# Requirement Engineering

## Lecture 8: Requirements Documentation
### Model-based Requirements Documentation

Prof. Dr. Benjamin Leiding

M.Sc. Anant Sujatanagarjuna

# Reminder

## Bonus Task

- Will be published on:  21.12.2022 - 4:00 PM

- Submission Deadline: 25.01.2023 – 1:59 PM

- Submission Location: Moodle

# General Requirements Engineering Process

## Overview

| Requirements Engineering | | | | | |
|---|---|---|---|---|---|
| Requirements Analysis | | | | Requirements Management | |
| Elicitation | Negotiation | Documentation | Validation | Change Management | Tracing |

# MODEL-BASED REQUIREMENTS DOCUMENTATION TECHNIQUES

**Lecture 8: Requirements Documentation**

**Content**

1. Model-based Requirements Documentation Techniques
   1. **Models in General**
   2. Goal Models
   3. Use Cases
   4. Data | Functional | Behavioral Perspective
   5. Agent-oriented Modelling

# Model-based Requirements Documentation Techniques

## Models in General – Requirements Model vs. Design Model

- Models are frequently used for system design
  - "Design Models"
  - E.g., architectural models

- Considerable difference between requirements models and design models
  - Requirements models depict aspects of the underlying problem
  - Design models document solutions chosen during system development

# Model-based Requirements Documentation Techniques

## Models in General – The Term "Model"

- According to Merriam-Webster:
  - *Structural design*
  - *A usually miniature representation of something*
  - *A system of postulates, data, and inferences presented as a mathematical description of an entity or state of affairs*

- We use the following definition in this lecture:
  - *A model is an abstract representation of an existing reality or a reality to be created.*

# Model-based Requirements Documentation Techniques

## Models in General – Properties of Models

- Mapping of reality
  - Aspects of the observed reality are mapped onto model elements
  - Descriptive model creation → Model documents the existing reality
  - Prescriptive model creation → Model prototypes fictious reality
  - Models can be both descriptive and prescriptive at the same time
    - Describes a stakeholder
    - Prescribes a use case of a system

# Model-based Requirements Documentation Techniques

## Models in General – Properties of Models

- Reduction of Reality
  - Models do not capture the complete reality
  - Instead, the models reduce the captured reality
  - Only particular aspects of the system are modeled
  - Subject matter is summarized during compression

- Pragmatic Property
  - Models serve a special purpose
  - Models are within a special context
    - **NOT general purpose!**
  - Purpose affects the construction of models and the reduction of the reality
  - Ideally contains only information pertaining to its purpose

# Model-based Requirements Documentation Techniques

## Models in General – Properties of Models

- Defined through **syntax** and **semantics**

- <u>Syntax</u>
  - Defines the modeling elements to be used
  - Specifies their valid combinations

- <u>Semantics</u>
  - Defines the meaning of the individual model elements
  - Foundation for the interpretation of the models

- Can be **formal**, **informal**, and **semiformal**
  - Depends on the magnitude of formal definitions

# Model-based Requirements Documentation Techniques

## Models in General – Advantages of Models

- Humans handle graphically depicted information better
  - Perceived faster
  - Memorized faster
  - Also true for requirements models

- Strictly defined focus
  - Everything not part of the focus of the model is removed → Removal of noise

- Harmonized level of abstraction
  - Modeling elements dictate the level of abstraction

# Model-based Requirements Documentation Techniques

## Models in General – Suppression of Details

- Complexity is reduced by abstraction

- Three main mechanisms
  - Selection
    - Selects a particular aspect to be depicted by the model
    - Other aspects are ignored completely, i.e., not part of the model
  - Aggregation
    - Combines aspects into aggregated aspects
    - Condenses information
  - Classification/generalization
    - Identifies common features
    - Suppresses differences between the common features
    - Commonalities are represented as generalized information

# Model-based Requirements Documentation Techniques

## Models in General – UML

- Object Management Group (OMG) standard
  - Current version UML 2.5.1

- Graphical notation for the analysis, design, and documentation of object-oriented systems

- UML is **not**
  - a development process
  - specialized for a certain topic
  - complete & formal
    - Cannot be complied without additional information
    - → Semiformal
  - Capable of semantics
    - UML only provides a syntax
    - Semantics depend on the reader of the document

# Model-based Requirements Documentation Techniques

## Goal Models – Goals in General

- Goals are the stakeholders description of system properties
  - What they want from the system


- Effort for goal considerations usually minimal


- Positive impact of goal modeling is high
  - Especially concerning the comprehensiveness and quality

**Lecture 8: Requirements Documentation**

**Content**

1. Model-based Requirements Documentation Techniques
    1. Models in General
    **2. Goal Models**
    3. Use Cases
    4. Data | Functional | Behavioral Perspective
    5. Agent-oriented Modelling

