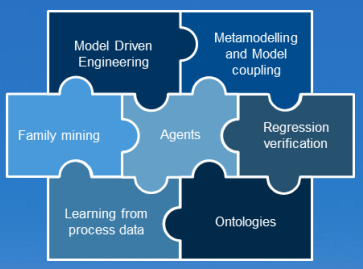
# SUMMER SCHOOL
# Enabling Technologies – Part I
# (Agents, Modeling, Notations for Automation)




**Univ.-Prof. Dr.-Ing. Birgit Vogel-Heuser**

Ordinaria
Automation and Information Systems (AIS)
Mechanical engineering,
Technische Universität München
www.ais.mw.tum.de; vogel-heuser@tum.de

**TUM ASIA SUMMER SCHOOL**
**24TH – 30TH August 2017**

Automation
and Information Systems
Technical University of Munich

## Agenda – TUM Asia Summer School 29th August

| | 29th August | 30th August |
|---|---|---|
| 9:00AM – 10:30AM | **Comparison of Industry 4.0, IoT, Smart Factory, Smart Data** | **Case Studies & Successful Demostrators: Applying Enabling Technologies** |
| 10:30AM – 11:00AM | MORNING TEA BREAK | |
| 11:00AM – 12:30PM | **PART I: Enabling Technologies (Agents, Modelling Notations for Automation)** | **Smart Data Enabled Learning During Operation** |
| 12:30PM – 01:30PM | LUNCH BREAK | |
| 01:30PM – 03:00PM | **PART II: Enabling Technologies (Agents, Modelling Notations for Automation)** | **Security and Human in the Loop** |

Complete Agenda: https://tum-asia.edu.sg/i4ss/

Automation
and Information Systems
Technical University of Munich

## Outline of part I

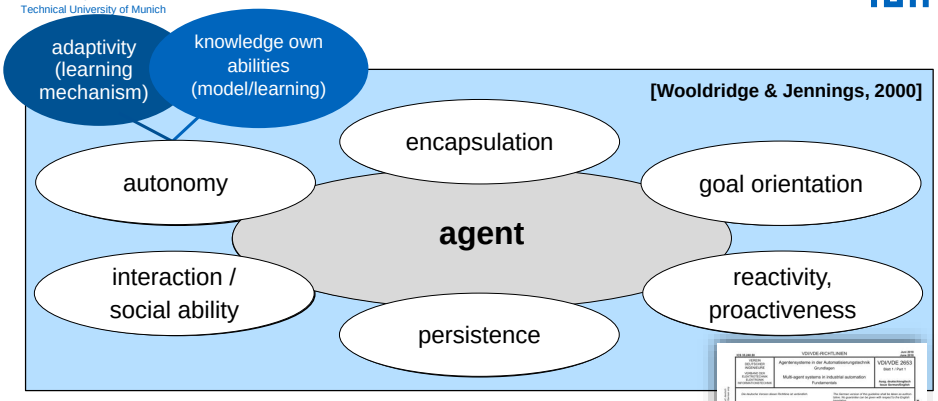
1. Principles and characteristics of agents
2. Application areas of software agents
3. Model Driven Engineering using Systems Modeling Language (SysML)
4. How to generate code out of a Model

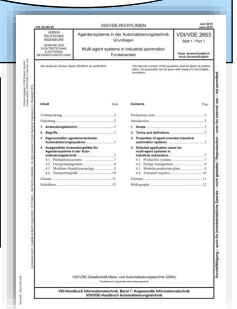




Automation and
Information Systems
Technical University of Municipal

## Properties of technical agents



A **technical agent** is an **encapsulated** (hardware/software) entity with **specified objectives**.
An agent endeavours to reach these objectives through its **autonomous behaviour**, in **interacting** with its **environment** and with **other agents**.

**Source:** VDI-Standard 2653 Sheet 1, 2010

## Terms of Agent Systems

| | |
|---|---|
| **Encapsulation** | • State and behavior encapsulated in the agent |
| **Target orientation** | • Agent-oriented behavior aiming a specific goal<br>• Objectives either set by the developer (for example, to optimize the agent's state permanently) or transferred from an user as an order at the execution time |
| **Reactivity** | • Ability to perceive the environment and react appropriately |
| **Autonomy** | • Control of internal state and behavior<br>• Determines behavior mainly by agents themselves and not fixable from "outside"<br>• Precondition for proactivity |
| **Proactivity** | • Ability to act targeted and with foresight without direct influence from the outside (self-initiative)<br>• Basis for proactivity is the existence of goals (see target orientation) |
| **Interaction** | • Agents interact with each other at a high level of interaction (e.g. negotiations)<br>• Basis for the interactions between agents is the underlying organizational context<br>• Interactions of agents in order to achieve individual goals or to handle interdependencies |
| **Persistence** | • Agent has continuous control flow, which depends on external activation<br>• Ability to maintain its internal state during its life cycle |







## What are the possible problems and benefits of the control of the production by software agents?

Issues:

## Advantages:

Automation and
Information Systems
Technical University of Munich

## Morphological box for classifying agents- Which agent types are most appropriate for field level agents?

| Property | characteristic | | | | | |
|---|---|---|---|---|---|---|
| autonomy | autonomous | | passive | | | |
| architecture | reactive | | proactive | | | |
| | memoryless | stored | FIPA | BDI | InteRRaP | … |
| communication | synchronous | | asynchronous | | | |
| language | jointly defined | | interpretive | | | |
| protocol definition | Petri nets | automata | text | | | |
| world model | together | | Individually with intersection | | | |
| adaptivity | Not adaptively | learning mechanism | | | | |
| | | imitative | self-critically | rule-based | | |
| mobility | mobile | | stationary | | | |
| competition | cooperative | | competitive | | | |
| method of cooperation | negotiating | | dominant | | | |
| knowledge of their own abilities | Non | resources | skills | | | |
| perception of the environment | through communication | | through observation | | | |

**Source**: A. Wannagat und B. Vogel-Heuser: Kopplung von regelungstechnischer Analyse und Agentensystemen. *Holleczek, P.; Vogel-Heuser, B. (Hrsg.): Echtzeitsysteme im Alltag, Informatik aktuell, Springer Berlin Heidelberg*, 36-45.

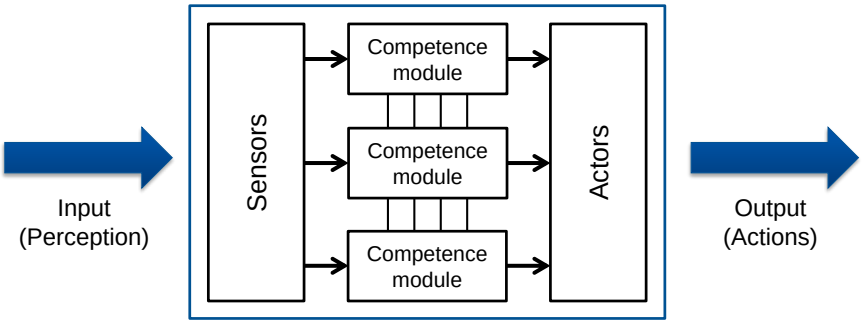




Automation and
Information Systems
Technical University of Munich

## Architecture of reactive agents

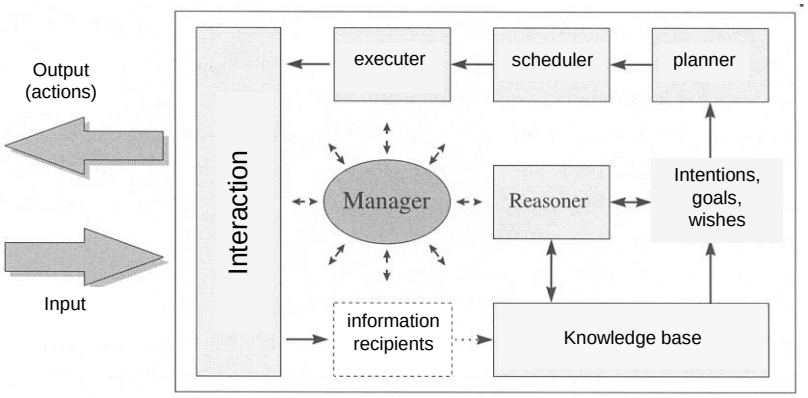
- Reactive agents (stimulus-response)
- No internal symbolic model of the environment
- No complex conclusion processes
- Compact, fault-tolerant and flexible agents
- Intelligence due to environment interactions
- Through continuous interaction intelligence arises and increases




[Source: Brenner, W., Zarnekow, R., Wittig, H.: Intelligente Softwareagenten, Heidelberg; 1998]

## Architecture of a deliberative agent

- Deliberative Agent
- Explicit, symbolic model of the environment
- Ability of logical conclusion as a basis for intelligent action
- Modeling the environment in first, knowledge base of agents






[Source: Brenner, W., Zarnekow, R., Wittig, H.: Intelligente Softwareagenten, Heidelberg; 1998]



