



ASSIGNMENT 4: FORMAL REQUIREMENTS SPECIFICATION

2008 Dec 1 – Stefan Schurgast, Michael Sommerhalder, Krzysztof Dabkowski

1. Set-based formal specification methods

2. State-based formal specification methods

3. Comparison



Set-based formal specification techniques

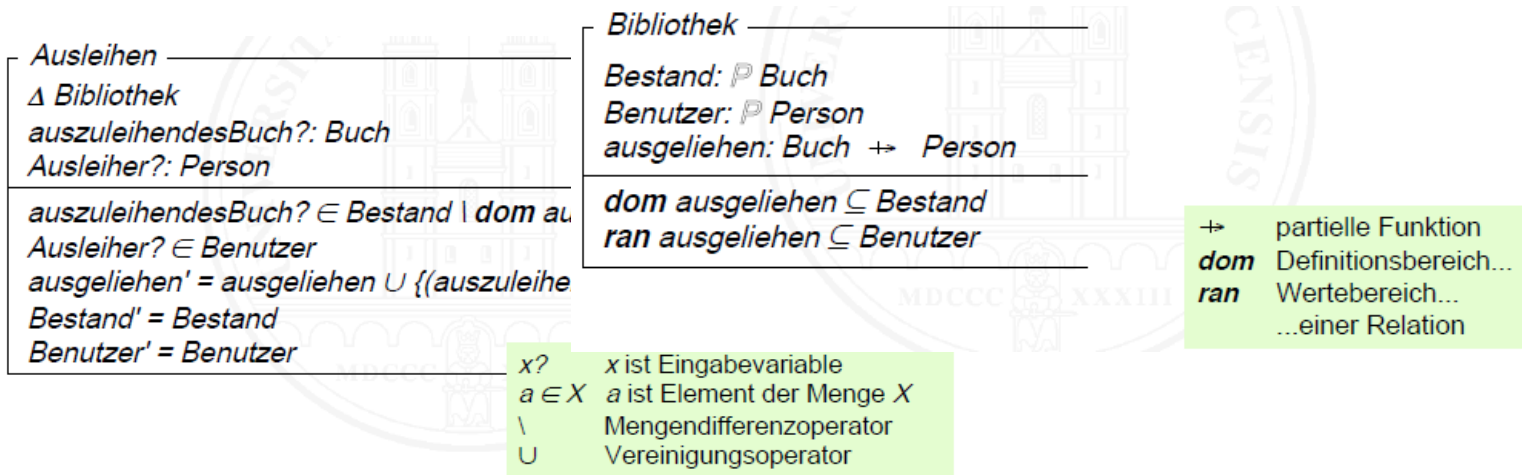
- Based on set theory and logical statements (able to express everything)
- Mathematical model of system (states and transitions)
 - Basic set
 - Correlation between sets (i.e. relations and functions)
 - Invariants (predicates)
 - Change in states
- Uses predicate logic and assertions
- Statements that can be true or false
- No fuzzy statements (mutually exclusive)



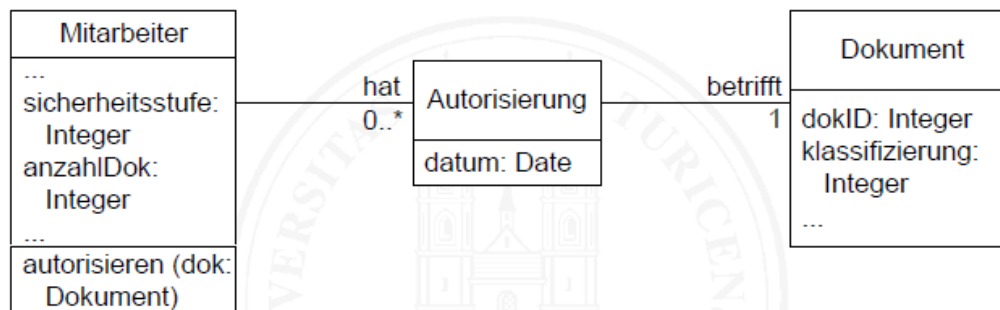
Set-based formal specification techniques

Most popular examples: Z, OCL, VDM, ...

