课程：软件工程方法与实践

Requirements Engineering

—Requirements Artefacts

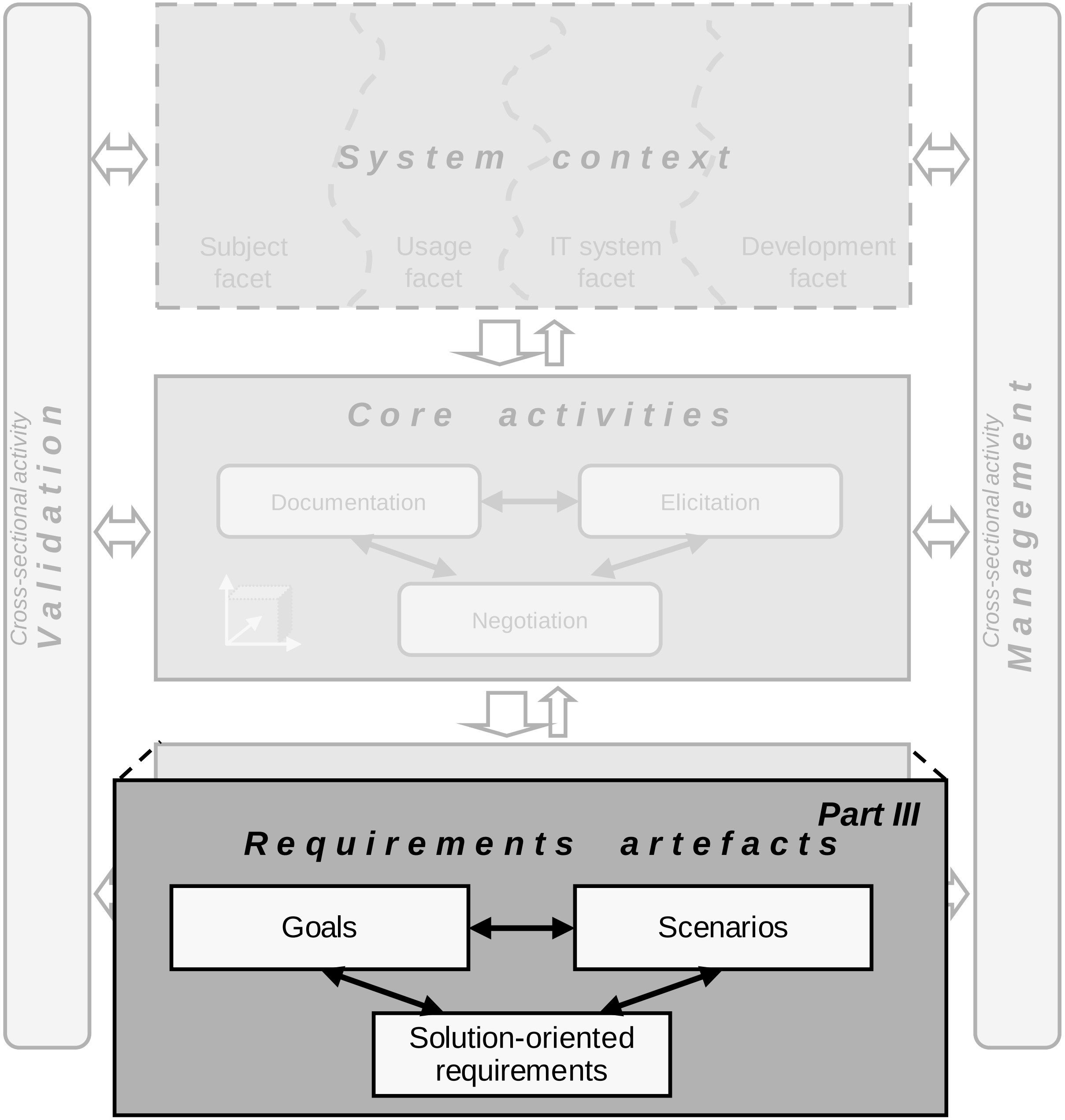
董　威

wdong@nudt.edu.cn

Department of Computer Science

National University of Defense Technology

Textbook: Klaus Pohl. “Requirements Engineering: Fundamentals, Principles, and Techniques”. Springer, 2010



Cross-sectional activity

Validation

System context

Subject facet

Usage facet

IT system facet

Development facet

Core activities

Documentation

Elicitation

Negotiation

Cross-sectional activity

Management

Part III

Requirements artefacts

Goals

Scenarios

Solution-oriented requirements

7 FUNDAMENTALS OF GOAL ORIENTATION

Motivation

- Facilitate common understanding of the system
- Support requirements elicitation with goals
- Identify and evaluate alternative realisations
- Detect irrelevant requirements
- Justification of requriements with rationales
- Proof of completeness for requirements specifications
- Goals have greater stability than requirements

The Term ”Goal”

An intention with regard to the objectives, properties or use of the system

AND/OR Goal Decomposition

- AND-decomposition of a goal:
  - decomposition of a goal G into a set of sub-goals G_1, …, G_n
  - n>1
  - Goal G is satisfied if and only if all sub-goals are satisfied
- OR-decomposition of a goal:
  - decomposition of a goal into a set of sub-goals G_1, …, G_n
  - n >1
  - Goal G is satisfied if one of sub-goals is satisfied

Example

目标“舒适、快速的目的地导航”，AND分解为：

- G1：方便地选择目的地
- G2：根据用户特定的参数自动进行路线规划
- G3：显示交通拥堵信息，并且能自动重新规划路线以避免拥堵

目标“能够定位汽车的位置”，OR分解为：

- G1：通过手机定位汽车
- G2：通过GPS定位汽车

Goal Dependencies

- ”Requires”-dependency
  - G_1 requires G_2 if the satisfaction of G_2 is a prerequisite for satisfying G_1
- ”Support”-dependency
  - G_1 supports G_2 if the satisfation of G_1 contributes positively to satisfying G_2

Example

- G1：系统在交通拥堵时为驾驶员提供导航
- G2：系统能够接受交通信息

G1需要G2

- G1：导航系统应能按需下载电子地图
- G2：系统应当允许方便地选择目的地

G1支持G2

AND和OR分解隐含支持依赖

Goal Dependencies (cont‘d)

- ”Obstruction” dependency
  - G_1 obstructs G_2 if satisfying of G_1 hinders the satisfaction of G_2
- ”Conflict” dependency
  - A conflict between G_1 and G_2 exists if satisfying G_1 excludes satisfying G_2 and vice-versa
- Goal equivalence
  - Satisfying G_1 leads to the satisfaction of the G_2 and vice-versa

Example

- G1：导航系统应当能通过GSM网络按需下载电子地图
- G2：导航系统所产生的GSM网络数据流量应尽可能低

G1阻碍G2

- G1：能够通过GPS定位汽车
- G2：遵守特定国家的隐私法律

G1和G2是冲突的

- G1：系统应符合A国的汽车安全法规
- G2：系统应符合B国的汽车安全法规

如果两个国家汽车安全法规完全一致，G1和G2是等价的

Goal Modelling Languages and Methods

- Model-based goal documentation
  - helps understanding and communicating goals
  - complements template-based documentation (each technique provides a different level of abstraction)
- Common goal modelling languages include Goal-oriented Requirements Language (GRL), i* and KAOS
- Goal modeling method consists of language, rules, guidelines and management practices
  - Common goal modelling methods include Goal-Based Requirements Analysis Method (GBRAM), Goal-Driven Change method (GDC), the i* framework, the KAOS framework, the Non-Functional Rquirements (NFR) framework

Documenting Goals Using AND/OR Trees and AND/OR Graphs

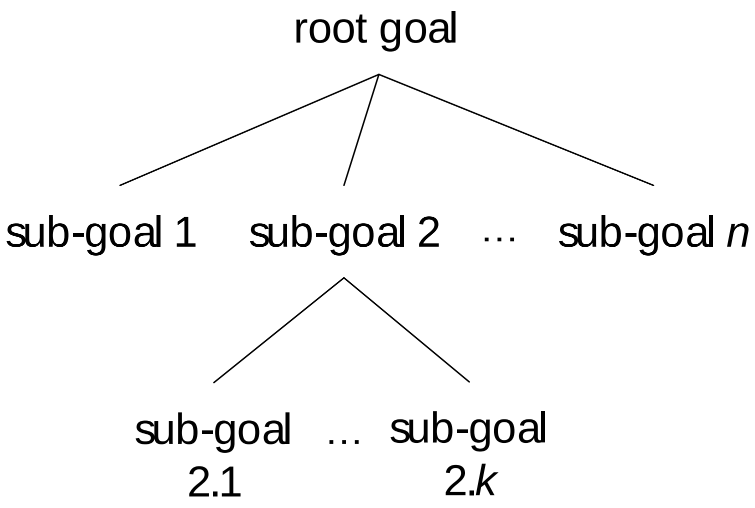
- AND/OR trees
  - Consist of nodes representing goal decompositions
  - Hierarchical, each node has exactly one super-goal
  - Graphical notation indicates type of decomposition (AND/OR)
  - Feature models provide a similar approach
- AND/OR graphs
  - Some sub-goals contribute to the satisfaction of more than one super goal
  - AND/OR graphs are acyclic

Documenting Goals Using AND/OR Trees and AND/OR Graphs

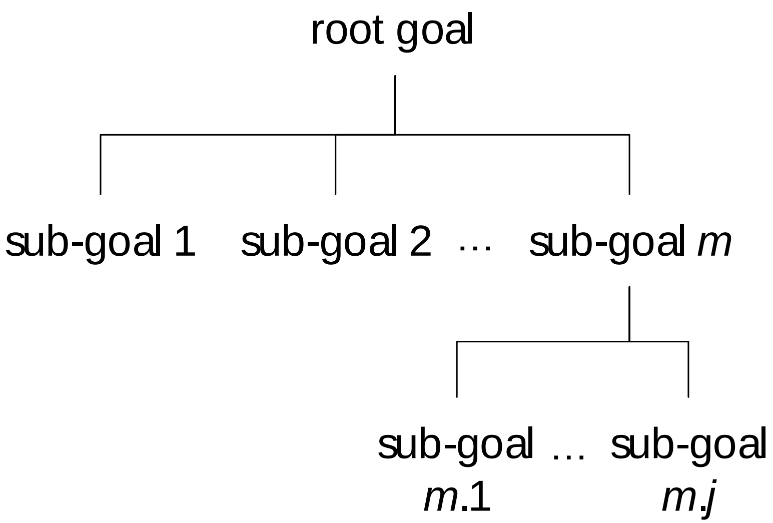
- Additional goal dependencies to extend AND/OR graphs
  - ”Requires” relationship
  - ”Conflict” relationship

