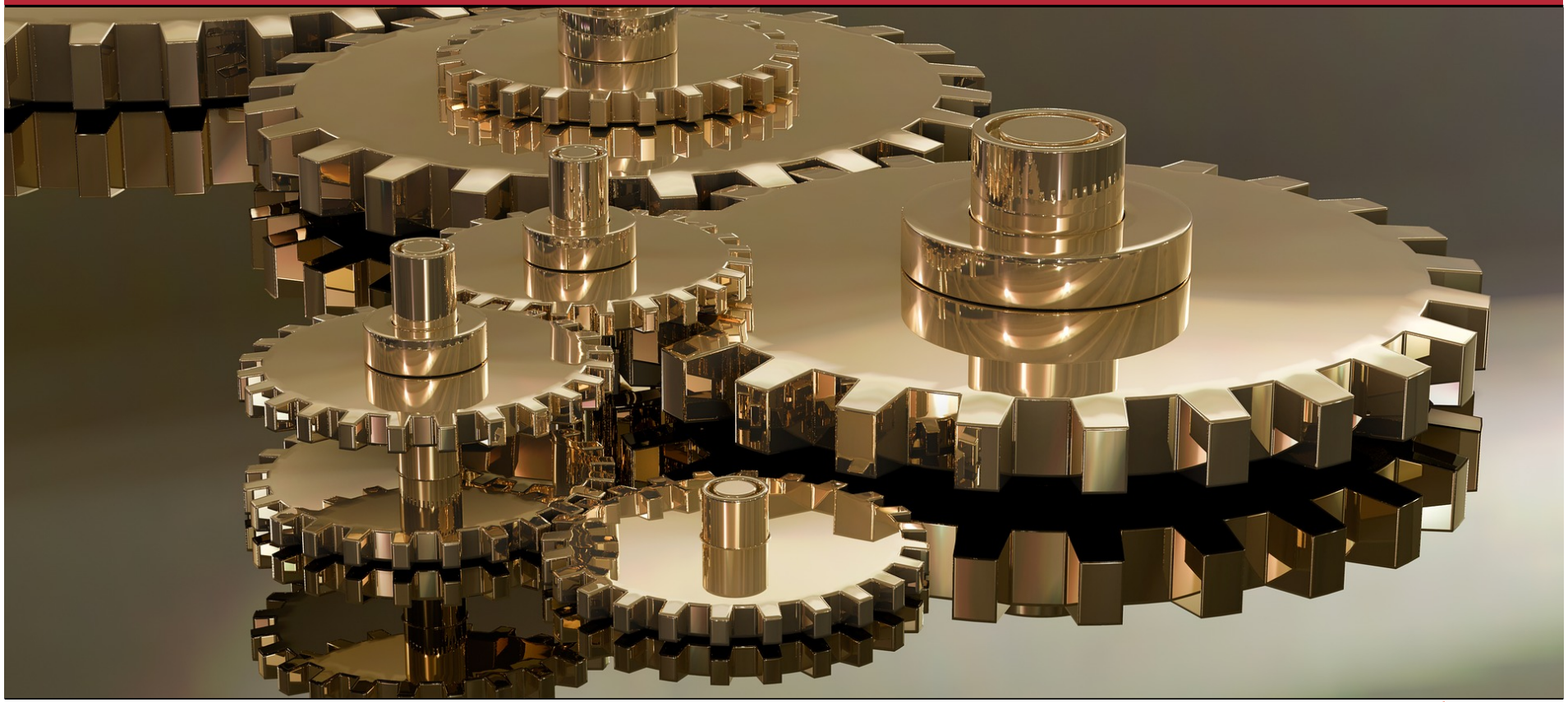
Software Engineering

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Cases

Prof. Dr. Reiner Hähnle
