# Statecharts

- Used to model *behaviour* of Reactive Systems
  (event driven, react to internal and external stimuli)
- Visual formalism but formal and rigourous for
  1. documentation
  2. analysis
  3. simulation
  4. code generation
- Solves FSA problems:
  1. flat $\Rightarrow$ hierarchy ($\Rightarrow$ re-use)
  2. represent large number of transitions concisely
  3. represent large number of states concisely

# Statecharts references

- David Harel. **Statecharts: A Visual Formalism for Complex Systems.** Science of Computer Programming. vol. 8. 1987. pp. 231 - 274.
- David Harel and Amnon Naamad. **The STATEMATE semantics of statecharts.** ACM Transactions on Software Engineering and Methodology (TOSEM) vol. 5. Issue 4. October 1996. pp.293 - 333.
- Michael von der Beeck. **A comparison of statechart variants.** In Formal Techniques in Real-Time and Fault-Tolerant Systems. L. de Roever and J. Vytopil, Eds. Lecture Notes in Computer Science vol. 863. Springer-Verlag, New York, pp. 128 - 148. 1994
- Michael von der Beeck. **A structured operational semantics for UML-statecharts.** Software and Systems Modeling. Volume 1, No. 2 pp. 130 - 141 . December 2002.
- Statechart revisions in UML 2.0 (`www.omg.org`).

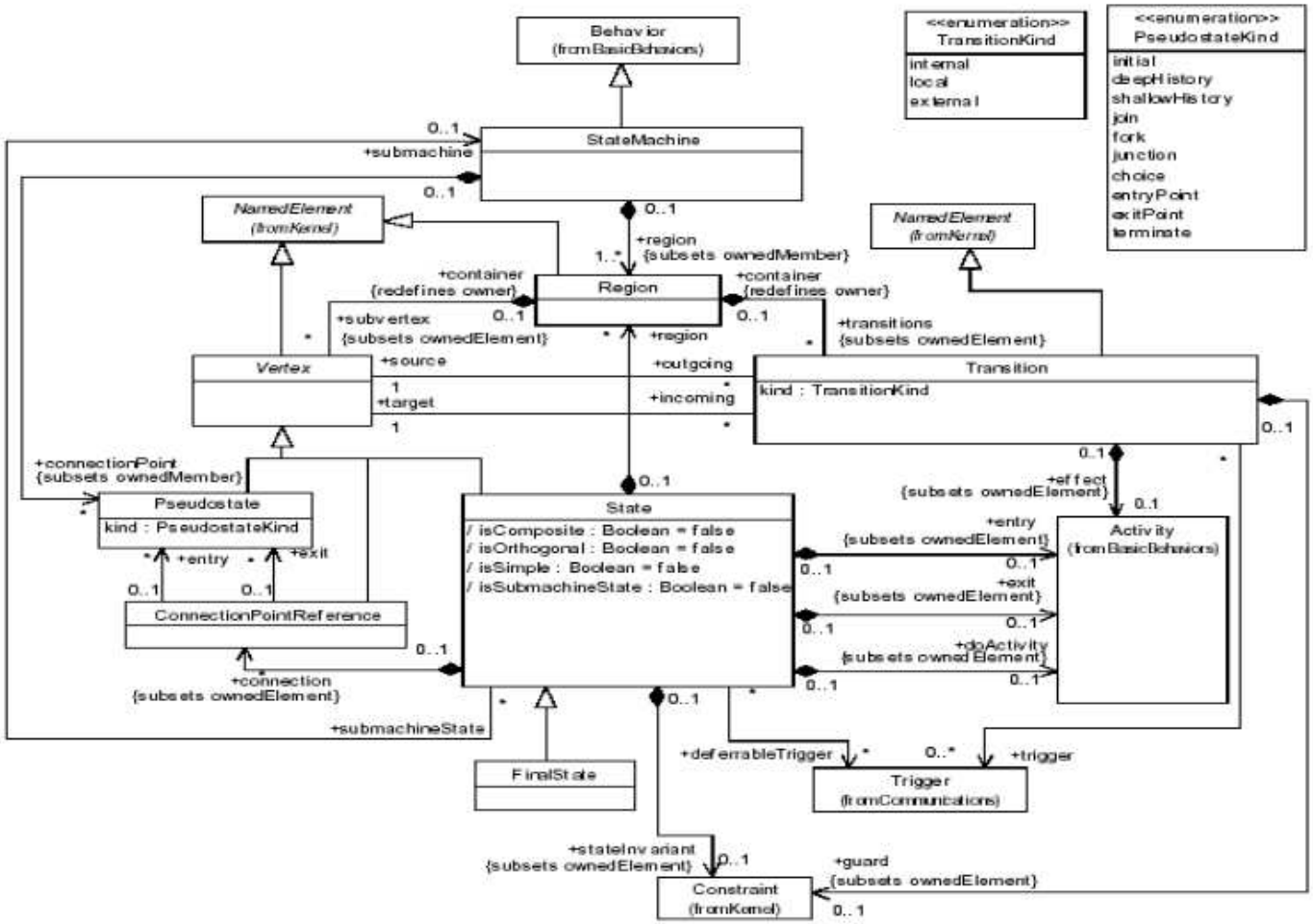
# Statecharts tools

- STATEMATE [http://www.ilogix.com/products/magnum/index.cfm]
- Rhapsody [http://www.ilogix.com/products/rhapsody/rhap_inc.cfm]
- Rational Rose [http://www.rational.com]
- Stateflow [http://www.mathworks.com/products/stateflow/]
- BetterState Lite [http://www.windriver.com/products/html/betterstate.html]
- XJTek [http://www.xjtek.com/products/xjcharts/]
- Poseidon for UML [http://www.GentleWare.com]
- ArgoUML [http://argouml.tigris.org/]
- visualSTATE [http://www.iar.com/Products/VS/]
- SVM [http://msdl.cs.mcgill.ca/people/tfeng/research.html]