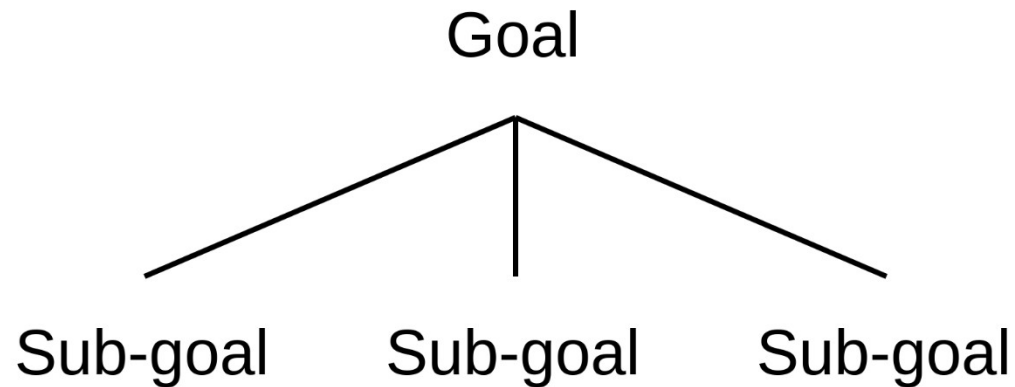**Model-based Requirements Documentation Techniques**

**Goal Models – AND / OR Trees**

- Documents hierarchical decompositions of goals into sub-goals

- Two types of decompositions
  - AND → All sub-goals must be fulfilled
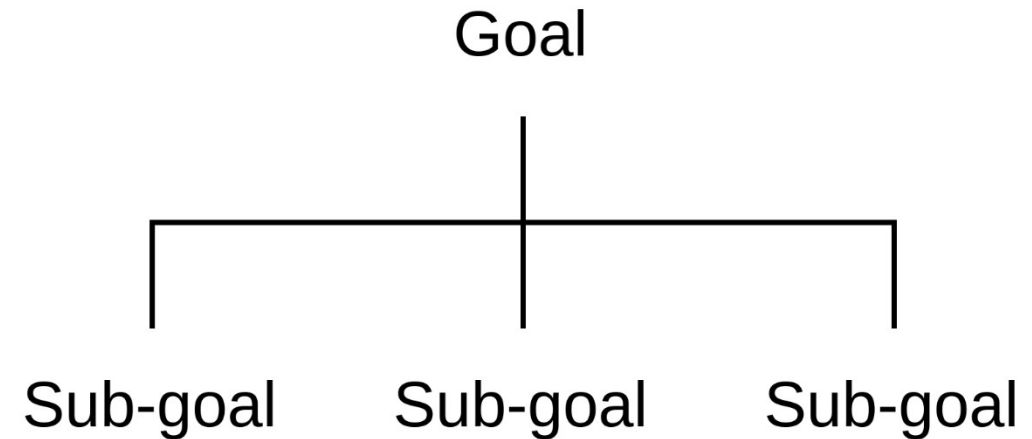  - OR → At least one sub-goal must be fulfilled

# Model-based Requirements Documentation Techniques

## Goal Models – AND / OR Trees



**OR-decomposition**

Goal

Sub-goal    Sub-goal    Sub-goal

**AND-decomposition**

Goal

Sub-goal    Sub-goal    Sub-goal

**Lecture 8: Requirements Documentation**

**Content**

1. Model-based Requirements Documentation Techniques
   1. Models in General
   2. Goal Models
   3. **Use Cases**
   4. Data | Functional | Behavioral Perspective
   5. Agent-oriented Modelling

# Model-based Requirements Documentation Techniques

## Use Cases – Overview

- Method to document functionalities
  - Planned
  - Of existing system


- Relatively simple models


- Two concepts
  - Use case diagrams
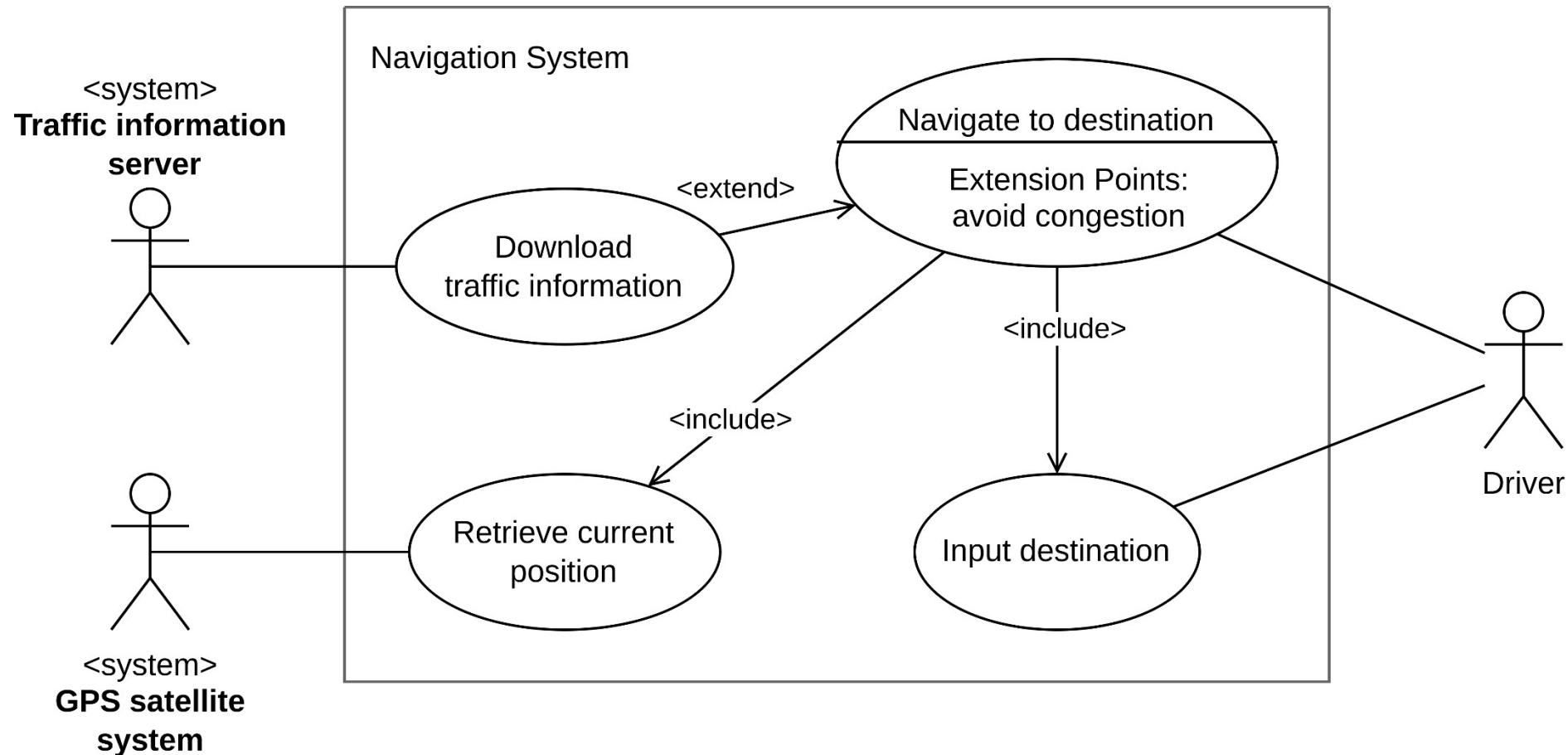  - Use case specification
- Both should be used in conjunction