Z:



OCL:



```

context Mitarbeiter::autorisieren (dok: Dokument)
pre: self.sicherheitsstufe ≥ dok.klassifizierung
post: anzahlDok = anzahlDok@pre + 1
and
self.hat->exists (a: Autorisierung | a.betrifft = dok)
    
```



Advantages

- Short notation
- Unambiguous or identifies ambiguity
- Can be proofed with logic proofing methods
- Automated proofs possible , e.g. with refutation, (therefore often used in security critical problems like in aviation)

Disadvantages

- Not always easy to understand, needs time to be read
- Must be written by someone with a good mathematical background
- Can get very complex with increasing size of system
- Hardly ever specification of complete system (which makes completeness proofs difficult or impossible)



state-based formal specification techniques

- Represents dynamic behavior of system
 - Time dimension:
 - Point of time
 - Variation in time
- Consist of states with transitions
- Normally described with graphs

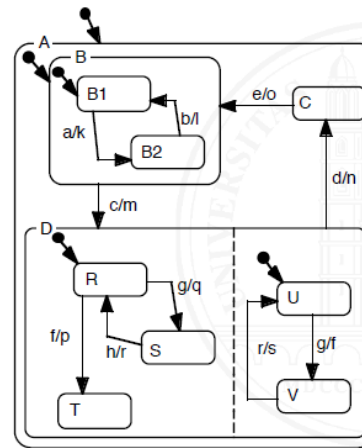


Set-based formal specification techniques

Popular examples:

Statecharts:

- Parallelity
- Hierarchical (or similar notations like requirements-oriented statecharts [ROSC])

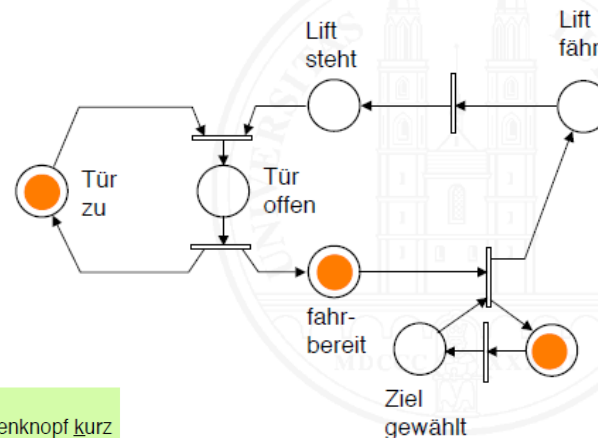


Notation:

- Zustand oder Statechart
- Zustandsübergang
- Initialzustand
- Zustand besteht aus parallelen Statecharts
- auslösendes Ereignis e und ausgelöste Aktion a
- Endzustand

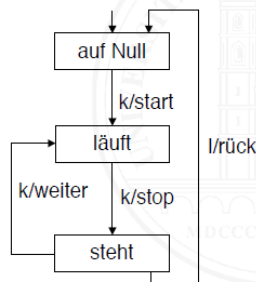
Petrinets:

- Directed bipartite graph
- Nodes, places, directed arcs



- Stelle
- Transition
- Marke

Finite state machines:



Ereignisse:
k Bedienknopf kurz gedrückt
l Bedienknopf lang gedrückt

Aktionen:
start Stoppuhr starten
stop Stoppuhr anhalten
weiter Stoppuhr weiterlaufen lassen
rück Stoppuhr auf Null stellen



Advantages (state based)

- Small
- Easy to understand
- more intuitive than non-graphical representations
- Unambiguous or identifies ambiguity
- Graphical notation reduces complexity
- Simulations possible
- Graph proofs sometimes possible (e.g. completeness analysis)

Disadvantages

- Not always easy to see completeness since complexity normally rises by at least the size of the system



COMPARISON OF SET- AND STATE-BASED METHODS



Set-based

- Unambiguous
- Usually text representation
- One needs mathematical background to be able to read notation
- Not easy to check for completeness
- Scalable
- No simulations possible

State-based

- Unambiguous
- Usually graphical representation
- Interpretation easier
- Completeness checks easier
- State charts can be decomposed and therefore scale
- Simulations can be made



Facit

Set-based formal specification methods:

- Good for describing systems in an object oriented manner
- High level view of system

State-based formal specification methods:

- Good for describing systems with their dynamics
(as states and transitions)
- Natural programmer perspective