Notation of AND/OR goal trees

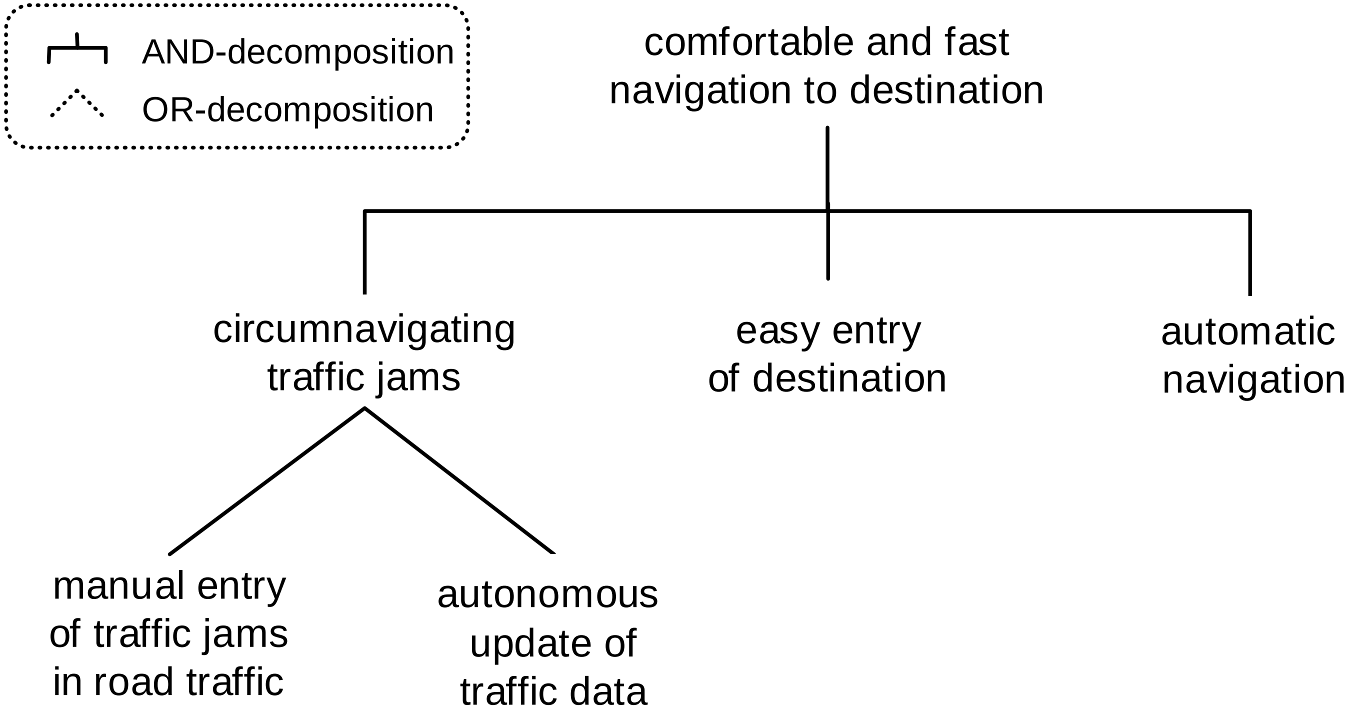
OR-decomposition



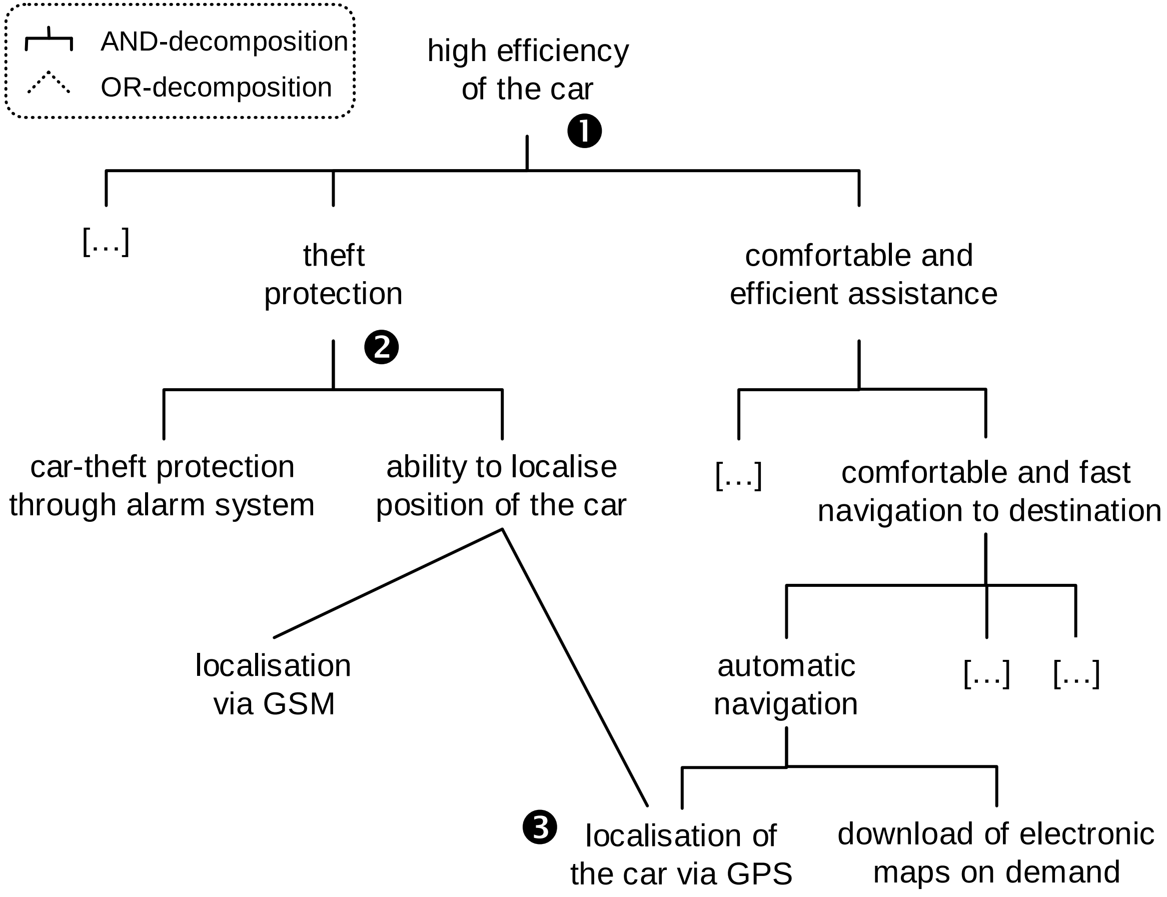
AND-decomposition



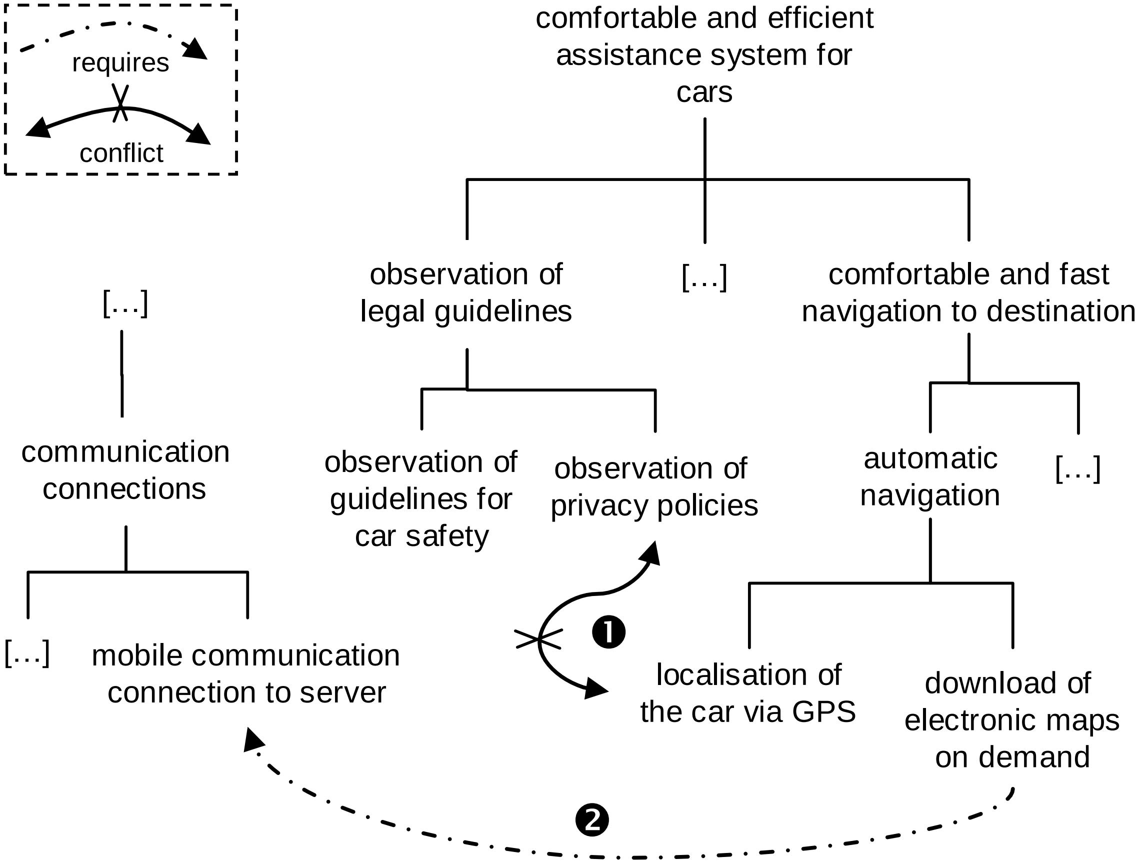
Example of goal modelling using AND/OR trees