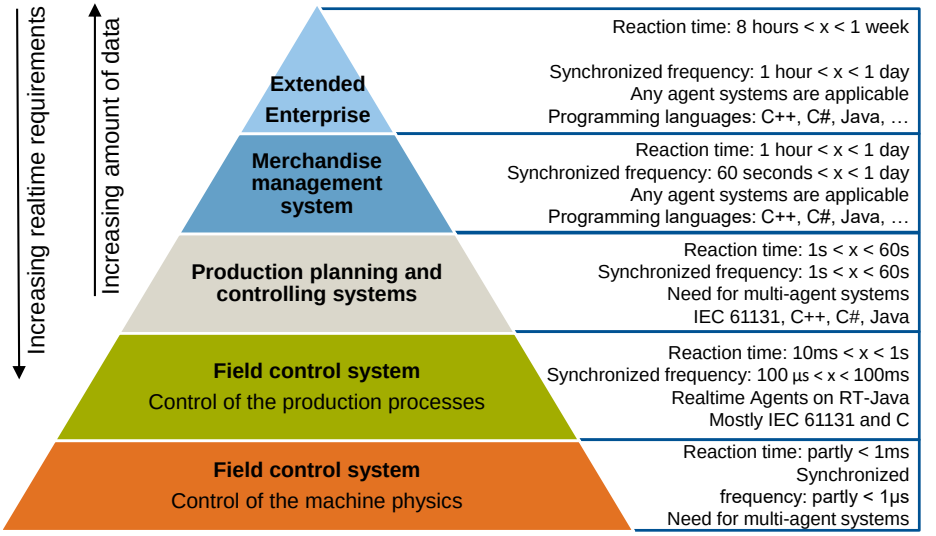


## Outline of part I

1. Principles and characteristics of agents
2. Application areas of software agents
3. Model Driven Engineering using Systems Modeling Language (SysML)
4. How to generate code out of a Model

Automation and
Information Systems
Technical University of Munich

## Fields of application for agents



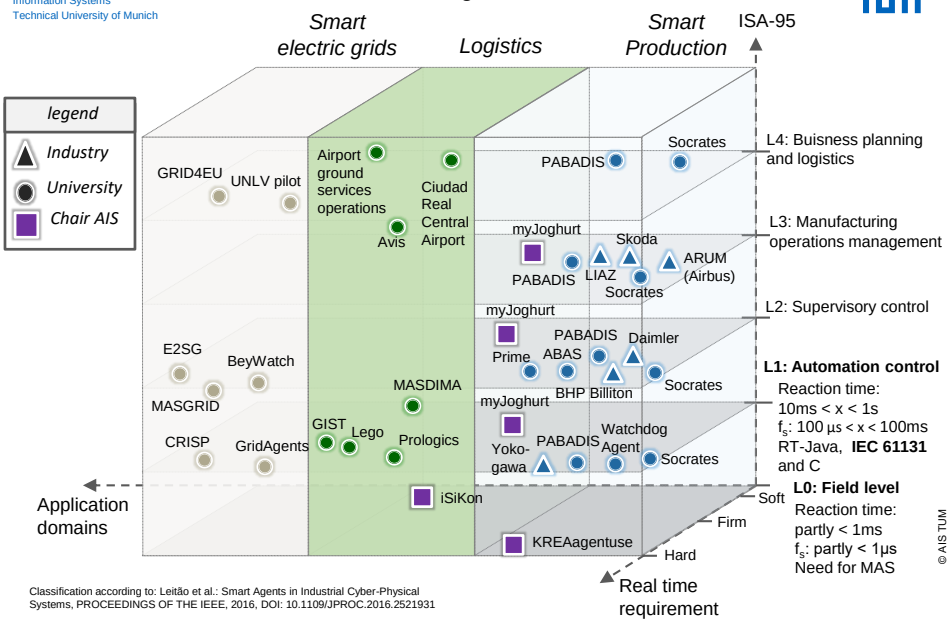



**Source:** See Lüder, A. Möglichkeiten und Grenzen Agentenbasierter Steuerungssysteme; 2006



Automation and
Information Systems
Technical University of Munich

## State of the art- agents in CPS/CPPS

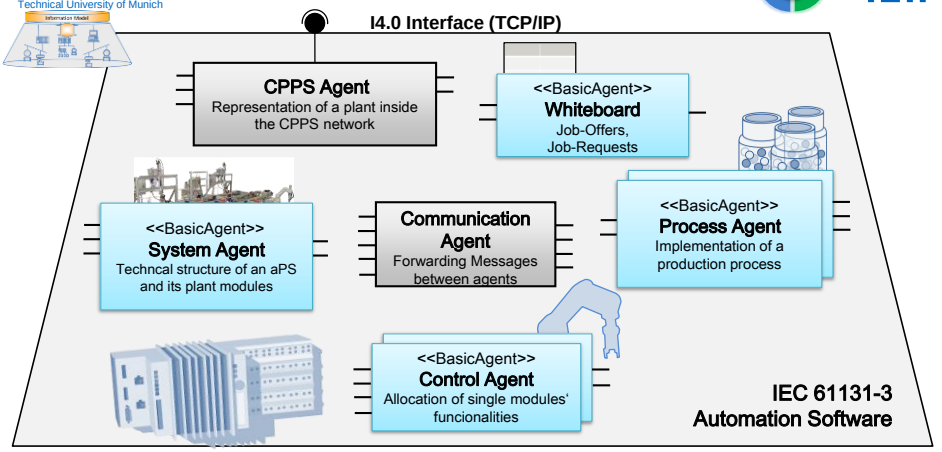





Classification according to: Leitão et al.: Smart Agents in Industrial Cyber-Physical Systems, PROCEEDINGS OF THE IEEE, 2016, DOI: 10.1109/JPROC.2016.2521931

Automation and
Information Systems
Technical University of Munich

## Definition of an agent




**CPPS Platform (Library)**

A **technical agent** is an **encapsulated** (hardware/software) entity with **specified objectives**. An agent endeavours to reach these objectives through its **autonomous behaviour**, in **interacting** with its **environment** and with **other agents**.

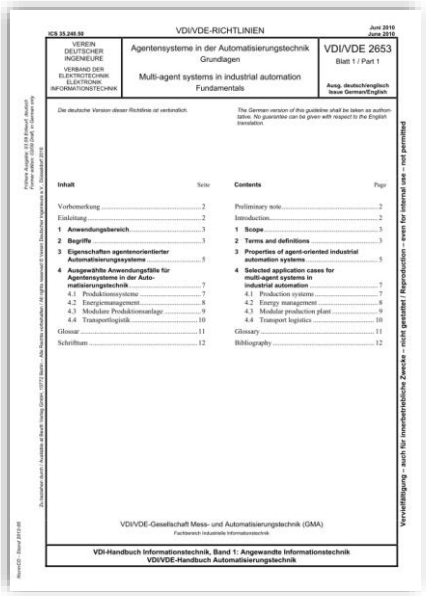
**Source:** VDI-Standard 2653 Sheet 1, 2010





Automation and
Information Systems
Technical University of Munich

## Guideline VDI/VDE 2653 – Part 1



VDI/VDE-RICHTLINIEN

ICS 35.240.50 — Juni 2010 / June 2010

VEREIN DEUTSCHER INGENIEURE
VERBAND DER ELEKTROTECHNIK ELEKTRONIK INFORMATIONSTECHNIK

Agentensysteme in der Automatisierungstechnik
Grundlagen

Multi-agent systems in industrial automation
Fundamentals

VDI/VDE 2653
Blatt 1 / Part 1

Ausg. deutsch/englisch
Issue German/English

*Die deutsche Version dieser Richtlinie ist verbindlich.*

*The German version of this guideline shall be taken as authoritative. No guarantee can be given with respect to the English translation.*

**Inhalt** Seite

Vorbemerkung ..... 2
Einleitung ..... 2
**1 Anwendungsbereich** ..... 3
**2 Begriffe** ..... 3
**3 Eigenschaften agentenorientierter Automatisierungssysteme** ..... 5
**4 Ausgewählte Anwendungsfälle für Agentensysteme in der Automatisierungstechnik** ..... 7
4.1 Produktionssysteme ..... 7
4.2 Energiemanagement ..... 8
4.3 Modulare Produktionsanlage ..... 9
4.4 Transportlogistik ..... 10
Glossar ..... 11
Schrifttum ..... 12

**Contents** Page

Preliminary note ..... 2
Introduction ..... 2
**1 Scope** ..... 3
**2 Terms and definitions** ..... 3
**3 Properties of agent-oriented industrial automation systems** ..... 5
**4 Selected application cases for multi-agent systems in industrial automation** ..... 7
4.1 Production systems ..... 7
4.2 Energy management ..... 8
4.3 Modular production plant ..... 9
4.4 Transport logistics ..... 10
Glossary ..... 11
Bibliography ..... 12

VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik (GMA)
Fachbereich Industrielle Informationstechnik

VDI-Handbuch Informationstechnik, Band 1: Angewandte Informationstechnik
VDI/VDE-Handbuch Automatisierungstechnik

Vervielfältigung – auch für innerbetriebliche Zwecke – nicht gestattet / Reproduction – even for internal use – not permitted

<u>Multi-agent systems in industrial automation</u>

### Part 1: Fundamentals
Terminology for agents used in automation, basic concepts and properties of agent systems
User groups: Operators and developers

