OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG

INFORMATIK UND MATHEMATIK

Modeling and Verification (MOV)

Organizational issues and Introduction

Prof. Dr. Carsten Kern

Ostbayerische Technische Hochschule
Fakultät Informatik und Mathematik

Information about lecturer

Acad. studies:	RWTH Aachen University
	Computer Science (Economies as minor)
	Mathematics
Abroad:	Madrid (Erasmus scolarship, 1 year)
Dissertation:	RWTH Aachen (Formal methods and software engineeging)
Professions:	- 2 years technical project manager and overall project manager
	- Subsequently freelancer
	- Founding a startup (BMWi grant) called MathComm
	- CEO of NubiFactum GmbH
	- Professor at OTH-Regensburg since 2015 (focus: Software Engineering, ML)

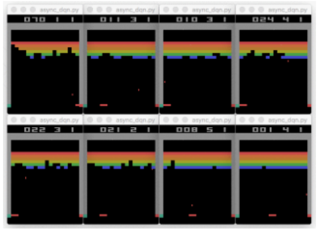
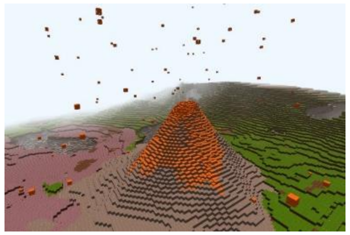
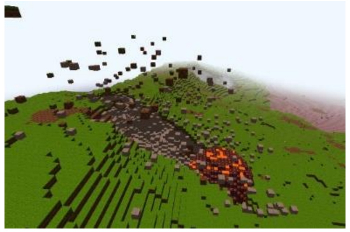
Research interests

Research topics (also possible for HSPs or Master's theses, cf website):

- Software Engineering processes/architecture
- Machine Learning and AI algorithms
- Model-driven and test-driven approaches
- (New) modeling techniques/languages and their formal background

→ That's what we're gonna talk about from now on! ;)

Game algorithms: **3D Framework:** **VR programs:** **Table soccer:**

Lectures and assignments/practical courses

- Discussion: Tuesdays, 11:45 am – 01:15 pm, Zoom
  maybe Wednesdays (if necessary),
- Assignments: Wednesdays, 08:15 am – 09:45 am, K006
- Office hours: Tuedays, 10:00-11:00 am, Zoom

- Course material:
  - Slides and assignments via G.R.I.P.S.
  - Recommended literature later in this lecture
- Final exam:
  - 90 minutes written exam at the end of the semester
- Bonus points for exam (up to 9 of ~90 points):
  - Cf. slide 5

[Important remark about bonus points (given by OTH president):
- **Bonus points do not save you from failing the exam**
- **Bonus points are only granted if you pass the exam (without incorporating them)]**

Prerequisites

- The following knowledge is required/recommended:
  - Object Oriented Programming
    - Extended Java knowledge (e.g. to implement Java web services)
  - Software Engineering
    - Knowledge of modeling software systems (usually UML)
  - Theoretical Computer Science
    - Knowledge of automata theory and formal languages (DFA, NFA, CFG etc.)
    - Logics
- A notebook/tablet is recommended, to:
  - Solve the in-class exercises
  - Actively follow the live modeling examples
  - Solve assignments and present the solution during the assignment courses

Lectures and assignments

- "What do we do within the lectures?"
  - Listen to what I say ;)
  - Discuss theoretic foundations of models
  - Get to know many model elements
  - Do small exercises to strengthen theoretic/practical knowledge $\Longrightarrow$

- "What do we do for and within the assignment classes?"
  - Every Thursday there will be a new assignment sheet online
  - You prepare solutions for the next assignment class
  - We will discuss solutions/problems etc. in small groups within the assignment classes
  - You will finish the assignment in/after the assignment class according to the discussions
  - [Listen to your presentations (cf. next slide)]

Bonus points

- Possibility of giving short (~10-15 minutes) presentations of interesting topics concerning modeling, model-driven development, verification, etc. (contact me)

- Possible topics (not exhaustive) are, e.g., Presenting:
  - Comparisons of two (or more) BPMN process engines
  - Integration of web services into process engines (e.g., Activiti or Camunda)
  - OCL/PDL/CTL, model checking tools (NuSMV, etc.)
  - …
- "Why should I do that?"
  - One way of gaining bonus points for the exam, or
  - Find possible topics for HSPs or your Master thesis
- "Are there other possibilities for gaining bonuses?"
  - Yes, e.g.:
    - doing extra or very elaborate assignment tasks (cf. assignments sheets)
    - writing (HTML) tutorials for existing tools
    - Implementing small tools (model checker etc.)

Your experience and expectations?

Your expectations concerning "Modeling and Verification“

!?

OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG
INFORMATIK UND MATHEMATIK

Content: rough, preliminary overview

No.	Topic	Content
0/1	Introduction + BPMN	Organizational topics, first glimpse onto modeling, Basic BPMN notation, elements and semantics
2	BPMN+Impl.	Extended BPMN notation, process engines, modeling conventions and integration of web services
3	EPCs	Modeling using Event-driven process chains
4	EPCs	Transforming EPCs to BPMN
5	Petri nets	Modeling distributed Systems using Petri nets
6	Petri nets	Transforming EPCs to Petri nets
7	MSCs	Modeling distributed systems using MSCs
8	MSGs, CFMs	Properties of distributed systems regarding implementability
9	PDL	Reasoning about events in and properties of MSCs
10	OCL	Using constraints in UML class diagrams
11/12	CTL-MC	Verifying system properties: Computation-Tree-Logic Model Checking
12/13	Symb. MC	Verifying system properties: Symbolic Model Checking using BDDs
12/13	???	??? e.g.: Questions + Test Exercises for Exam

Goals of this course

- Independent modeling and implementation of (business) processes
  - EPCs (event-driven process chains)
  - Mastering the BPMN language, optimizing processes
  - Handling of Process-Engines (e.g., Activiti, Camunda process engine)
  - Implementing scripts and web services running on a process engine
- Understanding the theory, handling and modeling using other advanced modeling notations such as, e.g.:
  - (Context-free gram.: only new for Business Information Technology students)
  - Petri nets
  - Message Sequence Charts/Graphs, Communicating Automata
- Employing logics to express properties of modeled systems
  - Object Constraint Language (OCL)
  - Propositional Dynamic Logic (PDL)
  - Computation Tree Logic (CTL)
- Understanding and applying a basic Model-Checking framework

My expectations towards you

- **Active participation**, because:
  - What remains active in your brain concerning a lecture …

▪ Right after a lecture:	~50% of lecture content
▪ After a week:	~15% of lecture content
▪ At the end of the term:	$\varepsilon \geq 0$

  - A quotation of Konfuzius (551-479 AD)
    - I hear and I forget
    - I see and I remember
    - I do and I understand
- Thus:
  - Within the lectures, we will regularly solve small tasks to consolidate currently attained knowledge
  - This requires active participation of all attendees

... And this makes the course more fun for both of us ;)

Today’s questions and goals

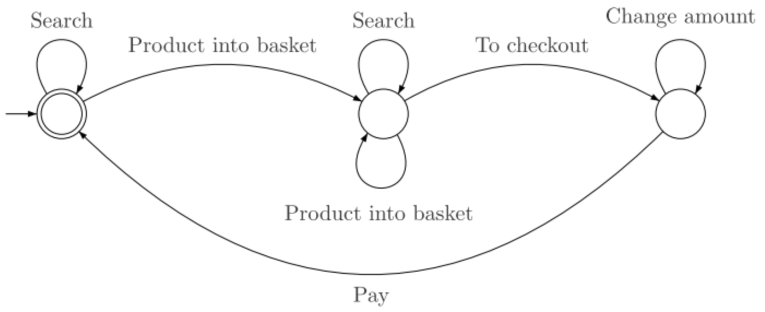
- What are *models*?
- What do we need *models* and *modeling* for?
- Business processes: what are they and what do we need them for?
- Introduction to BPMN
- First examples to BPMN

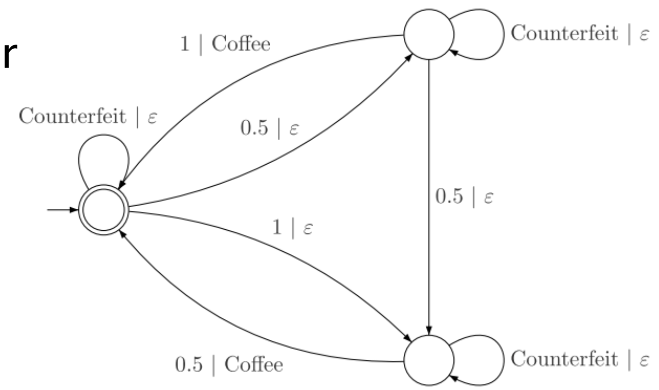
Rough definition of "What is a model"

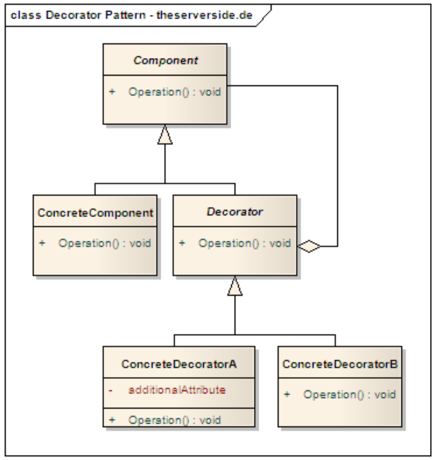
What is a model?

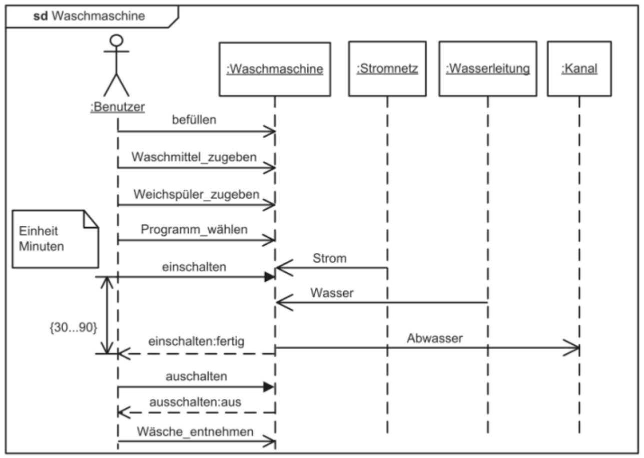
Which models do you know?