AND-decomposition
OR-decomposition
comfortable and fast navigation to destination
circumnavigating traffic jams
easy entry of destination
automatic navigation
manual entry of traffic jams in road traffic
autonomous update of traffic data

Example of a goal model documented using an AND/OR graph


AND-decomposition
OR-decomposition
high efficiency of the car
❶
[…]
theft protection
comfortable and efficient assistance
❷
car-theft protection through alarm system
ability to localise position of the car
[…]
comfortable and fast navigation to destination
localisation via GSM
automatic navigation
[…]
[…]
❸
localisation of the car via GPS
download of electronic maps on demand

Example of goal modelling with extended AND/OR graphs

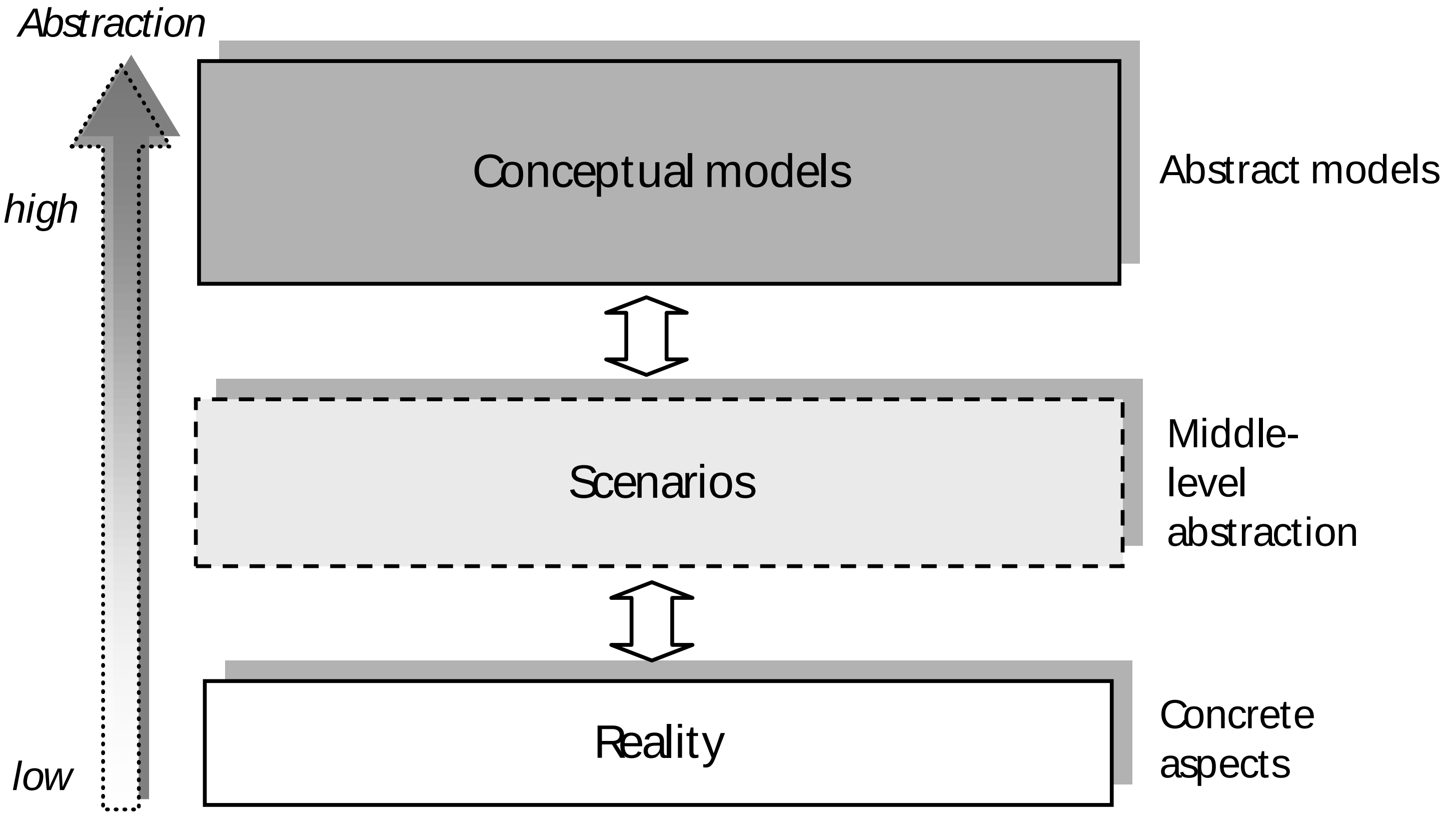

requires
conflict
comfortable and efficient assistance system for cars
observation of legal guidelines
[…]
comfortable and fast navigation to destination
[…]
communication connections
observation of guidelines for car safety
observation of privacy policies
automatic navigation
[…]
[…]
mobile communication connection to server
❶
localisation of the car via GPS
download of electronic maps on demand
❷

8 FUNDAMENTALS OF SCENARIOS

Scenarios as Middle-Level Abstractions

- Goals alone do not sufficiently support requirements elicitation
- Scenarios are intermediary abstractions between abstract models and reality
- Aspects documented in scenarios have different abstraction levels
- Scenarios may contain aspects
  - very close to the considered fragment of reality
  - abstraction level may also come close to the abstraction level of conceptual level

Scenarios as middle-level abstractions



Scenarios as a Means for Putting Requirements in Context

- Scenarios are well suited for documenting context information
- Link between requirements and the relevant context aspects
- Context information can be structured in a certain way
- Different types of scenarios depend on the extent of context information
  - System-internal scenarios
  - Interaction scenarios
  - Context Scenarios

Scenarios as a Means for Putting Requirements in Context (cont‘d)

- Kinds of context information can be characterised as follows
  - Actors: Persons or systems interacting with the system
  - Roles: Specific class of actors
  - Goals: Scenarios illustrate satisfaction of goals
  - Precondition: define conditions that must hold before executing the scenario
  - Postconditions: must hold within the system or context after executing the scenario
  - Resources: special preconditions referring to persons, information, financial or other material resources needed for a scenario
  - Location: real or fictional place where the scenario is executed

场景与上下文信息示例

“自动刹车控制”场景：

Carl正驾驶汽车以每小时50英里的速度在高速公路上行驶。Peter驾驶另一辆汽车在Carl前方行驶并开始减速。当Carl发现Peter减速后也踩下刹车踏板。此时Carl的车载系统检测到与前方车辆的距离已不在安全距离内，因此向驾驶员发出警告。接着，两车距离继续拉近。此时，为了帮助驾驶员，车载系统会启动自动全刹车，并通知Carl开启了自动全刹车。当两辆车距离停止减少后，系统会终止全刹车动作。但是，系统会继续控制Carl车辆的速度，直到与前车的距离拉大到安全距离，然后终止此次动作并通知Carl。

该场景的上下文信息：

- ■ 参与者：Carl
- ■ 角色：驾驶员
- ■ 目标：保持安全距离
- ■ 前置条件：汽车以高于每小时50英里的速度行驶
- ■ 后置条件：未发生追尾事故，并恢复与前车间的安全距离
- ■ 资源：与Peter汽车的距离
- ■ 场所：在高速公路上

Developing Scenarios for Each Context Facet

- **Subject facet**: scenarios may be used to document information about relevant context objects
- **Usage facet**: scenarios are mainly used for the specification of usage workflows and therefore contain usage information
- **IT system facet**: scenarios may be used in order to specify relevant procedures (e.g. maintainance)
- **Development facet**: scenarios may be used for documenting system modifications

Example of Negative Scenarios

允许的负面场景：

Chris把她的银行卡插入ATM机插槽，输入用户密码以及取款金额。ATM机通知她由于账户余额不足，无法支取所制定的金额。

禁止的负面场景：

Jack把他的银行卡插入ATM机插槽，输入用户密码以及取款金额。ATM机从Jack的账户上扣减了支取金额。接下来当ATM机吐钞时，吐钞机发生了失效。

Example of Misused Scenarios

汽车安全系统的不当使用场景：

另一辆汽车的司机故意穿插到Carl的正前方以迫使Carl的汽车紧急刹车。Carl在刹车过程中受伤。

Example of Descriptive Scenarios

描述性场景“输入目的地”：

Carl想开车到Plymouth市的Union Street。他使用汽车导航系统查询最短路径。他选择“输入目的地”，导航系统显示“请使用语音或手工方式输入目的地”。他想通过语音输入，因此说“Plymouth”。由于背景噪音以及发音不标准，系统无法识别，识别结果显示最有可能的目的地“Portsmouth”。系统同时显示“您的输入无法精确识别”，并提供如下选项：

（1）确认此目的地（是/否）；

（2）重新输入（新地点）；

（3）显示相似地名（相似）；

（4）手工输入（按键M）。

Carl说“相似”，系统列出相似地点的列表，包括“Portsmouth”、”Plymouths”等。Carl选择”Plymouths”。

Example of Exploratory Scenarios

探索性场景“输入起点”：

Carl想使用车上的导航系统驾驶到某一个地点。要解决的一个首要问题是行程的起点是否总是当前汽车所在位置，或者Carl是否可以自己定义起点。可能方式包括

（1）自动选择当前位置作为起点，避免额外交互，并支持快速导航。

（2）把非当前位置作为起点，可以作为旅行规划的一种手段。

（3）允许用户选择“导航（当前位置作为起点）”或者“输入起点模式的导航”，其中前一个被作为默认设置。

Example of Explanatory Scenarios

解释性场景“自动刹车控制”：

如果两车之间的距离快速接近，那么将存在很大的追尾风险。当汽车在高速公路上以55英里以上时速行驶时，快速变更车道可能导致打滑或旋转。因此，变更车道之前必须减速。因此，机载计算机启动自动紧急刹车。由于防抱死系统能确保汽车在刹车过程中的可操控性，Carl可以在刹车的同时安全地变更车道。为了避免驾驶员在自动刹车中受到惊吓，系统提醒Carl启动了自动刹车。与前车恢复安全距离后，车载计算机将驾驶控制交还给Carl。系统通知Carl控制权进行了转交，以便他做好准备接手汽车操控。