### Part 2: Development
Criteria for the selection and for the comparison of agent-oriented development methods and platforms
Analysis of existing agent-oriented development methods
User groups: Those who are concerned with the development of multi-agent systems

### Part 3: Application
Reports (industry and universities) using multi-agent systems in different application fields
Definition of the problem and the problem solution with agents as well as the advantage of using agents
User groups: Those who are concerned with the development of multi-agent systems

Automation and
Information Systems
Technical University of Munich

## Revision of the guideline VDI/VDE 2653 – Part 1

- New chapter „Learning“:
  - Integration of rule-based learning approaches, e.g. learning classifier systems, for critial but comprehensible control tasks
  - Classifier System reviews the applicated rules. → Good rules are kept in rule set. Unusable rules are deleted in rule set
  - Initial learning process can be reduced through a priori rules
  - Using rules based on fuzzy logic improves the adaptation of the rules
- New chapter „Energy“:
  - New sub-chapters „Approach for controlling small distributed energy systems“, „Multimodal energy systems“, „Home automation“, „Microgrids“, „Virtual power plants“, „Energy management in vehicles“
  - Switch to renewable energies requires small and distributed energy conversion systems
  - Realisation through multi-agent system
- New chapter „Mobile transport robots“
  - Using multi-agent systems for the control and communication of mobile transport robots
  - Collaboration between MAS to accomplish joint tasks
- New chapter „Data analysis “
  - Use the resulting data from different systems to generate more information and knowledge for the expansion of business areas and models from the collected data
  - Horizontal and vertical coupling of heterogeneous systems
  - If damage was recorded, compensation strategies are applicable
- New chapter „Smart Environment“
  - Complexes and hierarchical control structures are implemented by decentralized intelligent algorithms
  - Information about the building and the current situation are stored in a knowledge-based, semantic building model

– **New part 4: metrics**





Automation
and Information Systems
Technical University of Munich

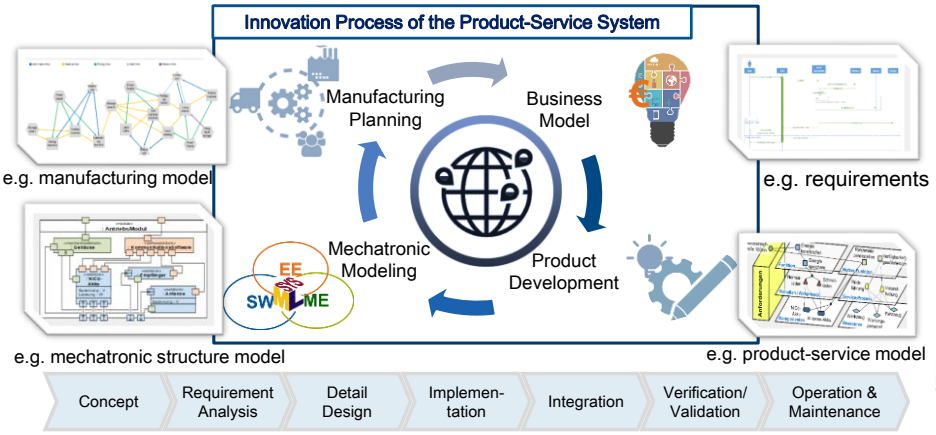
## Outline of part I

1. Principles and characteristics of agents
2. Application areas of software agents
3. Model Driven Engineering using Systems Modeling Language (SysML)
4. How to generate code out of a Model

Automation and
Information Systems
Technical University of Munich

## Product-Service-Systems Innovations

- Distributed and Collaborative Development
- Cross-Domain Dependency
- High Flexibility and Variability in Innovation Processes

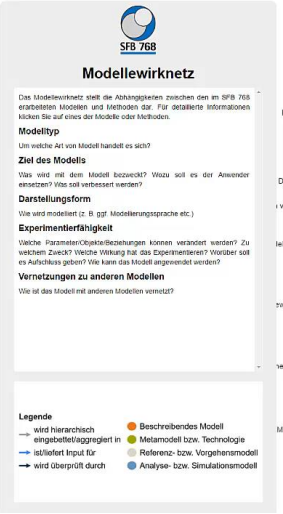



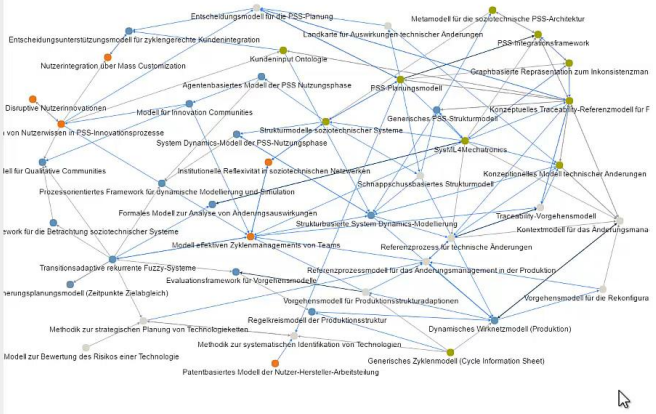





Automation and
Information Systems
Technical University of Munich

## Overview on the CRC 768 Model Network

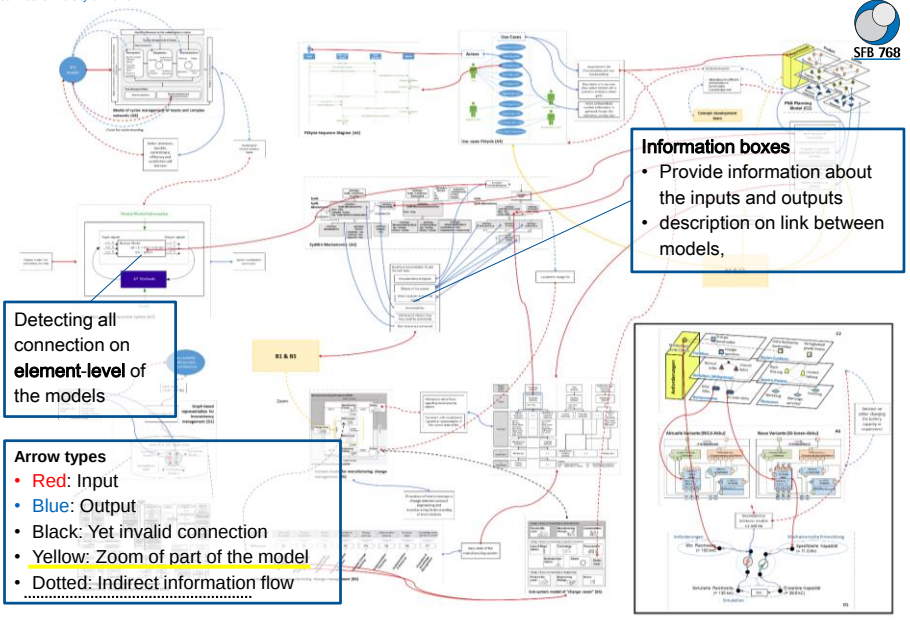


Automation and
Information Systems
Technical University of Munich

## Visualization of the inconsistencies in coupled engineering models

SFB 768



**Information boxes**
- Provide information about the inputs and outputs
- description on link between models,

Detecting all connection on **element-level** of the models

**Arrow types**
- Red: Input
- Blue: Output
- Black: Yet invalid connection
- Yellow: Zoom of part of the model
- Dotted: Indirect information flow

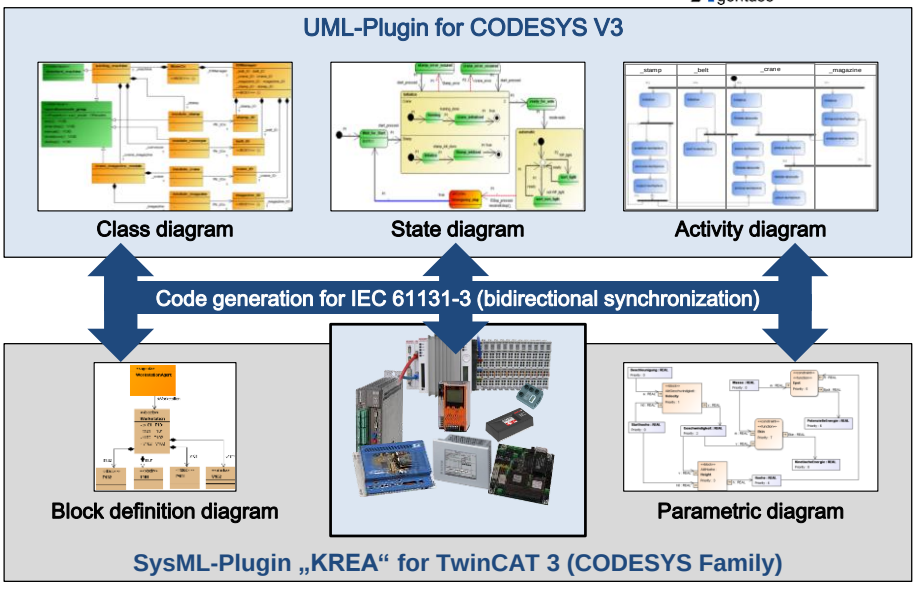




Automation and
Information Systems
Technical University of Munich

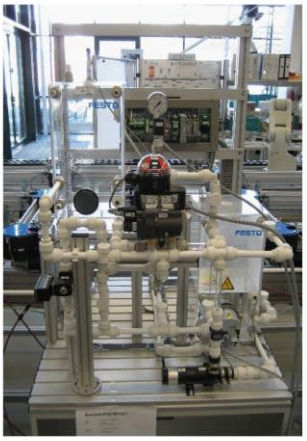
## UML-based implementation of IEC 61131-3 code with SysML expansion for software agents