Fachgebiet Software Engineering



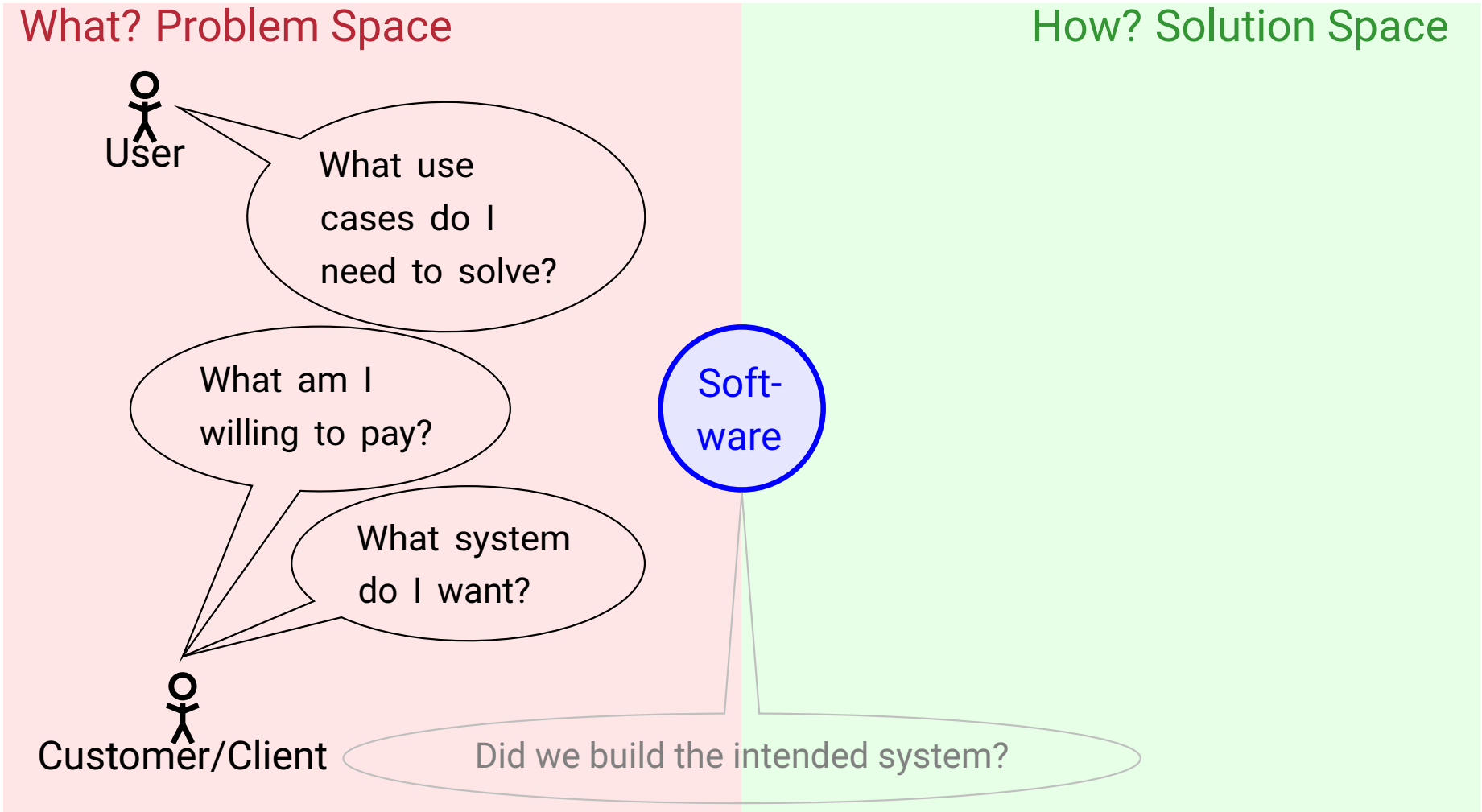
Software Engineering Group

Part I

Use Case Analysis

What? vs How?

