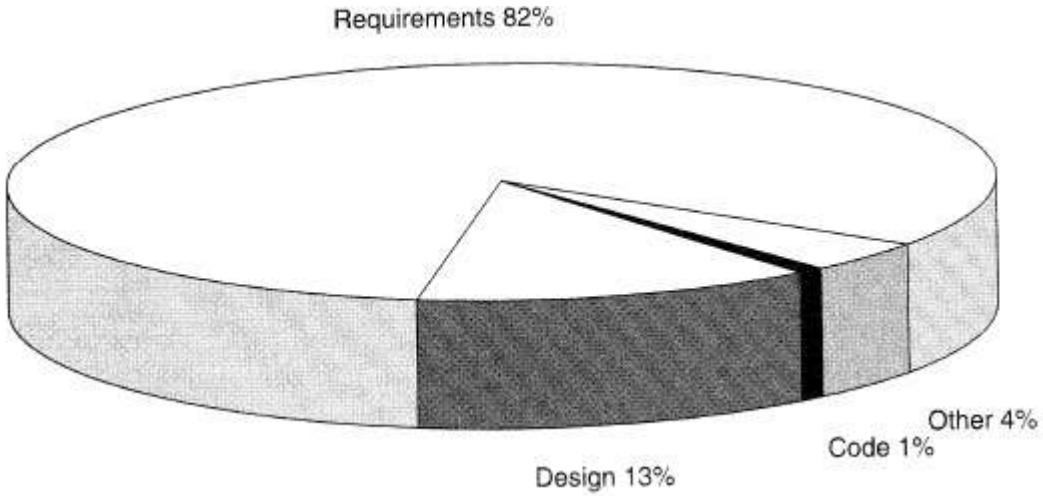
HOCHSCHULE
HAMM-LIPPSTADT

Stefan Henkler E-Mail: stefan.henkler@hshl.de

HOCHSCHULE HAMM-LIPPSTADT

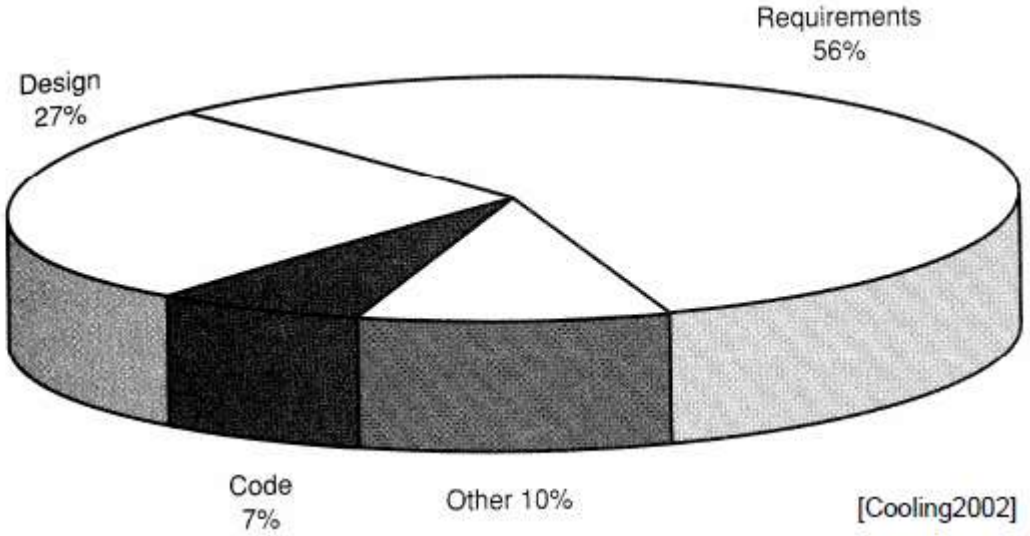
1. Requirements Engineering
2. Requirement Specification
3. Approach: SysML
4. Discussion & Summary
5. Bibliography

3 Why are Requirements so important?



Distribution of software errors

Cost of rectifying errors



[Cooling2002]

HOCHSCHULE HAMM-LIPPSTADT

- **Requirement Engineering (RE)** is the science and discipline concerned with analyzing and documenting requirements.

Requirement:

- (1) A condition or capability needed by a user to solve a problem or achieve an objective.
- (2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.
- (3) A documented representation of a condition or capability as in (1) or (2).

[IEEE-Std-610.12-1990]

HOCHSCHULE HAMM-LIPPSTADT

Input or output	Type	Description
Existing system information	Input	Information about the functionality of systems to be replaced or other systems which interact with the system being specified
Stakeholder needs	Input	Descriptions of what stakeholders need from the system to support their work
Organisational standards	Input	Standards used in an organisation regarding system development practice, quality management, etc
Regulations	Input	External regulations such as health and safety regulations with which the system must comply
Domain information	Input	General information about the application domain of the system
Agreed requirements	Output	A description of the system requirements which is understandable by stakeholders and which has been agreed by them

[Kotonya&Sommerville1998]

HOCHSCHULE HAMM-LIPPSTADT

a) **Requirement Elicitation**
b) **Requirement Analysis**
c) **Requirement Specification**
d) **Requirement Validation**
e) **Requirement Management**

[Thayer&Dorfman1997]

7 a) Requirement Elicitation

HOCHSCHULE HAMM-LIPPSTADT

- **Requirement Elicitation:** the process through which the customer and developer discover, review, articulate, and understand the users' needs and constraints on the software and development activities
- Requirements elicitation is about discovering what requirements a system should be based upon
- This doesn't involve just asking stakeholders what they **want.** It requires a careful analysis of:
  - The organisation
  - The application domain
  - Organisation processes where the system will be used
- To determine the stakeholders **need.**

[Thayer&Dorfman1997]

HOCHSCHULE HAMM-LIPPSTADT

- Establish objectives
  - Business goals
  - Problem to be solved
  - System constraints
- Understand background
  - Organizational structure
  - Application domain
  - Existing systems
- Organize knowledge
  - Stakeholder identification
  - Goal prioritization
  - Domain knowledge filtering
- Collect requirements
  - Stakeholder requirements
  - Domain requirements
  - Organizational requirements

HOCHSCHULE HAMM-LIPPSTADT

Who are they?