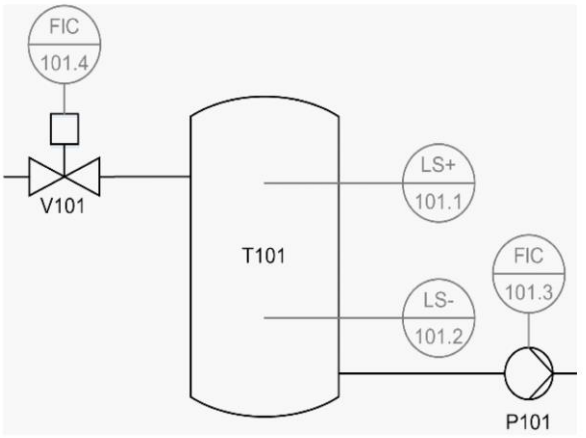
KREA gentuse



Automation and
Information Systems
Technical University of Munich

## Tank with upper and lower filling level sensors, valve and pump








**Source:** P. Hehenberger, B. Vogel-Heuser, D. Bradley, B. Eynard, T. Tomiyama, S. Achiche: „Design, Modelling, Simulation and Integration of Cyber Physical Systems: Methods and Applications“, 2016

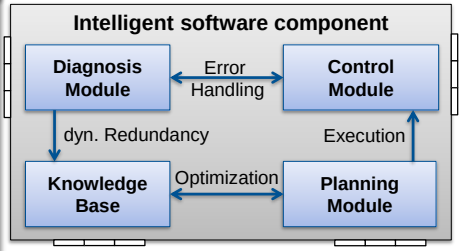


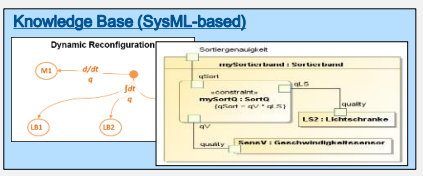
Automation and
Information Systems
Technical University of Munich

## Level 0 - fault tolerant sensor (Model based with agent)

**Application in process control**

**Intelligent software component**




**Templates referred to the MOFM2T-standard**

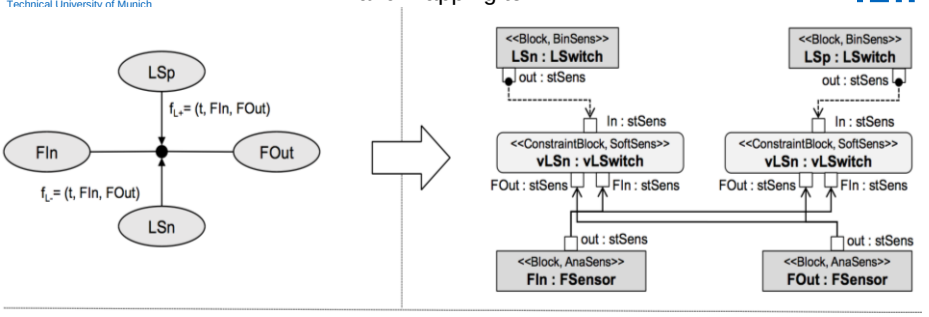
PLC (IEC 61131-3)

| | | | |
|---|---|---|---|
| ADR (A1.out) | ADR (A2.out) | ADR (B1.out) | ADR (B2.out) |
| | ADR (VA2.out) | | |
| | | ADR (VB1.out) | |
| | | | ADR (VB2.out) |

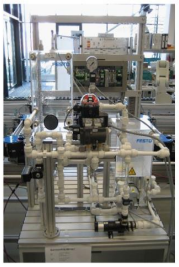
**Implementation of redundancy matrix**

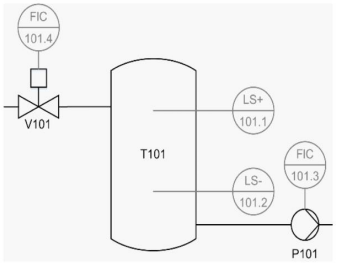
Automation and
Information Systems
Technical University of Munich

## Redundancy Model of Tank according to [SWL+13] and mapping to PAR




FIn = FIC 101.4 FOut = FIC 101.3 LSn = LS- 101.2 LSp = LS+ 101.1





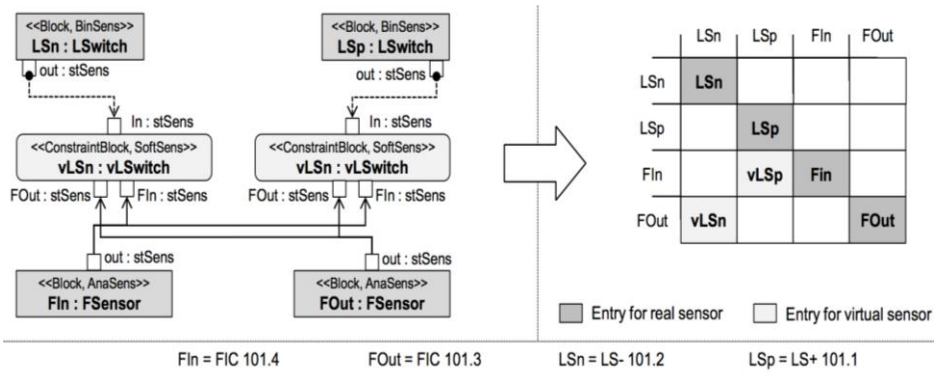

**Source:** P. Hehenberger, B. Vogel-Heuser, D. Bradley, B. Eynard, T. Tomiyama, S. Achiche: „Design, Modelling, Simulation and Integration of Cyber Physical Systems: Methods and Applications“, 2016





Automation and
Information Systems
Technical University of Munich

## Mapping of PAR onto initialization of the redundancy matrix



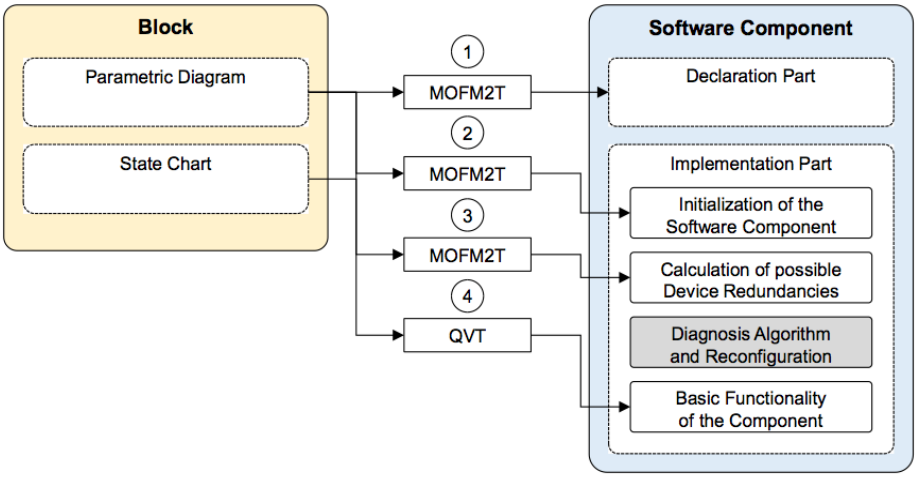

FIn = FIC 101.4 FOut = FIC 101.3 LSn = LS- 101.2 LSp = LS+ 101.1



**Source:** P. Hehenberger, B. Vogel-Heuser, D. Bradley, B. Eynard, T. Tomiyama, S. Achiche: „Design, Modelling, Simulation and Integration of Cyber Physical Systems: Methods and Applications“, 2016

Automation and
Information Systems
Technical University of Munich

## Agent-based self-aware, self-describing CPPS-module



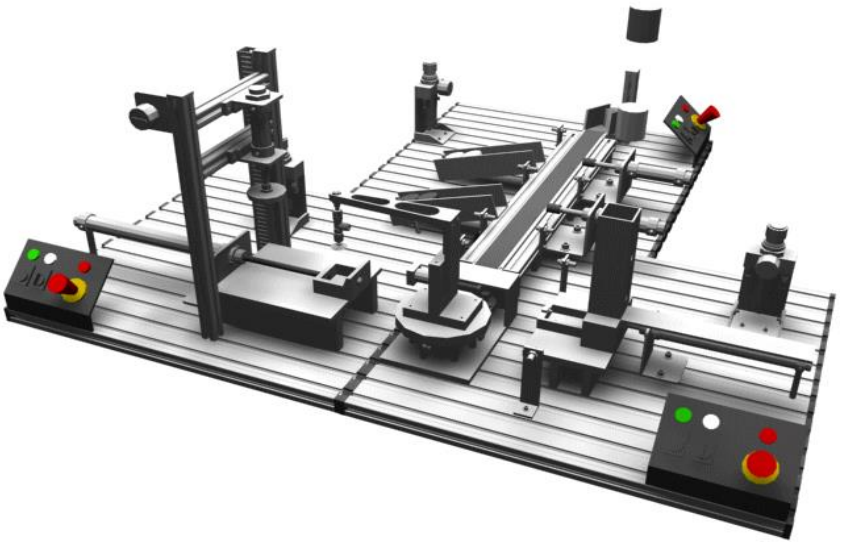

**Source:** P. Hehenberger, B. Vogel-Heuser, D. Bradley, B. Eynard, T. Tomiyama, S. Achiche: „Design, Modelling, Simulation and Integration of Cyber Physical Systems: Methods and Applications“, 2016