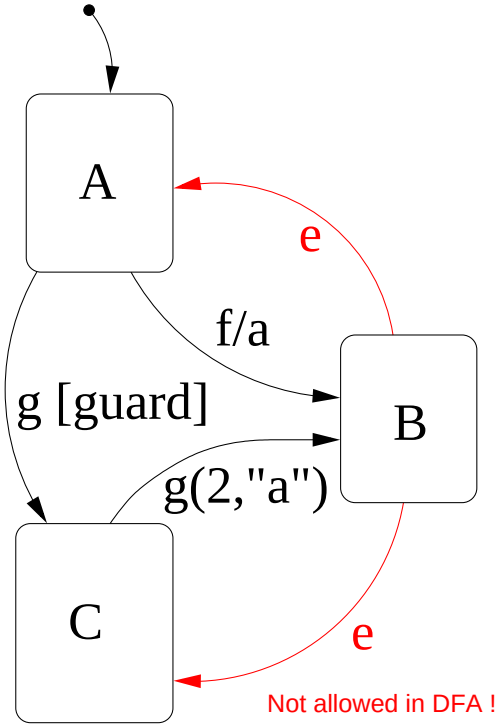
# Statecharts =

1. Deterministic Finite State Automata (FSA, DFA)
2. Depth
3. Orthogonality
4. Broadcast
5. History
6. Time
7. syntactic sugar: entry/exit actions

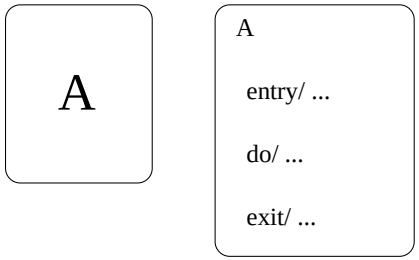
# UML 2.0 Statecharts abstract syntax



# Deterministic Finite State Automata (FSA, DFA)

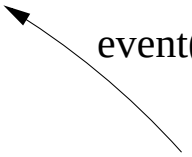



State




Default state (pseudo state)

transition



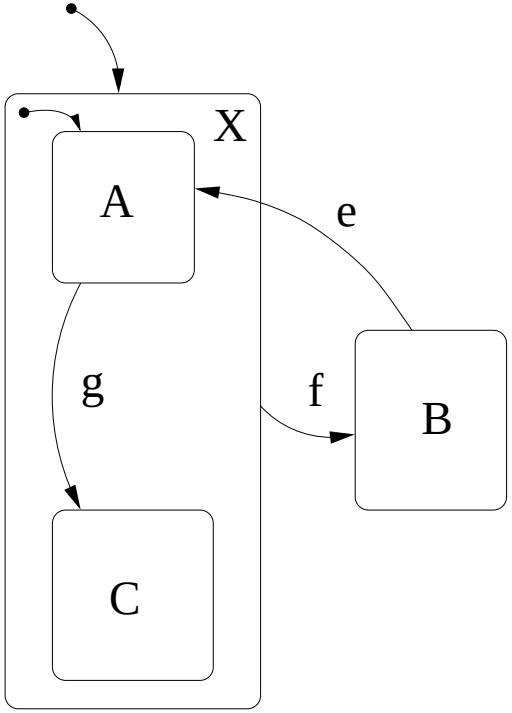
event(params) [guard] /output_action(params)

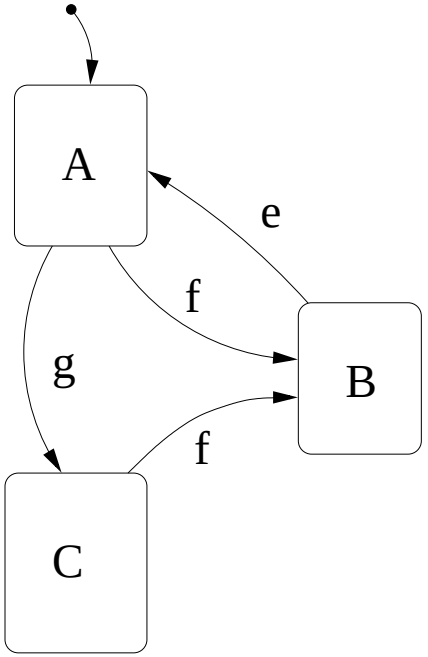
# Depth (XOR)

Statechart with depth

give meaning
by "flattening"