# Model-based Requirements Documentation Techniques

## Use Cases – UML Use Case Diagrams

- Models to schematically depict:
  - Functions from a user's point of view
  - Interrelations of functions of a system
  - Relations between functions and their environment

- **We do not cover all concepts of use case diagrams in this lecture**
  - Additional information can be found in the literature

# Model-based Requirements Documentation Techniques

## Use Cases – UML Use Case Diagram (Example)

# Model-based Requirements Documentation Techniques

## Use Cases – Issues of UML Use Case Diagrams

- Diagrams do not contain details
  - Very high level
  - Very abstract

- Examples for open questions
  - How does the driver communicate with the Navigate to destination use case?
  - Is there an order in the inclusion of the use cases Retrieve current location and Input destination?

# Model-based Requirements Documentation Techniques

## Use Cases – Use Case Specifications

- Use case specifications provide details to the diagrams

- Specifications documented textually

- Not simple prose, but in form of templates (usually tabular)

- The template defines the concrete information contained in the use case specification

# Model-based Requirements Documentation Techniques

## Use Cases – Use Case Specification Template

- Template prescribes the following information
  - Attributes for unique identification of use cases
  - Management attributes
  - Attributes for the description of the use case
  - Specific use case attributes, e.g.,
    - the trigger event,
    - actors,
    - pre- and post-conditions,
    - the result of the use case,
    - the main scenario,
    - alternative and exception scenarios,
    - cross references,
    - quality requirements

# Model-based Requirements Documentation Techniques

## Use Cases – Use Case Specification Template (Example)

| Section | Content |
|---|---|
| Designation | UC-12-37 |
| Name | Navigate to destination |
| Authors | John Smith, Sandra Miller |
| Priority | Importance for system success : high Technological risk : high |
| Criticality | High |
| Source | C. Warner (domain expert for navigation systems |
| Person Responsible | J. Smith |
| Description | The driver of the vehicle types the name of the destination. The navigation system guides the drive to the desired destination. |
| Trigger event | The driver wishes to navigate to his destination |
| Actors | Driver, traffic information system, GPS satellite system |

# Model-based Requirements Documentation Techniques

## Use Cases – Use Case Specification Template (Example)

| Section | Content |
|---|---|
| Pre-conditions | The navigation system is activated |
| Post-conditions | The driver has reached his destination |
| Result | Route guidance |
| Main scenario | 1. The navigation system asks for the desired destination<br>2. The driver enters the desired destination<br>3. The navigation system pinpoints the destination in its maps<br>4. On the basis of the current position and the desired destination, the navigation system calculates a suitable route<br>5. The navigation system compiles a list of waypoints<br>6. The navigation system shows a map of the current position and shows the route to the next waypoint<br>7. When the last waypoint is reached, the navigation system shows "destination reached" on the screen |

# Model-based Requirements Documentation Techniques

## Use Cases – Use Case Specification Template (Example)

| Section | Content |
|---------|---------|
| Alternative scenario | 4a.   Calculation of the route must honor traffic information and avoid traffic congestions.<br>4a1. The navigation system queries the server for updated traffic information.<br>4a2. The navigation system calculates a route that does not contain any traffic congestions. |
| Exception scenarios | Trigger event: The navigation system does not receive GPS signal from the GPS satellite system. |
| Qualities | → QR.04 (reaction time upon user input)<br>→ QR.15 (operating comfort) |

Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level

Requirements Engineering – TU Clausthal

28

**Lecture 8: Requirements Documentation**

**Content**

1. Model-based Requirements Documentation Techniques
   1. Models in General
   2. Goal Models
   3. Use Cases
   4. **Data | Functional | Behavioral Perspective**
   5. Agent-oriented Modelling