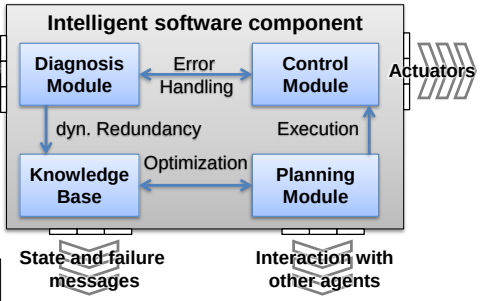
Automation and
Information Systems
Technical University of Munich

## Introduction of the small lab scale production system pick-and-place-unit (PPU)



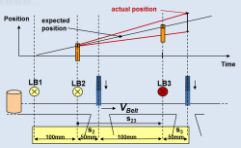
Automation and
Information Systems
Technical University of Munich

## Intelligent Software Components on PLCs



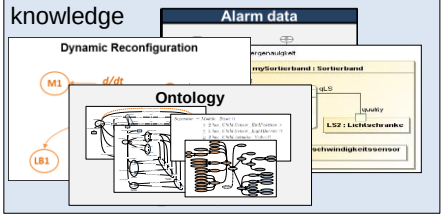

### Diagnosis Module

- Evaluation of sensors values
- Execution of failure diagnosis



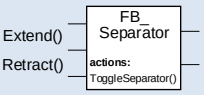

### Knowledge Base

Models of the agents' local knowledge




### Control Module

Control of the plant module or other subagents

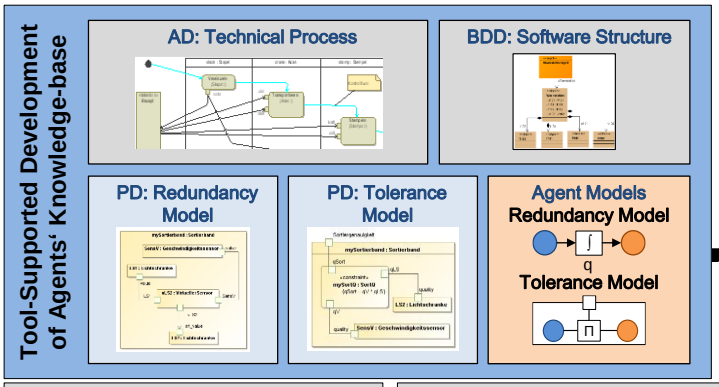






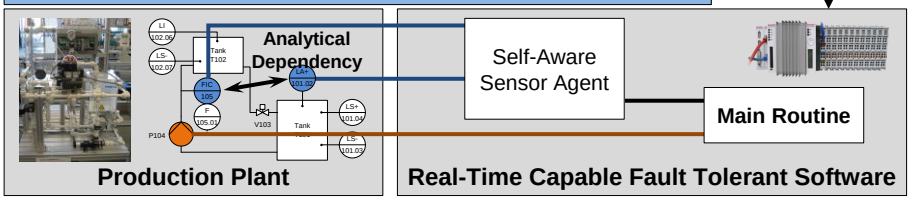

Automation and
Information Systems
Technical University of Munich

## KREAagentuse: SysML-based automation software development




- Based on project AVE
- Implementation of tool support

Model Transformation/ Code Generation

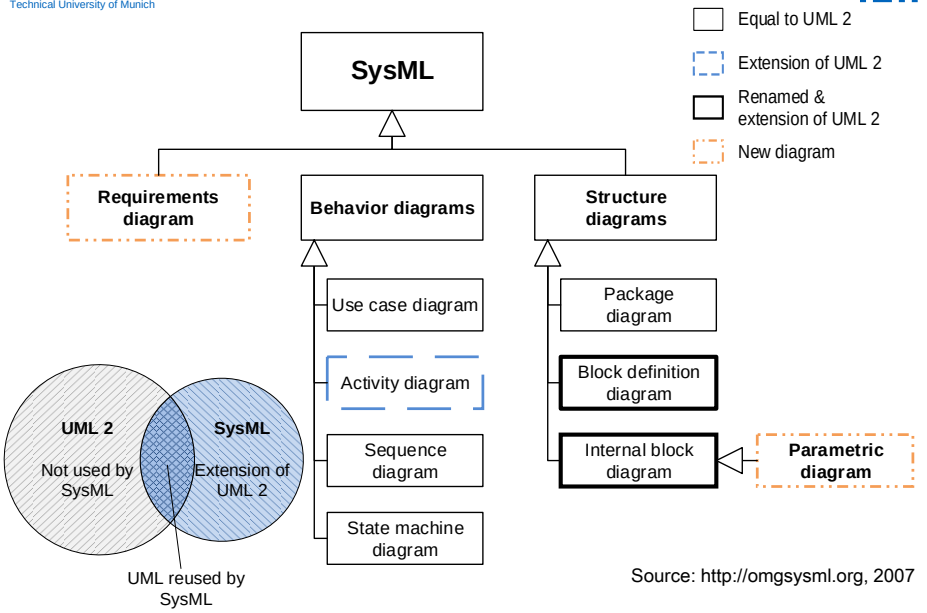





**Source:** Frank et al. 2011, Schütz et al. 2012, DFG funded project KREAagentuse

Automation and
Information Systems
Technical University of Munich

## Construction of SysML diagrams



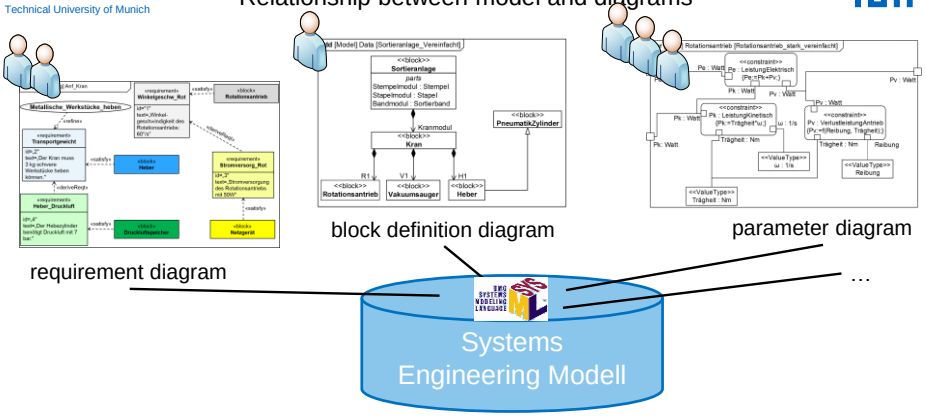

Source: http://omgsysml.org, 2007





Automation and
Information Systems
Technical University of Munich

## Systems Modeling Language (SysML)
## Relationship between model and diagrams




SysML offers **9 types of diagrams** (structure and behavior) for the representation of **different system aspects** for different stakeholders

→ Base, to develop mechatronic PSS **model-based** and **consistently**

→ Diagrams to support the modeling from the **early requirement design** to the **realization**

Automation and
Information Systems
Technical University of Munich

## Requirement diagram of the stamping process

Requirement in text form:
The workpiece must be stamped for at least 10s with a force of at least 10N.