Example of System-Internal Scenarios

系统内部场景：

“导航控制”组件向“定位”组件请求GPS坐标。“定位”组件为“导航控制”组件提供坐标。“导航控制”组件调用“显示控制”组件，向其传递当前位置和目的地信息。“屏幕输入”组件发送路径参数给“导航控制”组件。随后“导航控制”组件计算最终路径。

Example of Interaction Scenarios

场景“输入目的地”：

Carl想开车到Plymouth市的Union Street。他使用汽车导航系统查询最短路径。他选择“输入目的地”，导航系统显示“请使用语音或手工方式输入目的地”。他想通过语音输入，因此说“Plymouth”。由于背景噪音以及发音不标准，系统无法识别，识别结果显示最有可能的目的地“Portsmouth”。系统同时显示“您的输入无法精确识别”，并提供如下选项：

（1）确认此目的地（是/否）；

（2）重新输入（新地点）；

（3）显示相似地名（相似）；

（4）手工输入（按键M）。

Carl说“相似”，系统列出相似地点的列表，包括“Portsmouth”、”Plymouths”等。Carl选择”Plymouths”。

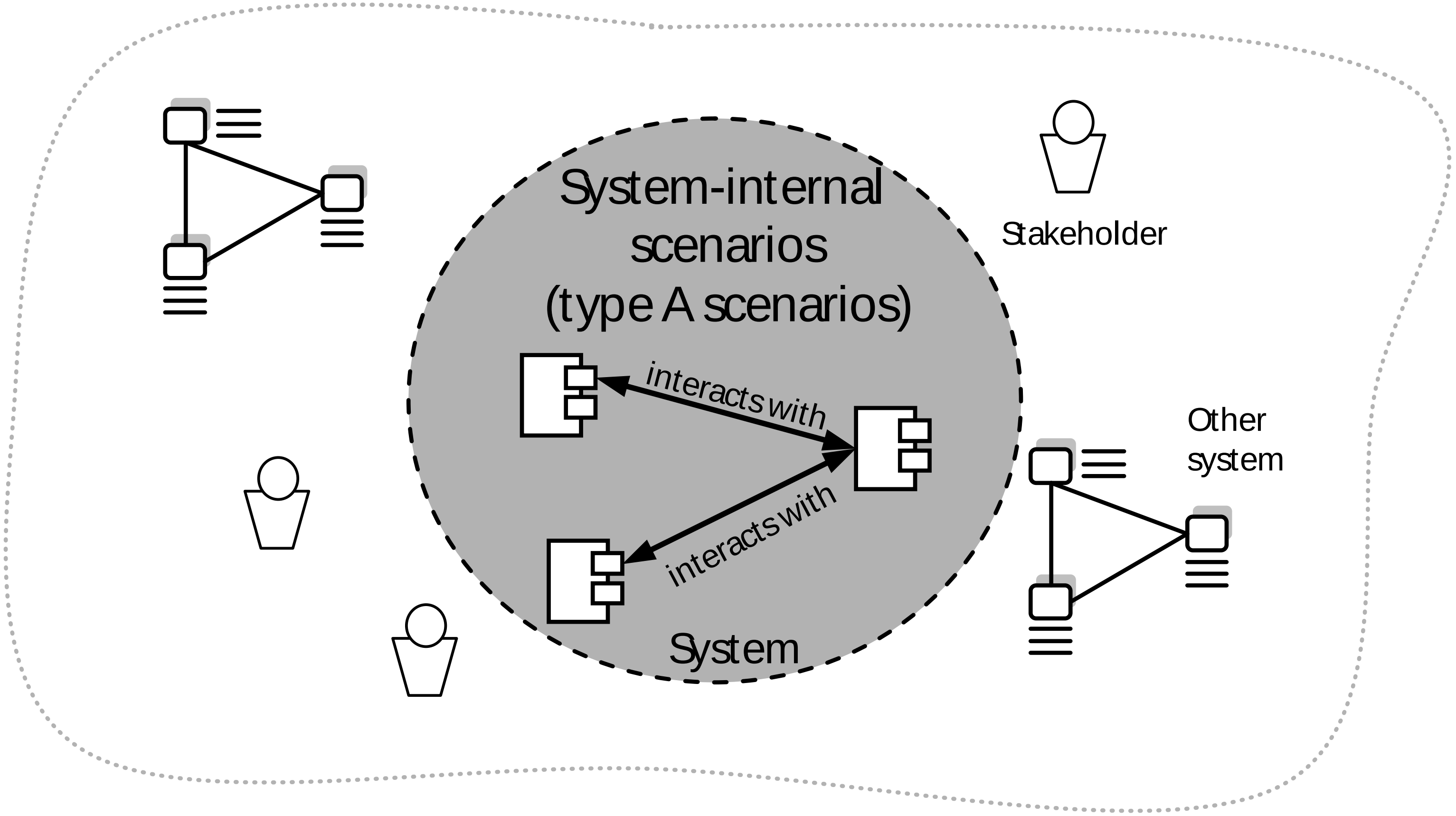
Example of Context Scenarios

上下文场景：

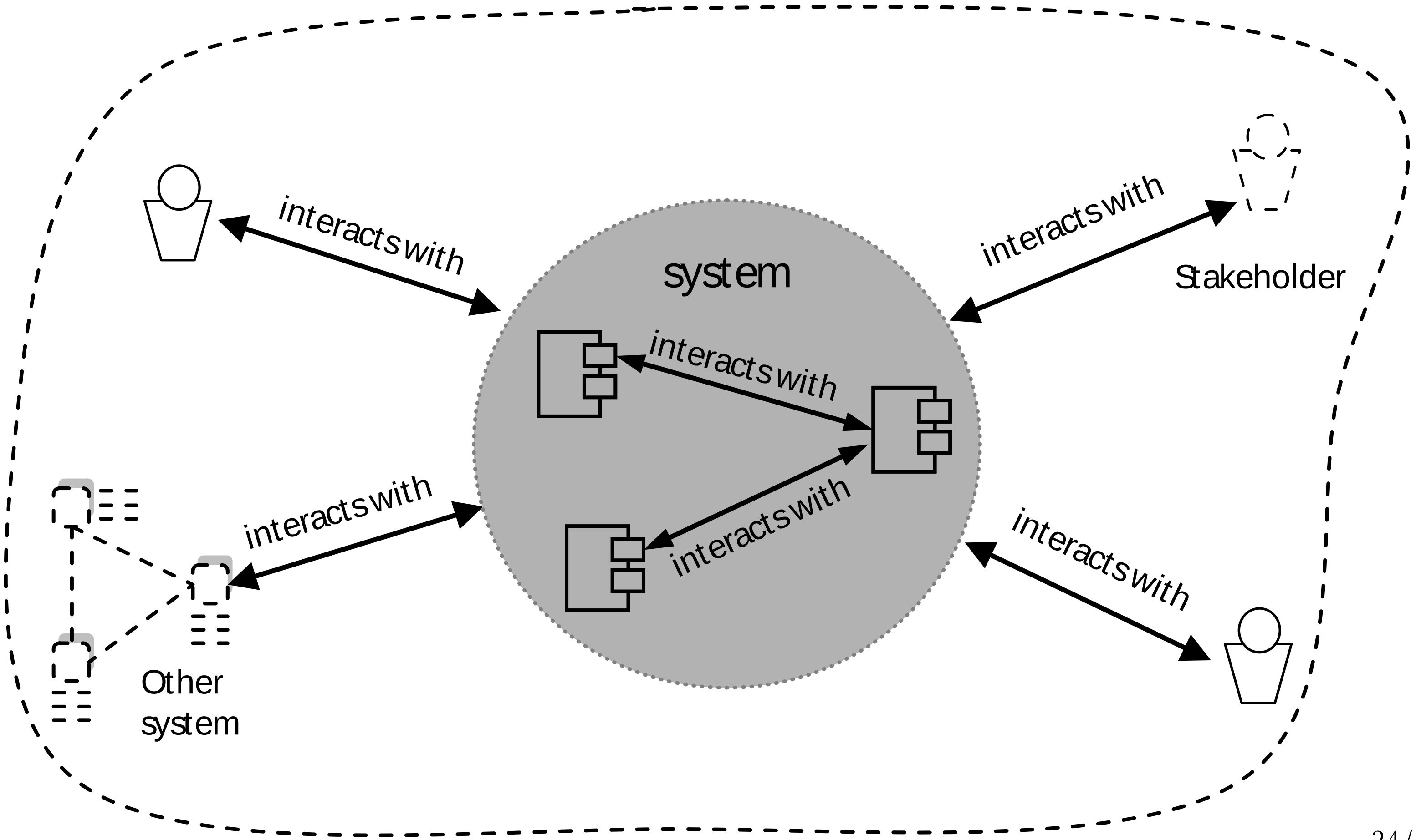
驾驶员想要开车到导航系统已安装地图以外的某个地点。由于导航系统在没有地图的情况下无法进行导航，驾驶员决定通过移动网络运营商进行导航。

驾驶员通过手机建立与移动网络运营商之间的连接，并使用运营商提供的在线路径规划服务。驾驶员在手机上的用户对话框中输入起点、目的地、导航参数（最短路径）。路径规划系统计算路径，并以标准的格式向手机传送路径及相应的地图。驾驶员建立与汽车导航系统之间的数据连接，传送所接收到的数据。然后，导航系统就可以导航到目的地。