# Model-based Requirements Documentation Techniques

## Modelling Requirements in the Three Perspectives

- Different perspective → Different models


- **Data perspective**
  - Entity-relationship diagrams
  - UML class diagrams
- **Functional perspective**
  - Data flow diagrams
  - UML activity diagrams
- **Behavioral perspective**
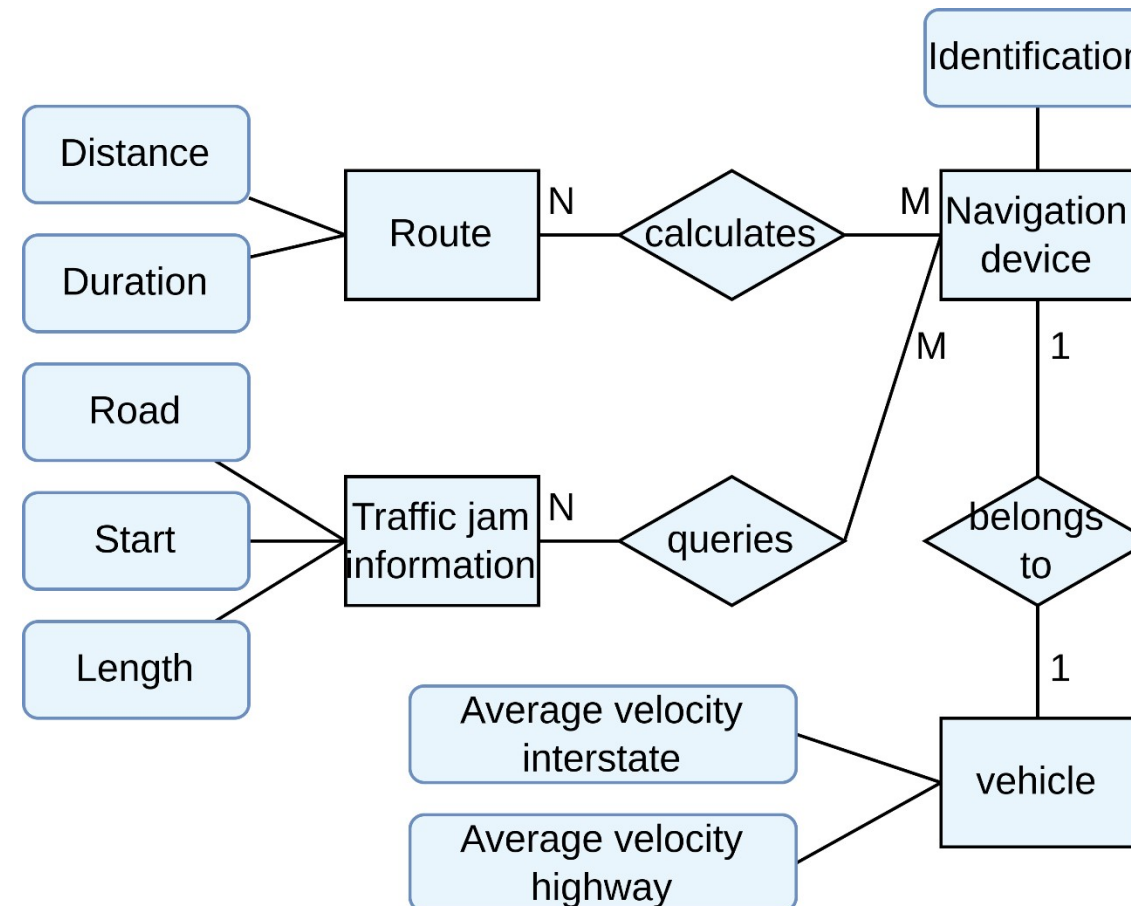  - Statecharts
  - UML state machine diagrams

# Model-based Requirements Documentation Techniques

## Data Perspective – Entity-relationship Diagrams

- Concept from the world of databases

- Used to model data (entities) and their relationships

- Extensions of entity-relationship diagrams developed over the years
  - Min/max notations for cardinalities
  - Inheritance mechanism
  - ...

- (Extensions out of scope in this lecture)

# Model-based Requirements Documentation Techniques

## Data Perspective – Entity-relationship Diagrams (Example)

Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level

Requirements Engineering – TU Clausthal

32

**Model-based Requirements Documentation Techniques**

**Data Perspective – UML Class Diagrams**

- Consists of classes and their associations

- In principle, similar to entity-relationship diagrams
  - Classes ~ entity types
  - Associations ~ relation types

