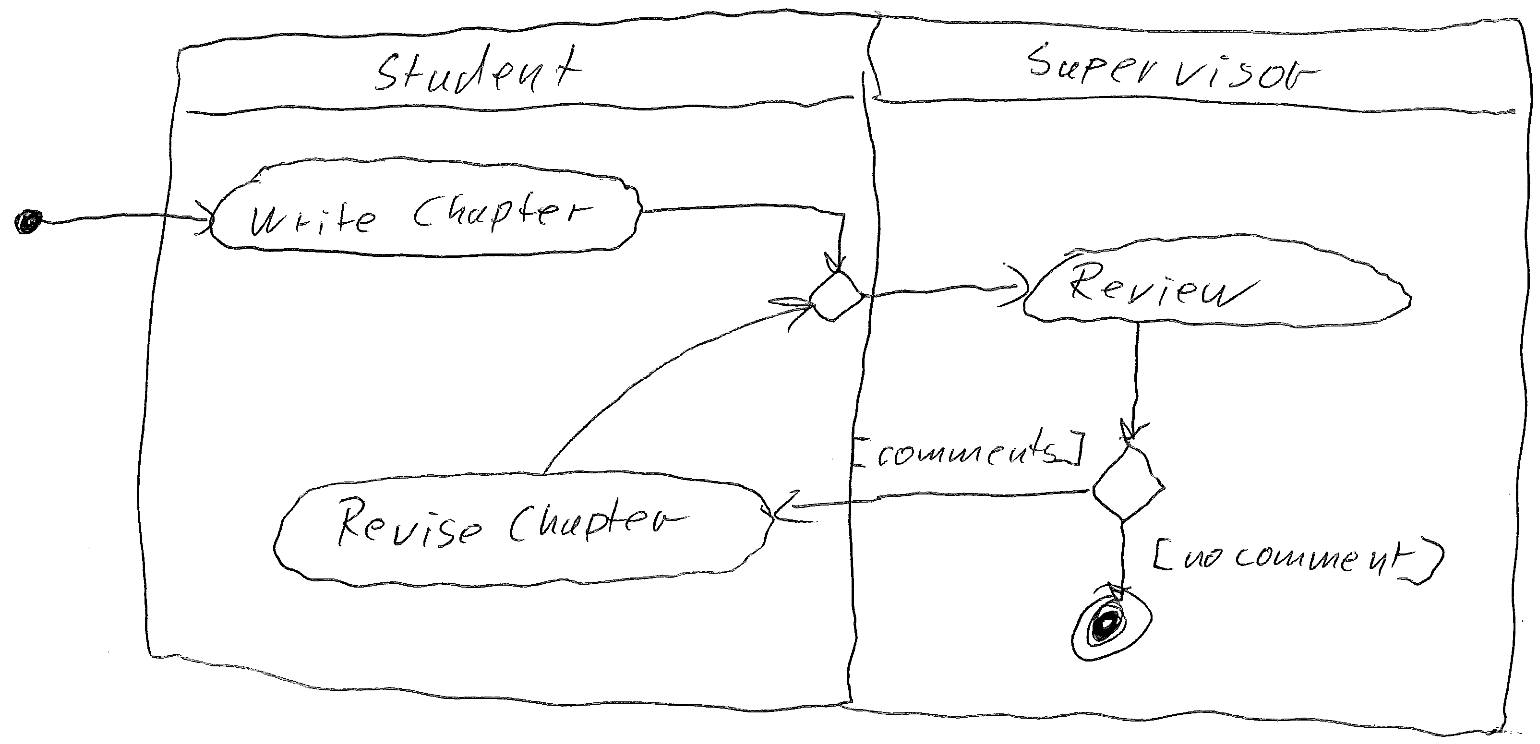
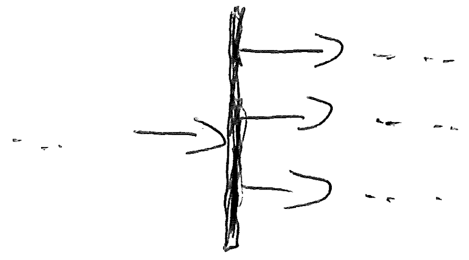
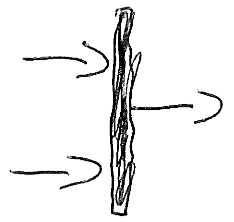
Example of Conditional Activities



