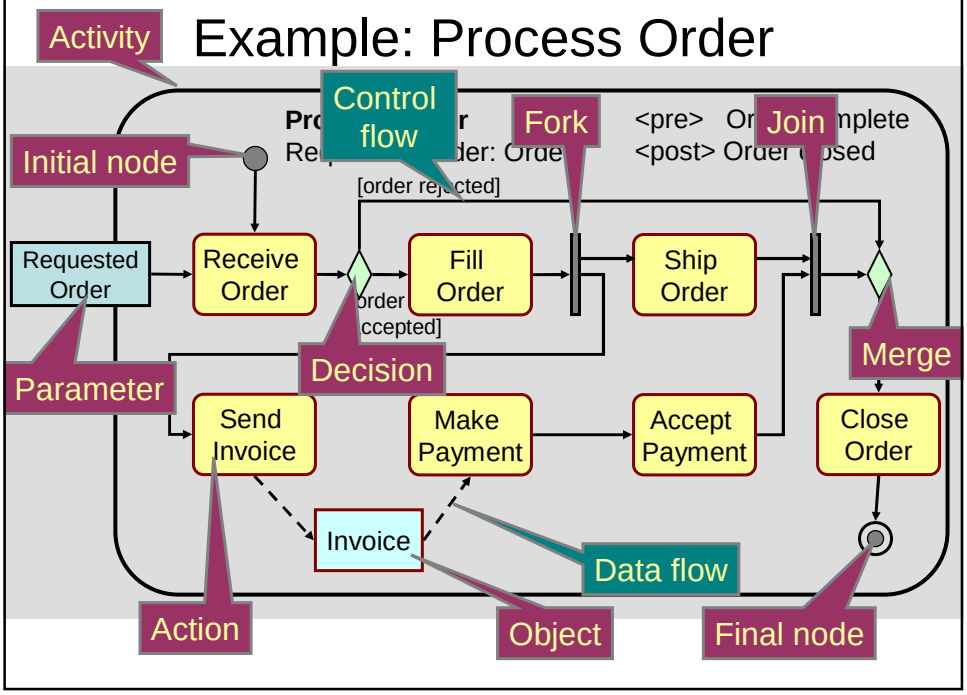
- Graph model:
  - Activity nodes:
    - Object (objektum)
    - Action (akció)
    - Control (vezérlés)
      - decision, merge (döntés)
      - fork, join (párh. tevékenység)
      - initial (kezdőpont)
      - final (végpont – minden út)
      - flow final (token nyelő – egy út)
  - Activity edges: flow
    - Data/object (Adatfolyam)
    - Control (Vezérlési folyam)



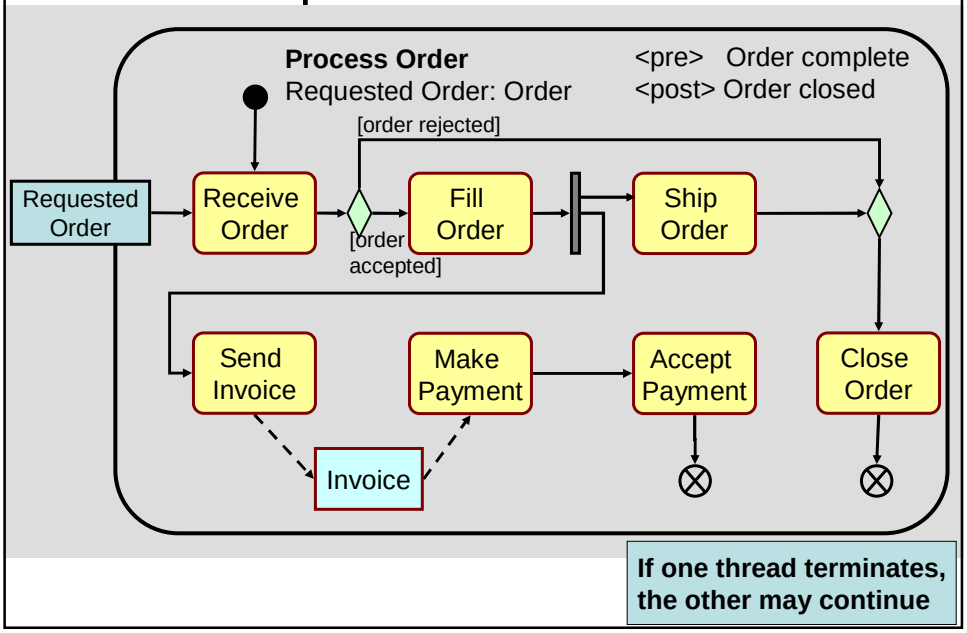
Example: Process Order



Example: Process Order



Example: Process Order II.