**Chart name in the tool**

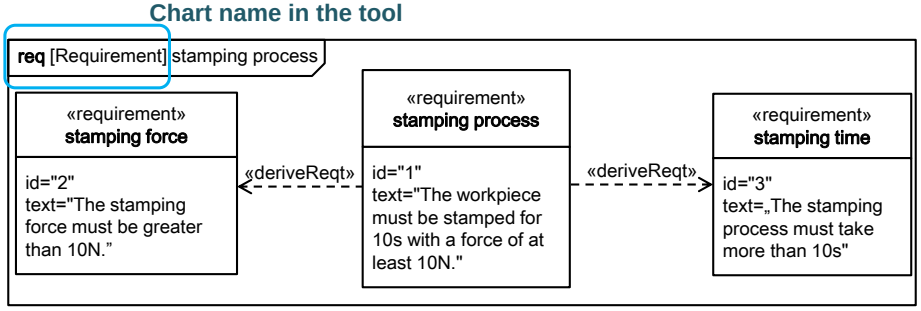


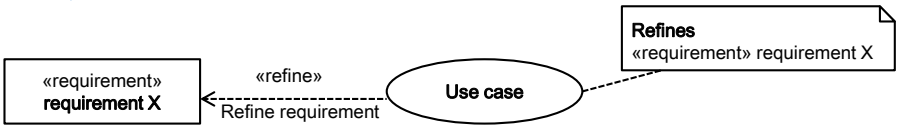

**table** [Requirement] Stempelprozess

| id | name | text |
|---|---|---|
| 1 | stamping process | "The workpiece must be stamped for 10s with a force of at least 10N" |
| 2 | stamping force | "The stamping force must be greater than 10N" |
| 3 | stamping time | "The stamping process must take more than 10s" |

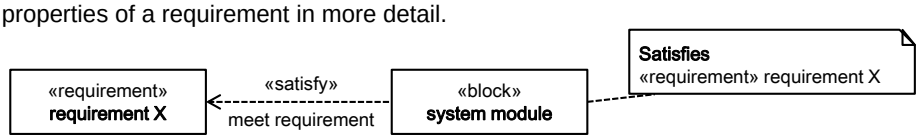


Automation and
Information Systems
Technical University of Munich

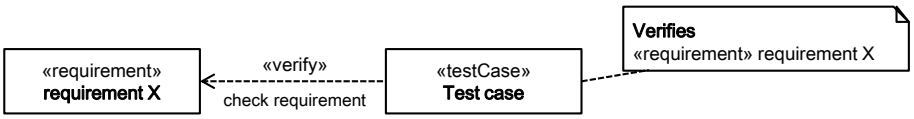
## Requirement diagram – Relationships between requirements




**«refine»** : Refine relationship describes that a model element describes the properties of a requirement in more detail.



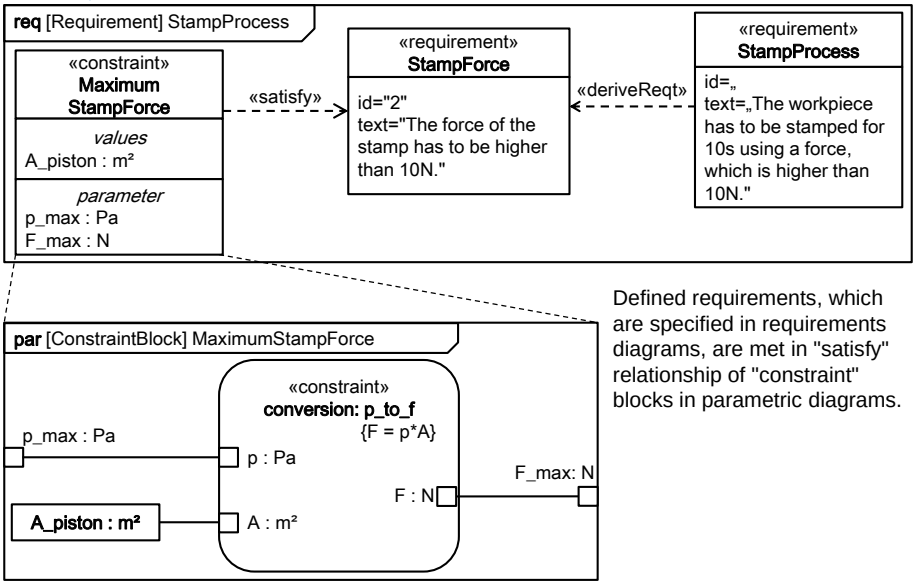

**«satisfy»** : Satisfy relationship describes design elements that fulfills a requirement.




**«verify»** : Verify relationship combines a test case with the requirement that is checked by the test case.

Automation and
Information Systems
Technical University of Munich

## Relationship between requirements diagram and parametric diagram




Defined requirements, which are specified in requirements diagrams, are met in "satisfy" relationship of "constraint" blocks in parametric diagrams.





Automation and
Information Systems
Technical University of Munich

## Structural modeling in SysML Overview BDD and IBD

**Gross design**

- The **Block Definition Diagram (BDD)** provides a "black box" view of a system block. It also describes the hierarchy of the sub-blocks of the system building block.
- The BDD can be compared with the first page of a building instruction for a piece of furniture. It shows the type and number of all parts in the package.

**Detailed design**

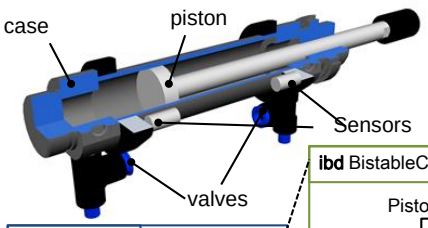
- The **Internal Block Diagram (IBD)** gives a "white box" or internal view to a system block. It shows how all parts of the system block shown are connected.
- The IBD can be compared with the last page of a building instruction for a piece of furniture. It shows how the individual parts are concretely connected.

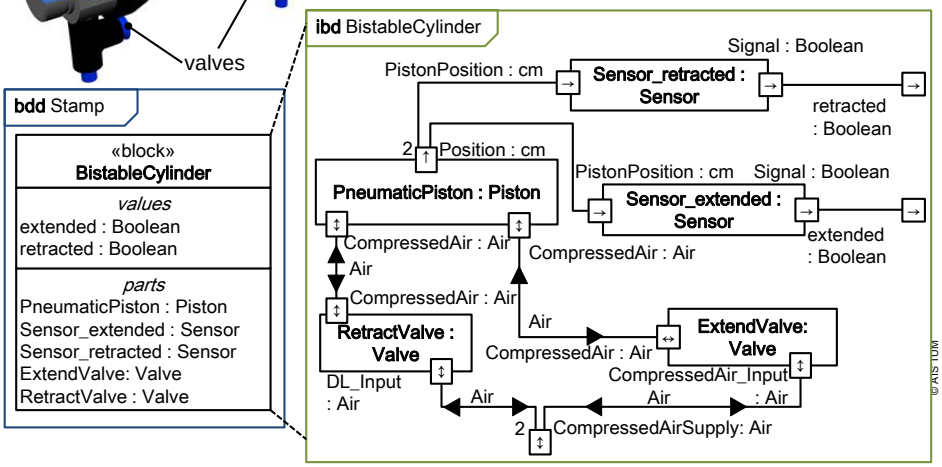
The **structure** of a system block is described in the **Block Definition Diagram** and the role of its **system components** is described in the **Internal Block Diagram**.

Automation and
Information Systems
Technical University of Munich

## Relationship between block definition diagram (BDD) and internal block diagram (IBD)

The relationships between parts of a block defined in the BDD are represented in the IBD.



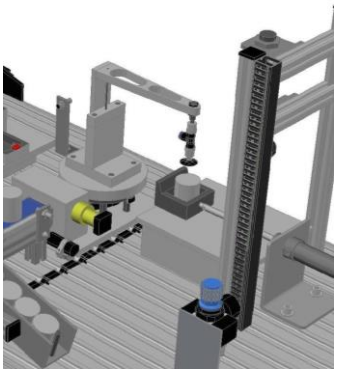





Automation and
Information Systems
Technical University of Munich

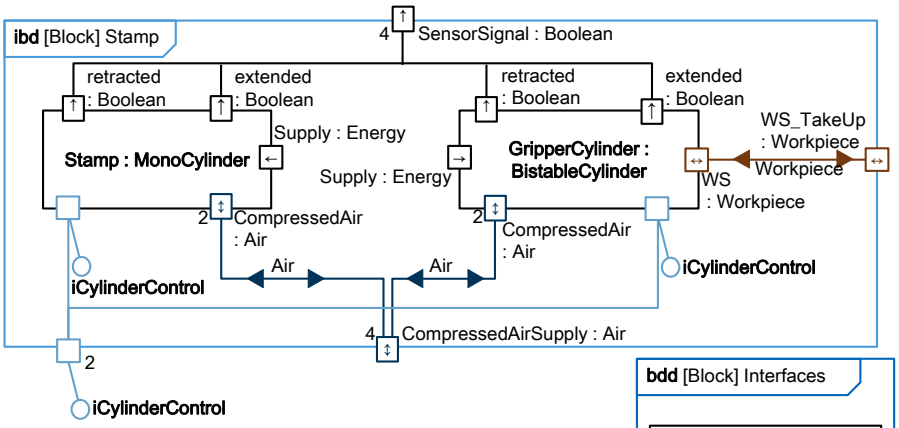
## Brainstorming

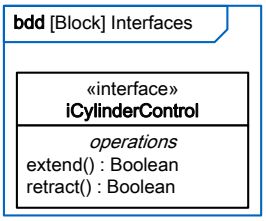
**Which ports, which interfaces, which object flows have to be specified for the stamp of the PPU?**



