SWEN 223
Software Engineering Analysis

UML State Diagrams

Thomas Kühne
Victoria University of Wellington
Thomas.Kuehne@ecs.vuw.ac.nz, Ext. 5443, Room Cotton 233

State Diagrams

Applicability

- can be used to specify the states exhibited by an object or the system
- determine the responses (state transitions) to outside stimuli (events)
- are concerned with when operations execute, rather than what operations do, or how they are implemented

Applications

Reactive Behaviour

- Events
  - » make object/system transition between states
  - » are enabled/disabled, depending on state
  - » yield different actions/transition, depending on state

Areas

- Broad Range of Applications
  - » computer programs, business processes, protocols, web page navigation, …

Foundation

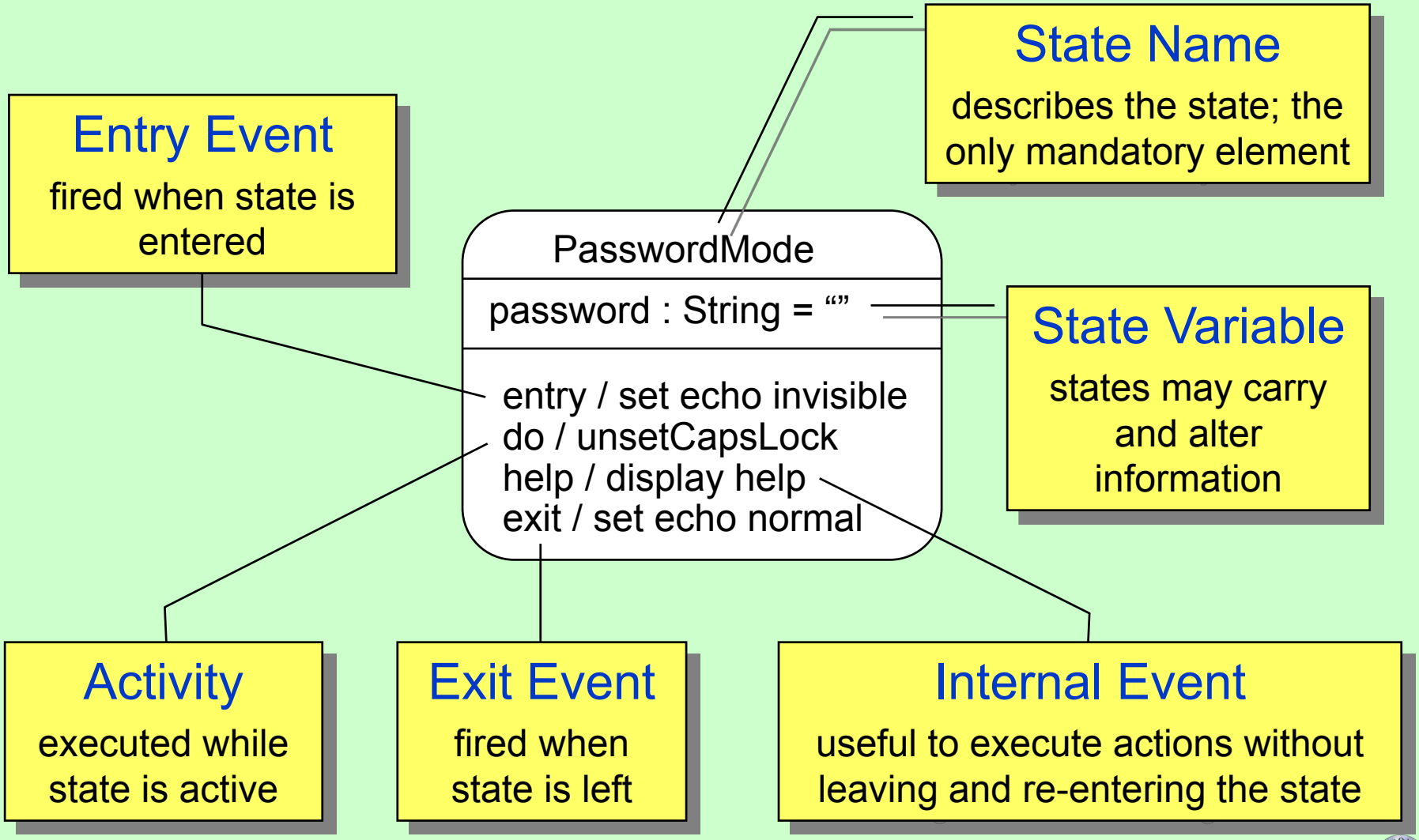
Finite State Machines

- States and Events
  - » limited expressiveness
  - » can “recognise” regular languages
  - » can also be represented as state transition tables