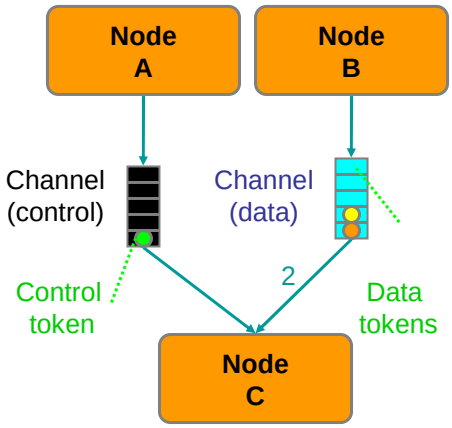


Data flow and Control flow

- Data Flow: data token
  - Object node ⇒ Action node
    - An object node is a channel / queue
    - An object may be linked to multiple action nodes
    - Output actions are competing for the data token (i.e. the object)
  - Type conformance: object type < input type of action
- Control flow: control token
  (ordering constraint between two actions)
  - All predecessor actions should be terminated prior to starting the current action
  - The current action should terminate prior to starting any of the successor actions

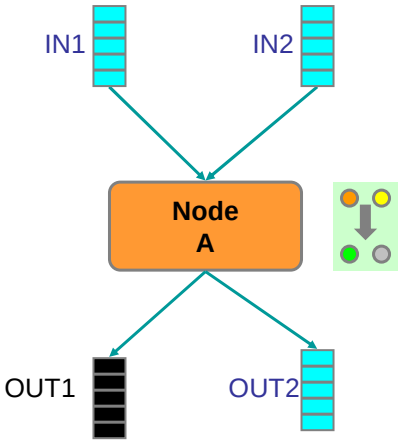
Semantics: Dataflow Networks

- Tokens:
  - control + several data
- Channel: object node
  - Stores the tokens
- Node: action node
  - Processing tokens
- Edges:
  - Flow of tokens
  - weights: how many tokens are in the flow at a time?
- Firing rule:
  - Behaviour of a node



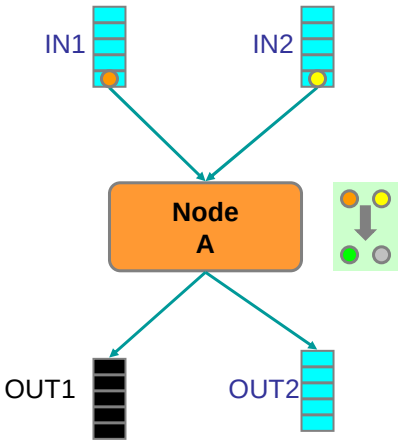
Semantics: Dataflow Networks

- Firing rule (cont.):
  - precondition:
    - input tokens + curr. state
  - postcondition
    - output tokens + new state



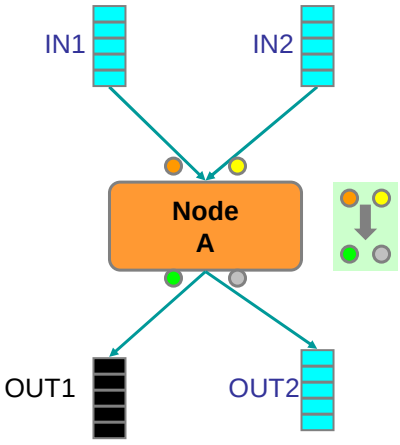
Semantics: Dataflow Networks

- Firing rule (cont.):
  - precondition:
    - input tokens + curr. state
  - postcondition
    - output tokens + new state
- Execution of a firing:
  - Is there token on all inputs with
    - Right amount?
    - Right type?



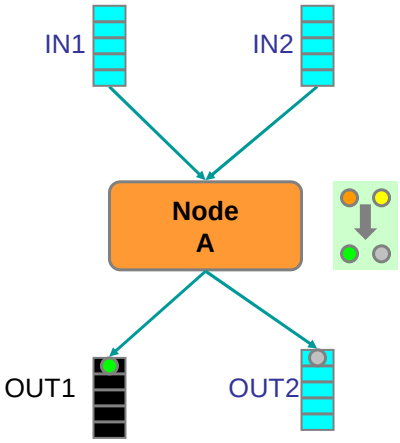
Semantics: Dataflow Networks

- Firing rule (cont.):
  - precondition:
    - input tokens + curr. state
  - postcondition
    - output tokens + new state
- Execution of a firing:
  - Is there token on all inputs with
    - Right amount?
    - Right type?
  - Execution of action