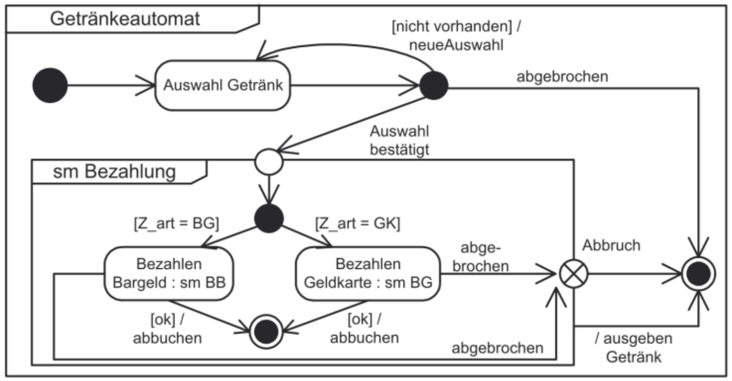
- Model:
  - Abstract representation of a (software) system
  - Contains objects
  - Describes their relation, interaction and behavior











[source: UML 2 glasklar]

OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG
INFORMATIK UND MATHEMATIK

Goals of modeling

Why do we need models/what do we use models for?

- Common language
- Create system design
- Improved understanding of systems
- Visualization
- Analysis of system properties
- Validation/verification
- Optimization
- Simulation
- Automatic code generation
- …

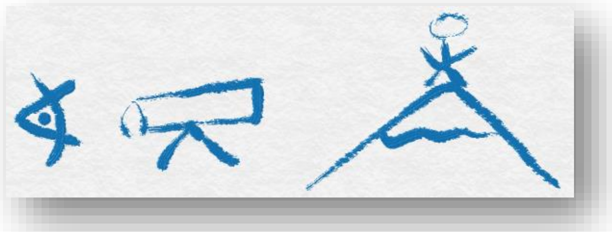
Why are these goals fundamental to software development?

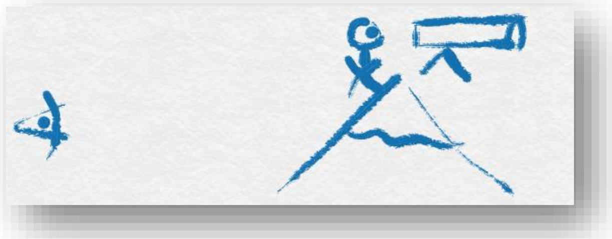
- And within the software development cycle especially also:
  - Quality intensification
  - Error prevention
  - Cost reduction
  - Improvement of documentation, reusability and extensibility

Different types of models

- Visual vs Textual
- Formal vs Non-formal/Informal
  - Problems of non-formal/informal models (e.g.: given in natural language)
    - Mostly ambiguous (as semantics are not clearly defined)
    - Question arises: how to interpret model
    - E.g.:

"I saw the man on the mountain using binoculars"





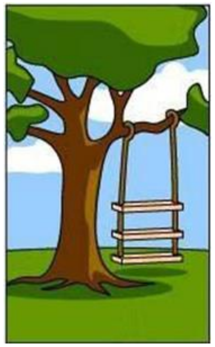
⋮

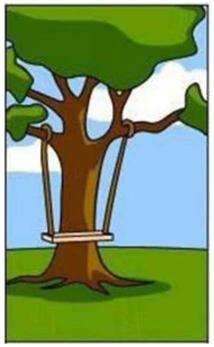
Only 2 out of 5 interpretations

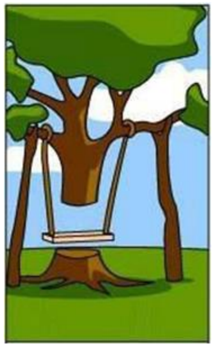
Example: the software development cycle

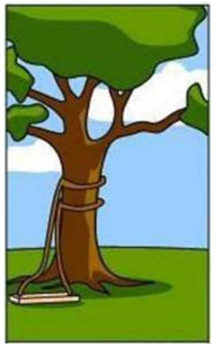
What problems - concerning natural language - arise during development?