- Anyone with a stake in creating or using a new system
  - Hands-on users
  - Their managers
  - The system administrator
  - The client (system owner)
  - The requirements engineer
  - Other Engineers
  - Software designers
  - Programmers

Stakeholders' backgrounds vary: they may:

- come from different departments
- be trained in different disciplines
- have different (possibly conflicting) goals
- be unwilling to consider other stakeholders' goals
- have more or less political influence over requirements decisions

10 Uncovered Knowledge

HOCHSCHULE HAMM-LIPPSTADT

- Application domain knowledge
  - E.g. knowledge about airport systems
- Context knowledge
  - E.g knowledge about Denver Airport
- Problem knowledge
  - E.g. knowledge about Denver's baggage-handling system
- Stakeholders needs and work processes to be supported

HOCHSCHULE HAMM-LIPPSTADT

- Interviews
- Questionnaires
- Examination of documentation
  - Standards
  - Systems manuals
  - Statement of requirements
- Prototyping
- Contextual Design
- Conversation and interaction analysis

b) Requirement Analysis

HOCHSCHULE HAMM-LIPPSTADT

- **Requirement Analysis**: the process of analyzing the customers' and users' needs to arrive at a definition of the requirements
- **requirement analysis:**
  - (1) The process of studying user needs to arrive at a definition of system, hardware, or software requirements.
  - (2) The process of studying and refining system, hardware, or software requirements.
- Analyze the results of elicitation
  - are the answers consistent?
  - identify trouble spots
  - identify boundaries
  - identify most important requirements
- possibly iterate over elicitation again

[IEEE-Std-610.12-1990]

HOCHSCHULE HAMM-LIPPSTADT

- The analysis has to establish an agreed set of requirements which are **complete**, **consistent**, and **unambiguous**. Such a set can be used as the basis for systems development.
- **Negotiation:** Stakeholders often disagree over requirements. Therefore they need to negotiate to try to reach agreement.

c) Requirement Specification

HOCHSCHULE HAMM-LIPPSTADT

- **Requirement Specification**: the development of a document that clearly and precisely records each of the requirements of the software.

Thayer&Dorfman1997]

HOCHSCHULE HAMM-LIPPSTADT

- Different specification methods have different levels of formality
  - the more formal, the more one can precisely state requirements and then verify the implemented system meets the requirements
  - the more formal, the more one might be able to analyze the requirements for consistency, etc.
  - the more formal, typically the more time is required for specification
    - not all projects want to spend a lot of time and effort in writing requirements precisely
    - particularly if requirements will change often

d) Requirement Validation

HOCHSCHULE HAMM-LIPPSTADT

- **Requirement Validation:** the process of ensuring that the requirement specification is in compliance with the user needs
- system requirements, conforms to document standards, and is an adequate basis for the architectural design.

[Thayer&Dorfman1997]

e) Requirement Management

HOCHSCHULE HAMM-LIPPSTADT

- **Requirement Management**: the planning and controlling of the requirements elicitation, specification, analysis, and verification activities.

[Thayer&Dorfman1997]

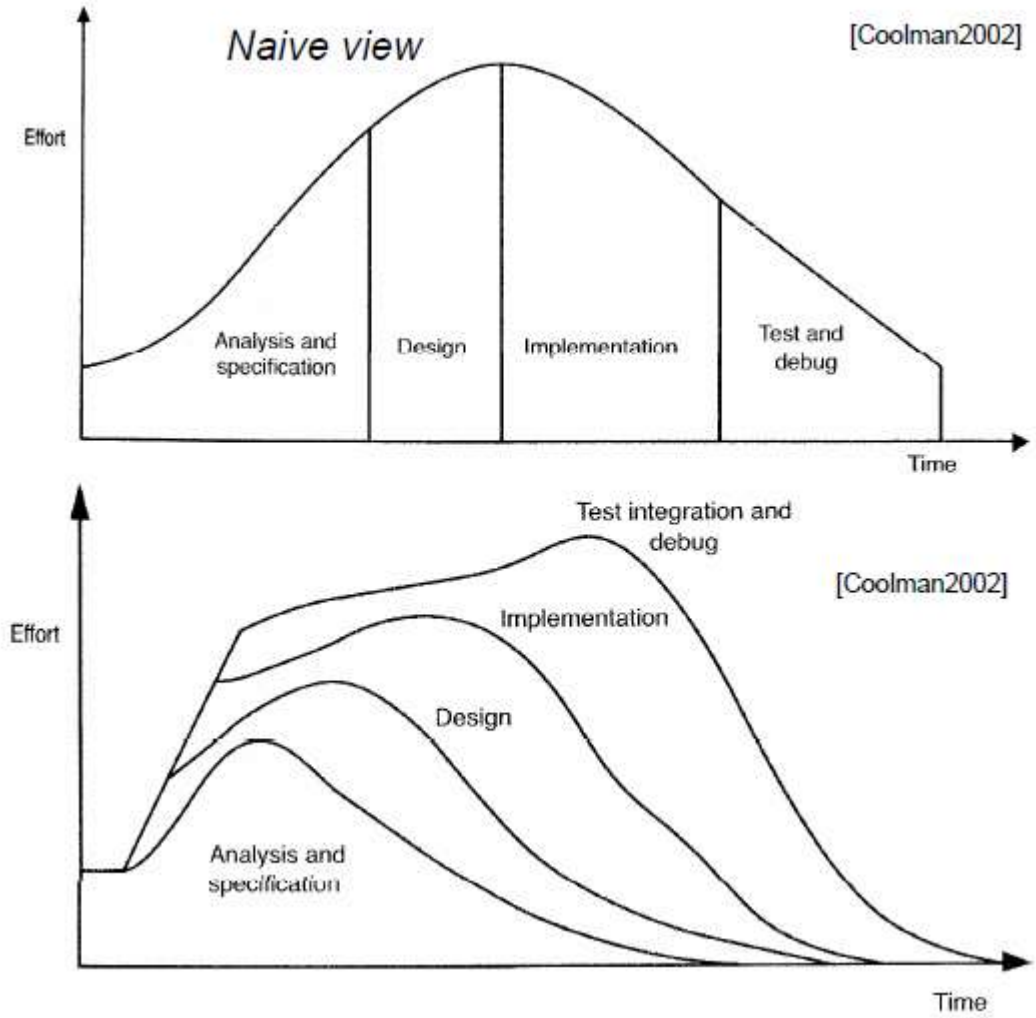
HOCHSCHULE HAMM-LIPPSTADT

- A system of any but the smallest size will be decomposed into a hierarchy of elements (partitioning):
- This is reflected at the requirement level by:
  - **(1) Allocation**: assigning requirements to elements
  - **(2) Flowdown**: requirements which respond to the allocated high level requirements
  - **(3) Traceability**: keep track of the dependencies

[Thayer&Dorfman1997]

▸19 Effort Distribution for Complex Systems

- For complex systems due to allocation and flowdown the Requirement engineering **continues** during the design (and implementation) phase



HOCHSCHULE HAMM-LIPPSTADT

Distinction:

[Thayer&Dorfman1997]

- Design solution: **HOW** to achieve something
- Requirements: **WHAT** to achieve

- Requirements and design solutions are sometimes mixed
  - The customer mandates a design solution as a requirement
    - Design solution (HOW): “provide a database for X”
    - Better: ask Why!
  - A derived requirement which is actually a design solution and not a requirement (depends on the perspective)
    - One person’s design is the next person’s requirements

HOCHSCHULE HAMM-LIPPSTADT

1. Requirements Engineering
2. **Requirement Specification**
3. Approach: SysML
4. Discussion & Summary
5. Bibliography

HOCHSCHULE HAMM-LIPPSTADT

requirements specification.