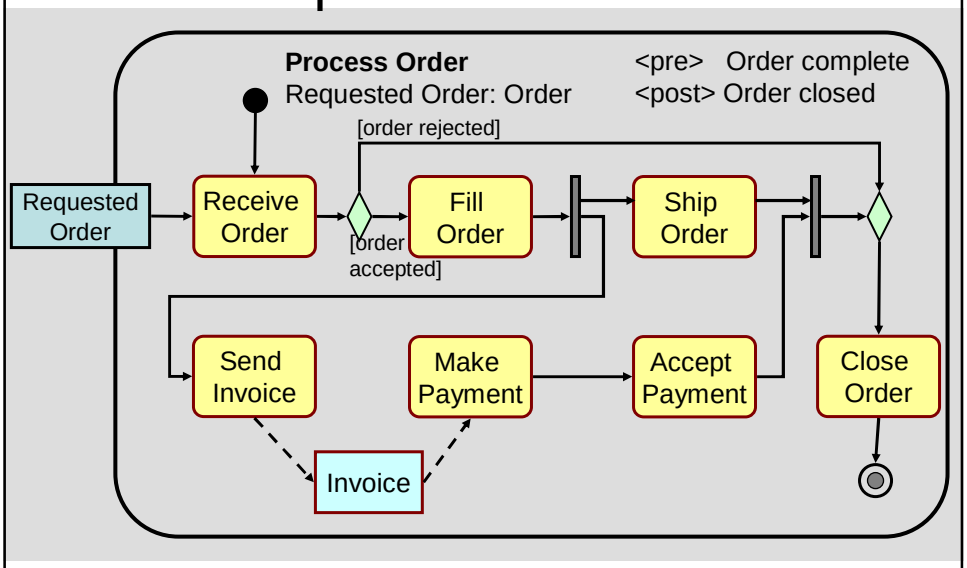


Semantics: Dataflow Networks

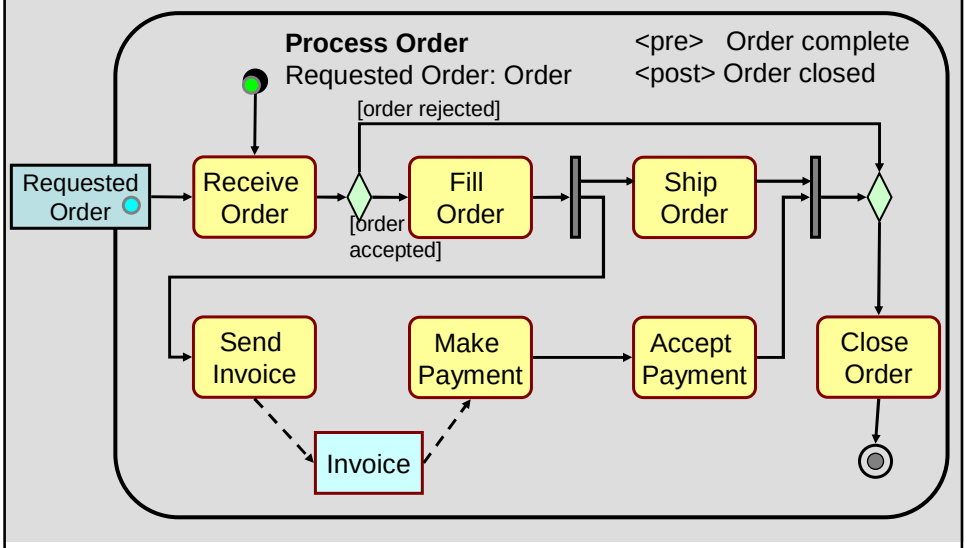
- Firing rule (cont.):
  - precondition:
    - input tokens + curr. state
  - postcondition
    - output tokens + new state
- Execution of a firing:
  - Is there token on all inputs with
    - Right amount?
    - Right type?
  - Execution of action
  - Sending the output tokens