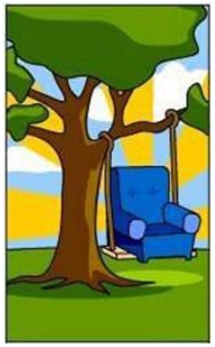
- Try to find the corresponding caption to the pictures below

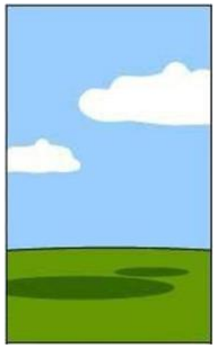

What customer explained

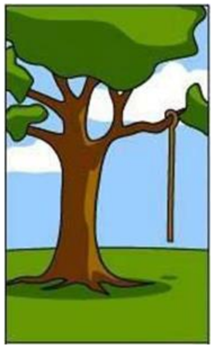

What project manager understood

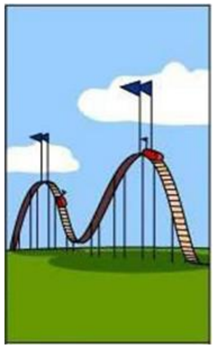

What software analyst designed

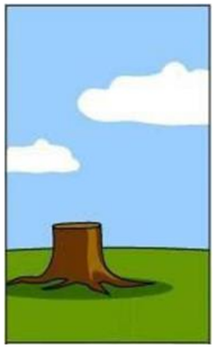

What was coded

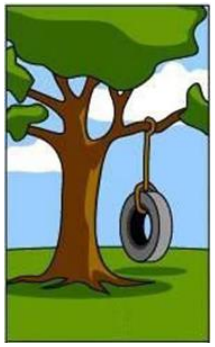

What consultant defined to be delivered


How project was documented


What was installed at customer side


What was billed


How the system was maintained


What the customer really needed

Different types of models (cont.)

- Visual vs Textual
- Formal vs Non-formal/Informal
  - Problems of non-formal/informal models (e.g.: given in natural language)
    - Mostly ambiguous (as semantics are not clearly defined)
    - Question arises: how to interpret model
- Qualitative vs Quantitative
  - Qualitative models describe:
    - Which objects exist
    - What happens
    - Why something happens
  - Quantitative models incorporate
    - Number of objects
    - Time
    - Probability
- Blackbox vs Whitebox
- …

BPMN Literature

Modeling languages

Scope of this course

Types of modeling (excerpt)

- Data modeling: (not our focus in this lecture)
  - Entity-Relationship-Models (databases)
  - Class diagrams (cf. UML)
- Process modeling:
  - Petri nets (PNs)
  - Event-driven process chains (EPCs)
  - Activity diagrams (cf. UML)
  - Business Process Model and Notation (BPMN)
- Distributed-system modeling:
  - Petri nets (PNs)
  - Message Sequence Charts/Graphs (MSCs, MSGs)
  - Communicating automata (CFMs)
  - Sequence diagrams (cf. UML)
  - State charts (cf. UML)
- …

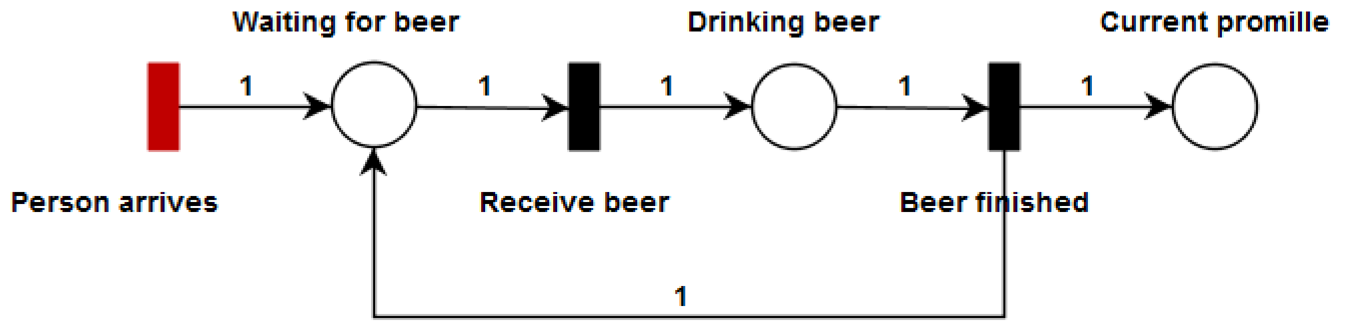
OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG
INFORMATIK UND MATHEMATIK

Short overview over our modeling languages: Petri nets

Petri nets (PNs):

- Petri nets are used for modeling distributed systems
- Elements of Petri nets:
  - Places, transitions (direct, timed, ...), token
  - Activated/enabled transitions
  - Firing transitions

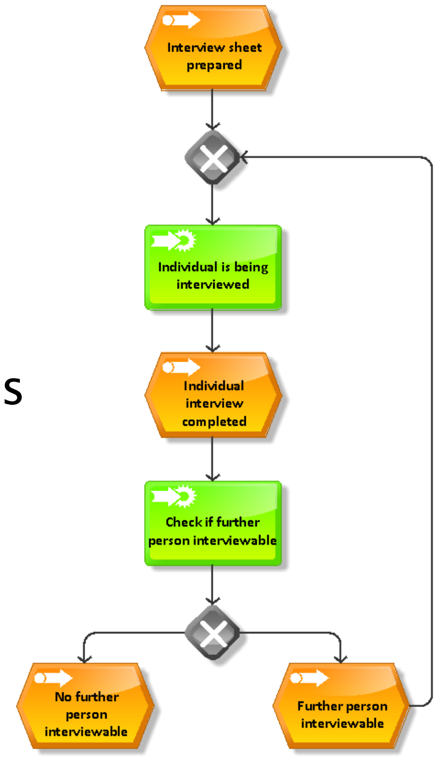
Drinking beer in a pub:



Short overview over our modeling languages: Event-driven process chains

Event-driven process chains (EPCs):

- EPCs are used for (business) process modeling
- EPCs are represented as a kind of flow chart, consisting of:
  - Events (start-, intermediate-, end events)
  - Functions
  - Logical relationships
  - Control-/information flow
  - …
- EPCs are easy to grasp
- EPCs can be seen as predecessors of BPMN diagrams
- Still heavily used in industry (e.g., at BMW)


Interview sheet prepared
Individual is being interviewed
Individual interview completed
Check if further person interviewable
No further person interviewable
Further person interviewable

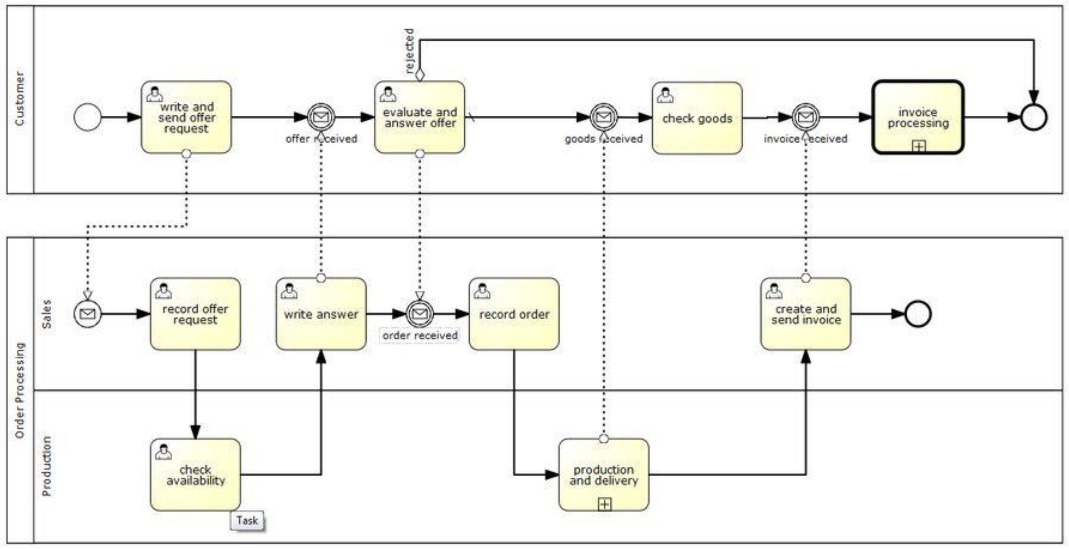
Short overview over our modeling languages: BPMN diagrams

Business Process Model and Notation (BPMN):

- BPMN is closely related to EPCs and UML activity diagrams
- BPMN is an OMG-standardized graphical notation with support of, e.g.:

IBM® SAP® ORACLE axway business. in motion accenture High performance. Delivered. software AG

- The standard notation contains >100 Elements



OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

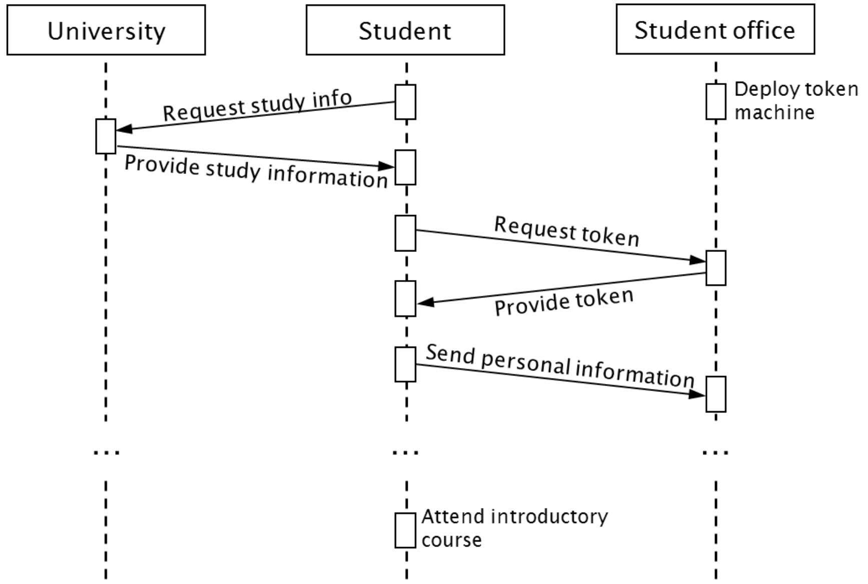
INFORMATIK UND
MATHEMATIK

Short overview over our modeling languages: Message Sequence Charts

Message Sequence Charts (MSCs):

What are main characteristics of UML Sequence diagrams?