Forking and Joining in Activity Diagrams





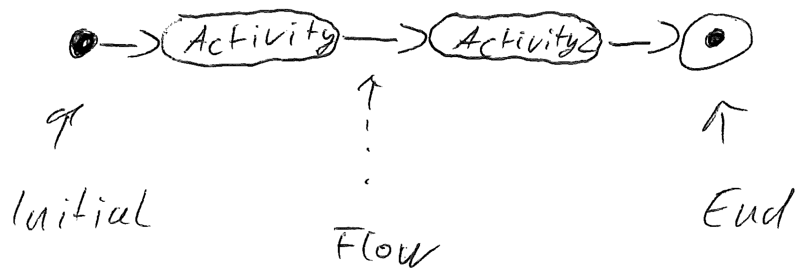
Forking and Joining [UML User Guide]

Motivation: model concurrent control flows (i.e., activities that run in parallel)
Forking: A **fork** (thick horizontal or vertical line) has exactly one incoming and two or more outgoing flows. (Gabelung)
Joining: A **join** (thick horizontal or vertical line) has two or more incoming and exactly one outgoing flow. (Vereinigung)

Further Rules for Activity Diagrams

- branched paths must be merged eventually (letztendlich)
- forked paths must be joined eventually
- only outgoing edges of branch nodes have guards

Activity Diagrams



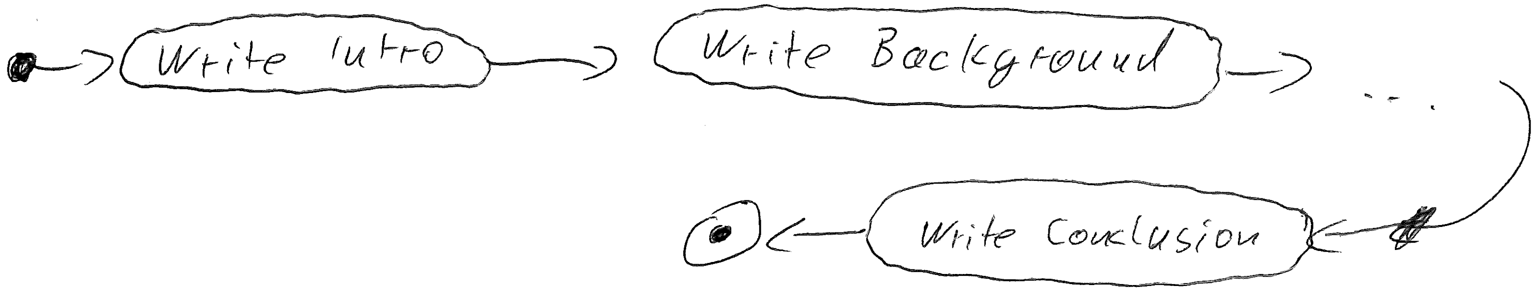
Activity Diagram (Aktivitätsdiagramm)

An **activity diagram** is a diagram visualizing activities and their order of execution. An activity diagram contains **activities** (rounded box) that are connected by means of **flows** (solid arrows). The execution begins at the **initialization** (filled circle) and ends with the **completion** node (bull's eye). (Aktivität, Fluss, Startzustand, Endzustand)

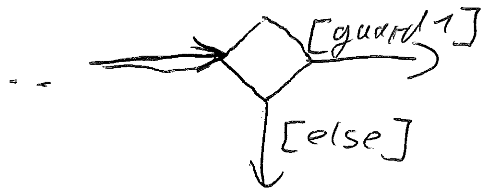
Rules for Activity Diagrams

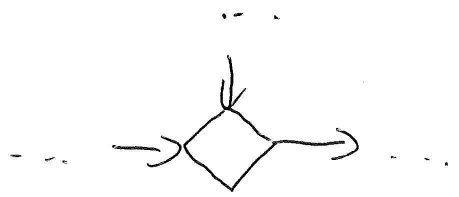
- exactly one initialization/completion node
- at least one activity
- every activity has one incoming and one outgoing flow
- every activity is reachable from initialization
- completion is reachable from every activity