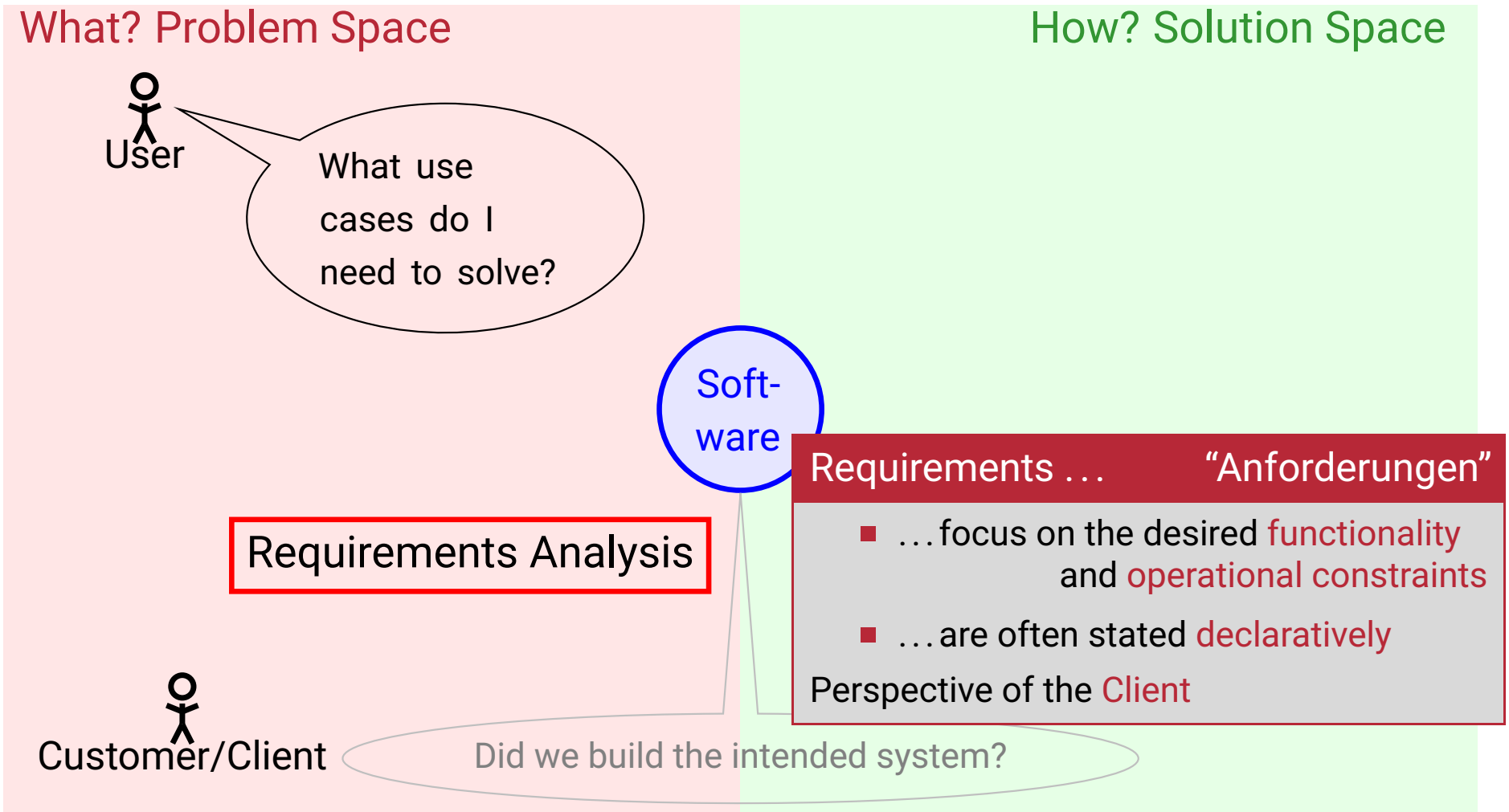
TECHNISCHE UNIVERSITÄT DARMSTADT



Software Engineering Group

What? vs How?