- MSCs are similar to UML Sequence diagrams
- An MSC represents a set of partially ordered events
- Can be used to model (simple) distributed scenarios
- Can be used to test or verify certain properties of distributed systems, e.g.:
  - Boundedness of buffers
  - Realizability
  - Timing constraints (race conditions)

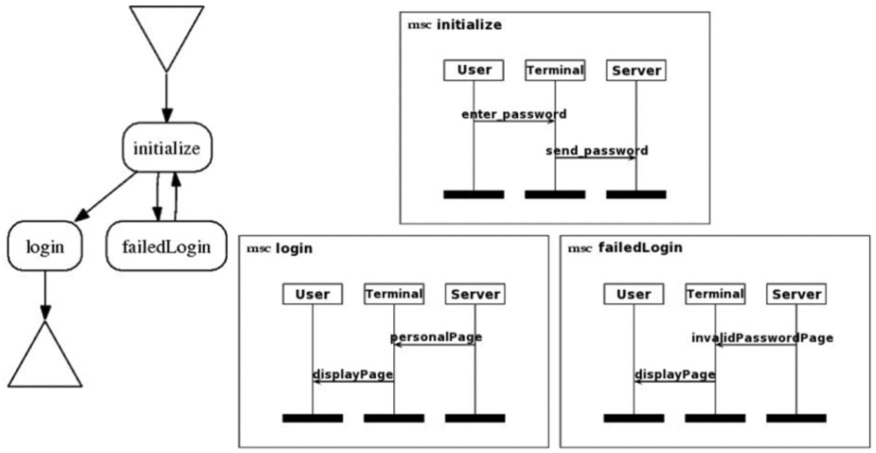


OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG
INFORMATIK UND MATHEMATIK

Short overview over our modeling languages: Message Sequence Graphs and CFMs

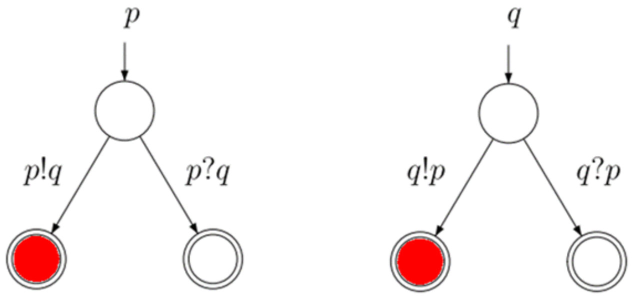
Message Sequence Graphs (MSGs):

- MSGs are graphs over MSCs
- An MSG visually represents a distributed system
- Properties of MSGs can be studied, to check a system's implementability



Communicating Finite-state Machines (CFMs):

- CFMs are sets of finite automata being able to communicate with each other
- A CFM formally represents a distributed system
- The model of CFMs is close to a code implementation
- Properties of CFMs can be studied, to check a system's implementability

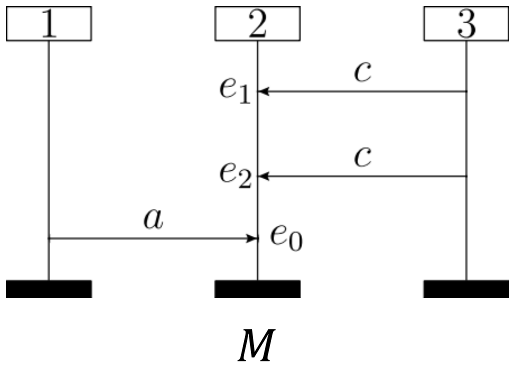


OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG
INFORMATIK UND MATHEMATIK

Short overview over our modeling languages: Propositional Dynamic Logic (PDL)

Propositional Dynamic Logic (PDL):

- A formal language (logic)
- A language for specifying constraints over MSCs/sequence diagrams
- Talks about events in your distributed system
- E.g.:



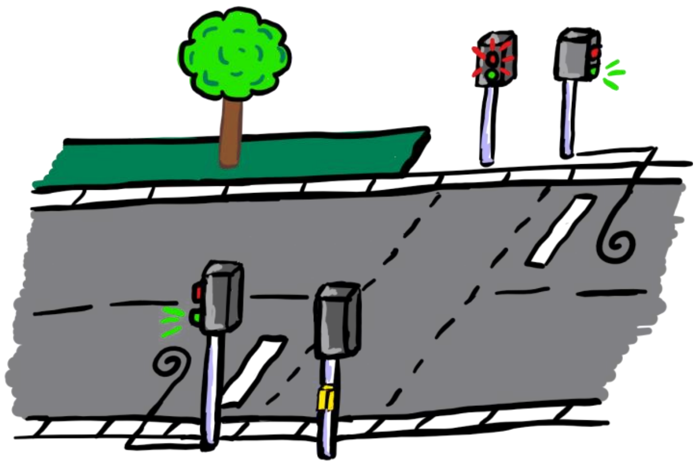
$$M, e_1 \vDash \text{procmin}$$
$$M, e_0 \nvDash \text{procmin}$$

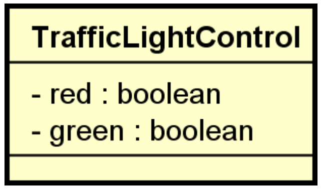
$$M \vDash \mathrm{E}\big(\text{procmax} \wedge ?\,(2,1,a)\big)$$