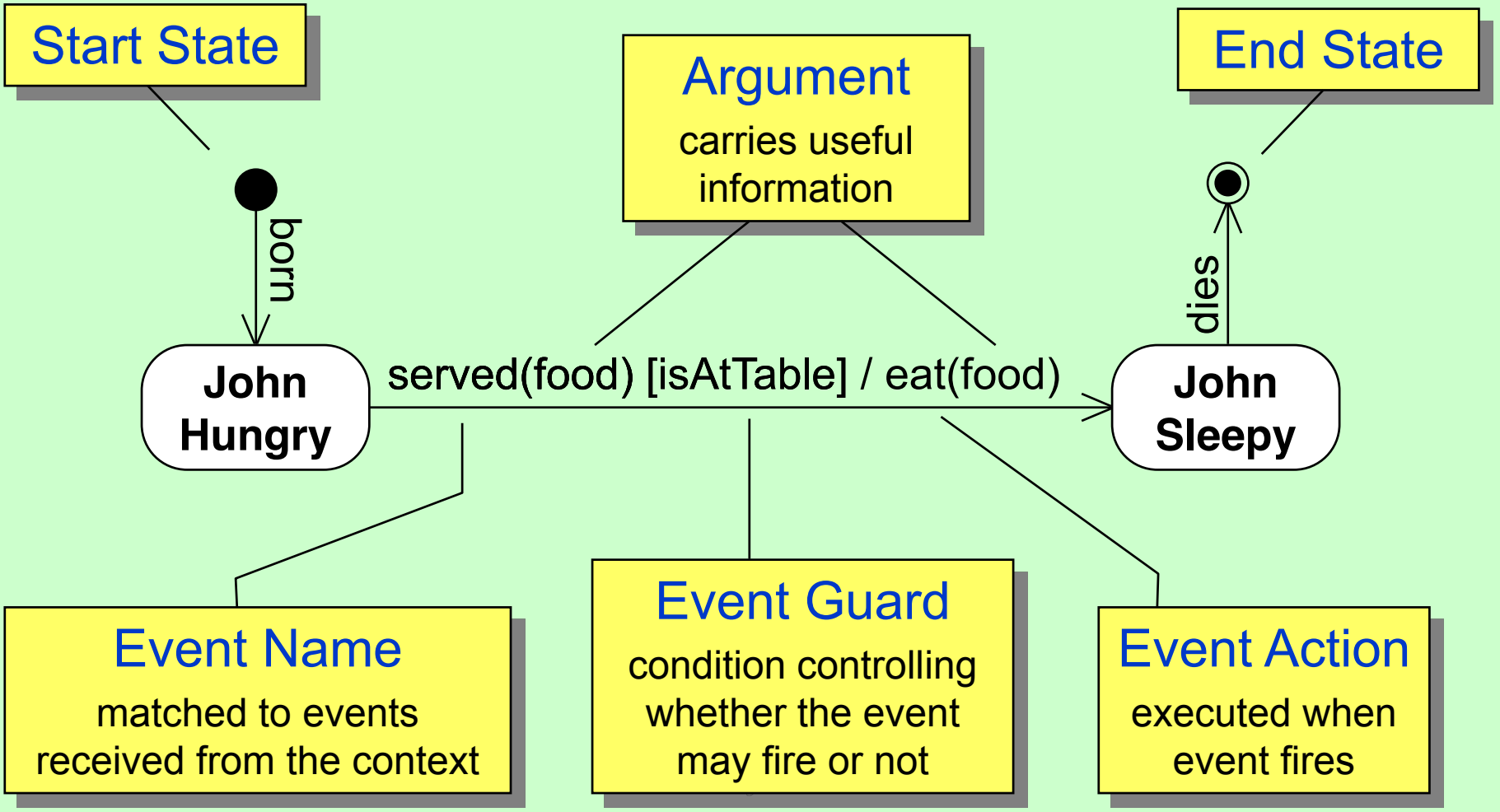
Harel Statecharts

- Multiple States
  - » extremely useful to structure state diagrams

States


State Name
describes the state; the only mandatory element
Entry Event
fired when state is entered
PasswordMode
password : String = “”
entry / set echo invisible
do / unsetCapsLock
help / display help
exit / set echo normal
State Variable
states may carry and alter information
Activity
executed while state is active
Exit Event
fired when state is left
Internal Event
useful to execute actions without leaving and re-entering the state