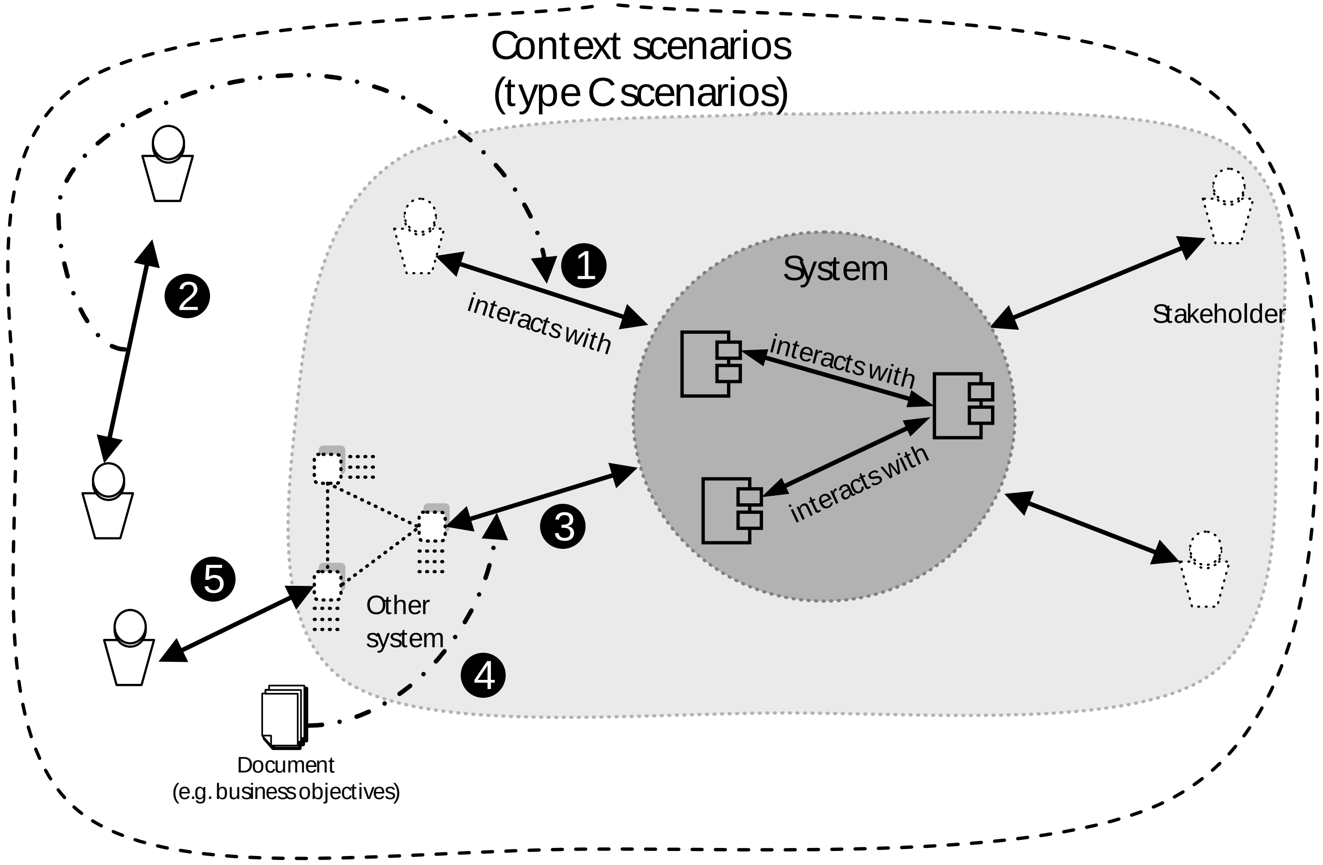
System-internal scenarios



Interaction scenarios



Context scenarios



Main Scenario, Alternative Scenarios, and Exception Scenarios

- Main scenario
  - Most common sequence of interactions for satisfying a goal
- Alternative scenario
  - Sequence of interactions that can be executed instead of main scenario
  - Results in satisfaction of the goals associated with the main scenario

Main Scenario, Alternative Scenarios, and Exception Scenarios (cont‘d)

- Exception scenario
  - Sequence of interactions
  - Executed instead of interactions documented in another scenario
  - Only executed when special event occurs
  - Leads to failure to satisfy one or multiple goals associated with the original scenario
- Documentation aspects
  - Alternative and exception scenarios can be documented as separate scenarios
  - Alternative and exception scenarios can be documented by replacing parts of the main scenario
  - Template exists for documenting them

Examples

可替换场景的定义

主场景的摘录：

步骤1：……

步骤2：驾驶员通过点击电子地图选择目的地；

步骤3：……

可替换场景的摘录：

步骤1：……

步骤2a：驾驶员从目的地列表中选择目的地；

步骤3：……

例外场景的定义

主场景的摘录：

步骤1：……

步骤2：系统确认目的地输入成功；

步骤3：系统通知驾驶员到目的地的路径计算成功；

步骤4：……

例外场景的摘录：

步骤1：……

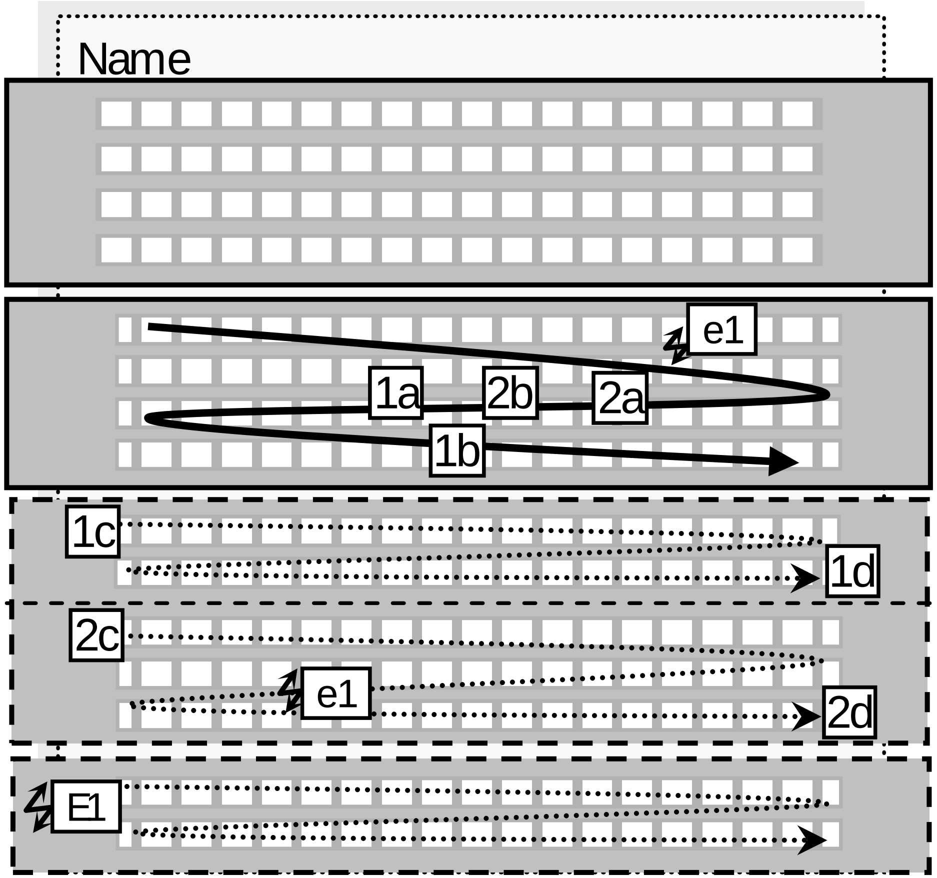
步骤2a：导航系统侦测到电源供应即将中断；

步骤3a：导航系统自动关闭；

Use Cases: Grouping Scenarios

- Use case: Specification of sequence of actions, including variant sequences and error sequences of a system
- Use case group main scenario with corresponding alternative and exception scenarios
- Use case contains:
  - Context information: e.g. the stakeholders‘ goals with respect to this use case as well as preconditions and postconditions for the execution of the use case
  - Main scenario: exactly one main scenario, describes the typical sequence of interaction steps satisfying the goal
  - Alternative scenarios (optional): one or multiple alternative scenarios; they replace parts of or the entire main scenario
  - Exception scenarios (optional): one or multiple exception scenarios, define how the system reacts to exceptions that occur during the execution of the alternative scenarios

Constituents of a use case



Use case context
(e.g. goals, preconditions,, postconditions)

Main scenario
(exactly one scenario)