- A document that specifies the requirements for a system or component. Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards. *Contrast with:* design description. *See also:* functional specification; performance specification.

[IEEE-Std-610.12-1990]

A Good Requirement Specification

HOCHSCHULE HAMM-LIPPSTADT

- Correct
- Unambiguous
- Complete
- Consistent
- Ranked for importance and/or stability
- Verifiable
- Modifiable
- Traceable

[Thayer&Dorfman1997]
[IEEE Std. 830-1993]
[IEEE Std. 830-1998]

▸24 Software Requirements Specification (1/4)

HOCHSCHULE HAMM-LIPPSTADT

Table of Contents

1. Introduction
1.1 Purpose
1.2 Scope
1.3 Definitions, acronyms, and abbreviations
1.4 References
1.5 Overview
2. Overall description
2.1 Product perspective
2.2 Product functions
2.3 User characteristics
2.4 Constraints
2.5 Assumptions and dependencies
3. Specific requirements
3.1 External interface requirements
3.2 Functional requirements
3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements
Appendixes
Index

[IEEE Std. 830-1998]

Software Requirements Specification (2/4)

HOCHSCHULE HAMM-LIPPSTADT

3.1 External interface requirements

3.1.1 User interfaces

3.1.2 Hardware interfaces

3.1.3 Software interfaces

3.1.4 Communications interfaces

- A detailed description of all inputs into and outputs from the software system.
- Complements the interface descriptions in 2.1 (not repeated information)

Content/format:

- Name of item;
- Description of purpose;
- Source of input or destination of output;
- Valid range, accuracy, and/or tolerance;
- Units of measure;
- Timing;
- Relationships to other inputs/outputs;
- Screen formats/organization;
- Window formats/organization;
- Data formats;
- Command formats;
- End messages.

[IEEE Std. 830-1998]

Software Requirements Specification (3/4)

HOCHSCHULE HAMM-LIPPSTADT

3.2 Functional requirements

3.2.1 Mode 1

3.2.1.1 Functional requirement 1.1

…

3.2.m Mode m

3.2.m.1 Functional requirement m.1

…

3.2.m.n Functional requirement m.n

Organized using system mode

- Some systems behave quite differently depending on the mode of operation (training, normal, or emergency)

Alternatives:

- User class
- Objects
- Feature
- Stimulus
- Response
- Functional hierarchy

[IEEE Std. 830-1998]

3.3 Performance requirements static and the dynamic numerical requirements placed on the software or on human interaction with the software
a) The number of terminals to be supported;
b) The number of simultaneous users to be supported;
c) Amount and type of information to be handled.
d) Amount of data to be processed within certain time periods for both normal and peak workload conditions.

- 3.4 Design constraints: This should specify design constraints that can be imposed by other standards, hardware limitations, etc.
- 3.5 Software system attributes
  - Examples
    - Reliability
    - Availability
    - Safety
    - Security
    - Maintainability
    - Portability
- 3.6 Other requirements

[IEEE Std. 830-1998]

HOCHSCHULE HAMM-LIPPSTADT

1. Requirements Engineering
2. Requirement Specification
3. Approach: SysML
4. Discussion & Summary
5. Bibliography

HOCHSCHULE HAMM-LIPPSTADT

System modeling language (SysML):

- defines a modeling language for systems engineering applications
- supports the specification, analysis, design, verification and validation of a range of complex systems
  - Includes the possibility of representing system requirements, non-software components (mechanics, hydraulics, sensors, etc.), physical equations, continuous flows (matter, energy, etc.) and allocations
- intended to assist in integrating systems and software methodologies

Who?

- International Council on Systems Engineering (INCOSE) joint with Object Management Group (OMG) in January 2001

Why Model based?

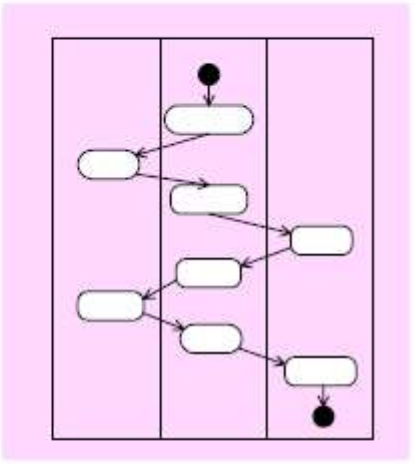
- Improved communications
- Reduced ambiguity
- Reduced errors
- More complete representation
- Enhanced knowledge capture

From Document centric to Model centric

Past

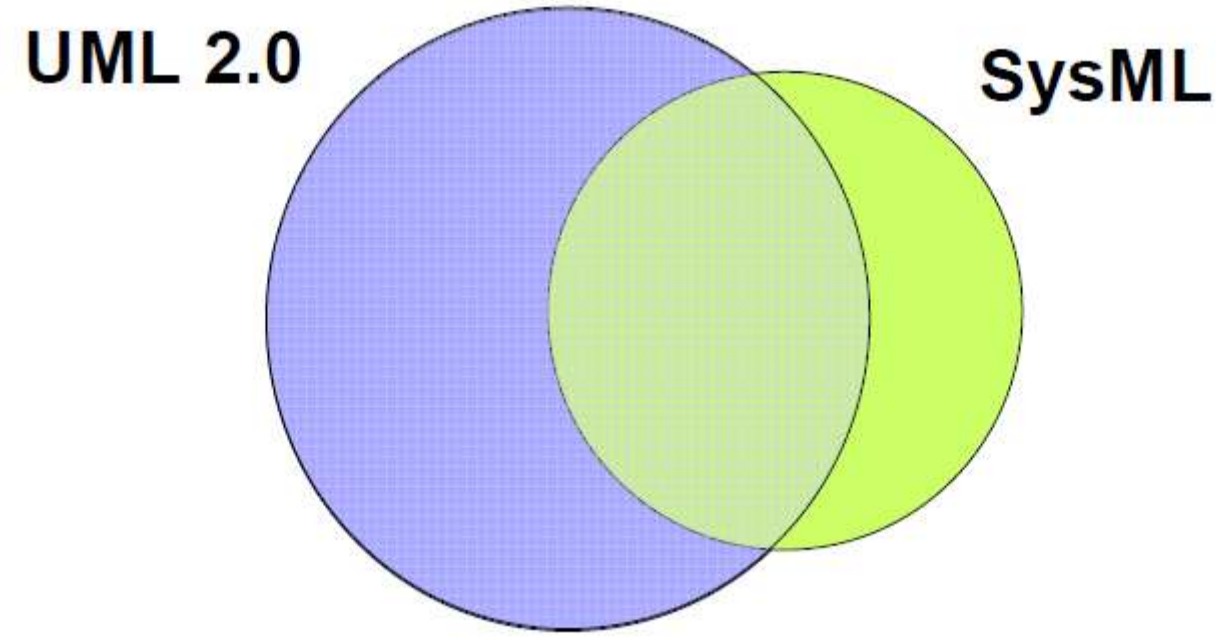
- Specifications
- Interface requirements
- System design
- Analysis & Trade-off
- Test plans

Future



Here: only Requirements!