- Class diagrams more powerful than entity-relationship diagrams

# Model-based Requirements Documentation Techniques

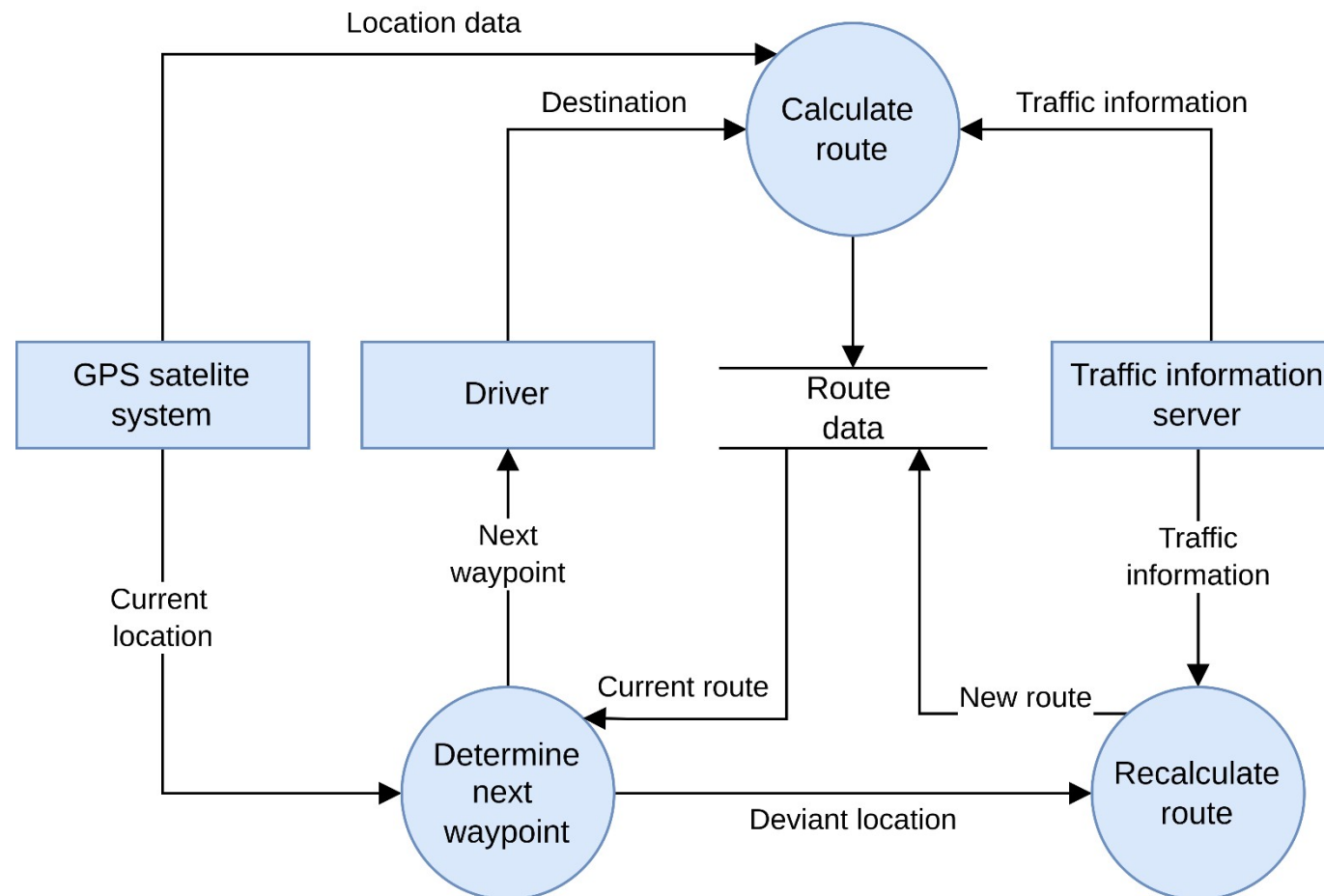## Data Perspective – UML Class Diagrams (Example)
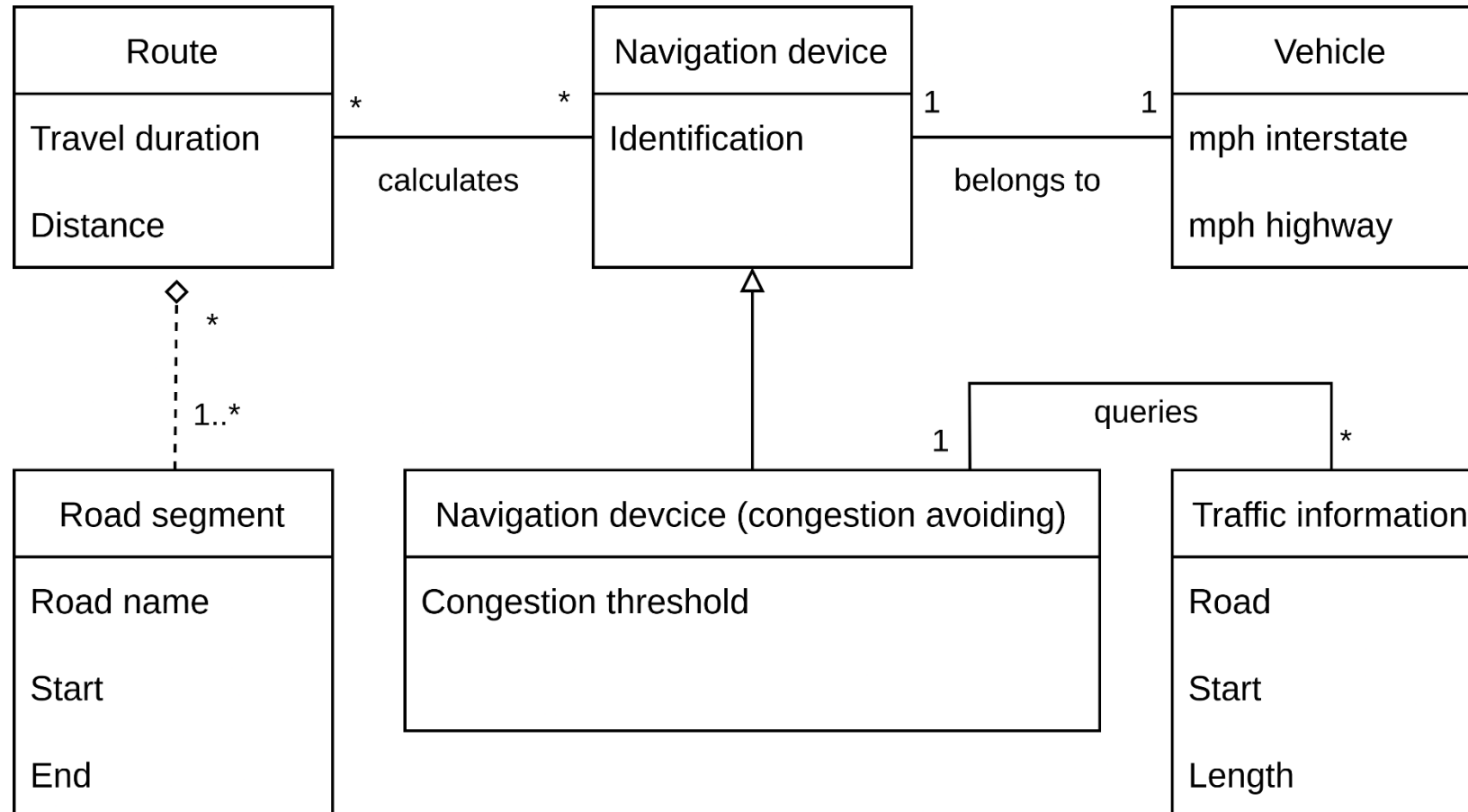
# Model-based Requirements Documentation Techniques