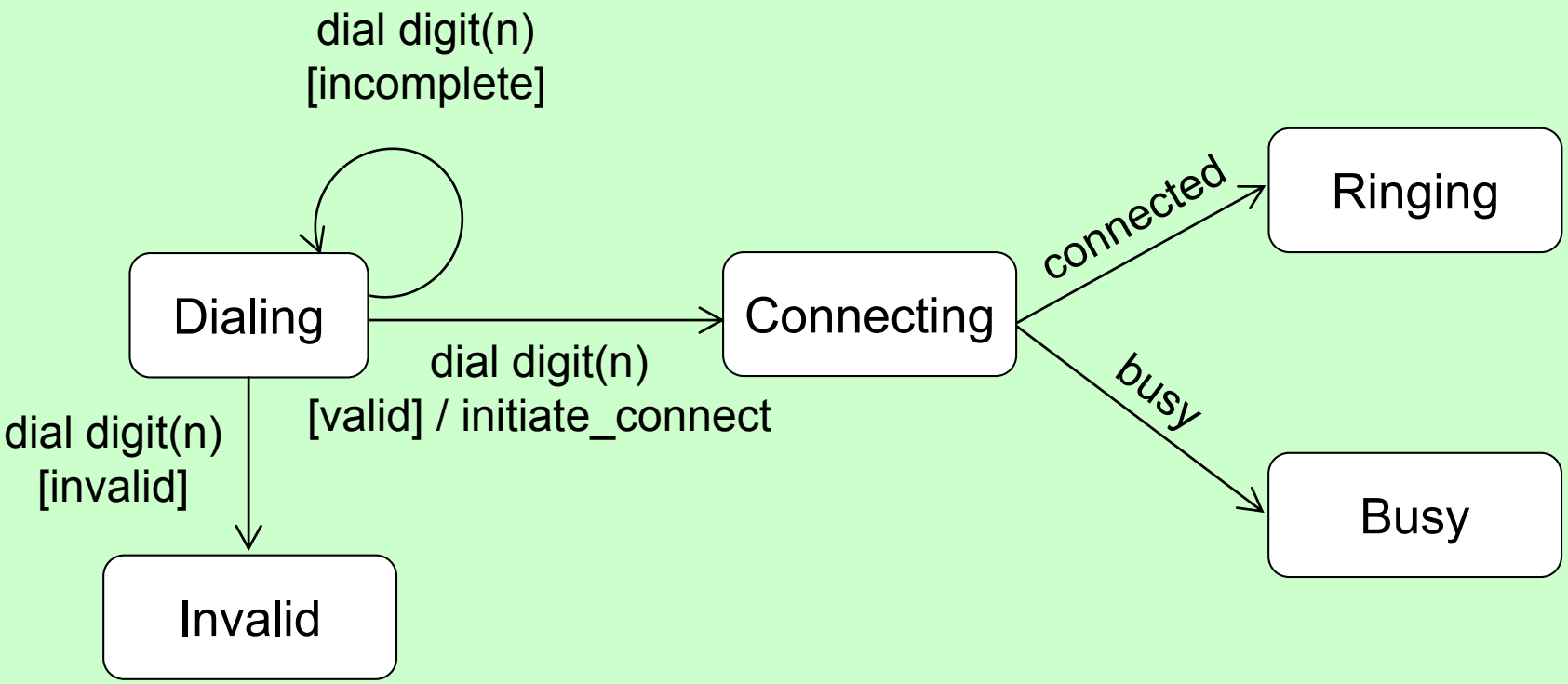
Transitions



(Some) Kinds of Events

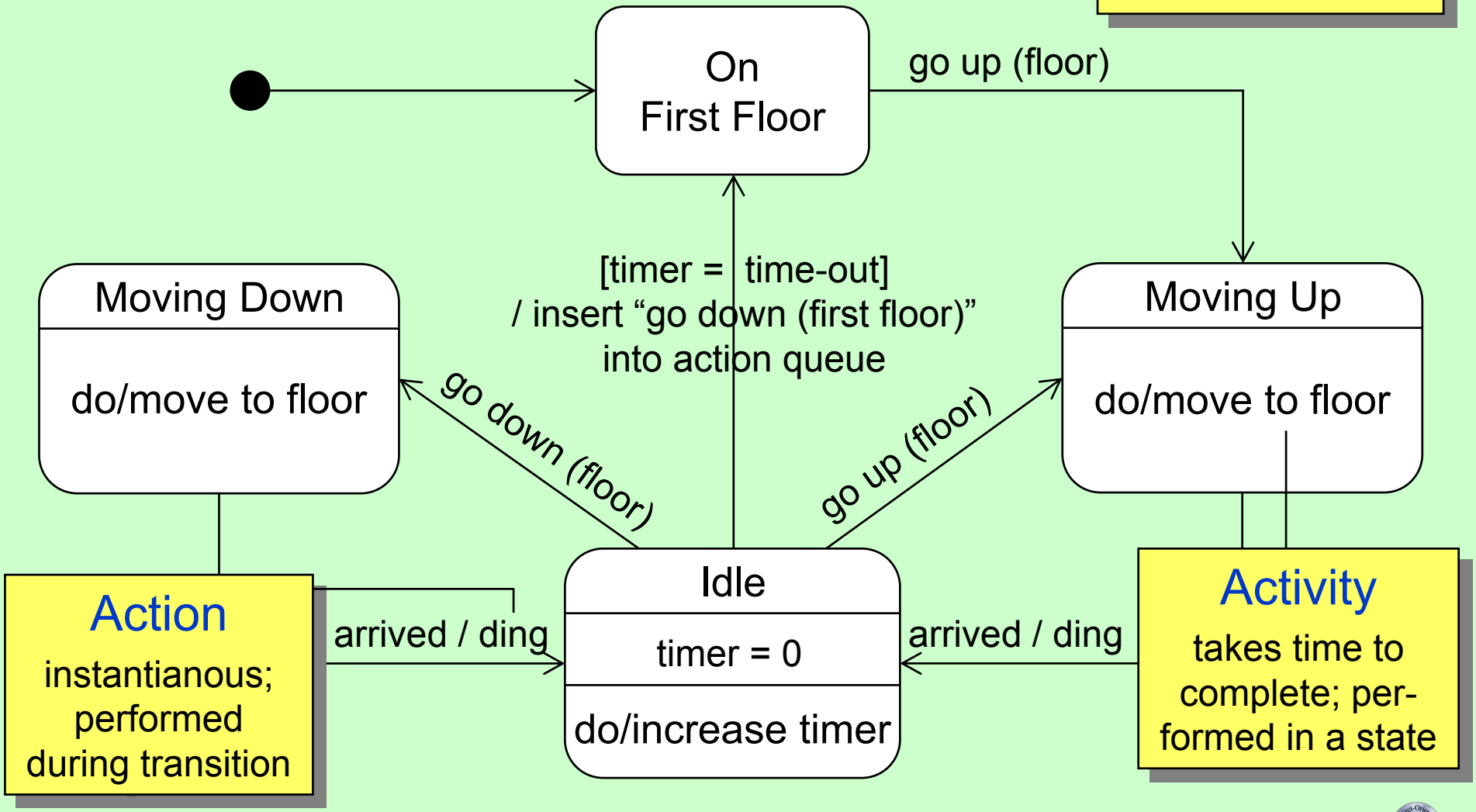
- SignalEvent
  - » represents the receipt of an asynchronous event and is queued by the receiver until it's ready to handle it
- CallEvent
  - » models the synchronous receipt of a message by an object, invoking a call of an operation
- TimeEvent
  - » after the specified time, the event occurs. The keyword after is often used in conjunction with time events