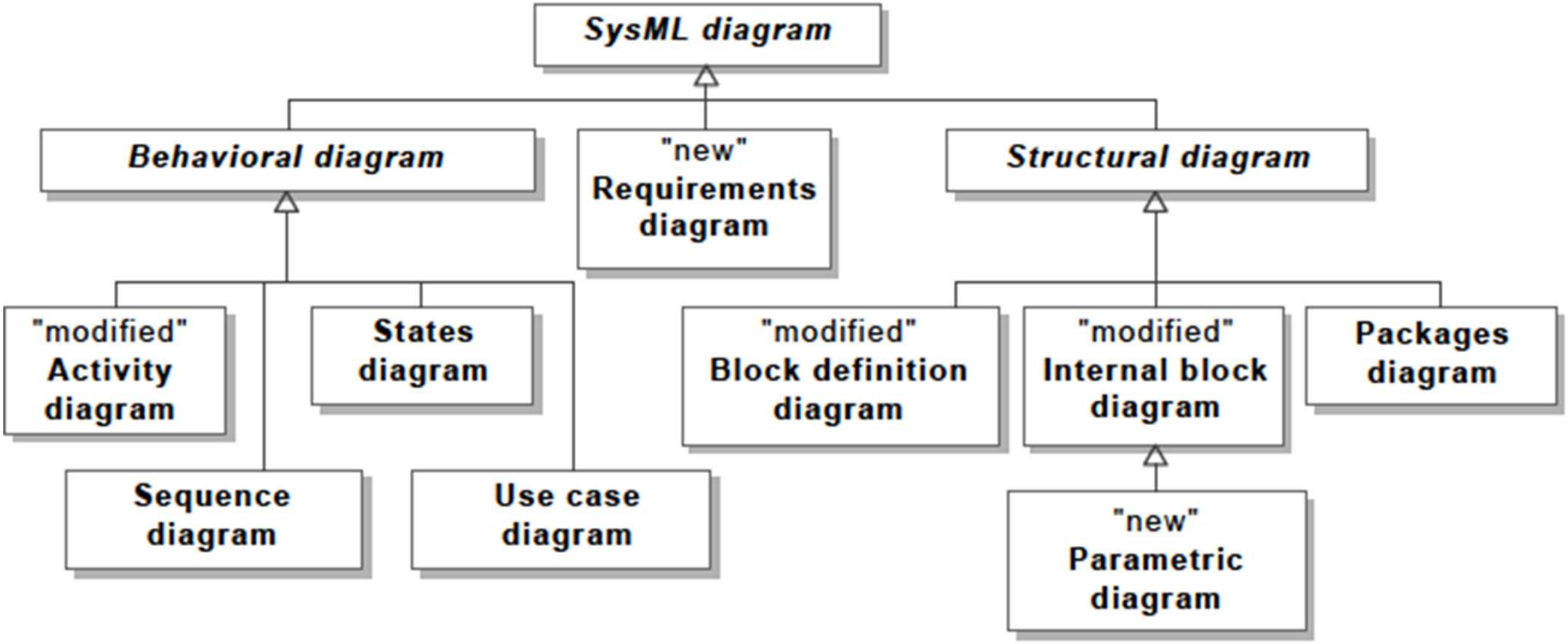
HOCHSCHULE HAMM-LIPPSTADT



Common diagrams: Activities, Block Definitions (UML2::Classes), Internal Blocks (UML2::Composite Structures), Sequences, State Machines, Use Cases

New diagrams: Allocations, Parametric Constraints, Requirements

▸32 SysML Diagrams



- (1)Requirement Diagrams
- (2)Use Case Diagrams
- (3)Scenarios: Sequence/Activity Diagrams