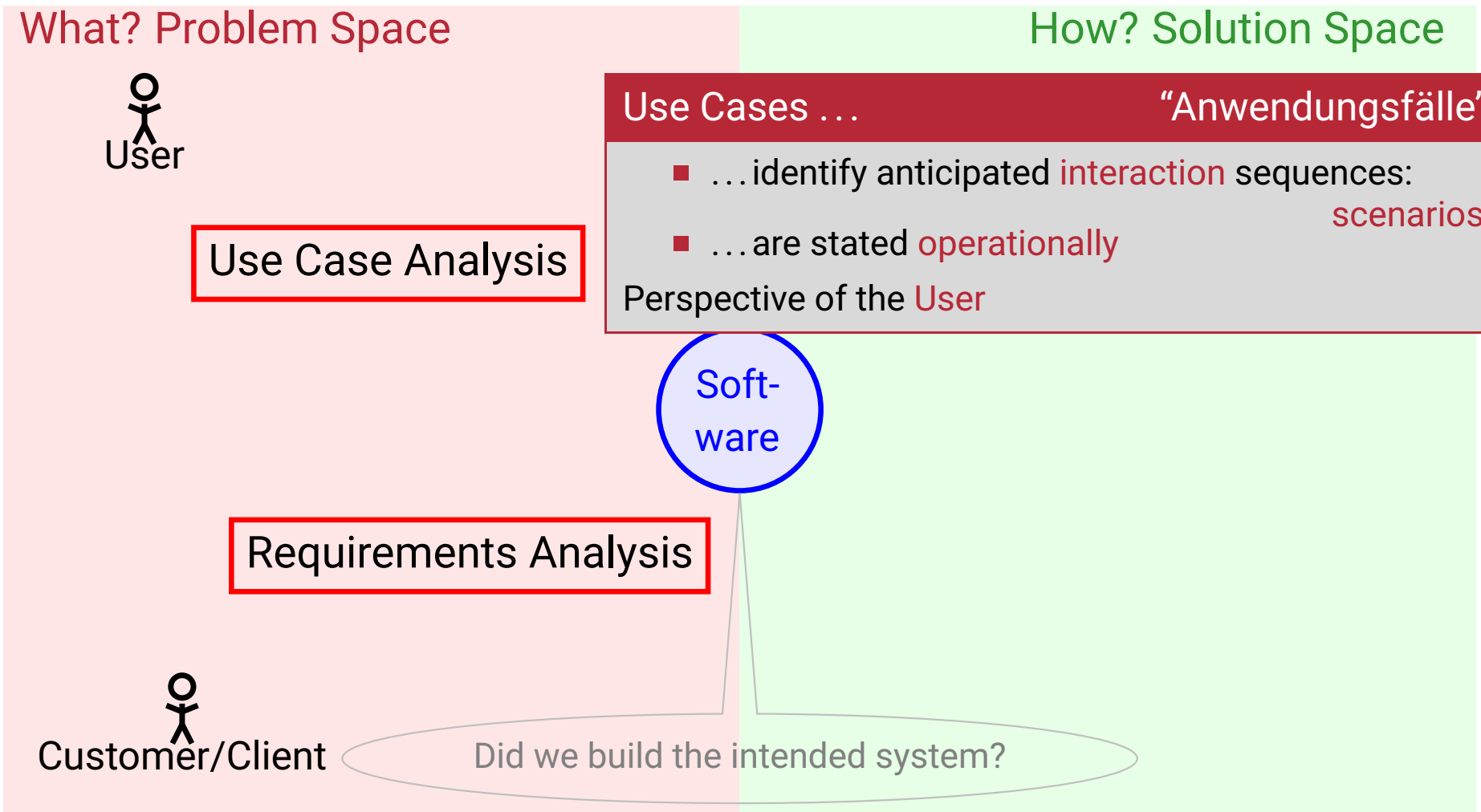
TECHNISCHE UNIVERSITÄT DARMSTADT



Software Engineering Group

What? vs How?