State automaton



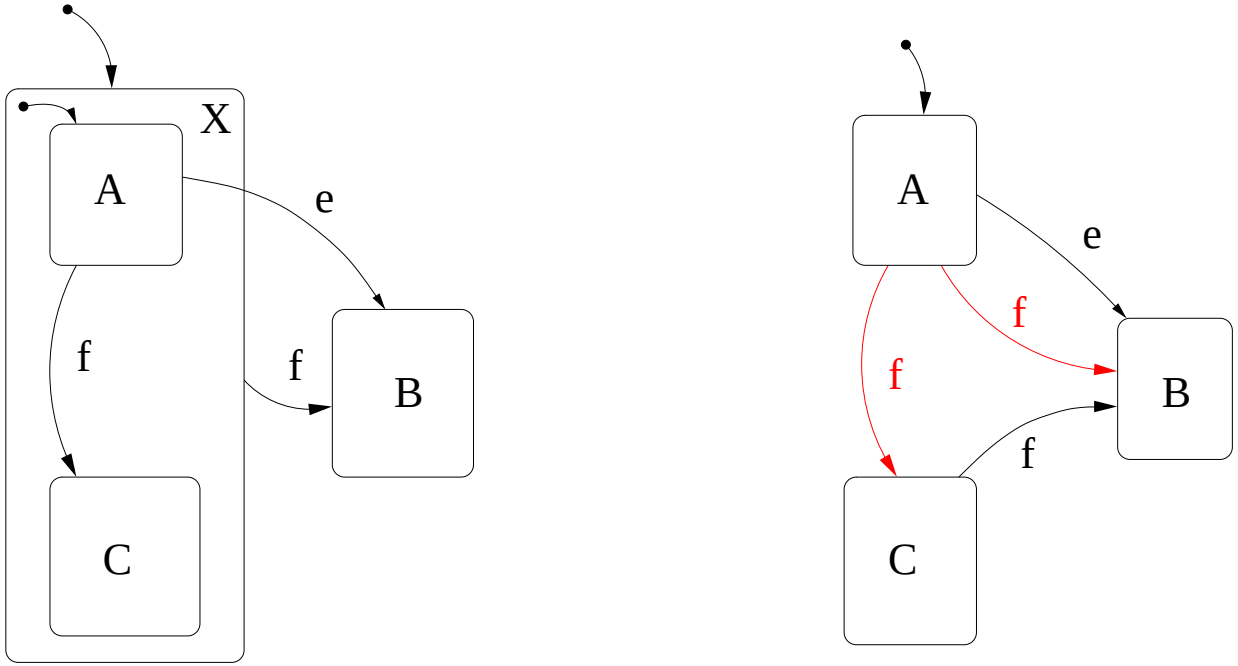



# Depth and determinism

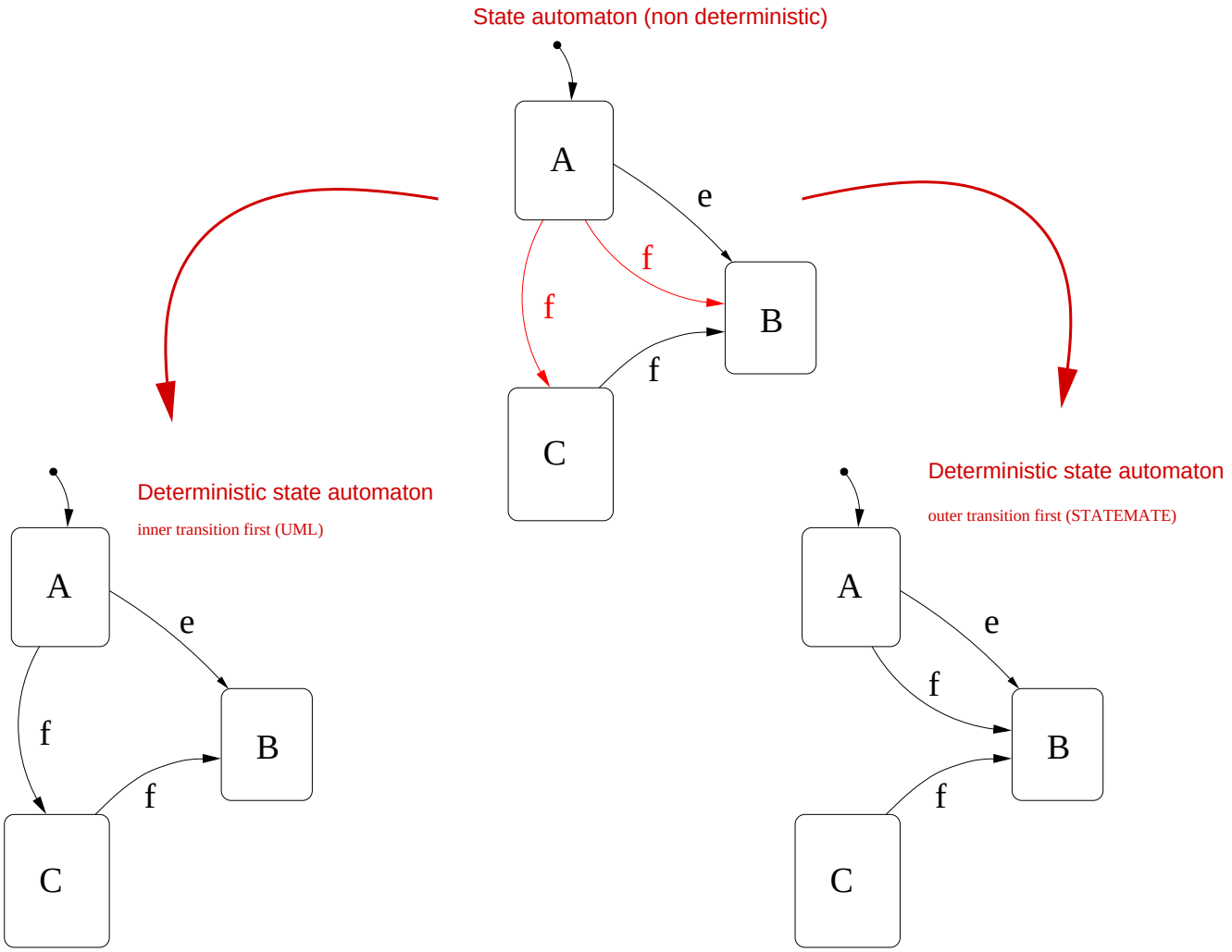