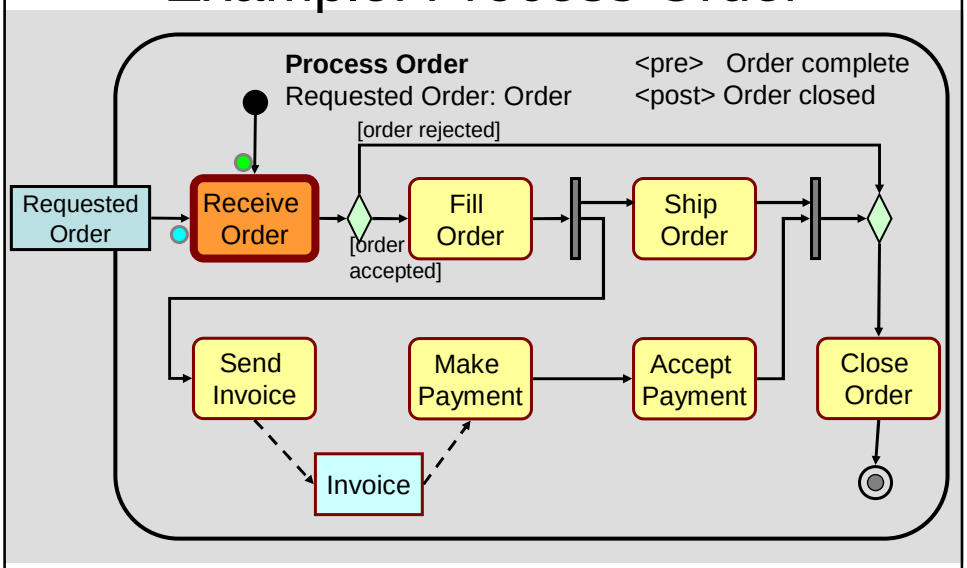
Example: Process Order



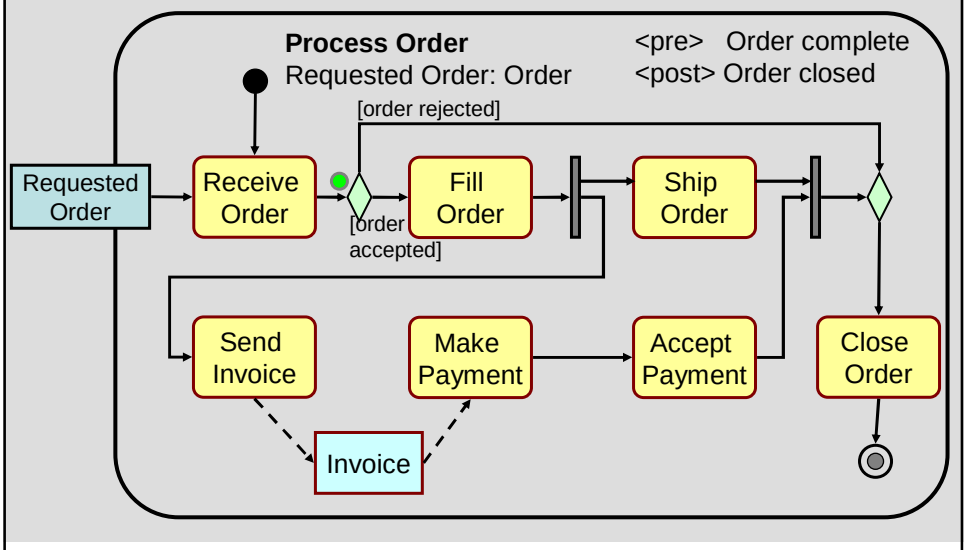
Example: Process Order



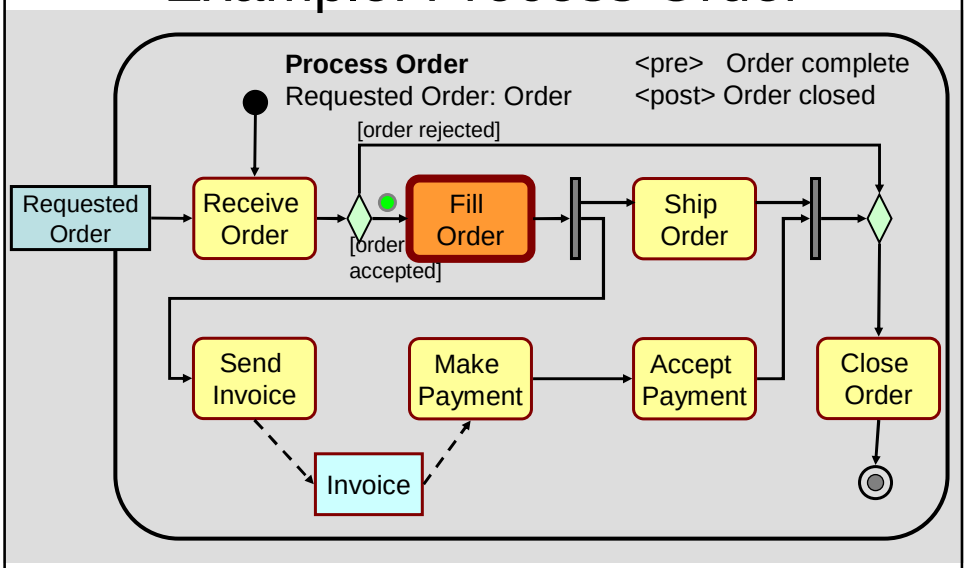
Example: Process Order