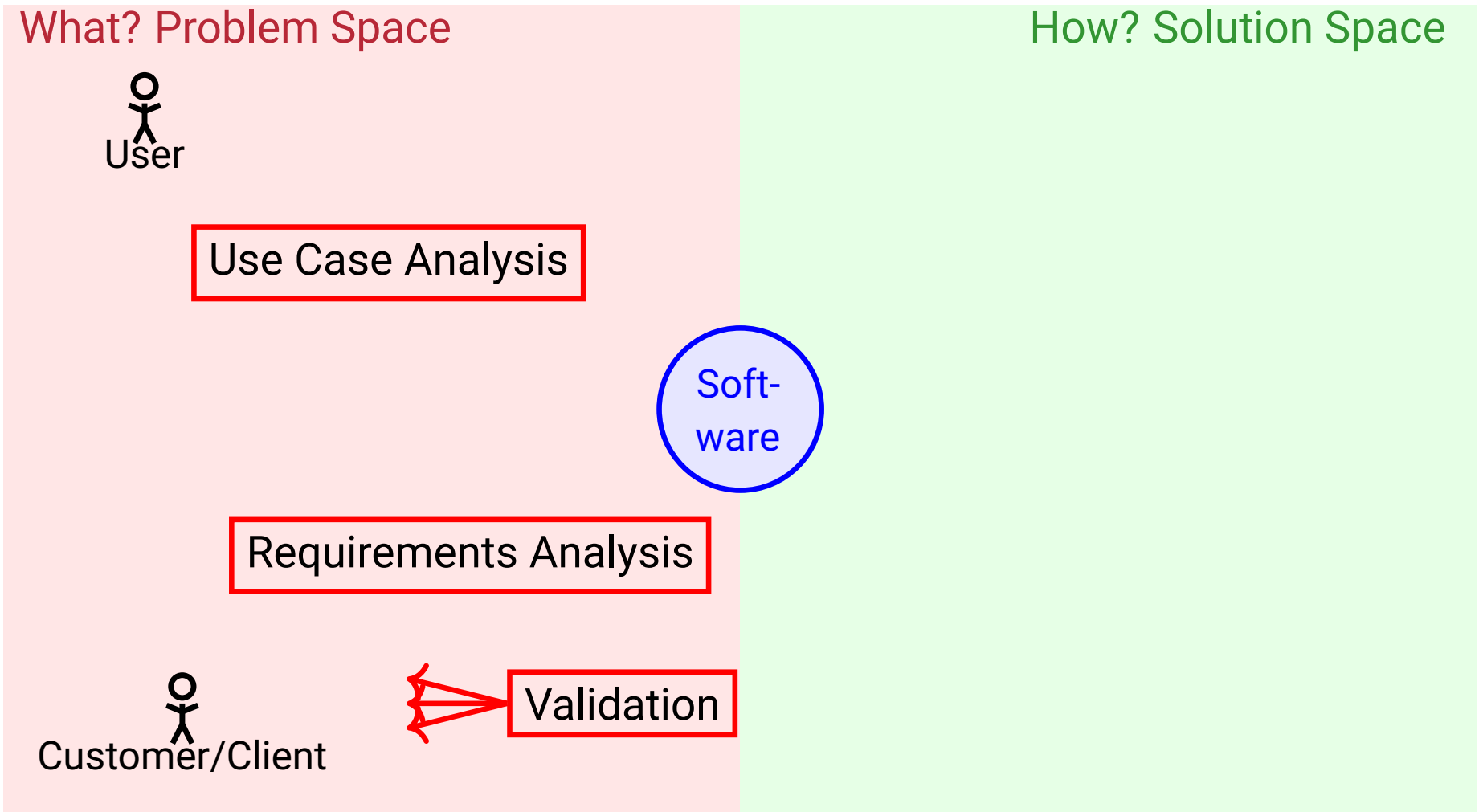
TECHNISCHE UNIVERSITÄT DARMSTADT



Software Engineering Group

What? vs How?

