Object Oriented Analysis and Design with Java

Prof. Swati Shah

Department of Computer Science and Engineering

Acknowledgements: Significant portions of the information in the slide sets presented through the course in the class, are extracted from the prescribed text books, information from the Internet. Since these are only intended for presentation for teaching within PESU, there was no explicit permission solicited. We would like to sincerely thank and acknowledge that the credit/rights remain with the original authors/creators only

PES UNIVERSITY ONLINE

Object Oriented Analysis and Design with Java

Advanced state Modelling

Prof. Swati Shah

Department of Computer Science and Engineering

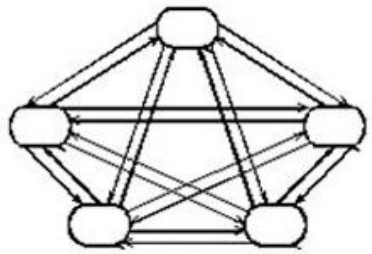
Advanced State Models

- Conventional state diagrams are **sufficient for describing simple systems but need additional power to handle large problems**.

- You can more richly model complex systems by using nest- ed state diagrams, nested states, signal generalization, and concurrency.

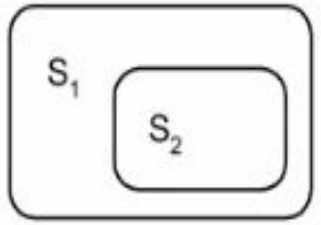
Problems with Flat State Models

- State diagrams have been often criticized because they allegedly are impractical for large problems.
- Unstructured state diagrams
- N independent boolean attributes that affect control. Representing such an object a single flat state diagram would require 2^n States. By partitioning the state into n independent state diagrams, however, only 2n states are required.
- State diagram in Figure in which n^2 transitions are needed to connect every state to every other state. It can be reduced as low as n transitions.

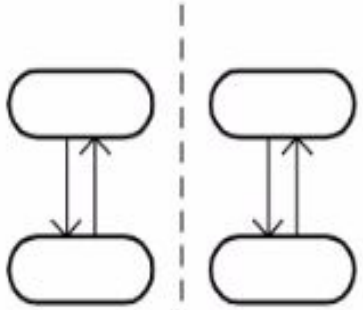


Features of Advanced state Diagram

- Two major features are introduced for controlling complexity and combinatorial explosion in state diagrams
  - **Nested** state diagrams
  - **Concurrent** state diagrams
- Many other features are also added
  - propagated transitions
  - broadcast messages
  - actions on state entry, exit
  - …



Nested State Diagrams



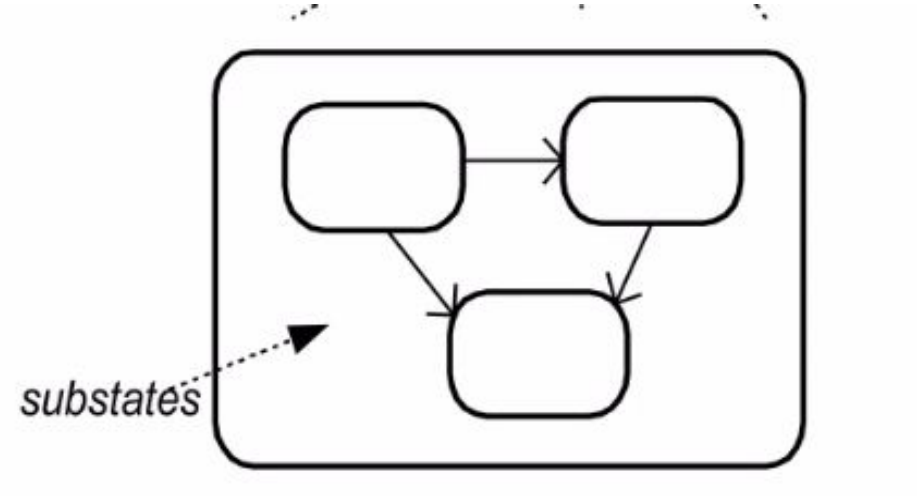
Concurrent State Diagrams

Nested State Diagram

- Activities in states are composite items denoting other lower-level state diagrams
- A lower-level state diagram corresponds to a sequence of lower-level states and events that are invisible in the higher-level diagram.