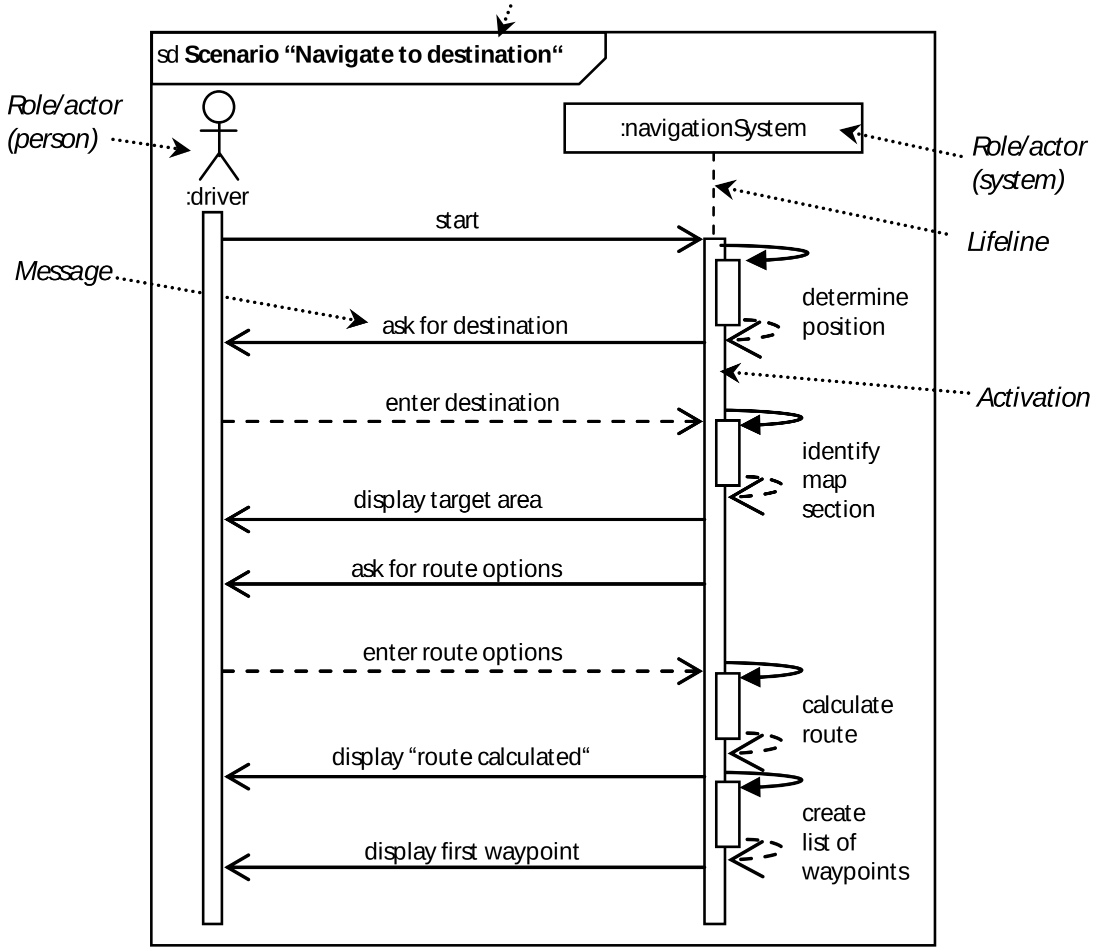
Alternative scenarios
(zero to multiple scenario)

Exception scenarios
(zero to multiple scenarios)

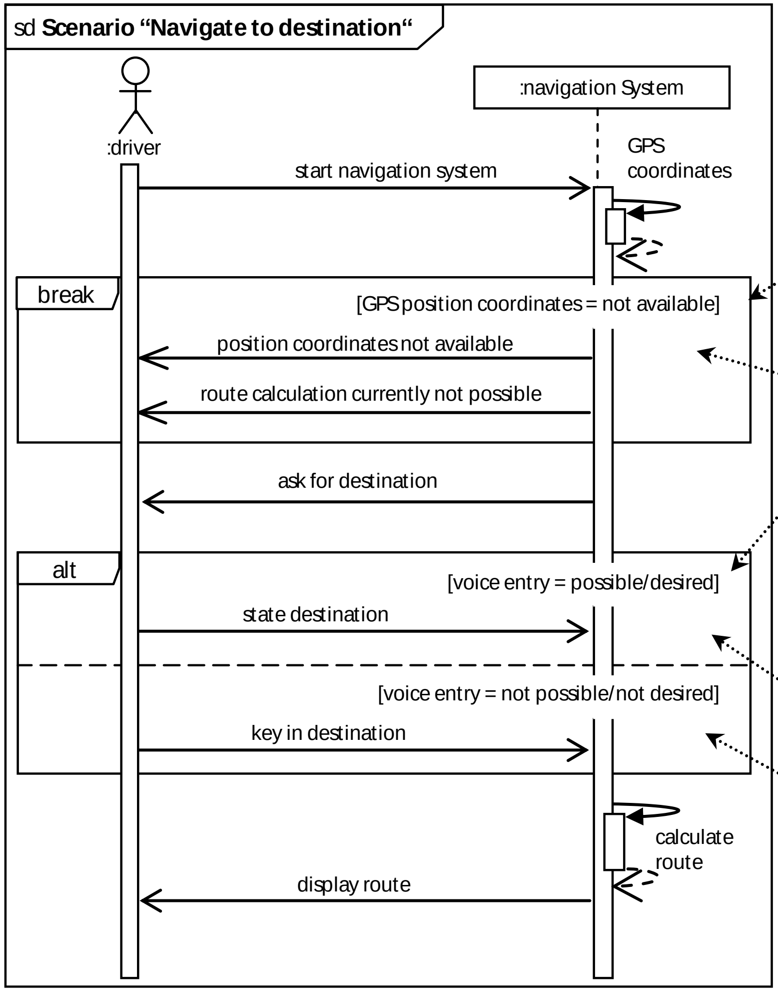
Sequence Diagrams

- Using models has several advantages over natural language
- UML supports doucmenting interactions by means of sequence diagrams
  - Message exchanges between a set of roles
  - Combined fragments for grouping messages
  - Exception scenarios using ”break” and ”alt” combined fragments
  - Alternative scenarios using ”alt” combined fragment

Documentation of a scenarios with sequence diagram



Alternative and exception scenarios in sequence diagrams


sd Scenario “Navigate to destination“
:driver
:navigation System
start navigation system
GPS coordinates
break
[GPS position coordinates = not available]
position coordinates not available
route calculation currently not possible
ask for destination
alt
[voice entry = possible/desired]
state destination
[voice entry = not possible/not desired]
key in destination
calculate route
display route

Condition (event) for the initiation of the exception scenario

Interactions of the exception scenario

Condition for the main scenario

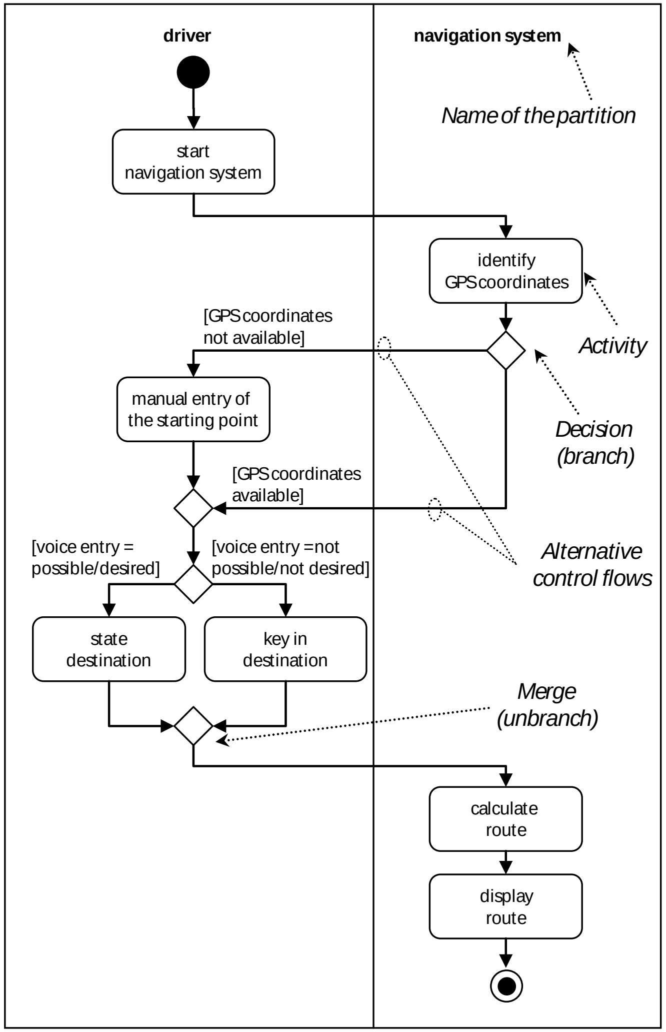
Interactions of the main scenario

Interactions of the alternative scenario

Activity Diagrams

- Main focus is the control flow
- Activity diagrams are control flow graphs
- Nodes represent execution of activities
- Alternative control flows are represented through decision nodes
- Activity diagrams especially well suited for documenting interrelations between main, alternative and exception scenarios

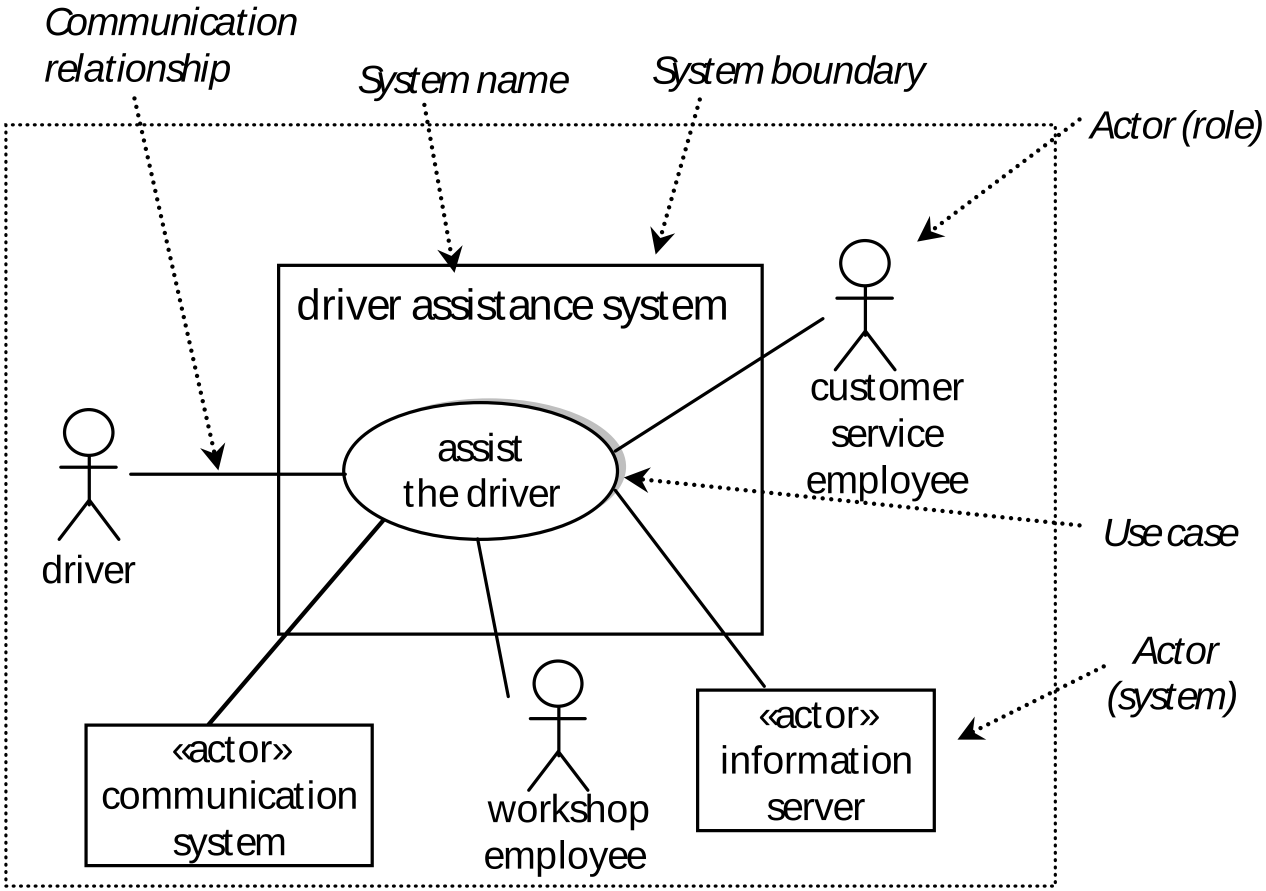
Documentation of the control flow of scenarios