Statechart with depth

give meaning
by "flattening"

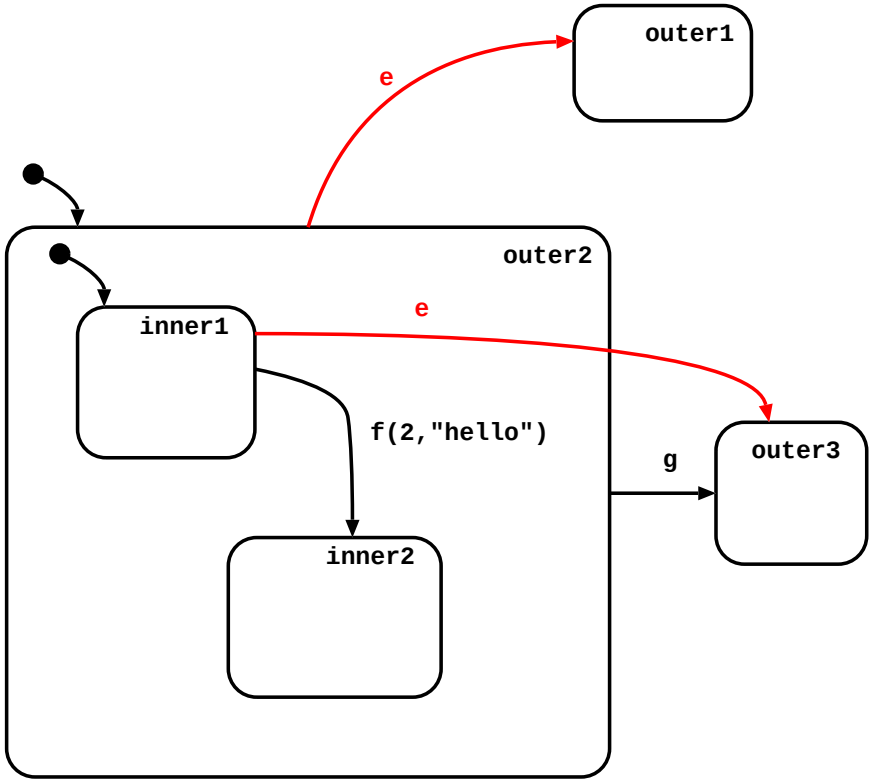
State automaton (non deterministic)