Super or Substate

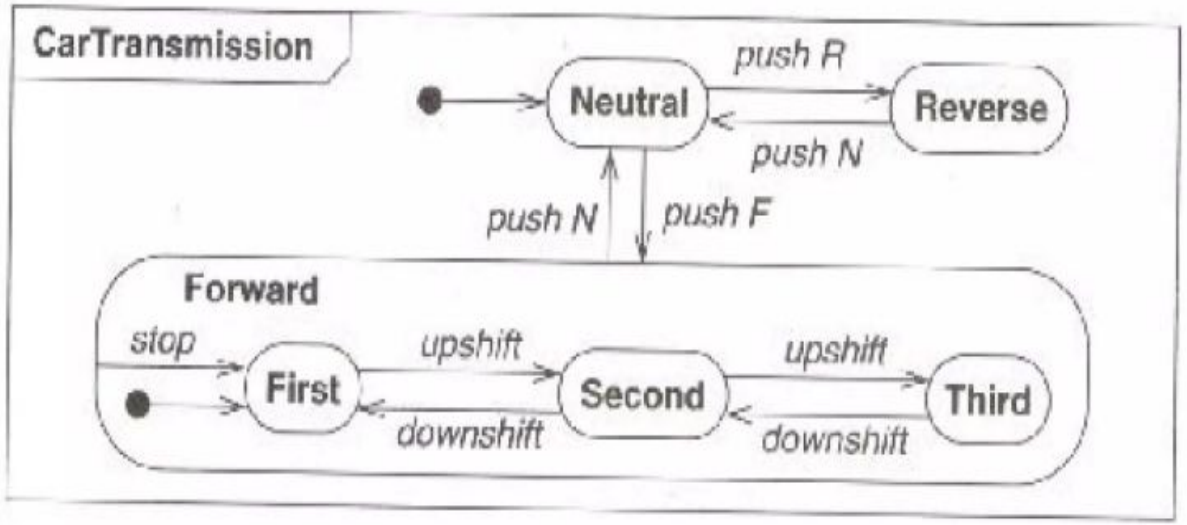
- When one state is complex, you can include substates in it.
  - drawn as nested rounded rectangles within the larger state



Nested state example

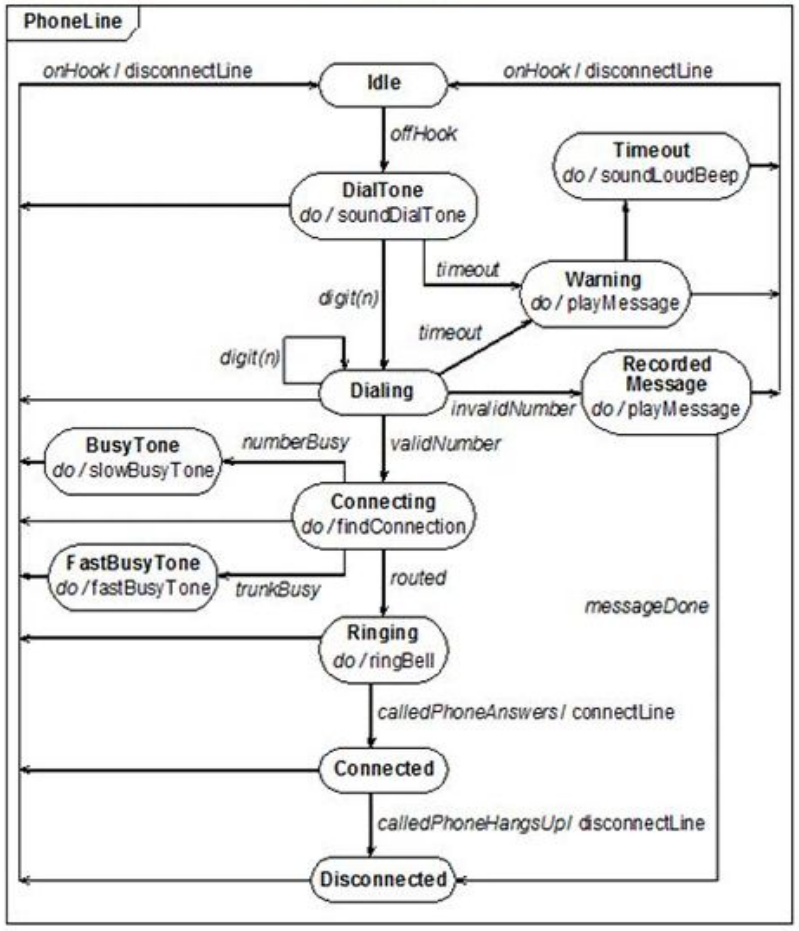
A state may be represented as nested substates.