State Chart Diagram

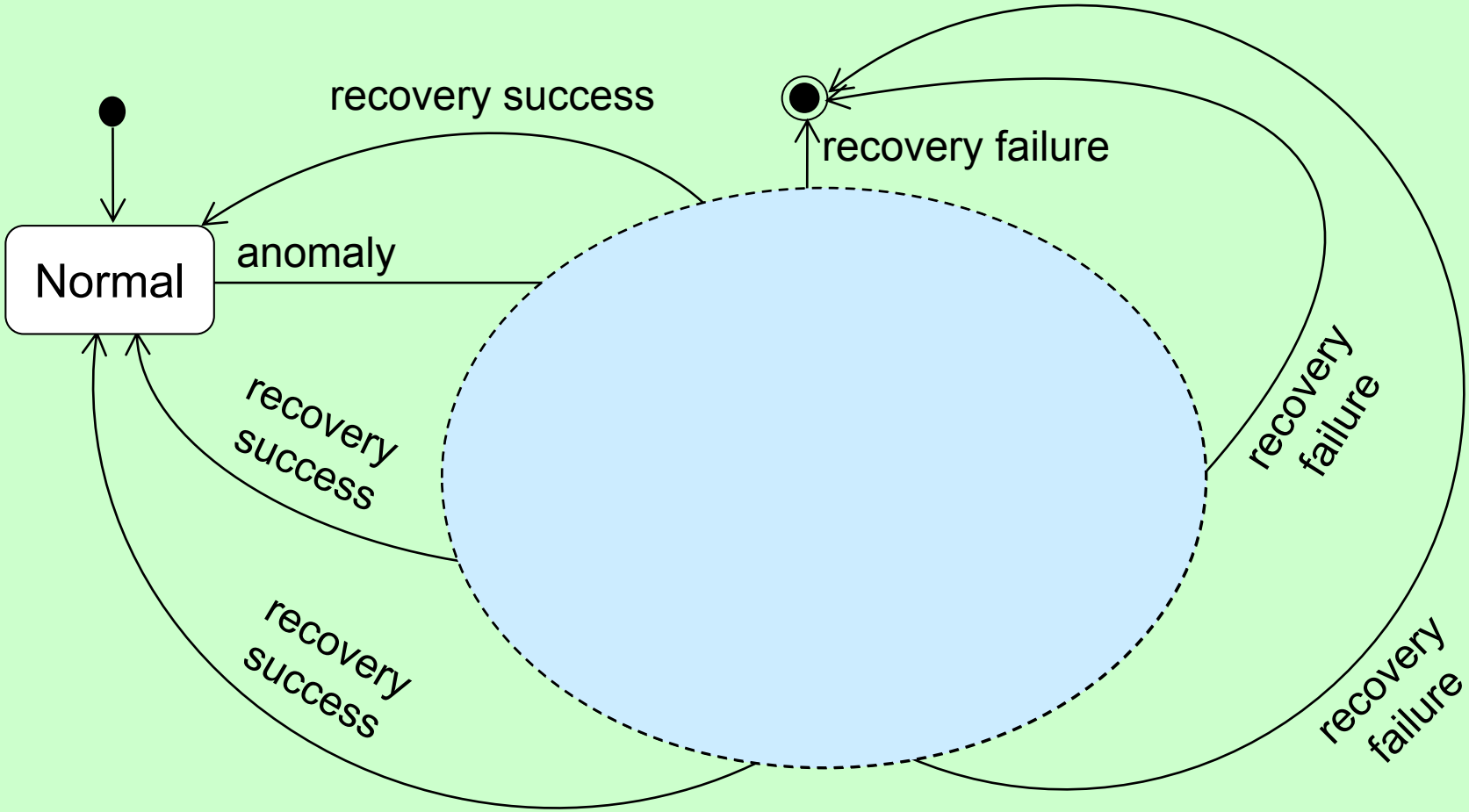


State Chart Diagram

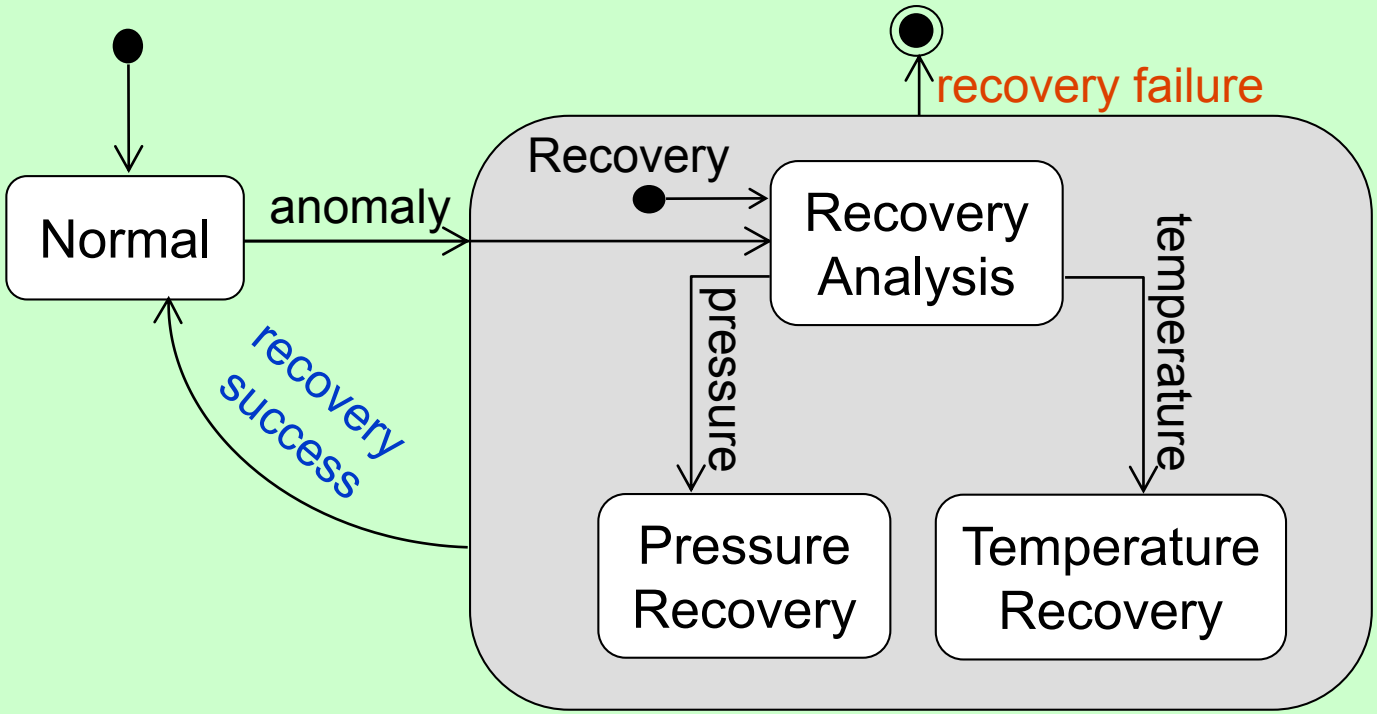