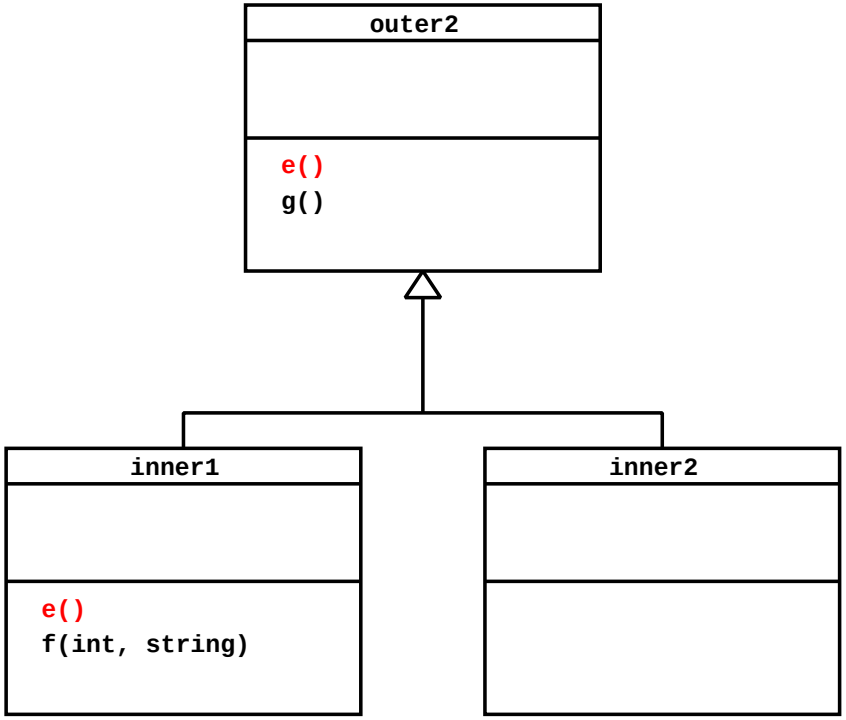


# Depth and determinism



# Why “inner-first” in UML Statecharts ?



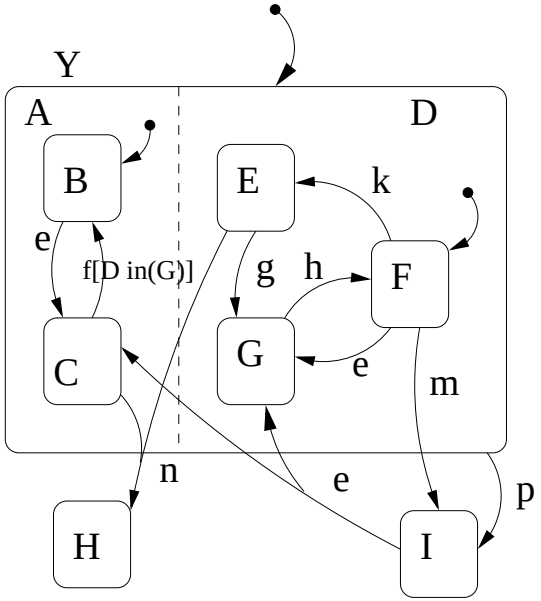



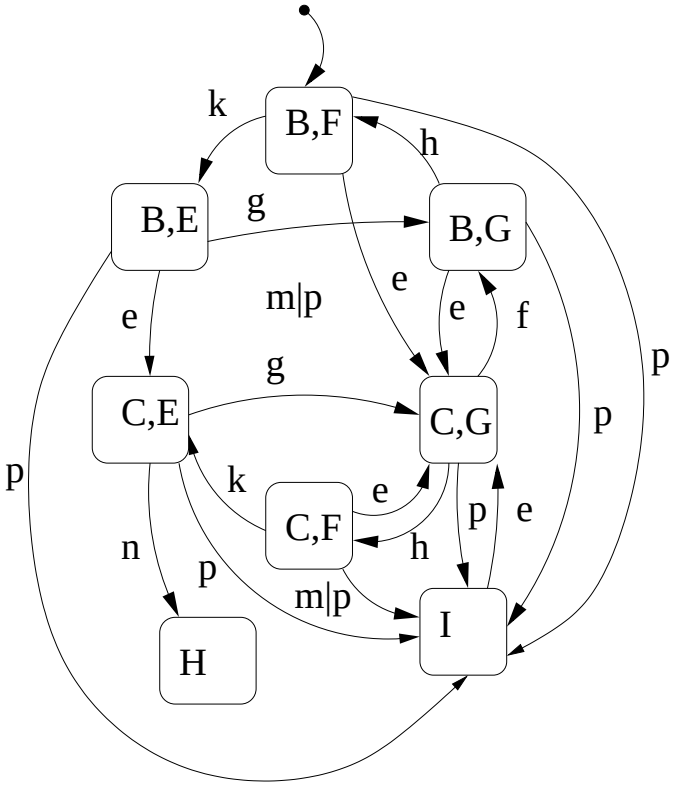
# Orthogonality (AND)