- In UML, substates are shown by nesting them in a superstate box.
- A substate inherits the transitions of its superstate.

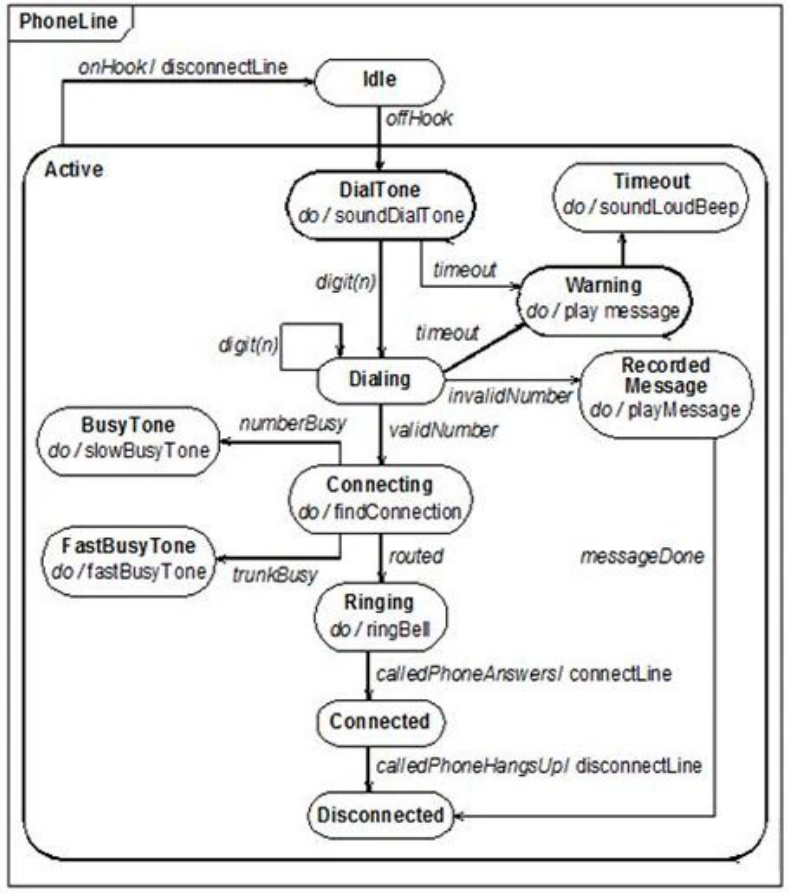


Simple state v/s Nested state

State Diagram for phone line with Activities

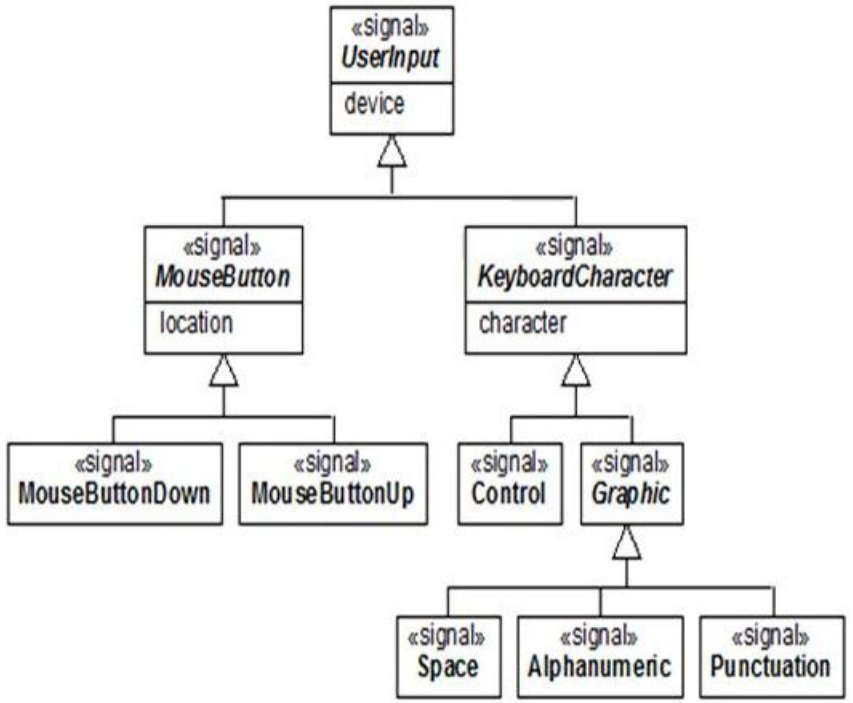