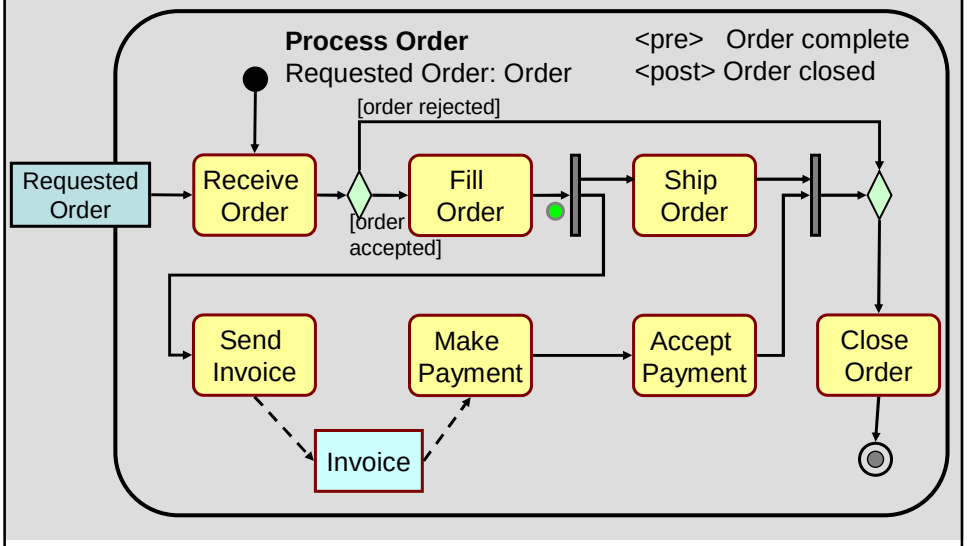
Example: Process Order



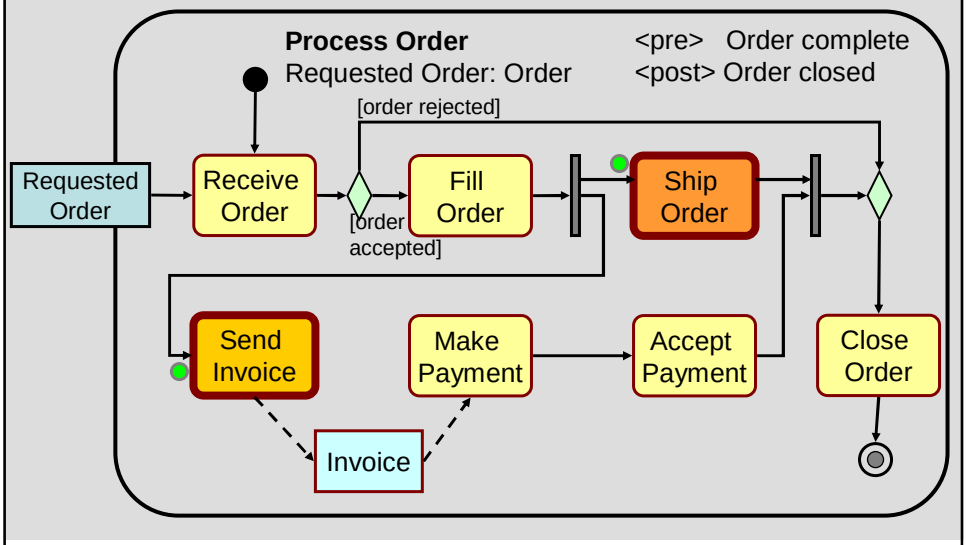
Example: Process Order



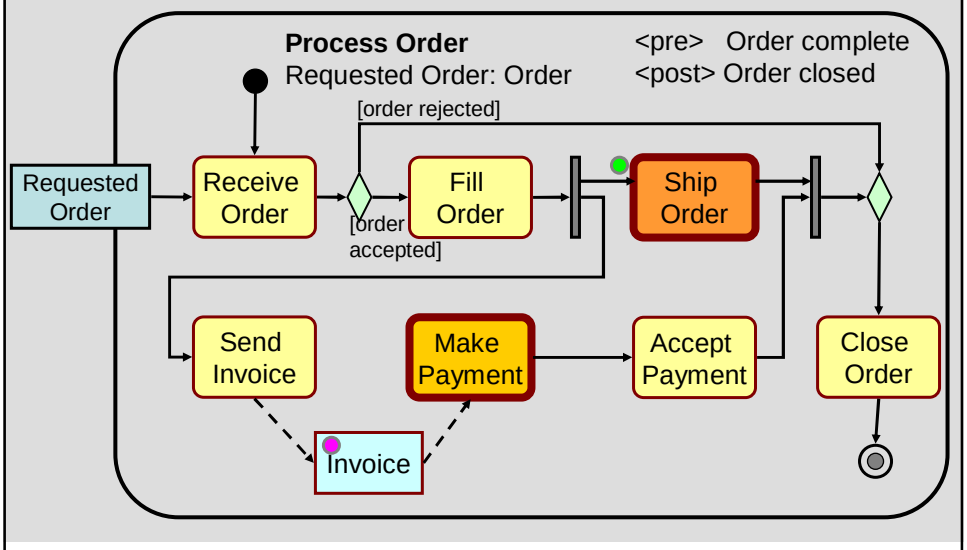
Example: Process Order



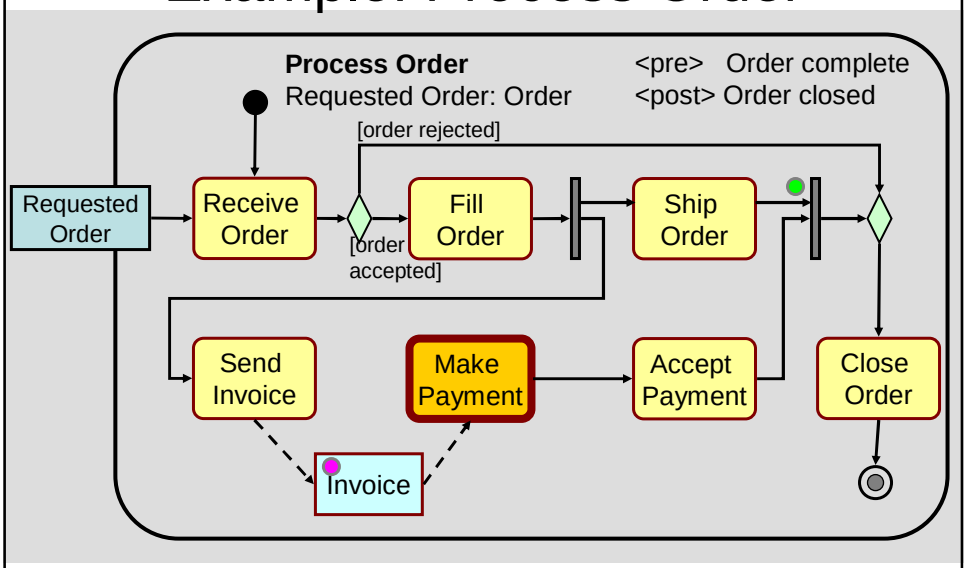