Statechart with Orthogonal Components

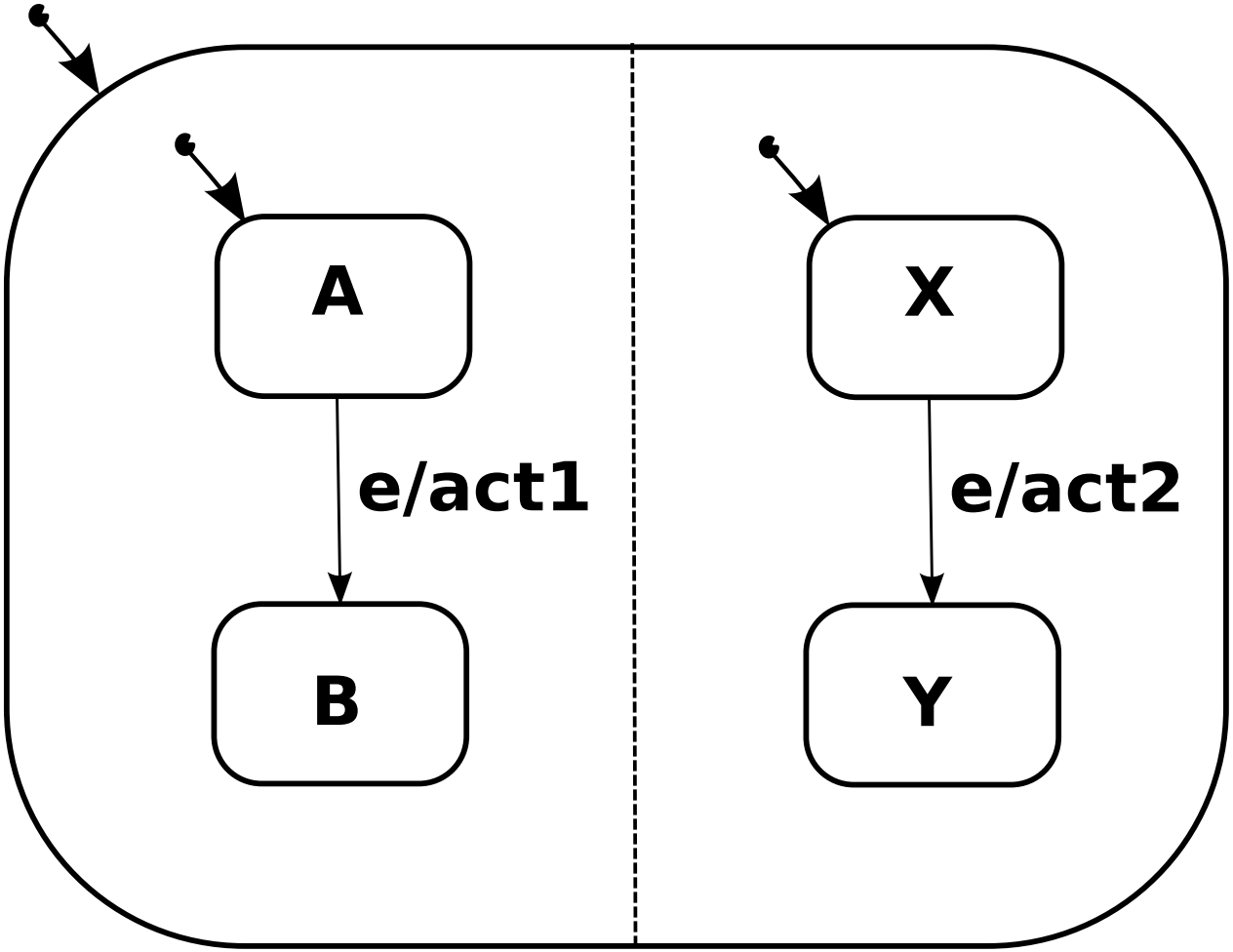
meaning

State Automaton

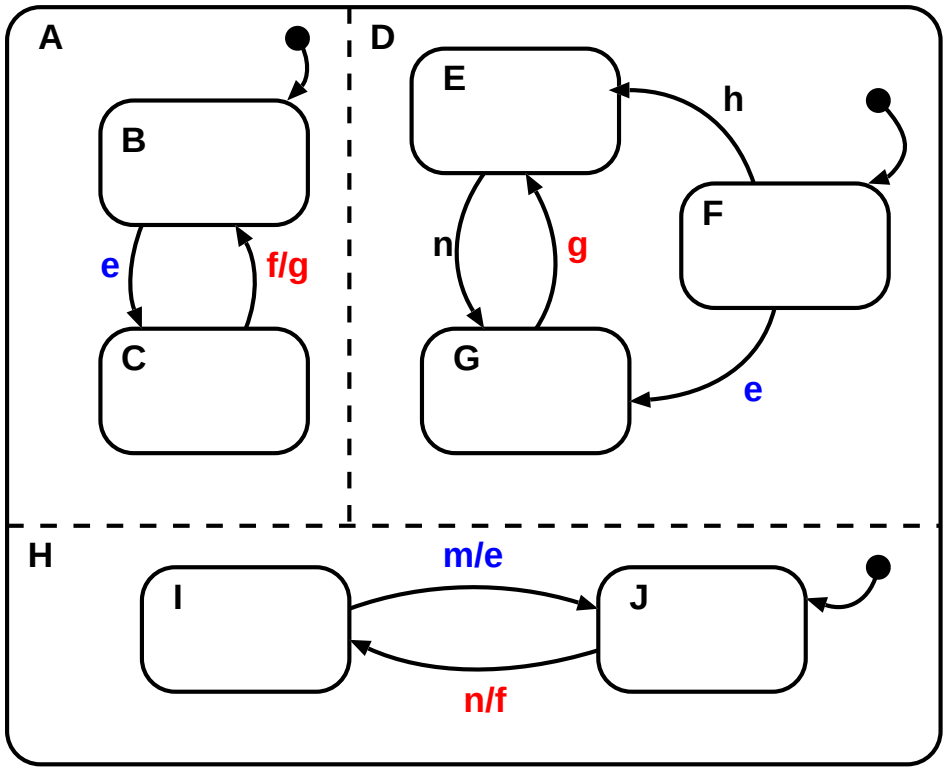


# Orthogonality and synchronization

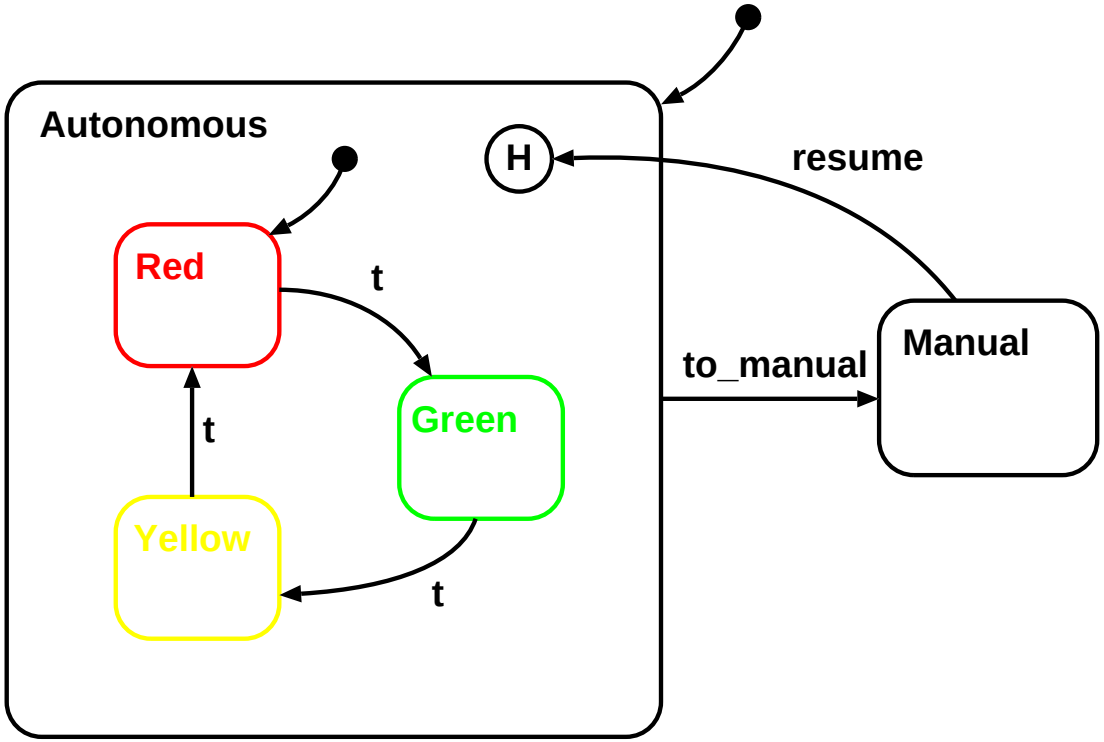


# Broadcasting (output events)

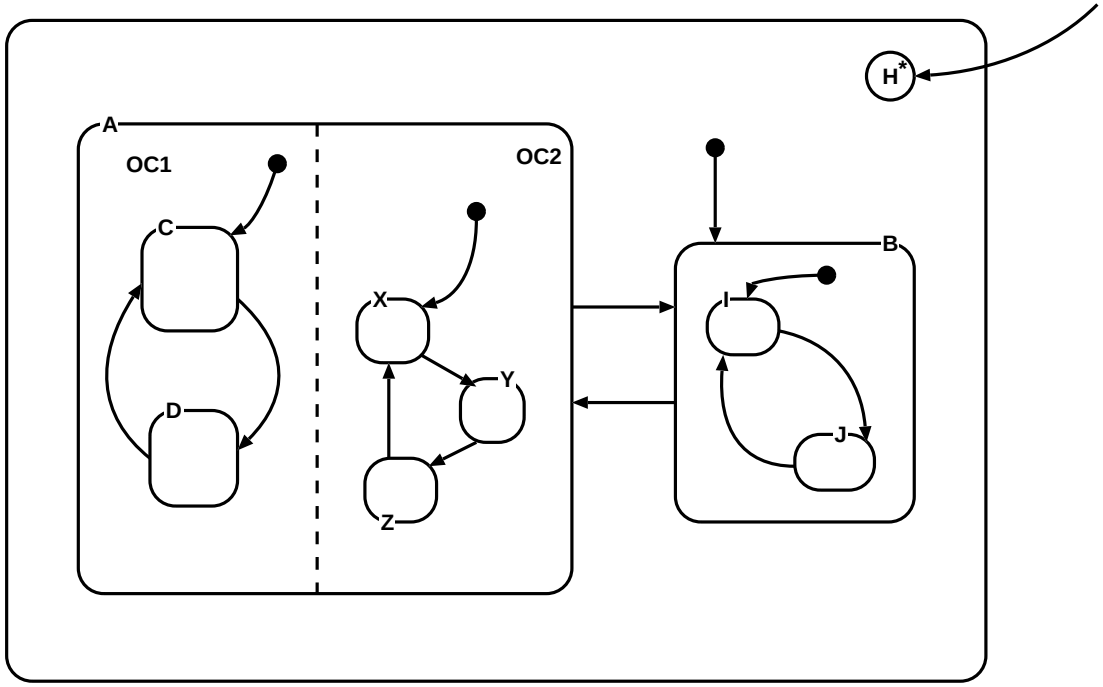



Input Segment: **nmnn**

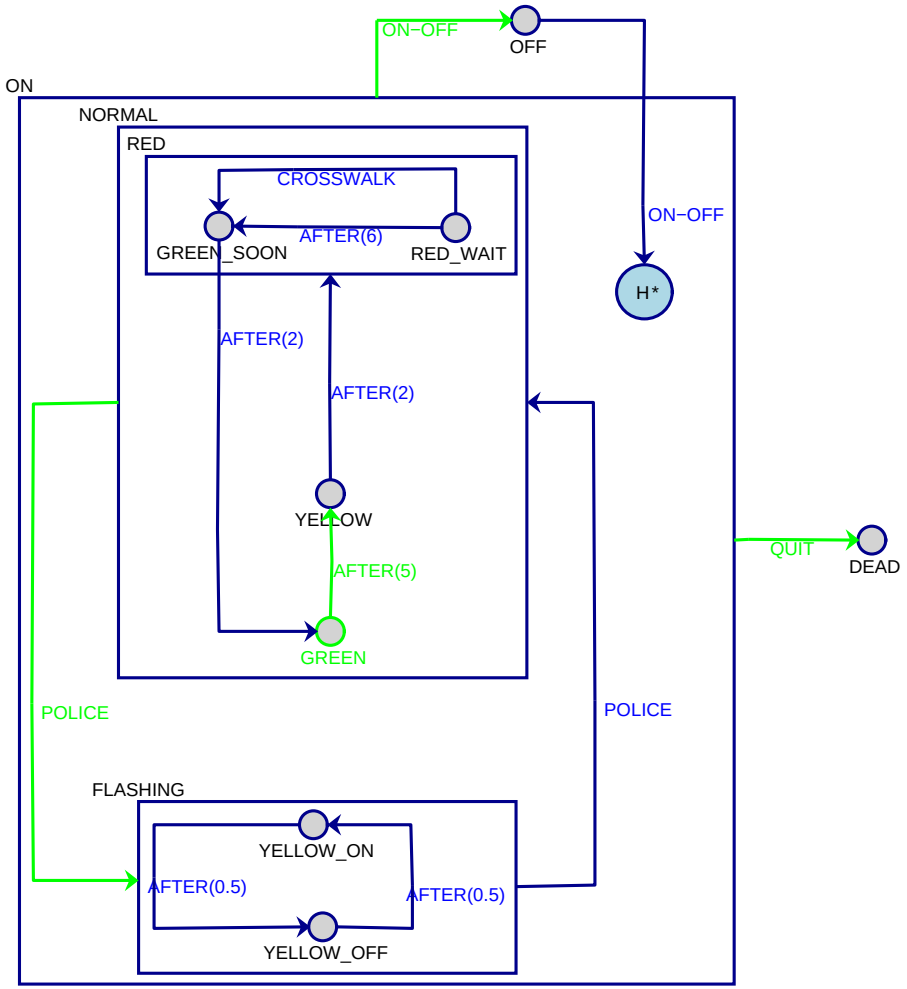
# History States



# Deep History



# Traffic Light Example, visual syntax (AToM$^3$)



# Traffic Light Example, textual syntax

```
# Statechart description generated by SVM-AToM3-plugin, written by Thomas Feng

STATECHART:
  OFF
  DEAD
  ON [DS] [HS*]
    NORMAL [DS]
      GREEN
      YELLOW
      RED [DS]
        RED_WAIT [DS]