Nested states for a phone line



Signal Concurrency

- Organize signals into a generalization hierarchy with inheritance of signal attributes.



Concurrency

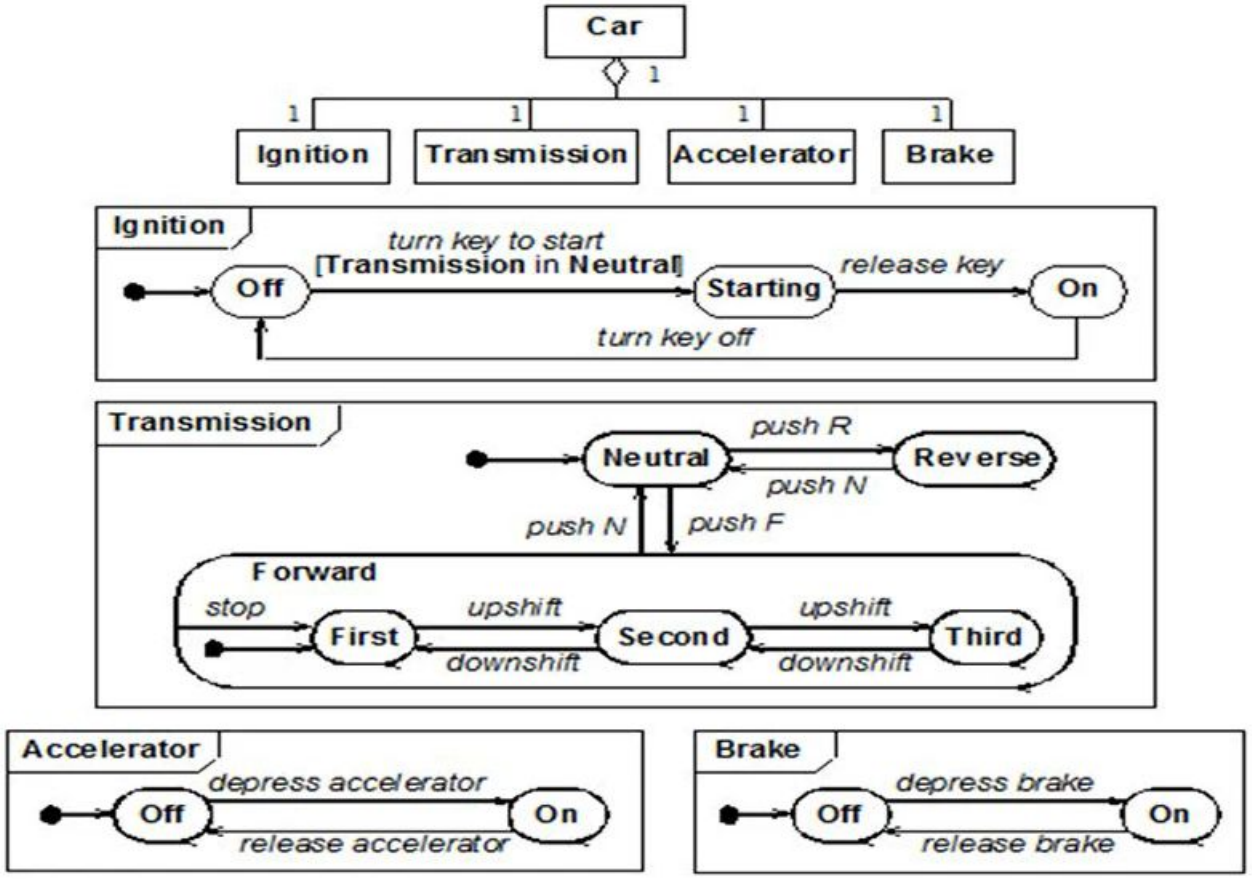
- State Models also Supports concurrency among objects. It supports two types of concurrency-