Example of Sequential Activities



Branching and Merging in Activity Diagrams





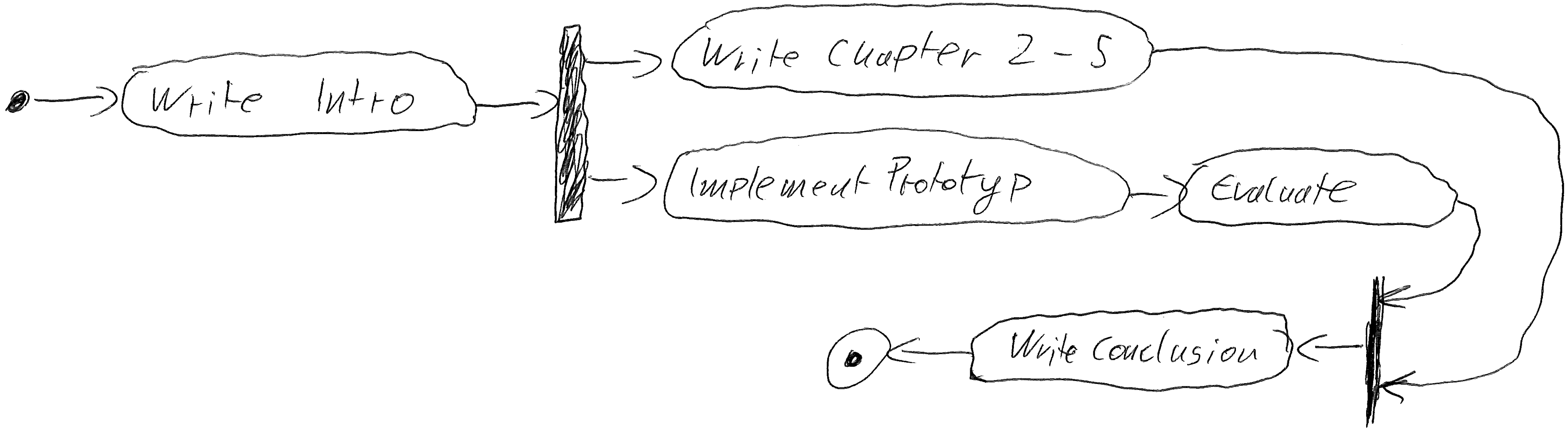
Branching and Merging [UML User Guide]

Motivation: model control flow that depends on certain conditions (i.e., actions that may happen)
Branching: A **branch** has exactly one incoming and two or more outgoing flows. Each outgoing flow has a Boolean expression called **guard**, which is evaluated on entering the branch. (Verzweigung)
Merging: A **merge** has two or more incoming and exactly one outgoing flow. (Zusammenführung)

Further Rules for Activity Diagrams

- guards on outgoing flows should not overlap (flow of control is unambiguous)
- guards should cover all possibilities (flow of control does not freeze)
- keyword **else** possible for one guard (sonst)

Example of Concurrent Activities



Swimlanes in Activity Diagrams

Swimlanes [UML User Guide]

Motivation: group activities according to responsibilities
Swimlane: An activity diagram may have no or at least two swimlanes. A **swimlane** (rectangle) represents a high-level responsibility activities within an activity diagram. (Verantwortlichkeitsbereiche)

Further Rules for Activity Diagrams

- each swimlane has a name unique within its diagram
- every activity belongs to exactly one swimlane
- only flows may cross swimlanes

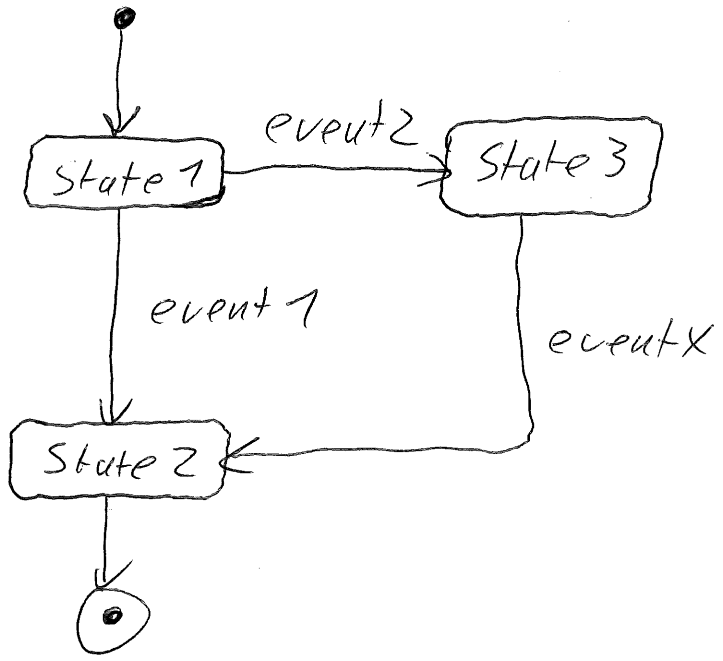
State Machine Diagrams

State Machine Diagram (Zustandsdiagramm)