## Functional Perspective – Data Flow Diagrams

- Model the flow of the data through the system
  - Input/Output data
  - Recipients of the data

- Can be applied on different levels of abstraction
  - Requirements on different levels of abstraction possible

# Model-based Requirements Documentation Techniques

## Functional Perspective – Data Flow Diagrams (Example)

Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level

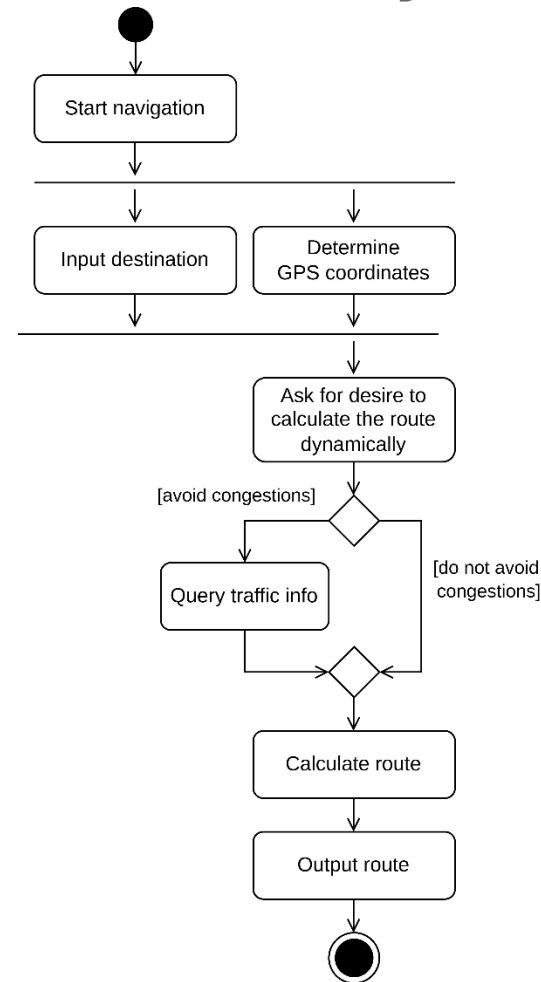Requirements Engineering – TU Clausthal

36

# Model-based Requirements Documentation Techniques

## Functional Perspective – UML Activity Diagrams

- Method to model action sequences

- Depict the control flow between activities and actions

- Can include the data flow (optional!)