Note that this is a „Wonkavator“



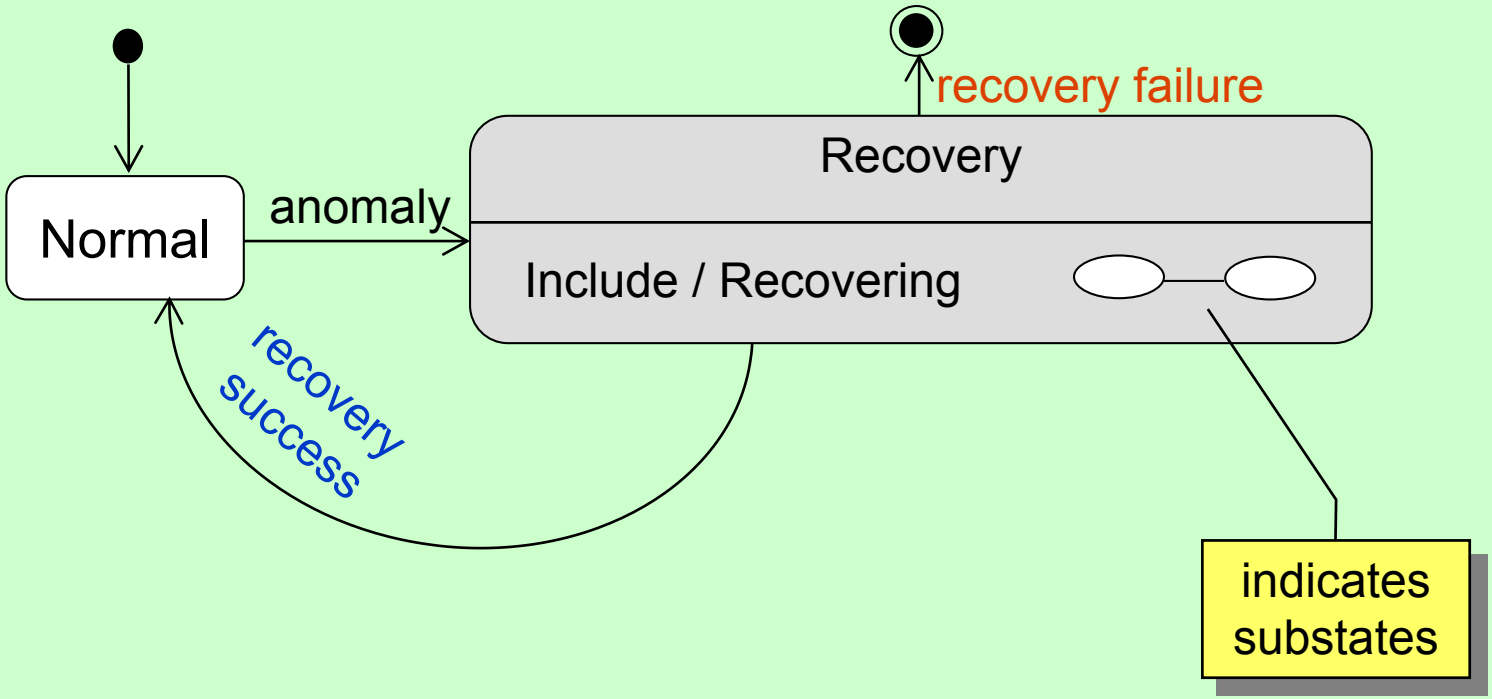
State Chart Diagram



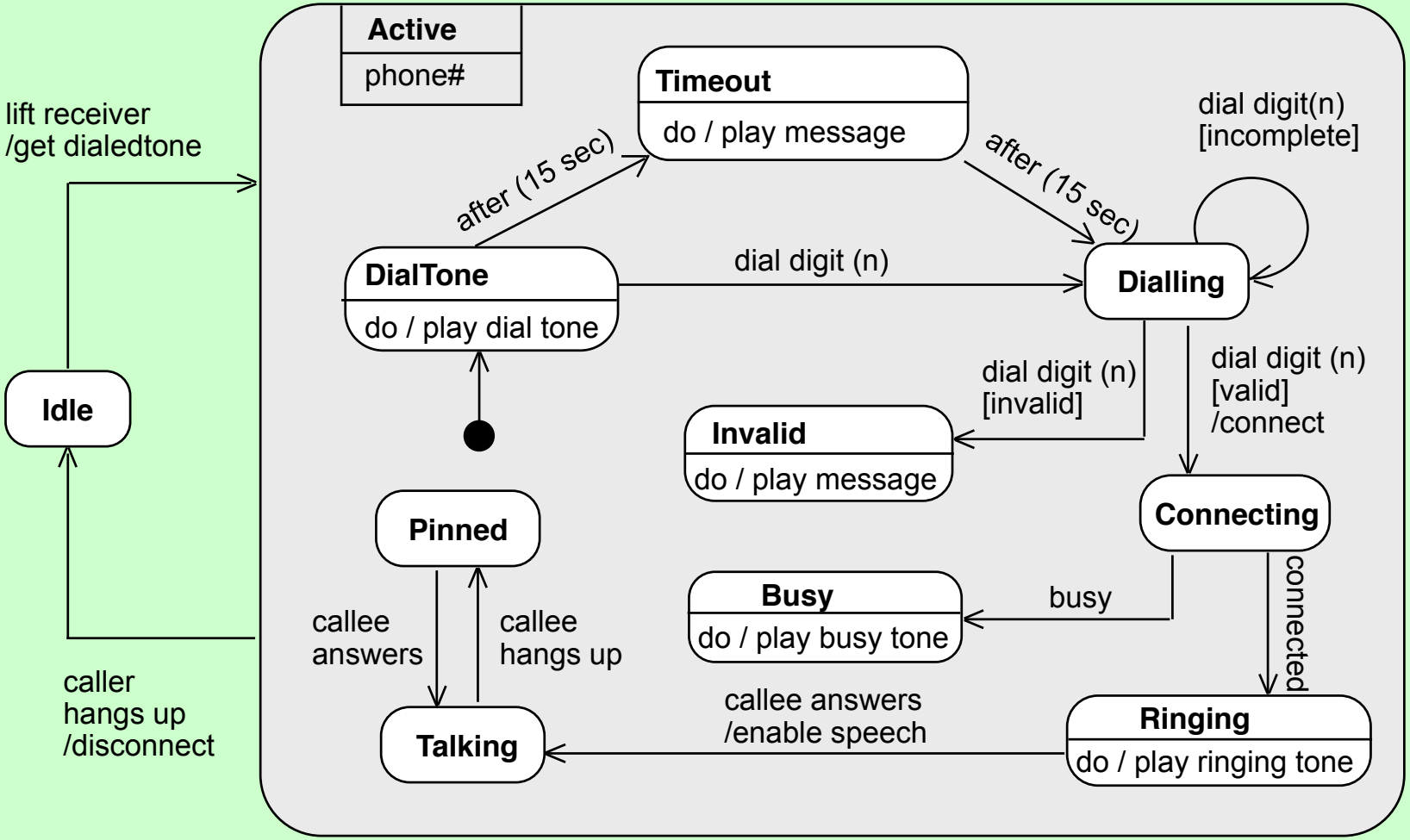
Superstate


recovery failure
Recovery
Normal
anomaly
Recovery Analysis
pressure
temperature
recovery success
Pressure Recovery
Temperature Recovery