Example: Process Order



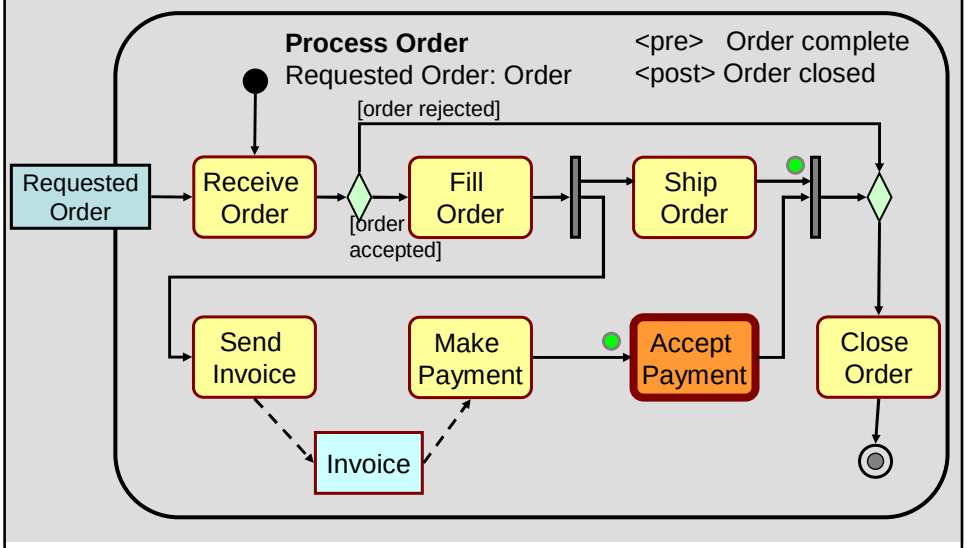
Example: Process Order



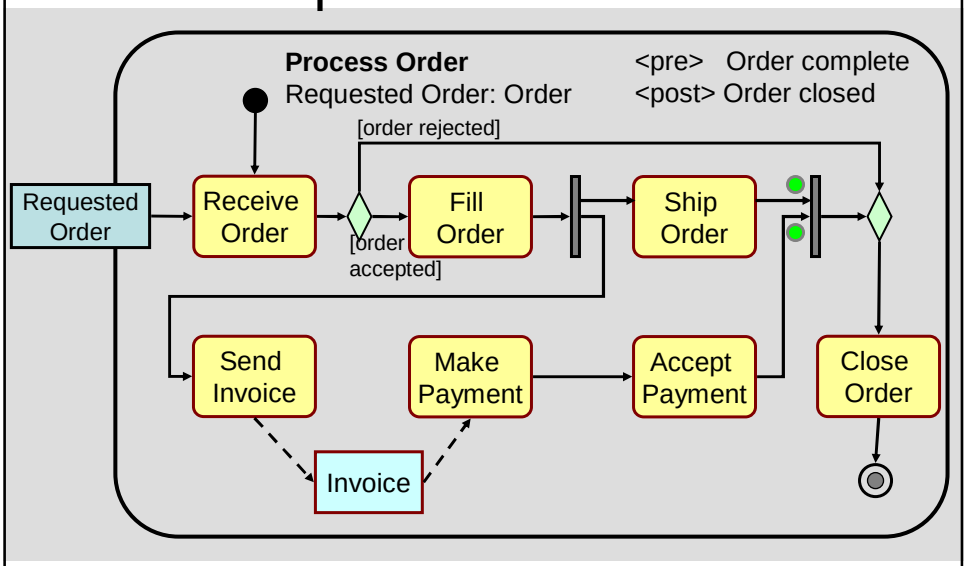
Example: Process Order



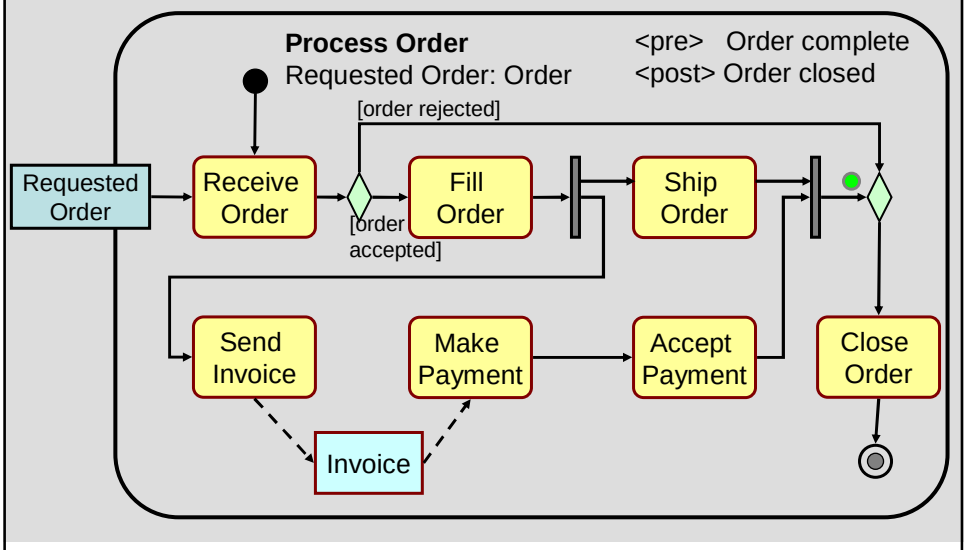