# Model-based Requirements Documentation Techniques

## Functional Perspective – UML Activity Diagrams (Example)

Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level
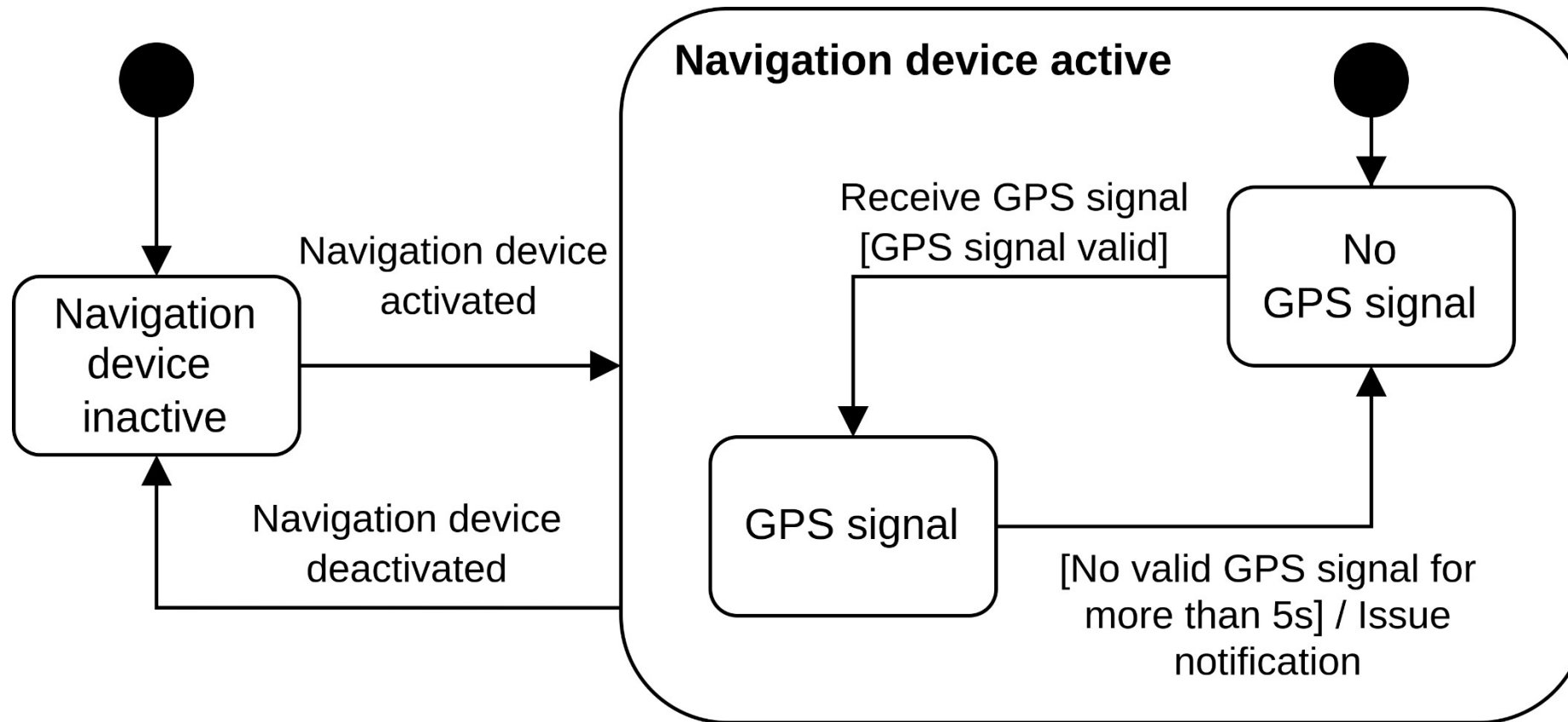
Requirements Engineering – TU Clausthal

38

# Model-based Requirements Documentation Techniques

## Behavioural Perspective – Statecharts

- Extension of finite automata

- Support hierarchization of states

- Allow concurrent behavior

# Model-based Requirements Documentation Techniques

## Behavioural Perspective – Statechart (Example)

Based on K. Pohl, C. Rupp (2011) – Requirements Engineering Fundamentals: A study guide for Requirements Engineering Foundation Level

Requirements Engineering – TU Clausthal

40

**Lecture 8: Requirements Documentation**

**Content**

1. Model-based Requirements Documentation Techniques
    1. Models in General
    2. Goal Models
    3. Use Cases
    4. Data | Functional | Behavioral Perspective
    5. **Agent-oriented Modelling**

# Agent-oriented Modelling

## Concepts and Definitions

- <u>Goal:</u> A situation description that refers to the intended state of the environment. Goals can:

  - be *functional* or *non-functional (quality).*

  - have sub-goals.

- "Goals are expressed by using nouns, verbs, and (optionally) adjectives. The nouns tend to be more of a state, and the verbs more into the activities that are needed to achieve a goal."

- e.g., if a *message* needs to be *transmitted securely*, the functional goal '*Transmit Message*' can be associated with the quality goal '*Securely*'

L. Sterling, K. Taveter (2009) – The Art of Agent-Oriented Modeling

# Agent-oriented Modelling

## Motivation: Why AOM?

- Most (if not all) processes in software systems are elicited by *Agents* playing a certain *Role* in the system, to achieve some *Goal.*

- AOM is a tool for *"modelling systems with multiple agents, both human and manmade, interacting with a diverse collection of hardware and software in a complex environment"*

- AOM models are clear and easily understandable for stakeholders → <u>useful for Requirements Engineering</u>

# Agent-oriented Modelling

## Goal vs. Requirement

| Goal | Requirement |
|---|---|
| Single desired result | Statement of need |
| One goal may consist of several requirements | One requirement may be related to many goals |

- No one to one mapping between goals and requirements is possible

# Agent-oriented Modelling

## Concepts and Definitions

- How does one identify functional and non-functional goals?

  - Functional goals usually describe **what** a system must accomplish = Identification depends heavily on the system.

  - Non-functional goals describe **how** the system must accomplish those goals, in terms of standards and quality = Identification can depend on functional goals.

  - **However**, there are many commonalities: Reliability, Availability, Security, …..

# Agent-oriented Modelling

## Concepts and Definitions

- Role: Some capacity or position that fascilitates the system to achieve it's goals. Roles express functions, expectations, and obligations of the agents enacting them.

  - eg. Network Administrator, Firewall

- Agent: An entity that can act in the environment, perceive events, and reason.

  - Can be human or software

L. Sterling, K. Taveter (2009) – The Art of Agent-Oriented Modeling

**Agent-oriented Modelling**

**Concepts and Definitions**

- <u>Activity</u>**:** Some action performed by an agent playing a role in pursuance of a system goal.