    1. Aggregation Concurrency
    2. Concurrency within an object

Aggregation Concurrency

- A state diagram for an assembly is a collection of state diagram, one for each part. The aggregate state corresponds to the combined states of all the parts.
- Aggregation is "**and-relationship**".
- Aggregate state is one state from the first diagram, and a state from second diagram and a state from each other diagram. In the more interesting cases, the part states interact.

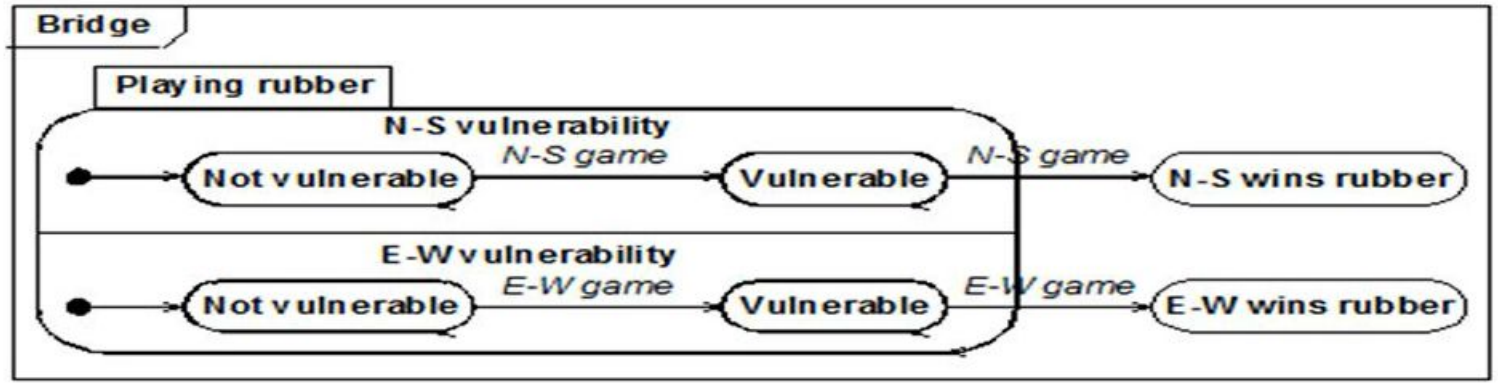
Aggregation Concurrency(Cont'd)



Concurrency within an object

- **The state model implicitly supports concurrency among objects**. In general, objects are autonomous entities that can act and change state independent of one another.

- Objects need not be completely independent and may be subject to shared constraints that cause some correspondence among their state changes.



THANK YOU

Prof Swati Shah

Department of Computer Science and Engineering

swatishahjaiswal@pes.edu