        GREEN_SOON
    FLASHING
      YELLOW_ON [DS]
      YELLOW_OFF

ENTER:
  N: OFF
  O: theglobals.GUI.send('DEAD')

ENTER:
  N: ON.NORMAL.YELLOW
  O: theglobals.GUI.send('YELLOW')

ENTER:
  N: ON.NORMAL.GREEN
  O: theglobals.GUI.send('GREEN')

ENTER:
  N: DEAD
  O: theglobals.GUI.tk.destroy()
     theglobals.EXECUTOR.shutdown()

ENTER:
  N: ON.FLASHING.YELLOW_OFF
  O: theglobals.GUI.send('DEAD')

ENTER:
  N: ON.FLASHING.YELLOW_ON
  O: theglobals.GUI.send('YELLOW')

ENTER:
  N: ON.NORMAL.RED
  O: theglobals.GUI.send('RED')

TRANSITION:
  S: ON.NORMAL.RED.RED_WAIT
  N: ON.NORMAL.RED.GREEN_SOON
  T: 6
  C: 1

TRANSITION:
  S: ON.NORMAL.RED.RED_WAIT
  N: ON.NORMAL.RED.GREEN_SOON
  E: CROSSWALK
  C: 1

TRANSITION:
  S: ON.NORMAL.GREEN
  N: ON.NORMAL.YELLOW
  T: 5
  C: 1

TRANSITION:
  S: ON.NORMAL.YELLOW
  N: ON.NORMAL.RED
  T: 2
  C: 1

TRANSITION:
  S: ON.NORMAL.RED.GREEN_SOON
  N: ON.NORMAL.GREEN
  T: 2
  C: 1

TRANSITION:
  S: ON.FLASHING.YELLOW_OFF
  N: ON.FLASHING.YELLOW_ON
  T: 0.5
  C: 1

TRANSITION: [HS]
  S: OFF
  N: ON
  E: ON-OFF
  C: 1

TRANSITION:
  S: ON
  N: OFF
  E: ON-OFF
  C: 1

TRANSITION:
  S: ON.FLASHING
  N: ON.NORMAL
  E: POLICE
  C: 1

TRANSITION:
  S: ON.NORMAL
  N: ON.FLASHING
  E: POLICE
  C: 1

TRANSITION:
  S: ON
  N: DEAD
  E: QUIT
  C: 1

TRANSITION:
  S: ON.FLASHING.YELLOW_ON
  N: ON.FLASHING.YELLOW_OFF
  T: 0.5
  C: 1
```

# UML 2.0: Class Diagrams and Statecharts