Example: Process Order



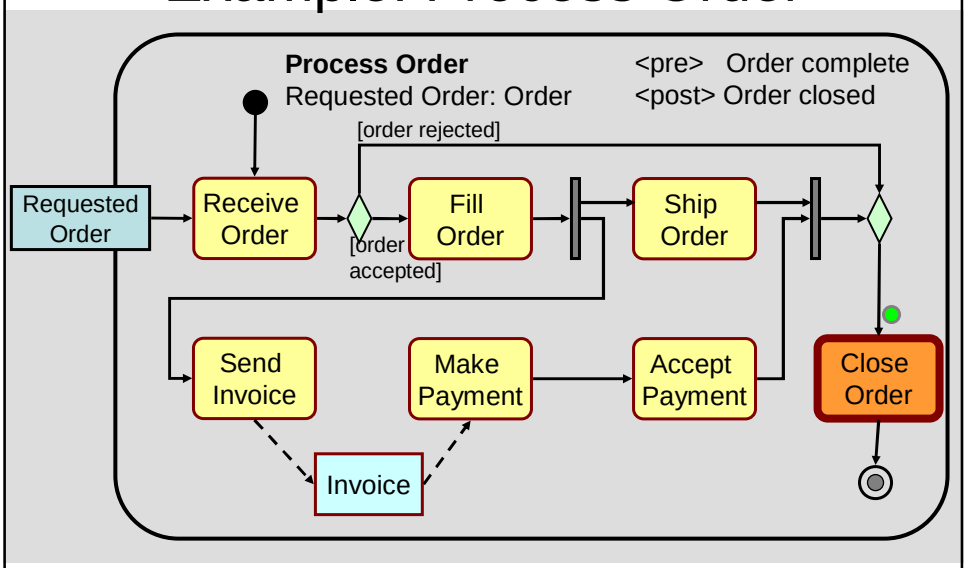
Example: Process Order



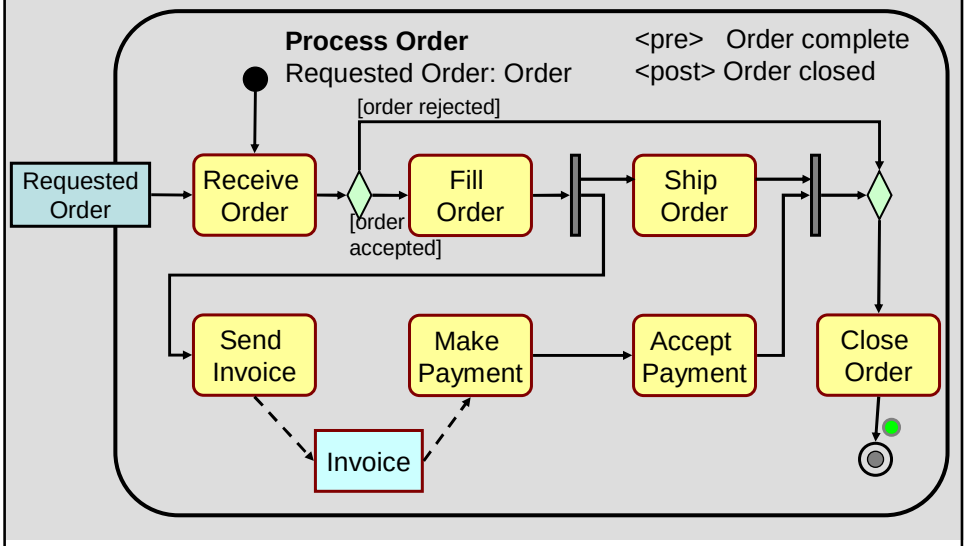
Example: Process Order



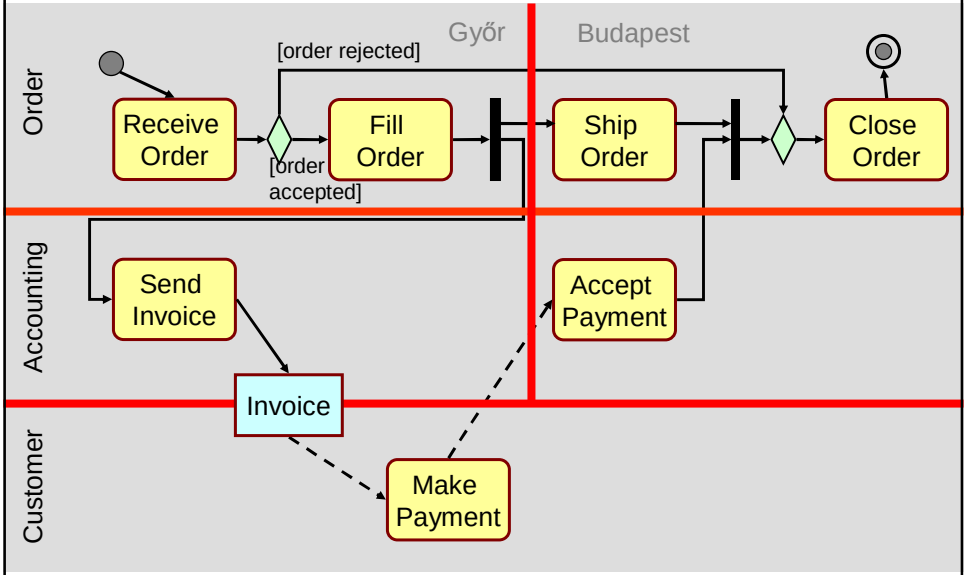
Example: Process Order