TECHNISCHE UNIVERSITÄT DARMSTADT



Software Engineering Group

Use Cases (“Anwendungsfälle”): Effort on the Client Side

TECHNISCHE UNIVERSITÄT DARMSTADT

Impossible to write good/complete uses cases without user involvement

Use Cases (“Anwendungsfälle”): Effort on the Client Side

TECHNISCHE UNIVERSITÄT DARMSTADT

Impossible to write good/complete uses cases without user involvement

A Common Misconception

Software cost ∼ development cost, not manufactoring cost

- Need to allocate at least 30–50 % of development cost at client side for requirements/use case analysis + validation
- In addition, allocate ongoing budget for maintenance

The CaSh Case Study

TECHNISCHE UNIVERSITÄT DARMSTADT

Main Roles & Functionalities (derived during requirements analysis)

General	■ Authentication
Administrator	■ Add/change new cars, rental locations ■ Billing
User	■ Check availability ■ Request booking ■ Change booking
Service Person	■ Take out vehicle for service

Software Engineering Group

Text Stories

TECHNISCHE UNIVERSITÄT DARMSTADT

Change Existing Booking

An authenticated customer wants to change an existing booking.

Software Engineering Group

Text Stories

TECHNISCHE UNIVERSITÄT DARMSTADT

Change Existing Booking

An authenticated customer wants to change an existing booking.

The customer selects the booking to change. The system displays the booking details, including adjacent reservations. The customer is offered to change the reserved duration, or else to cancel the reservation.

Text Stories

TECHNISCHE UNIVERSITÄT DARMSTADT

Change Existing Booking

An authenticated customer wants to change an existing booking.

The customer selects the booking to change. The system displays the booking details, including adjacent reservations. The customer is offered to change the reserved duration, or else to cancel the reservation.

When a new duration is entered, the system checks availability and records the change. In case of no availability, nothing happens and an information message is displayed.

Text Stories

TECHNISCHE UNIVERSITÄT DARMSTADT

Change Existing Booking

An authenticated customer wants to change an existing booking.

The customer selects the booking to change. The system displays the booking details, including adjacent reservations. The customer is offered to change the reserved duration, or else to cancel the reservation.

When a new duration is entered, the system checks availability and records the change. In case of no availability, nothing happens and an information message is displayed.

If the customer chooses to cancel the booking, then it is removed.

Text Stories

TECHNISCHE UNIVERSITÄT DARMSTADT

Change Existing Booking

An authenticated customer wants to change an existing booking.

The customer selects the booking to change. The system displays the booking details, including adjacent reservations. The customer is offered to change the reserved duration, or else to cancel the reservation.

When a new duration is entered, the system checks availability and records the change. In case of no availability, nothing happens and an information message is displayed.

If the customer chooses to cancel the booking, then it is removed.

Before any change happens, incurred cost is displayed, and a confirmation action is requested. After each performed change, a confirmation message is sent to the customer's preferred contact.

Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

Use cases are text stories used to discover and record requirements

Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

Use cases are text stories used to discover and record requirements

Use cases complement requirements analysis

Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

Use cases are text stories used to discover and record requirements

Use cases complement requirements analysis

Use cases provide operational requirements as a basis for system design

Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

Use cases are text stories used to discover and record requirements

Use cases complement requirements analysis

Use cases provide operational requirements as a basis for system design

Attention:

- Use cases $\neq$ User stories
  (a notion from Agile SW Development)
- Use cases are not a replacement of requirements analysis
  (do not capture non-functional requirements)

Constituents of Use Cases: Some Definitions

TECHNISCHE UNIVERSITÄT DARMSTADT

Actor ("Akteur") Someone or something with behavior:
a person, computer system, an organization, etc.