Automation and
Information Systems
Technical University of Munich

## Structure model – Internal block diagram (IBD)
## Example: Sorting plant: Block stamp



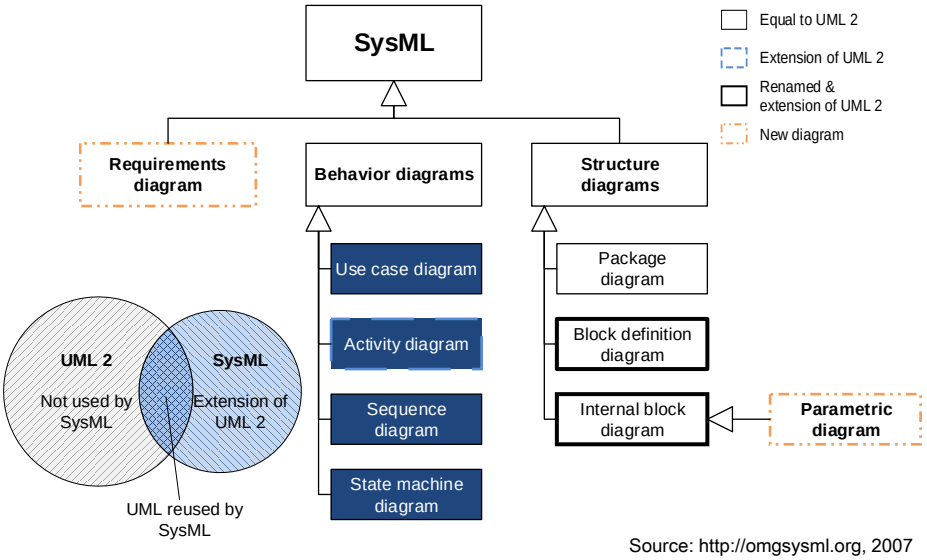

- Modelling power is neglected to retain the overview
- Air and workpiece are concrete objects, which are transmitted





Automation and
Information Systems
Technical University of Munich

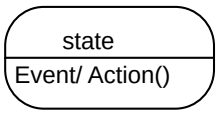
## Construction of SysML diagrams



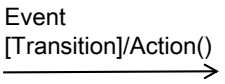



Source: http://omgsysml.org, 2007

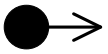
## Notation for state diagram




A simple state is a state that does not have substates - it has no regions and it has no submachine. The compartments of the state are: name compartment, internal activities compartment nad internal transitions compartment.

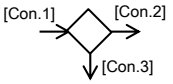



The transition is a directed relationship between a source state and a target state. Definition: 1. Trigger to execute state transition, 2. Condition to enter transition, 3. Behavior (action), which will be executed during the transition



The initial pseudostate. It shows on the first state. Outgoing transitions can only contain actions, but not contains

The final state.




The choice pseudostate enables to choice between transistions and to merge them.

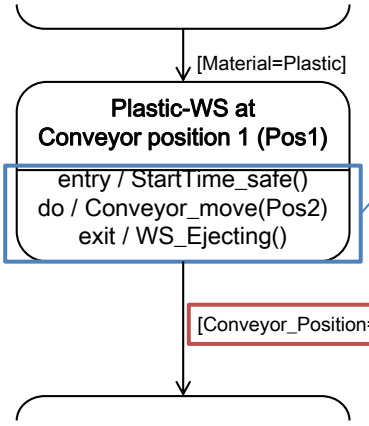
Source: Wellidens, 2006





Automation and
Information Systems
Technical University of Munich

## Behavior model – state diagram
## state description of the sorting plant




| Events / Actions() in state diagram | |
|---|---|
| entry / Action() | is executed, while entering the state |
| do / Action() | is executed as long as the condition is active, or until it is completed |
| exit / Action() | is executed, while exiting the state |

Transitions in state diagram:
Events [Condition]/Action

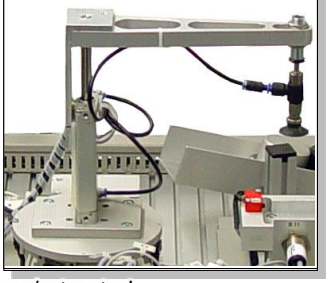


- In a state several methods of a class can be called (or other behavior diagrams)
- The state is exited if the subsequent transition is active: Then the Do-action is canceled and the Exit-action is executed once

Automation and
Information Systems
Technical University of Munich

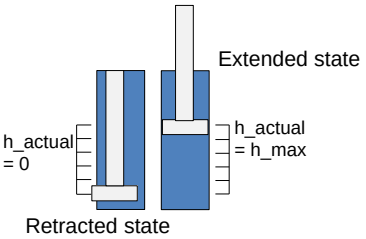
## State-based behavior– Important boundary conditions to model the crane

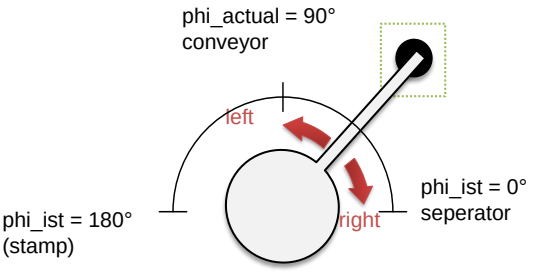
Up/ Extended



Down/retracted

- Crane is able to move to retracted and extended positions
- A pneumatic cylinder is used to extend the crane








## Behavior model: Notations of the activity diagram

An **activity** describes processes, which consist of several elementary actions.
<<precondition>>: activity is started only when the precondition is satisfied.
<<postcondition>>: After completing the activity, post-condition must be fulfilled.

Action1

**Action nodes** describe elementary process steps or subordinate activities. It is the most basic model element to describe behavior.

Object1

Alternatively, **object nodes** can be represented by rectangles. They can be used as incoming or outgoing parameters in activities.



**Activity edges** are directed relations between two nodes of an activity. A distinction is made between control and object flow. Both are graphically represented at the same way. Considering object flows, the arrow shows from or to an object node..

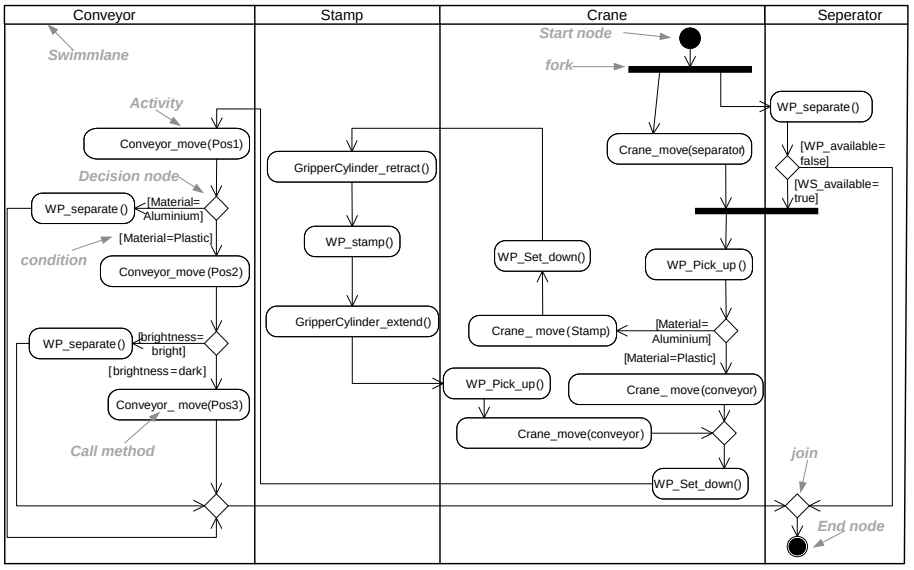


A **start node** is a starting point of a sequence in an activity diagram.

An **end node** ends the sequence, meaning all actions and control flows.

Automation and
Information Systems
Technical University of Munich

## Behavior model: Activity diagram
## Application example: Sorting process







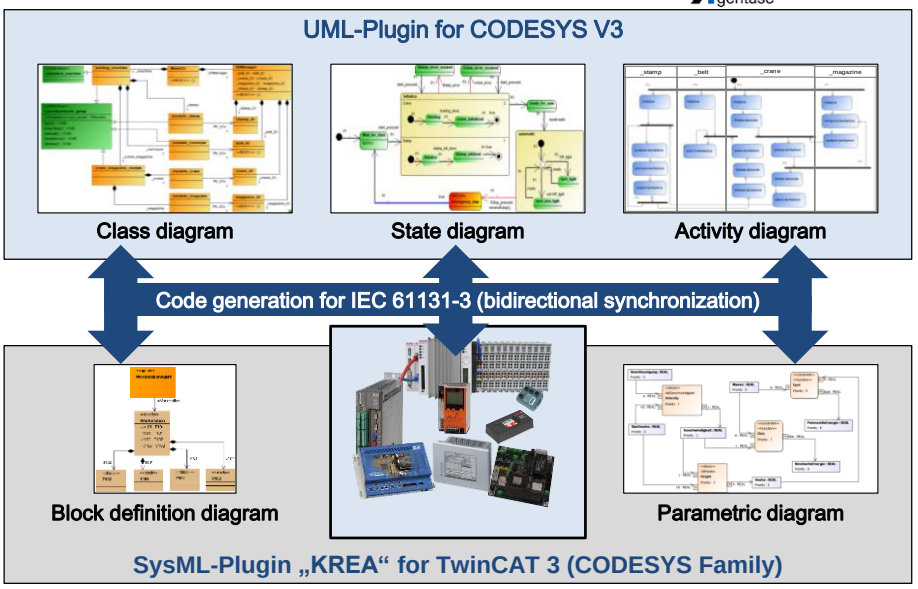

Automation
and Information Systems
Technical University of Munich

## Outline of part I

1. Principles and characteristics of agents
2. Application areas of software agents
3. Model Driven Engineering using Systems Modeling Language (SysML)
4. How to generate code out of a Model