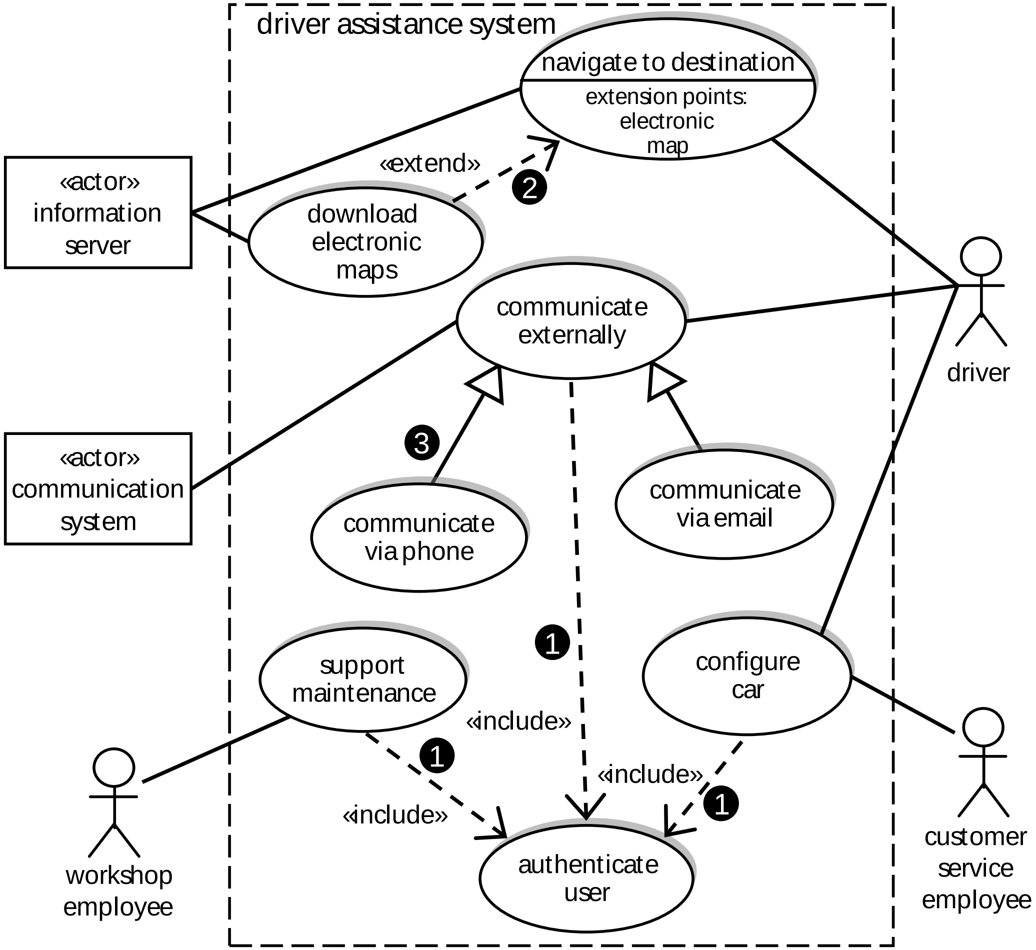
Use Case Diagrams

- Document relationships between use cases
- Modelling constructs of use case diagrams
  - Actors: represent systems or persons outside the system boundary
  - Use cases
  - System boundary: delimits the system from its environment
  - Relationships between actors and use cases: expresses that this actor interacts with the system in a use case
  - Relationships between use cases:
    - Generalisation: special use case inherits interactions of a general use case
    - ”Extend”: interaction sequence in use case extends the interaction sequence of another use case, depending on a defined condition
    - ”Include”: use case includes interaction sequence from anonther use case
  - Generalisation also between actors

Modelling constructs of use case diagrams in an example



Refined use case diagram of the driver assistance system



Use of the Different Scenario Types in the Requirements Engineering Process

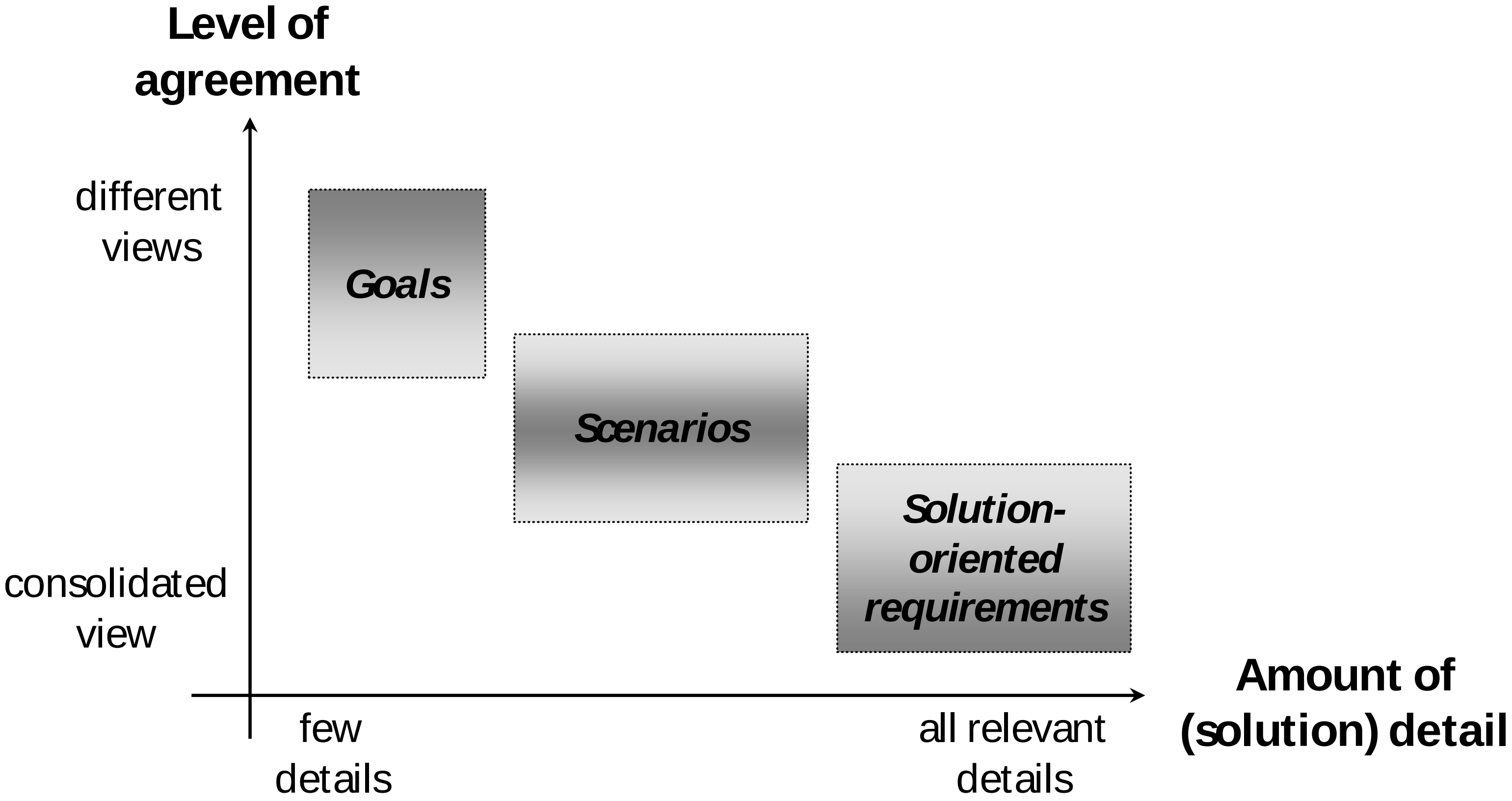
- Two kinds of usage
  - During the requirements engineering process to support or drive activities
  - In the specification, to be used for further development activities