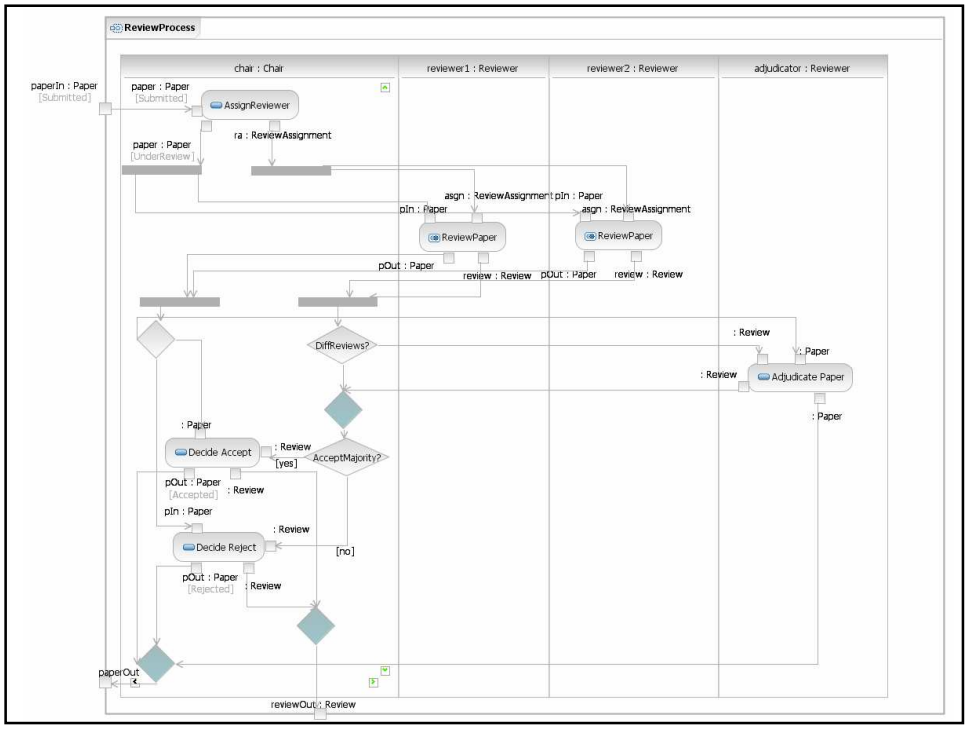
Example: Process Order



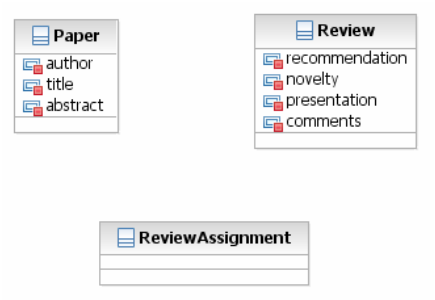
Partitioning of Activities



Example: Workflow of Paper review



Business Objects



Lifecycle of Business Objects