- <u>Environment:</u> An abstraction that provides the surrounding conditions for agents to exist and that mediates both the interaction among agents and the access to resources.
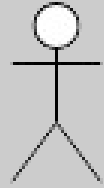
**Agent-oriented Modelling**

**Models**

Models that we will take a look at:

- Goal Models
- Behavioural Interface Models
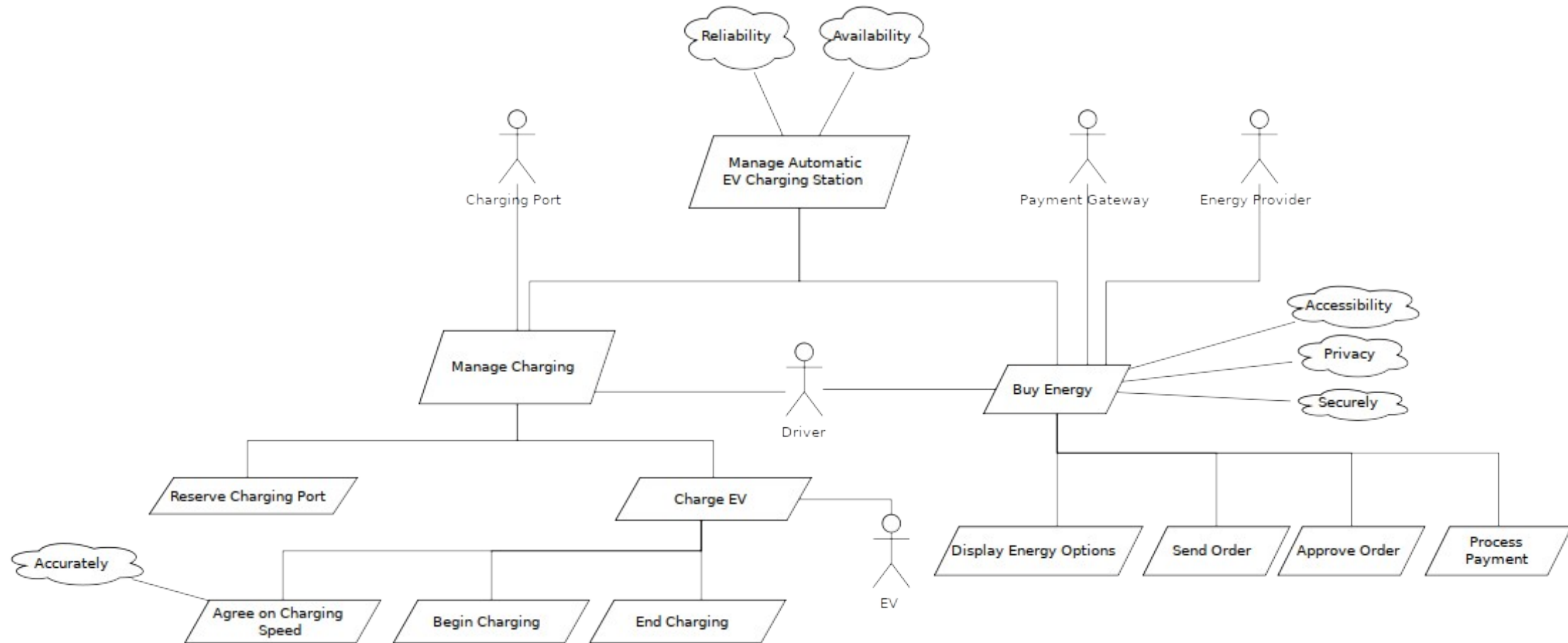
# Agent-oriented Modelling

## Goal Models

- Goal models hierarchically express the relationships between goals (functional and non-functional) and the roles played by various agents in pursuit of those goals.

- Sterling and Taveter's AOM Goal models omit AND/OR decomposition for simplicity.

| Symbol | Meaning |
|---|---|
|  | Goal |
|  | Quality Goal |
|  | Role |
|  | Reltionship between goals |
|  | Relationship between goals and quality goals |

# Agent-oriented Modelling

## Goal Model Example: Automated EV Charging Station

# Agent-oriented Modelling

## Behavioural Interface Models (BIM)

- Behavioral Interface Models model the behaviour of agents playing their roles
  - Models *Behavioural Units (= Activities)*
  - Represented as a table ↓

| Activity | Trigger(s) | Precondition(s) | Postcondition(s) |
|---|---|---|---|
| *Activity Name* | *Event(s) that trigger(s) the activity* | *Conditions for Activity to proceed* | *Conditions for Activity to be considered complete* |
| … | … | … | … |

## Agent-oriented Modelling

## BIM Example: Automated EV Charging Station (Manage Charging)

| Activity | Trigger(s) | Precondition(s) | Postcondition(s) |
|---|---|---|---|
| Reserve Charging Port(CP) | Driver wants to charge EV | Driver has bought energy, CP is free, EV is ready to charge | Driver has reserved CP |
| Charge EV | " | Driver has reserved CP | Driver has charged EV |
| Agree on charging speed | " | Driver has reserved CP, Max CP speed ≥ Min EV speed, Max EV speed ≥ Min CP speed | Charging speed is agreed upon |
| Begin Charging | " | Charging speed is agreed upon | EV has begun charging |
| End Charging | None | EV has competed charging | Driver has charged EV, CP is free |

# SUMMARY

## Summary

- Conceptual models as a means for requirements documentation
  - Abstraction and good overview vs. learning a modeling language
  - Different models for different purposes → Model needs to fit the purpose
- UML provides models for almost anything
  - We only covered a small part → Other UML models can also be useful for requirements documentation
  - UML is not the only answer → Other models work fine, too.

# Questions?