Primary actor The actor who requests a service from the system
(who initiates the use case)

Constituents of Use Cases: Some Definitions

TECHNISCHE UNIVERSITÄT DARMSTADT

Actor ("Akteur") Someone or something with behavior:
a person, computer system, an organization, etc.

Primary actor The actor who requests a service from the system (who initiates the use case)

Scenario (also known as a use case instance)
A specific sequence of actions and interactions between actors and the system

One particular story using a system

Software Engineering Group

Constituents of Use Cases: Some Definitions

TECHNISCHE UNIVERSITÄT DARMSTADT

Actor ("Akteur") Someone or something with behavior:
a person, computer system, an organization, etc.

Primary actor The actor who requests a service from the system (who initiates the use case)

Scenario (also known as a use case instance)
A specific sequence of actions and interactions between actors and the system

One particular story using a system

Use case A collection of related success and failure scenarios that describe an actor's usage of a system to achieve a goal

(compare "Change Booking" text story)

Software Engineering Group

Different Kinds of Use Cases
(Other distinctions exist)

TECHNISCHE UNIVERSITÄT DARMSTADT

White box vs. Black box — with whom does interaction take place?

White box (“Transparent”) use cases provide a detailed view of interaction of internals when satisfying a user goal

Black box use cases encapsulate the system and describe only interactions with external actors

Different Kinds of Use Cases
(Other distinctions exist)

TECHNISCHE UNIVERSITÄT DARMSTADT

White box vs. Black box — with whom does interaction take place?

White box (“Transparent”) use cases provide a detailed view of interaction of internals when satisfying a user goal

Black box use cases encapsulate the system and describe only interactions with external actors

Corporate vs. System

Corporate (“geschäftlich”) use cases describe a business process (often without mentioning the system under design)

System use cases are performed and described with respect to the system under design

Different Kinds of Use Cases
(Other distinctions exist)

TECHNISCHE UNIVERSITÄT DARMSTADT

White box vs. Black box — with whom does interaction take place?

White box (“Transparent”) use cases provide a detailed view of interaction of internals when satisfying a user goal

Black box use cases encapsulate the system and describe only interactions with external actors

Corporate vs. System

Corporate (“geschäftlich”) use cases describe a business process (often without mentioning the system under design)

System use cases are performed and described with respect to the system under design

Defaults

- Corporate use cases are usually white box use cases
- System use cases are in their vast majority black box use cases

Use Case Formats

Use Case Formats

TECHNISCHE UNIVERSITÄT DARMSTADT

Brief (“kurz”)
Terse one-paragraph summary, usually main success scenario

Software Engineering Group

Use Case Formats

TECHNISCHE UNIVERSITÄT DARMSTADT

Brief (“kurz”)
Terse one-paragraph summary, usually main success scenario

Casual (“informell”)
Informal paragraph format; multiple paragraphs that cover various scenarios (as seen in “Change Booking” text story)

Use Case Formats

TECHNISCHE UNIVERSITÄT DARMSTADT

Brief (“kurz”)
Terse one-paragraph summary, usually main success scenario

Casual (“informell”)
Informal paragraph format; multiple paragraphs that cover various scenarios (as seen in “Change Booking” text story)

Fully Dressed (“ausgearbeitet”)
All steps and variations written in detail; there are supporting sections, such as preconditions and success guarantees

Use Case Formats

TECHNISCHE UNIVERSITÄT DARMSTADT

Brief (“kurz”)
Terse one-paragraph summary, usually main success scenario

Casual (“informell”)
Informal paragraph format; multiple paragraphs that cover various scenarios (as seen in “Change Booking” text story)

Fully Dressed (“ausgearbeitet”)
All steps and variations written in detail; there are supporting sections, such as preconditions and success guarantees

Precision vs. Accuracy (“Detaillierungsgrad” bzw. “Zutreffendheit”)

Precision: Level of detail provided by the use case description: granularity

Accuracy: Correctness—is the description of the use case correct for the amount of detail given?