Superstate

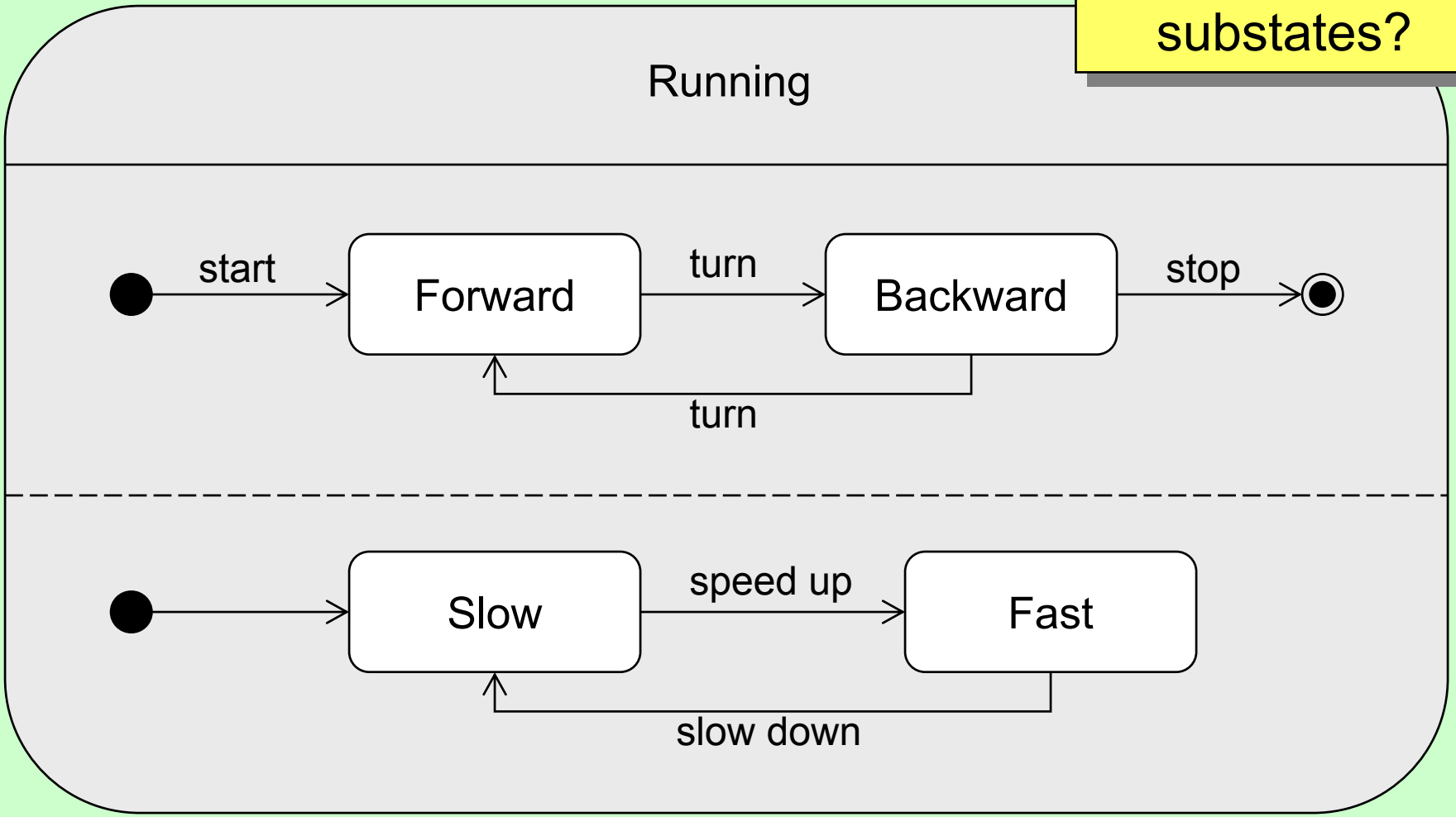

recovery failure
Recovery
Include / Recovering
Normal
anomaly
recovery
success
indicates
substates

Superstate


Active
phone#
Timeout
do / play message
lift receiver
/get dialedtone
after (15 sec)
after (15 sec)
dial digit(n)
[incomplete]
DialTone
do / play dial tone
dial digit (n)
Dialling
Idle
dial digit (n)
[invalid]
dial digit (n)
[valid]
/connect
Invalid
do / play message
Pinned
Connecting
Busy
do / play busy tone
busy
connected
callee
answers
callee
hangs up
caller
hangs up
/disconnect
callee answers
/enable speech
Talking
Ringing
do / play ringing tone

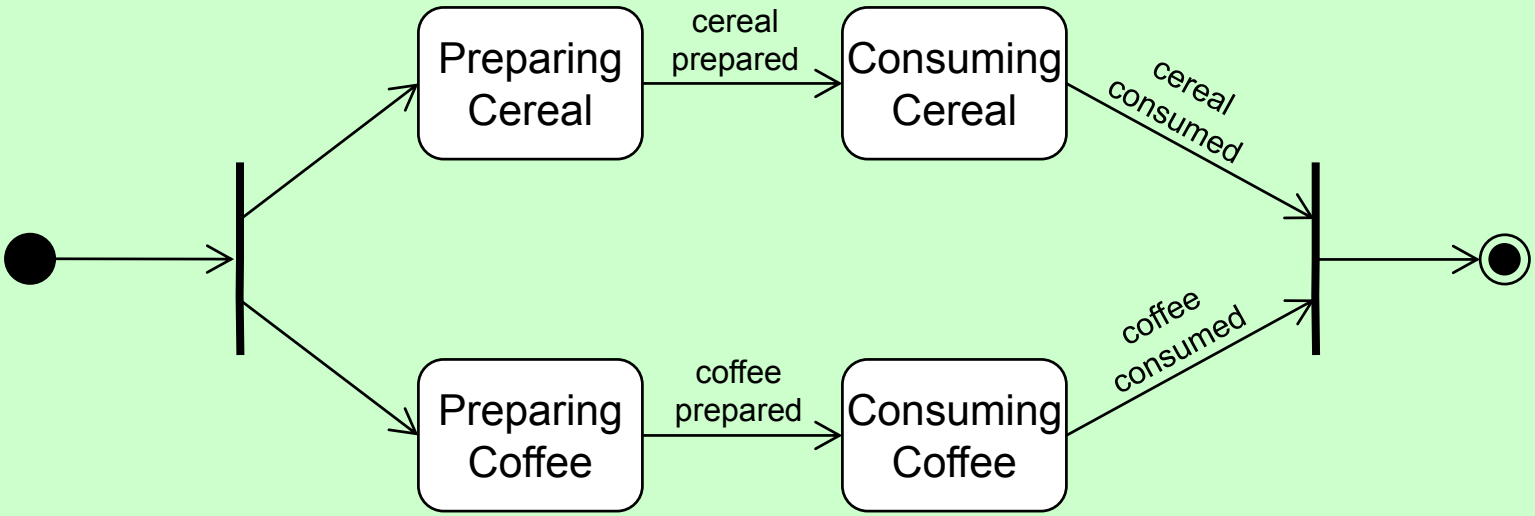
Concurrent Substates

How does this look like without concurrent substates?