Short overview over our modeling languages: Object Constraint Language (OCL)

Object Constraint Language (OCL):

- A formal language (sublanguage of the UML, i.e., standardized)
- A language for specifying constraints over UML-class/object and state diagrams
- Syntax similar to programming languages/SQL
- Talks about alive objects in your system
- E.g.: a traffic-light control system:





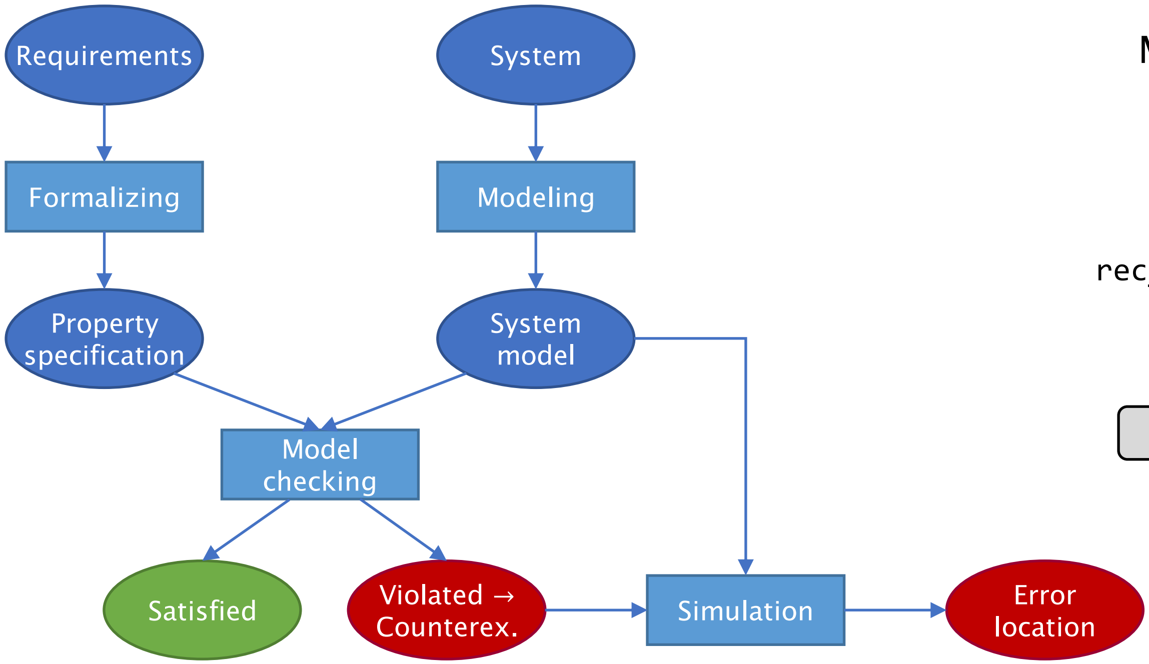
It may never happen that the traffic-light control allows:
`red=green=true`

```
context TrafficLightControl
inv: not (red and green)
```

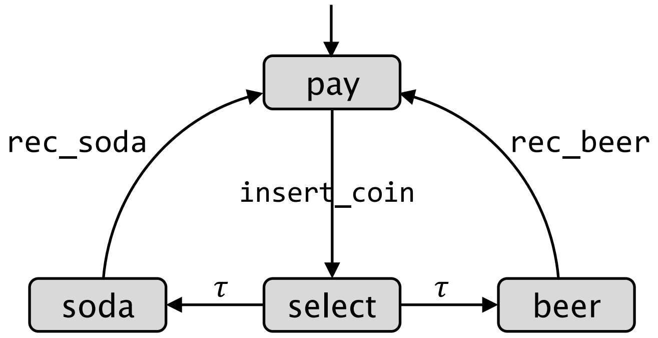
Short overview over our modeling languages: Model Checking

CTL Model Checking:

- A formalism for checking formal properties of a system
- Uses automata and logics (here CTL: Computation Tree Logic)
- E.g.: finding a bug in a beverage vending machine:



Model: DFA/NFA/Buchi-Automata



Literature

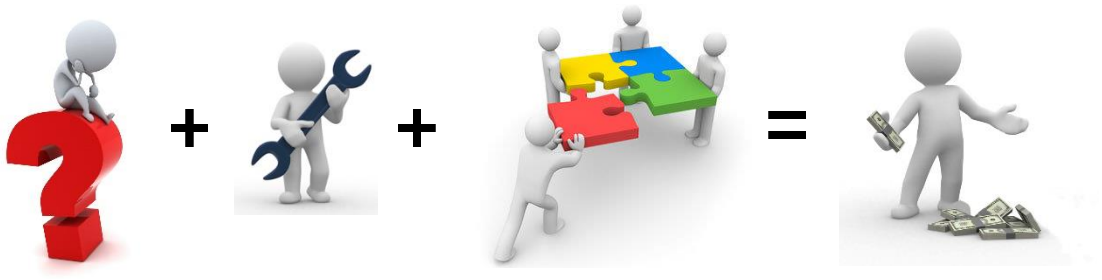
Business processes

Motivation

Definition: Business Process

There are several different definitions for what a business process is:

- Sequence of business activities
- Having a unique start point and defined end points
- Striving for a defined result (usually value creation for a customer)
- Supported and eased by using computers
- Shortly:
  "A set of dependent activities which create some kind of value"



Goal of business processes

Business processes describe value creation of an enterprise and therefore have to be executed as follows:

- Safe:
  - When repeating a business process the same result using the same amount of effort has to be achieved
- Economic:

  - Result has to be achieved with as low costs as possible

  → Automatization
- Efficient:

  - i.e., it should bind as little resources as possible

  → Automatization
- Fast:
  - i.e., result has to be created in lowest possible time

  → Automatization

Examples of business processes

Examples in general:	(German translation)
• Bid proposal	(Angebotserstellung)
• Contacting new customers	(Kontaktaufnahme Neukunden)
• Order transaction	(Auftragsabwicklung)
• Recruiting/Human resource development	(Personalbeschaffung/-entwicklung)
• Materials procurement	(Materialbeschaffung)
• Sales process	(Vertriebsprozess)
• Quality assurance	(Qualitätssicherung)
• Customer relations/support	(Beschwerde-, Feedbackbearbeitung)
• …	

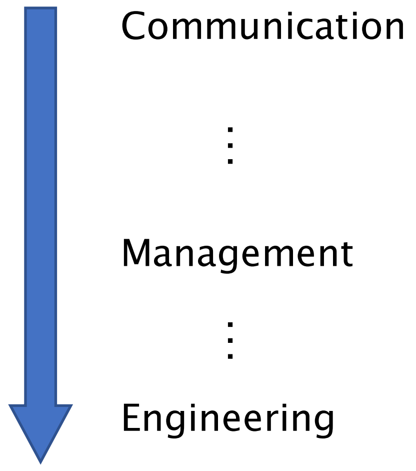
Concrete examples:	
• Registration at Facebook	
• Ordering a book at Amazon	
• Processing claims at Capitol insurance	(Schadenfallabw. bei der Capitol)
• Credit check at Sparkasse	
• Customer complaint process at Zalando	
• Processing a feature request at Threema	
• …	

OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

INFORMATIK UND
MATHEMATIK

Definition: Business process model

- Simplified illustration of a business process (at a certain level of abstraction)
- Describes a temporal and logical sequence of actions
- Serves for the following purposes:
  - Interdisciplinary information exchange about business process
  - Process design
  - Documentation
  - Analysis
  - Optimization
  - Implementation
  - Execution



OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG
INFORMATIK UND MATHEMATIK

Example: Business process model "Acquisition of a book at a bookstore" (in the past, i.e., before the internet)

Create a short process description at a rather high level of abstraction (~7 process steps):

1.
2.
3.
4.
5.
6.
7.

⇒ This is what we call a **current process** (German: Ist-Prozess)

Example: Business process model "Acquisition of a book at a bookstore" (today)

Create a short process description at a rather high level of abstraction (~7 process steps):

1.
2.
3.
4.
5.
6.
7.

⇒ This is what we can call a **target process** (German: Soll-Prozess)
- if it is not yet implemented and
- usually an optimization of the current process

Process engines: executing business processes

- Create interim report
- Call customer
- Generate vacancy

Manual steps + Automated steps = Process Engine

- Retrieve user data from DB
- Create invoice
- Book hotel via web site

Example: Business process management

Pit stop of a Formula-1racing car:

- The driver is the customer
- The customer expects an optimal service
- The process begins when the driver parks in the box
- Every engineer exactly knows his/her current area of operations
- The process is completed when the car leaves the box

Literature (German)

- Business Pprocess and Modeling Notation (BPMN):

  [1] Jakob Freund, Bernd Rücker, *Praxishandbuch BPMN 2.0*, Hanser, 2014, ISBN 978-3446442559. (available as eBook in library)

- Petri Nets (PNs):

  [2] Wolfgang Reisig, *Petrinetze: Modellierungstechnik, Analysemethoden, Fallstudien*, Vieweg + Teubner, 2010. (available as eBook in library)

- Message Sequence Charts/Graphs (MSCs/MSGs):

  [3] cf. course material on G.R.I.P.S.

Literature (English)

- Petri nets (PNs):

  [1] Wil van der Aalst, Christian Stahl: *Modeling Business Processes, A Petri Net-Oriented Approach*, MIT Press, 2011. (available as eBook in library)

  [2] Robert Gold, *Petri Nets in Software Engineering*, Working Papers FH Ingolstadt, 2004.

- Communicating automata (CFMs):

  [3] Carsten Kern: *Learning Communicating and Nondeterministic Automata*, Chapter 6.1, PhD Thesis, 2009.

- Propositional Dynamic Logic (PDL):

  [4] Carsten Kern: cf. [3], Chapter 7.1, PhD Thesis, 2009.

- Object Constraint Language (OCL):

  [5] OCL language specification, v2.4, Object Management Group, 2014

- Model Checking:

  [6] Chrisel Baier, Joost-Pieter Katoen: *Principles of Model Checking*, MIT Press, 2008.

  [7] Edmund Clarke: *Model Checking*, MIT Press, 2000.