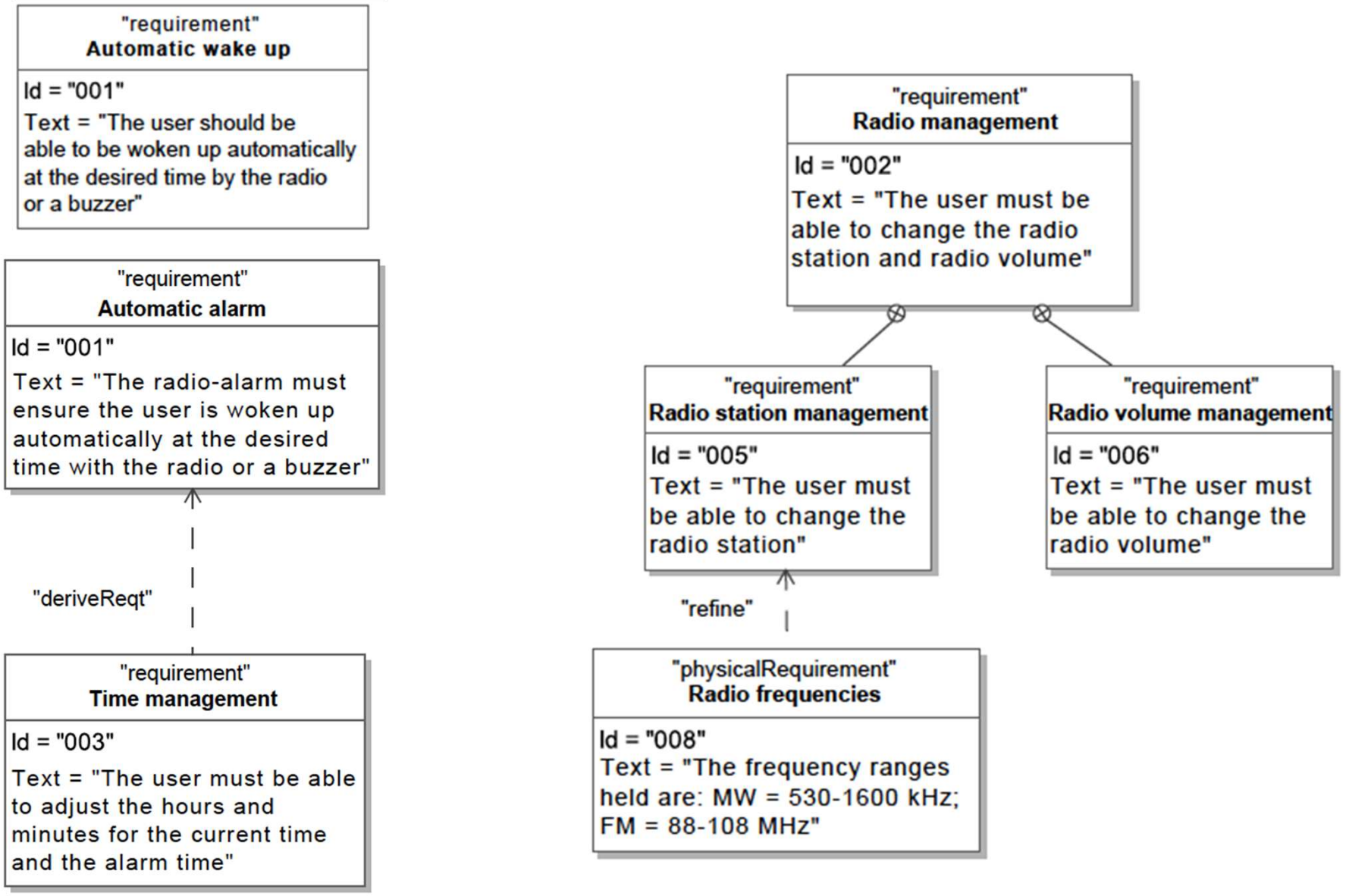
(1) Requirement Diagram

HOCHSCHULE HAMM-LIPPSTADT

- offers a graphical representation of the requirements
- Basic properties of a requirement
  - unique identifier
  - textual description of requirement
- Further properties of a requirement
  - priority (e.g. high, medium or low)
  - source (e.g. customer, marketing, technique and legislation)
  - risk (e.g. high, medium or low)
  - status (e.g. suggested, validated, implemented, tested and delivered)
  - verification method (e.g. analysis, demonstration and test)

