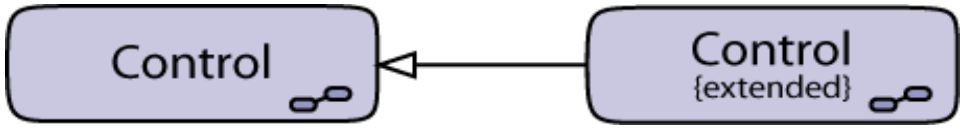
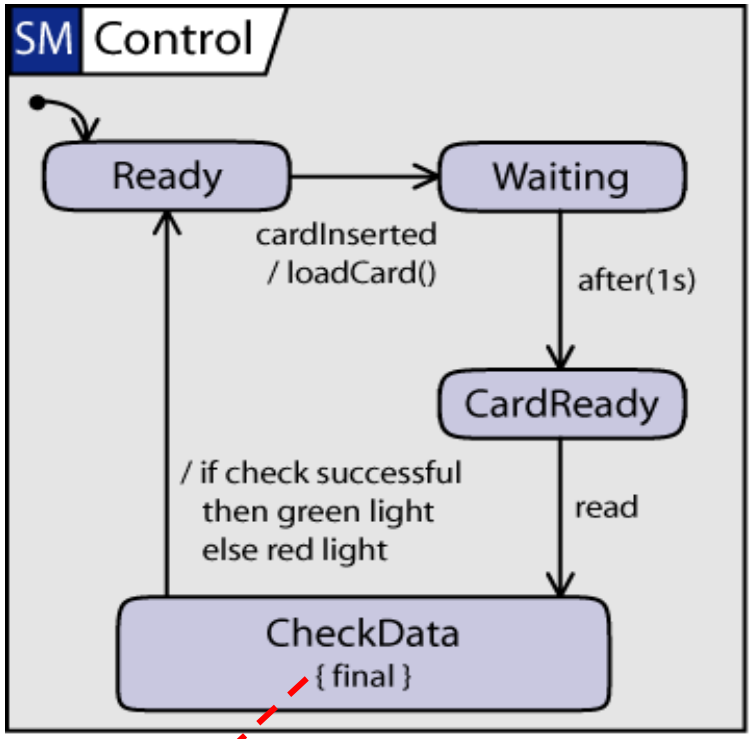
Semantic variation points

- Some semantic variation points have been mentioned before.
  - delays in event pool
  - handling of deferred events
  - entering of composite states without default entry
- Which events are prioritized?
  - completion events only
  - all internal events (completion, time, change)
- Which (additional) timing assumptions?
  - delays in communication
  - time for run-to-completion step
    - zero-time assumption

State machine refinement

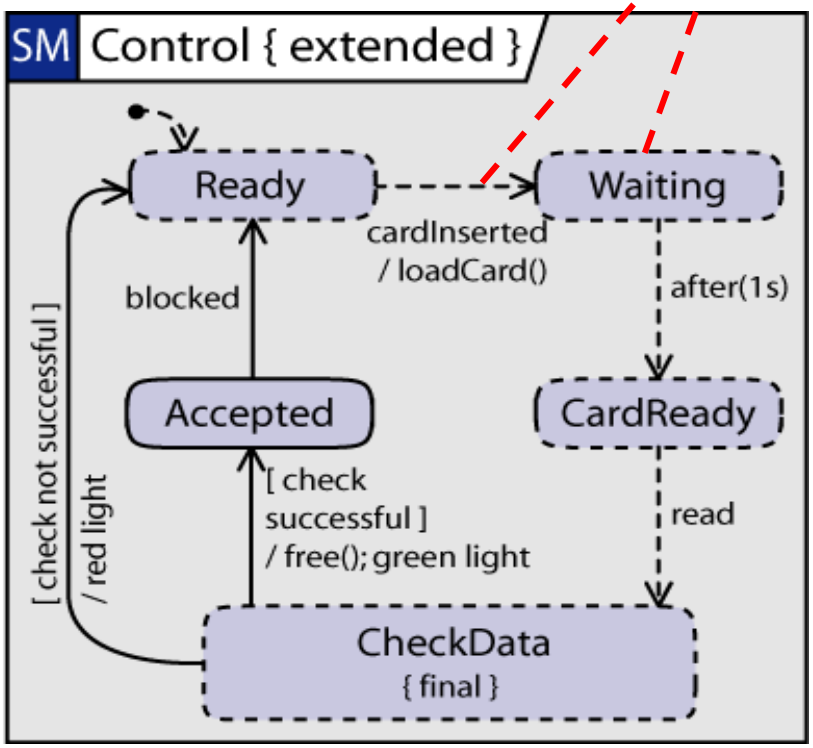
- State machines are behaviors and may thus be refined.