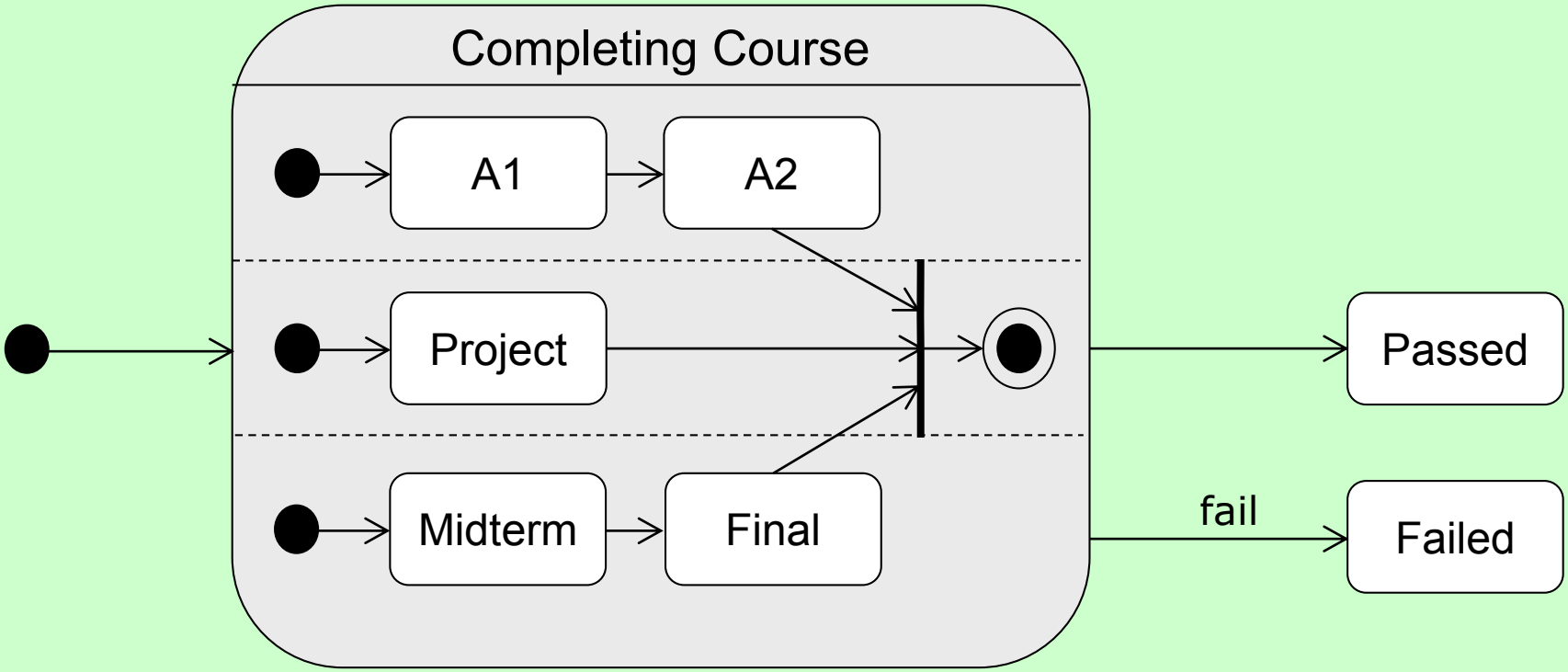


Fork and Join

Parallel Breakfast



Concurrent Substates



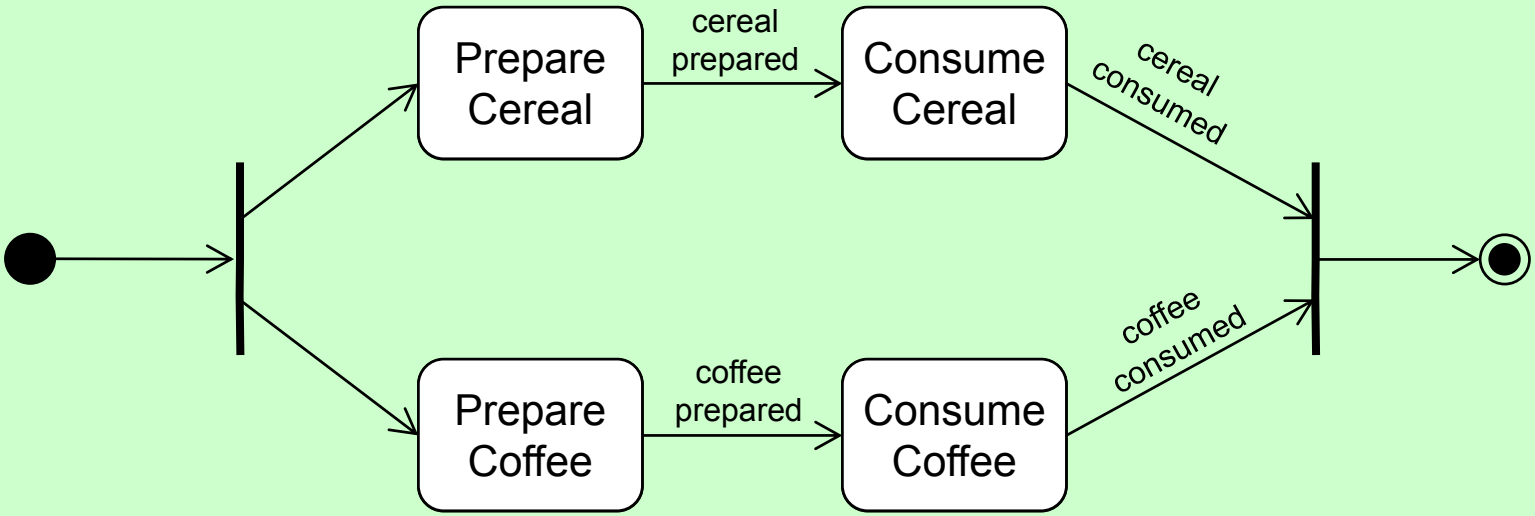
Multiple States

- Classic State Diagrams are XOR diagrams
  - » machine can only be in one of several states
- Statecharts are OR diagrams
  - » machine can be in multiple states
  - » extreme form are Petri Nets
  - » the possible combinations constitute new global states themselves
    - – different interpretation of a single substate
    - – way to concisely describe large finite state machines

Complexity Reduction Through…

- Superstates
  - » combination of several substates into one superstate
  - » reduces complexity by hiding
    - – aggregated states and multiplied transitions
- Concurrent Machines
  - » parallel execution
  - » reduces complexity by factorising
    - – multiplied states and corresponding transitions

Parallel Breakfast as Parallel Activities



Activity Diagrams Revisited

Activity Diagrams & State Diagrams

- A Correspondence
  - » Activities correspond to States
  - » Control flow lines correspond to (termination) Events
- Same Semantic Foundation
  - » token flow → Petri Nets
- Different interpretation and applications
  - » context with events vs automatic/implicit "next" events
  - » reactive behaviour vs control flow

Imprecise Natural Language

Informal "Specifications"

- "*I'm going to teach you how to solve the Rubik's cube in about 30min.*"
  - » does not appear to be a fast method, right?
- "*No eating or drinking from cups without lids.*"
  - » so I can eat from a plate?