9 SOLUTION-ORIENTED REQUIREMENTS

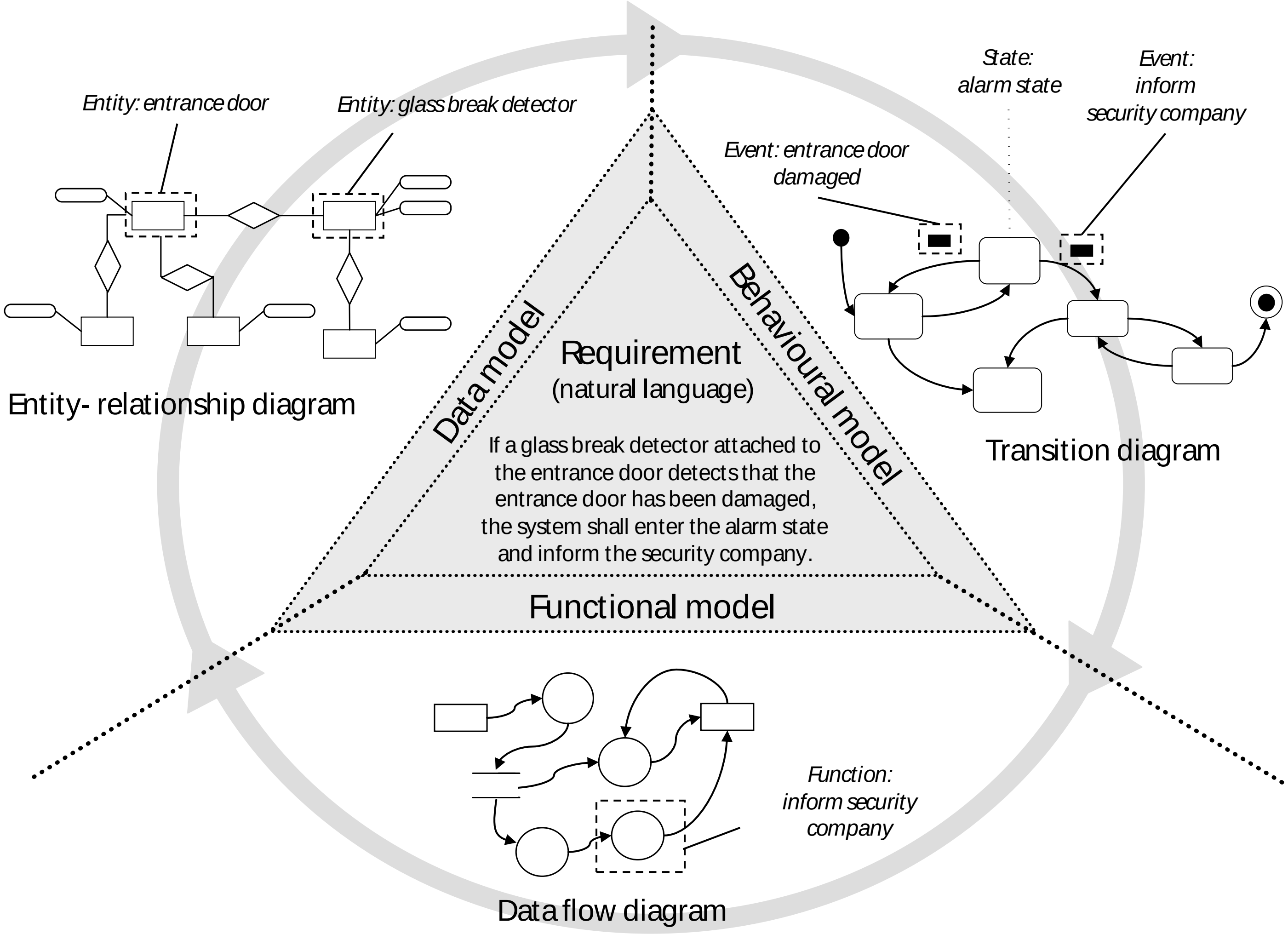
Three Perspectives on a Solution

- Data perspective
  - Defines data/information to be managed by the software-intensive system
- Functional perspective
  - Defines processes (functions) to be provided by the system
- Behavioural perspective
  - Defines the overall behaviour of the system

Coarse-grained characterisation of goals, scenarios and solution-oriented requirements



Model-based documentation of the three perspectives



Entity: entrance door

Entity: glass break detector

Entity- relationship diagram

Data model

Requirement

(natural language)

If a glass break detector attached to the entrance door detects that the entrance door has been damaged, the system shall enter the alarm state and inform the security company.

Behavioural model

Functional model

State: alarm state

Event: inform security company

Event: entrance door damaged

Transition diagram

Function: inform security company

Data flow diagram

Documenting Requirements in the Data Perspective

- Enhanced Entity-Relationship models
  - Entity types
  - Relationship types
  - Attributes (Properties of entity/relationship types)
  - Cardinality constraints
  - Generalisation/Specialisation of different types
    - Disjoint vs. overlapping
    - Total vs. partial

Documenting Requirements in the Data Perspective (cont‘d)

- UML class diagrams
  - Class:
    - Set of similar objects,
    - depicted as a rectangle with three compartments (name, attributes and operations)
  - Attribute
    - Lines of text in the attribute compartment
      - Type (class or data type)
      - Multiplicity (lower bound and upper bound)
      - Property string, e.g. {unique}, {read only}

Documenting Requirements in the Data Perspective (cont‘d)

- UML class diagrams
  - Association (relation of classes to each other)
    - Role name for each class in the association
    - Multiplicity (number of participating instances)
    - Property string, e.g. {unique} or {read only}
    - Association classes can be used to define attributes for associations
  - Aggregation and Composition
    - Specific types of associations, whole-part relationships
    - Composition: Strong whole-part relationship (parts have only one owner and cannot exist independently)

Documenting Requirements in the Data Perspective (cont‘d)

- UML class diagrams
  - Generalisation (relates sub-class to super-class)
    - Instances of sub-class are instances of super-class
    - Sub-class must be substitutable for its super-class
    - Sub-class defines additional attributes and operations
  - Abstract classes (class without direct instances)
    - Only instances of sub-classes allowed
  - Objects (instances of classes)
    - Concrete attribute values
    - Object diagrams for depicting objects

Documenting Requirements in Functional Perspective

- Data flow models
  - Process (function) represents a task or activity that the system shall provide or implement
  - Sources and sinks representing interactions of the system with external objects
  - Data stores are repositories of data
  - Data flows represent the transportation of information
  - Data dictionary contains definitions of data flows, data stores and their components
  - Mini-Specifications brriefly describe a process

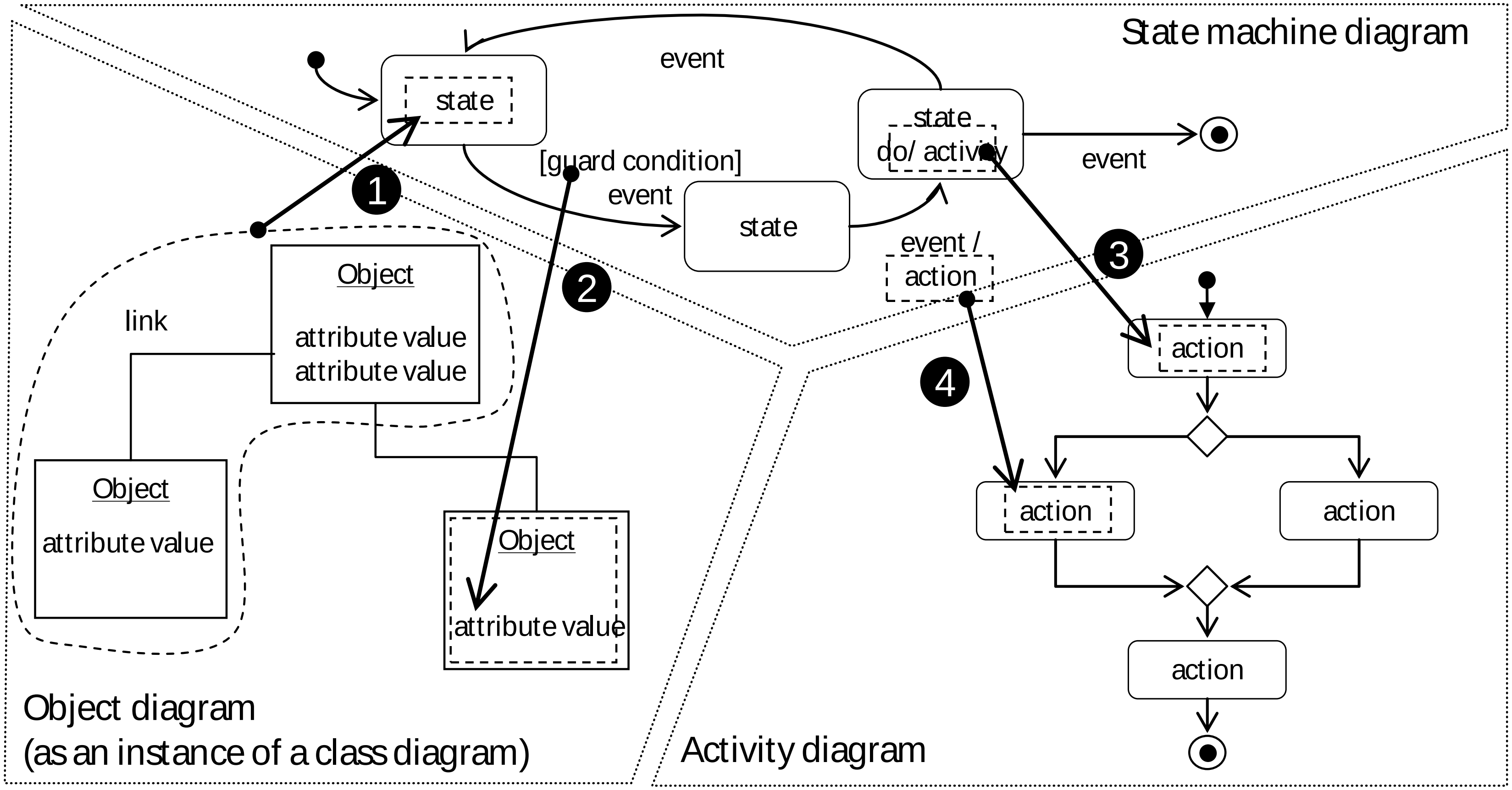
Documenting Requirements in the Behavioural Perspective

- Finite Automata
  - Deterministic and nondeterministic finite automata
- Mealy and Moore Automata
  - Mealy Automata: automata with input
  - Moore Automata: with input and output
- Statecharts
  - Actions in states and transitions
  - Conditional transitions
  - Hiearchical Refinement
  - Concurrency

Integration Using UML

- Integration of the perspectives
  - State machine can be related to a class
  - Guard condition of an action can refer to the attributes of a class
  - Action defined in state machine diagram can be related to an activity in an activity diagram

Interrelation of the three perspectives in UML



The end, thanks!