▸34 Requirement Diagram

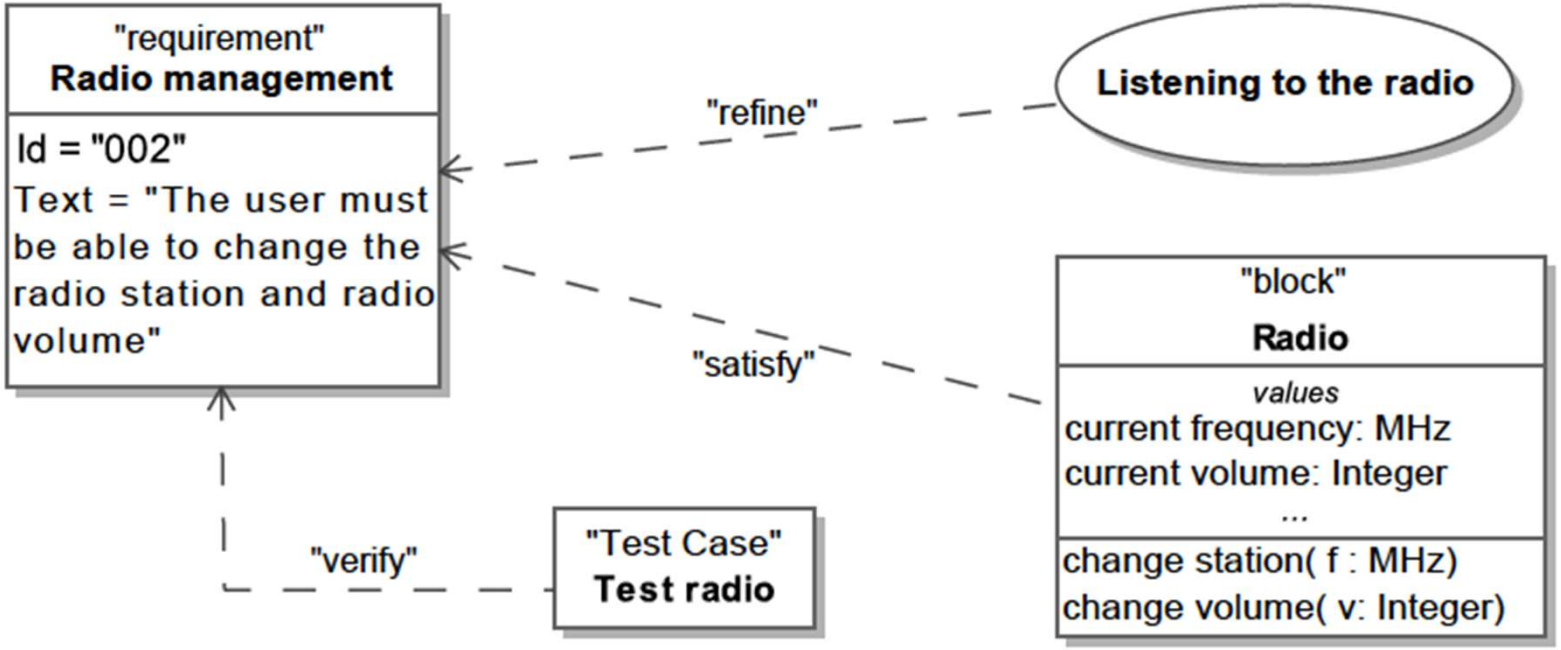
Example

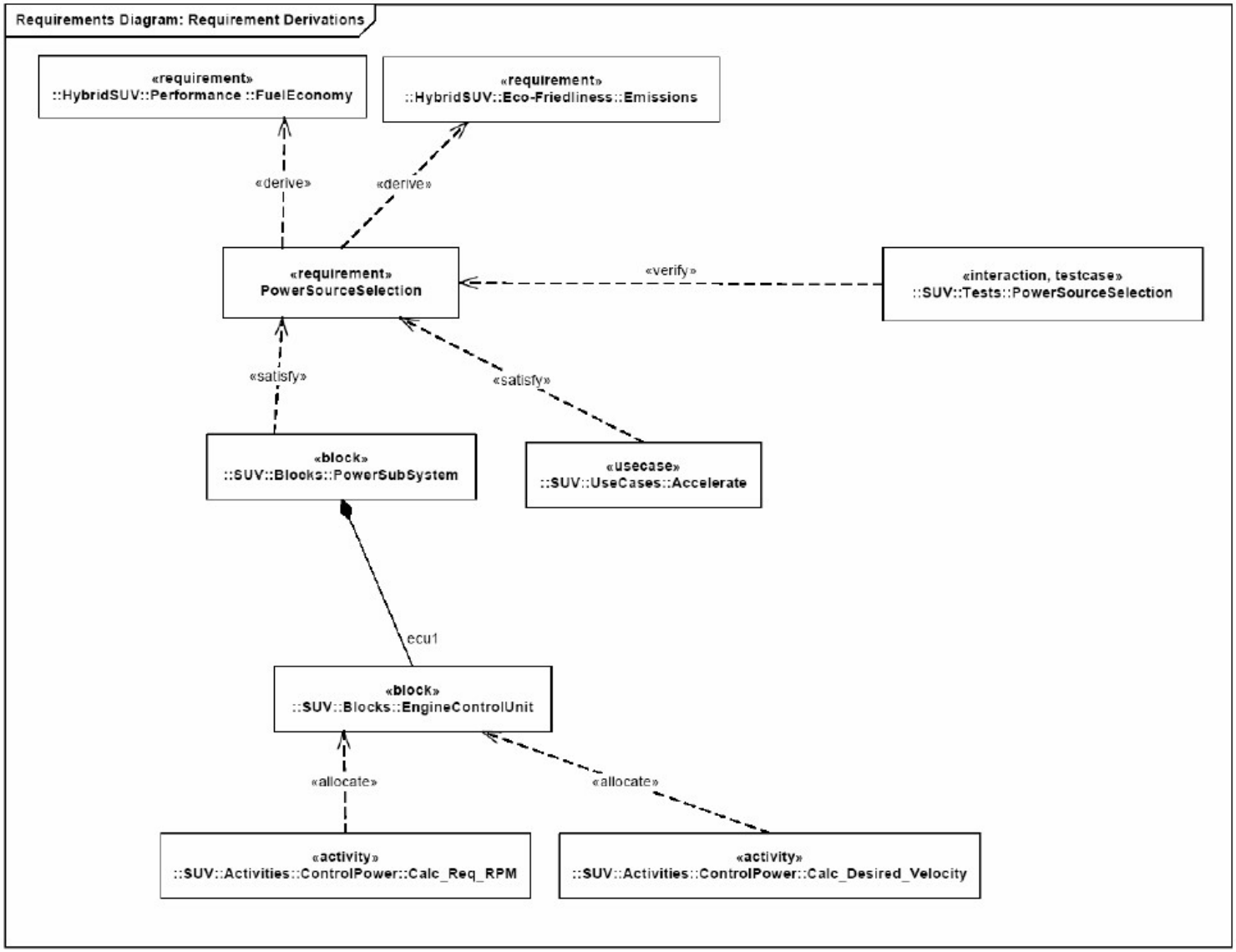


Relations – Requirement diagram specific

- Containment(): makes it possible to deconstruct a composite requirement into several single requirements, which are then easier to trace with regard to architecture and tests.
- Refinement (refine): consists of adding precision: for example, quantitative data.
- Derivation (deriveReqt): consists of connecting requirements from different levels: for example, connecting system requirements to requirements at the subsystem level. This generally involves making some choices in terms of architecture.

- The relation between a requirement and a behavioral element (use case, state diagram, etc.) is denoted by the keyword ***refine***.
- The relation between a requirement and an architecture block is denoted by the keyword ***satisfy***.
- The relation between a requirement and a test case is denoted by the keyword ***verify***.




Requirements Diagram: Requirement Derivations
«requirement»
::HybridSUV::Performance ::FuelEconomy
«requirement»
::HybridSUV::Eco-Friedliness::Emissions
«derive»
«derive»
«requirement»
PowerSourceSelection
«verify»
«interaction, testcase»
::SUV::Tests::PowerSourceSelection
«satisfy»
«satisfy»
«block»
::SUV::Blocks::PowerSubSystem
«usecase»
::SUV::UseCases::Accelerate
ecu1
«block»
::SUV::Blocks::EngineControlUnit
«allocate»
«allocate»
«activity»
::SUV::Activities::ControlPower::Calc_Req_RPM
«activity»
::SUV::Activities::ControlPower::Calc_Desired_Velocity

HOCHSCHULE HAMM-LIPPSTADT

- Requirements is a **crucial element** of developing systems as about 80% of the software errors are located in the requirements.
- The **system modeling language (SysML)** provides a modeling language for systems engineering applications that supports the specification and analysis of requirements and integrates systems and software methodologies.

HOCHSCHULE HAMM-LIPPSTADT

1. Requirements Engineering
2. Requirement Specification
3. Approach: SysML
4. Discussion & Summary
5. Bibliography

III.7 Bibliography

HOCHSCHULE HAMM-LIPPSTADT

[Anton+1998] A. Anton et al. The Use of Goals to Surface Requirements for Evolving Systems, International Conference on Software Engineering, pp157-166, IEEE Comp. Soc. Press, 1998

[Cooling2002] Jim Cooling, Software Engineering for Real-time Systems. Addison Wesley, November 2002
ISBN: 0201596202

[IEEE-Std-830-1993] Standards Coordinating Committee of the IEEE Computer Society, The Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017-2394, USA, IEEE Recommended Practice for Software Requirements Specifications, IEEE Std 830-1993.

[IEEE-Std-830-1998] Standards Coordinating Committee of the IEEE Computer Society, The Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017-2394, USA, IEEE Recommended Practice for Software Requirements Specifications, IEEE Std 830-1998.

[IEEE-Std-1471-2000] Standards Coordinating Committee of the IEEE Computer Society, The Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017-2394, USA, Recommended Practice for Architectural Description of Software-Intensive Systems, IEEE-Std-1471-2000.

Bibliography

[Kotonya&Sommerville1998] G. Kotonya and I. Sommerville, Requirements Engineering. John Wiley,1998. Systems Requirements Engineering, P. Loucopoulos et al., Mc Graw-Hill, 1995

[Lamsweerde2001] A. van Lamsweerde, Goal-Oriented Requirements Engineering: A Guided Tour, 5th International Symposium on Requirements Engineering, IEEE Computer Society Press, 2001

[Loucopoulos&Karakostas1995] Loucopoulos, P., and Karakostas, V., System Requirements Engineering, McGraw-Hill,1995

[Sommerville&Sawyer1997] Sommerville, I., and Sawyer, P., Requirements Engineering, John Wiley,1997

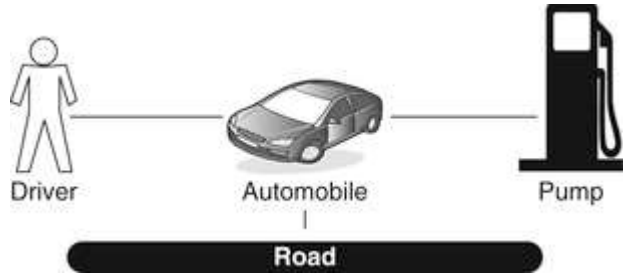
[Thayer&Dorfman1997] Software requirements engineering Thayer, Richard H. And Dorfman, Merlin [Hrsg.]:. - Los Alamitos, Calif. [u.a.] : IEEE Computer Soc. Press , 1997 .
ISBN: 0-8186-7738-4

HOCHSCHULE HAMM-LIPPSTADT

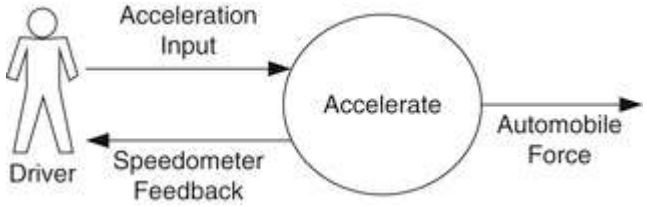
- [1] S. Friedenthal, A. Moore, und R. Steiner, *A practical guide to SysML: the systems modeling language*. Waltham, MA: Morgan Kaufmann, 2012.
  - Chapter 4