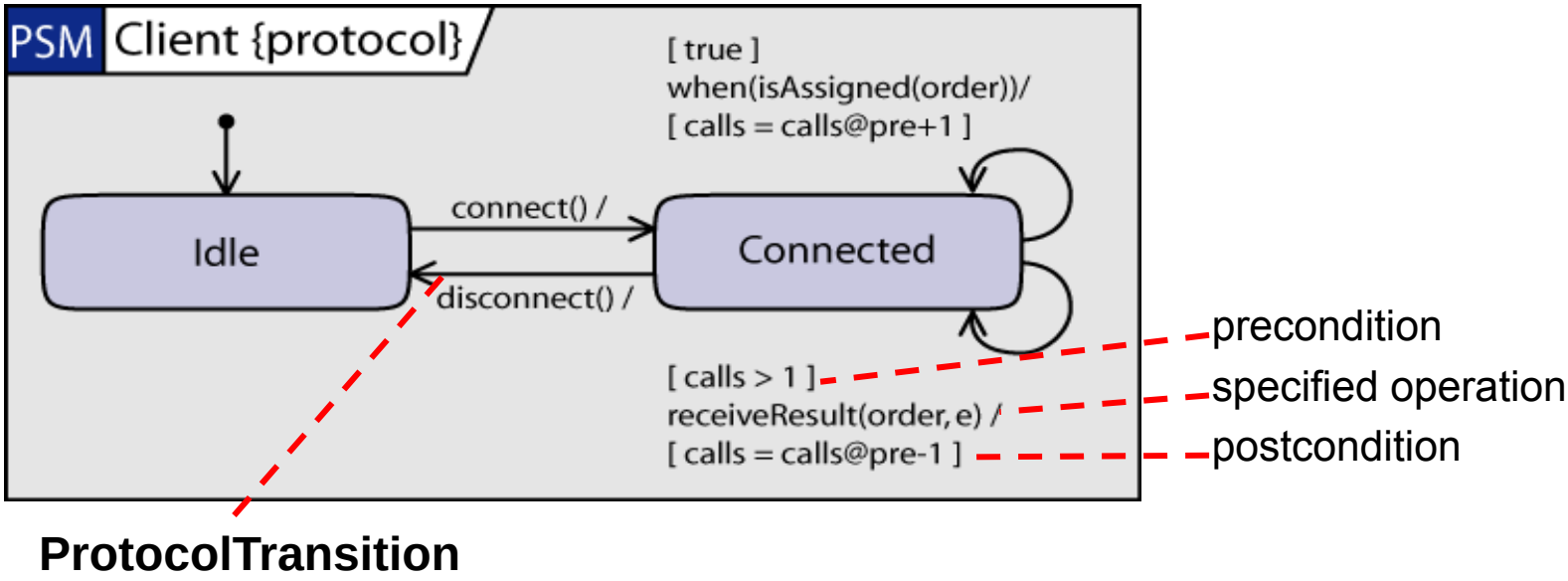
no refinement possible

not refined (may be omitted)



Protocol state machines

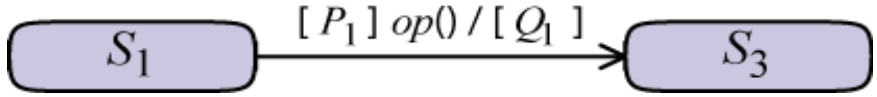
- Protocol state machines specify which behavioral features of a classifier can be called in which state and under which condition and what effects are expected.
  - particularly useful for object life cycles and ports
  - no effects on transitions, only effect descriptions



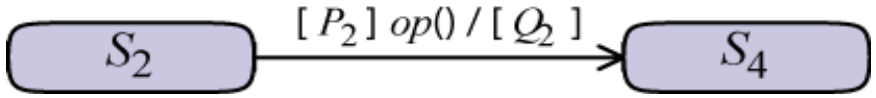
Protocol state machines

Several operation specifications are combined conjunctively:

context C::op()
pre: inState(S_1) **and** P_1
post: Q_1 **and** inState(S_3)



context C::op()
pre: inState(S_2) **and** P_2
post: Q_2 **and** inState(S_4)



results in

context C::op()
pre: (inState(S_1) **and** P_1) **or** (inState(S_2) **and** P_2)
post: (inState@pre(S_1) **and** P_1@pre) **implies** (Q_1 **and** inState(S_3))
and (inState@pre(S_2) **and** P_2@pre) **implies** (Q_2 **and** inState(S_4))

How things work together

- Static structure
  - sets the scene for state machine behavior
  - state machines refer to
    - properties
    - behavioral features (operations, receptions)
    - signals
- Interactions
  - may be used to exemplify the communication of state machines
  - refer to event occurrences used in state machines
- OCL
  - may be used to specify guards and pre-/post-conditions
  - refers to actions of state machines (`OclMessage`)
- Protocols and components
  - state machines may specify protocol roles

Wrap up

- State machines model behaviour
  - object and use case life cycles
  - control automata
  - protocols
- State machines consist of
  - Regions and …
  - … (Pseudo)States (with entry, exit, and do-activities) …
  - connected by Transitions (with triggers, guards, and effects)
- State machines communicate via event pools.
- State machines are executed by run-to-completion steps.