Automation and
Information Systems
Technical University of Munich

## UML-based implementation of IEC 61131-3 code with SysML expansion for software agents

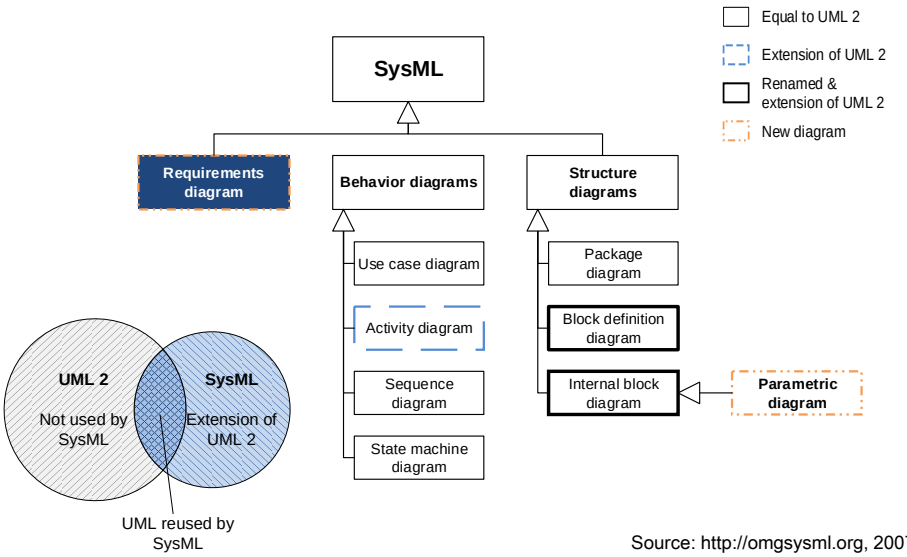







Automation and
Information Systems
Technical University of Munich

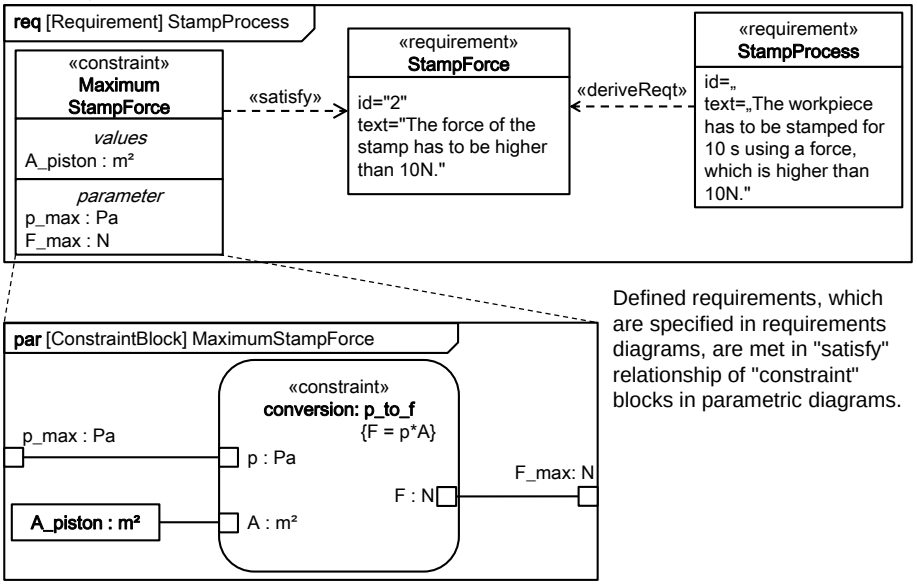
## Construction of SysML diagrams




Source: http://omgsysml.org, 2007

Automation and
Information Systems
Technical University of Munich

## Relationship between requirements diagram and parametric diagram




Defined requirements, which are specified in requirements diagrams, are met in "satisfy" relationship of "constraint" blocks in parametric diagrams.





Automation and
Information Systems
Technical University of Munich

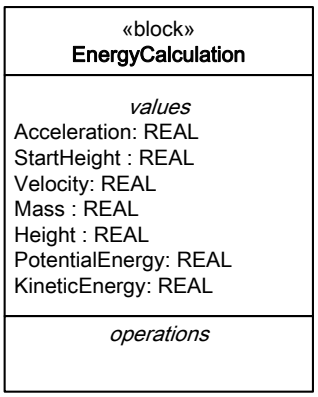
## Exemplary mapping between SysML and IEC 61131-3: block definition diagram

### block definition diagram

- Describes a system's structure in code conforming to IEC61131-3
- Visualization and structuring of the system, standard building blocks are stored centrally

### Example: energy calculation for the vertical fall

**SysML**




**IEC 61131-3**
**(Block → Function Block)**

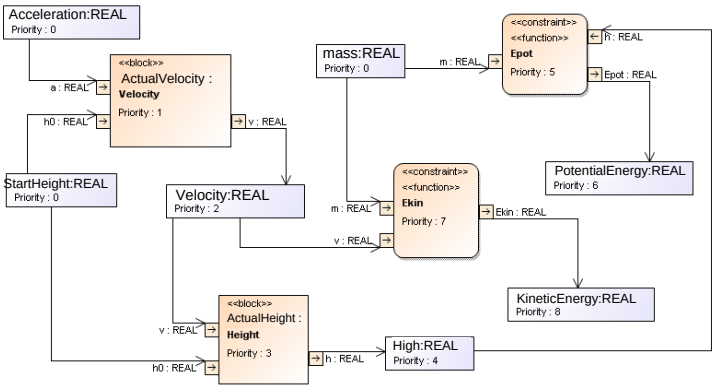
```
FUNCTION_BLOCK EnergyCalculation
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
    Acceleration: REAL;
    StartHeight : REAL;
    Velocity: REAL;
    Mass : REAL;
    Height : REAL;
    PotentialEnergy: REAL;
    KineticEnergy: REAL;
END_VAR
```

Automation and
Information Systems
Technical University of Munich

## Exemplary mapping between SysML and IEC 61131-3: Parametric diagram

### Parametric diagram
- Calling of IEC 61131-3 functions and function blocks
- Visualization and structuring of parametric relationships

### SysML





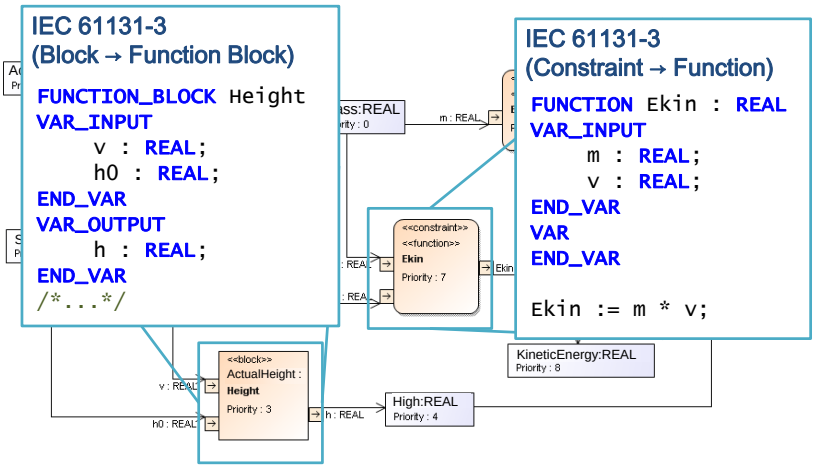



Automation and
Information Systems
Technical University of Munich

Exemplary mapping between SysML and IEC 61131-3: Parametric diagram

Parametric diagram
- Calling of IEC 61131-3 functions and function blocks
- Visualization and structuring of parametric relationships

### SysML

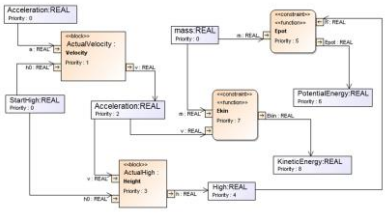


Automation and
Information Systems
Technical University of Munich

## Exemplary mapping between SysML and IEC 61131-3: Parametric diagram

**Parametric diagram**
- Calling of IEC 61131-3 functions and function blocks
- Visualization and structuring of parametric relationships

### SysML




**Problem**

What information is missing in pure SysML parametric diagram to automatically generate code?

### IEC 61131-3

```
ActualSpeed (a := Acceleration, h0 := StartHigh);
Speed:= ActualSpeed.h;
ActualHeight(v := Speed, h0 := StartHigh);
Height:= ActualHeight.h;
PotentialEnergy:= Epot(m := Mass, h := height);
KineticEnergy := Ekin(m := Mass, v := Speed);
```





Automation and
Information Systems
Technical University of Munich

## How can knowledge of software agents be modelled? Which kinds of knowledge have to be modelled?

### Which kinds of knowledge can be modelled with SysML?

### How can this knowledge be used during runtime?

### What is missing?

# Thank you for your attention!

**Univ.-Prof. Dr.-Ing. Birgit Vogel-Heuser**

Ordinaria
Automation and Information Systems (AIS)
Mechanical engineering,
Technische Universität München
www.ais.mw.tum.de; vogel-heuser@tum.de