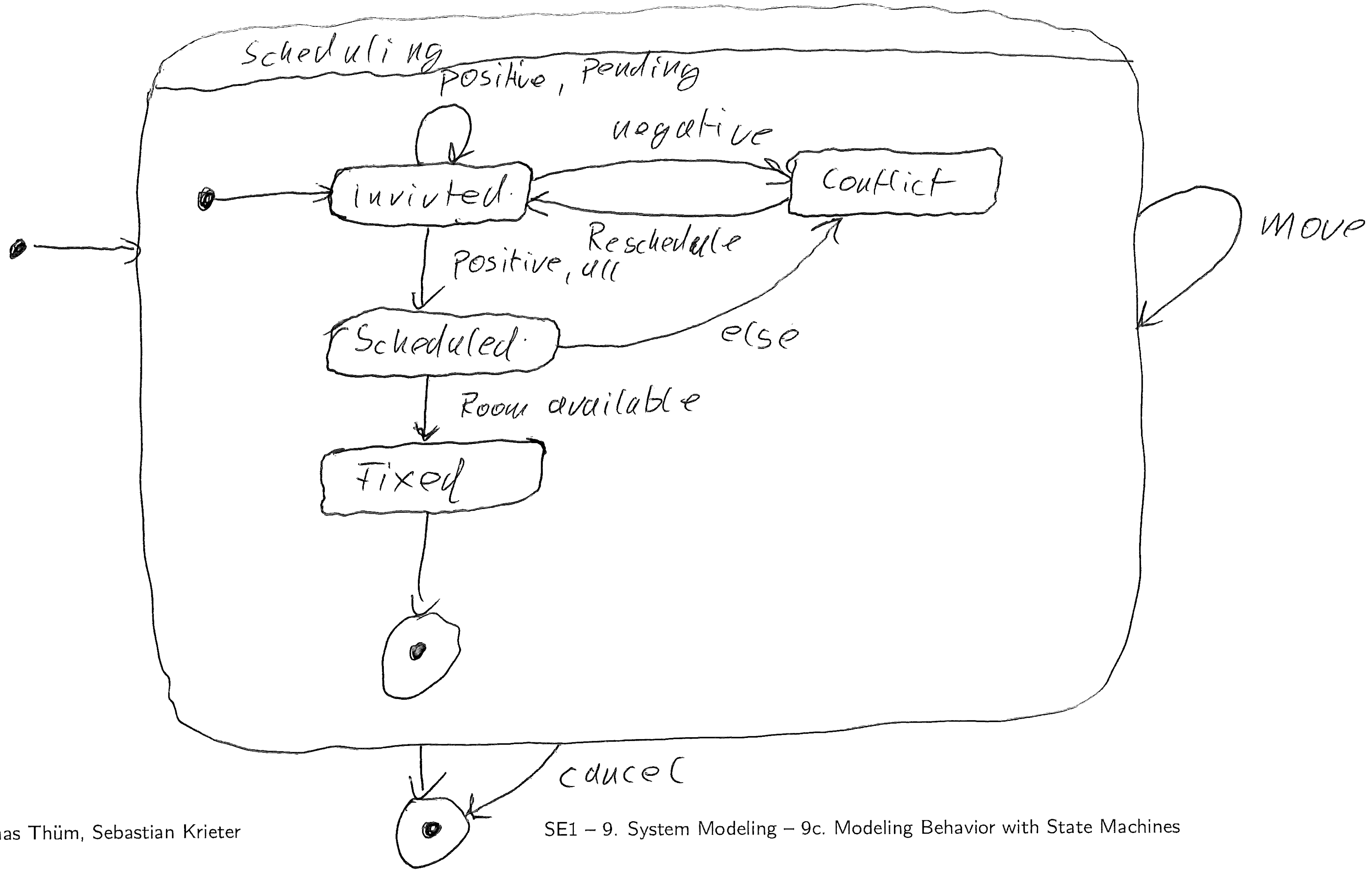
A **state machine diagram** specifies the sequences of states the (a part of) the system goes through during its lifetime in response to events, together with its responses to those events. Every **state** (rectangle with rounded corners) is characterized by a condition or situation. An **event** is an occurrence of a stimulus that can trigger a state transition. A **transition** (solid arrow) is a relationship between two states. (Zustand, Ereignis, Zustandsübergang) [adapted from UML User Guide]

Rules for State Machine Diagrams

there is a single **initial state** (filled circle) and a single **final state** (bull's eye) (Start- und Zielzustand) — see exception below



Example of a State Machine Diagram



Scheduling
positive, pending
negative
Invited
Conflict
Reschedule
positive, all
Scheduled
else
Room available
Fixed
move
cancel