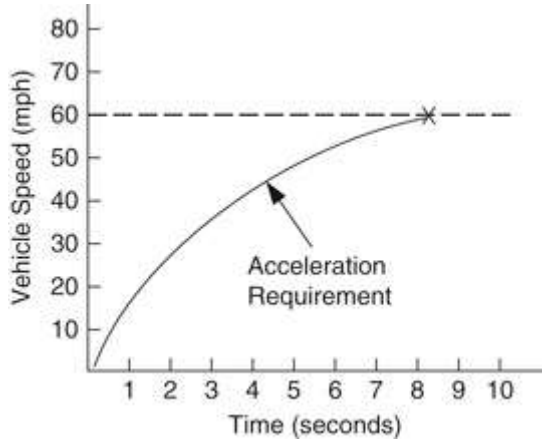
HOCHSCHULE HAMM-LIPPSTADT

- Boundary

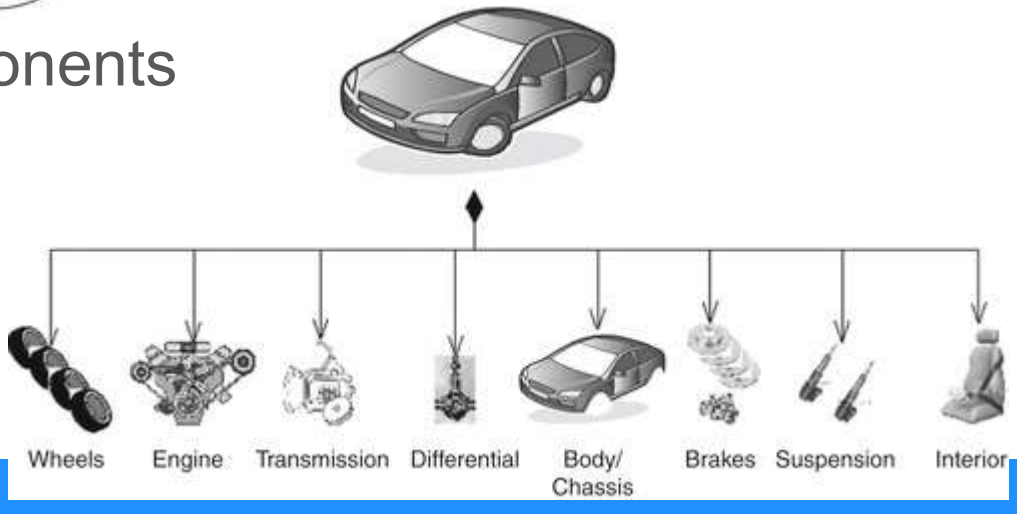


- Functional and performance requirements



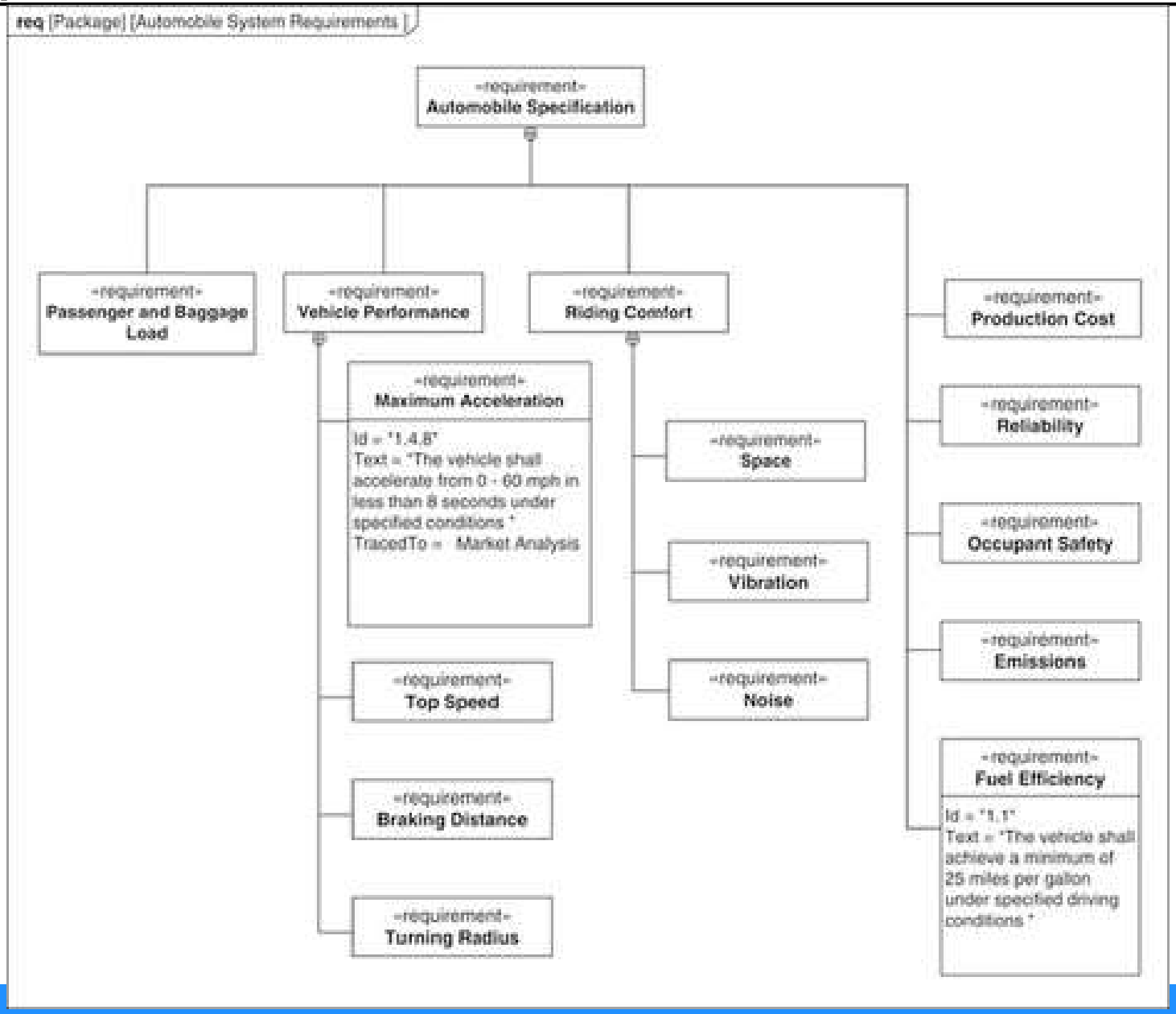


- System components



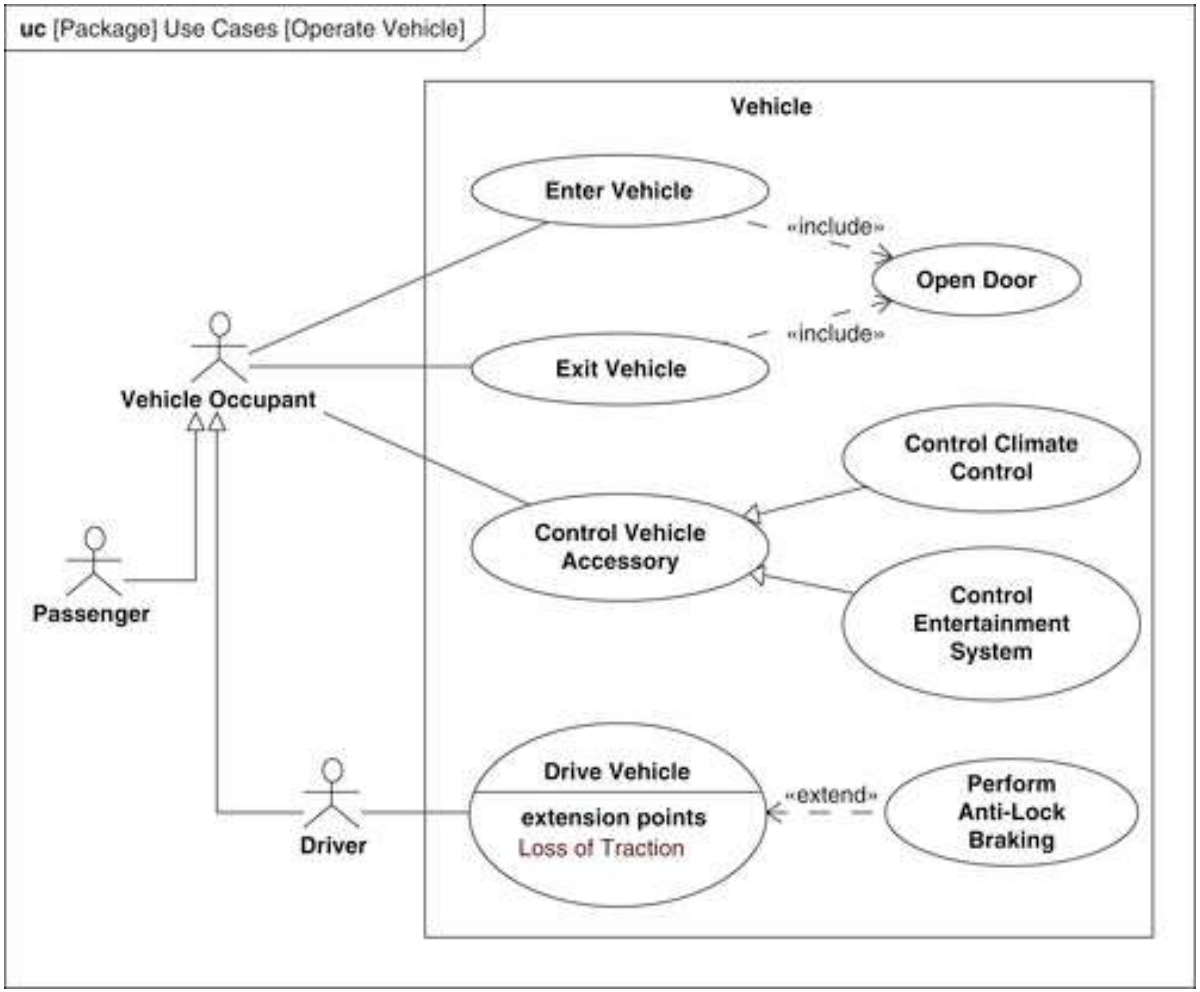
Automotive Case Study

Requirements



HOCHSCHULE HAMM-LIPPSTADT

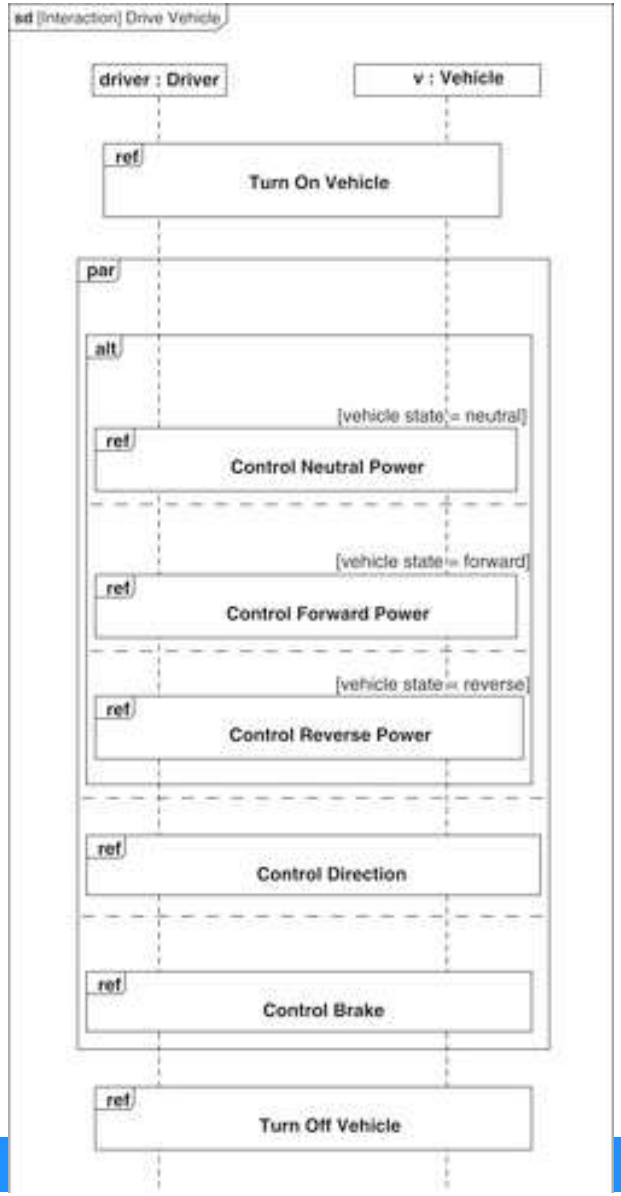
▸ Functional Requirements



Sequence Diagram

HOCHSCHULE HAMM-LIPPSTADT

- Interaction between driver and vehicle



Practical work

- Consider your Microcontroller project from last semester
  - Describe the requirements with the help of SysML Requirements
    - Use the reference possibilities of the SysML Requirement Diagram
  - If you have the possiblity install the SysML tool papyros
    - https://www.eclipse.org/papyrus/
    - Specify the requirements in papyrus
- Readings
  - Tim Weilkiens, “Systems Engineering with SysML/UML” (see: https://learning.oreilly.com/library/view/systems-engineering-with/9780123742742/)
    - Chapter 3. UML—Unified Modeling Language
    - 4.1 TO 4.3. THE REQUIREMENT DIAGRAM
    - 4.5. BLOCK DIAGRAMS