Template for Fully Dressed Use Case (Part I) ("Schema")

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines

Software Engineering Group

Template for Fully Dressed Use Case (Part I) ("Schema")

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Use Case Name	Start with a verb "Accomplish this task"

Software Engineering Group

Template for Fully Dressed Use Case (Part I) ("Schema")

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Use Case Name	Start with a verb "Accomplish this task"
Scope ("Umfang")	Corporate, system (better: name), subsystem

Two fundamental types of scope

Design Scope Specified in this row
Defines boundaries of system of the use case
(whole corporation, (sub-)system name)

Function Scope Limits functionality to be realized by system under design
Managed by a list of functions in and out of scope

Software Engineering Group

Template for Fully Dressed Use Case (Part I) ("Schema")

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Use Case Name	Start with a verb "Accomplish this task"
Scope ("Umfang")	Corporate, system (better: name), subsystem
Level ("Ebene")	User goal, summary goal, subfunction

Kinds of goals

User goal Most important elementary goal of user that produces value

Summary Goal Multiple user goals: context of system under design
or life-cycle sequence of related goals
or table of content for lower-level use cases

Subfunction: Use case being part of a user goal
factored out on a by-need basis, reusable in multiple goals

Template for Fully Dressed Use Case (Part I) ("Schema")

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Use Case Name	Start with a verb "Accomplish this task"
Scope ("Umfang")	Corporate, system (better: name), subsystem
Level ("Ebene")	User goal, summary goal, subfunction
Primary Actor	Initiates use case (invokes system to deliver service)

Template for Fully Dressed Use Case (Part I) ("Schema")

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Use Case Name	Start with a verb "Accomplish this task"
Scope ("Umfang")	Corporate, system (better: name), subsystem
Level ("Ebene")	User goal, summary goal, subfunction
Primary Actor	Initiates use case (invokes system to deliver service)
Stakeholders and Interests	Who cares about this use case? What do they want?

Stakeholder "Teilhaber"

Entities with interest in the behavior of the system under design:

- Company stakeholder
- Customer, vendor
- Regulatory agency, . . .

Software Engineering Group

Template for Fully Dressed Use Case (Part II) Guarantees

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines

Template for Fully Dressed Use Case (Part II) Guarantees

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Preconditions	What must be true at start & worth telling the reader

Preconditions "Vorbedingungen"

Spell out what the system will ensure at the start of the use case

Enforced by system and known to be true:
will not be checked again during use case execution
(For example, user authenticated)

Template for Fully Dressed Use Case (Part II) Guarantees

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Preconditions	What must be true at start & worth telling the reader
Minimal Guarantee	Fewest promises system makes to stakeholder

Minimal Guarantee

Smallest possible promise the system makes to stakeholders

In particular, if the primary actor's goal cannot be delivered:
(For example, the system logged all performed steps)

Cf. Minimal viable product (MVP)

Template for Fully Dressed Use Case (Part II) Guarantees

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Preconditions	What must be true at start & worth telling the reader
Minimal Guarantee	Fewest promises system makes to stakeholder

Software Engineering Group

Template for Fully Dressed Use Case (Part II) Guarantees

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Preconditions	What must be true at start & worth telling the reader
Minimal Guarantee	Fewest promises system makes to stakeholder
Success Guarantee	What must be true on successful completion & and is worth telling the reader

Success Guarantee

States the interests of the stakeholders to be satisfied after successful conclusion of the use case

At end of main success scenario or at end of a successful alternative path

Success guarantee stated in addition to minimal guarantee

Software Engineering Group

Template for Fully Dressed Use Case (Part III) Scenarios

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines

Software Engineering Group

Template for Fully Dressed Use Case (Part III) Scenarios

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Main Success Scenario	Representative scenario of successful use case execution

Main Success Scenario

Numbered list of successive steps executed to achieve the goal

Each step may reference to a sub-use case

Convention: first step specifies the trigger of the use case

Software Engineering Group

Template for Fully Dressed Use Case (Part III) Scenarios

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Main Success Scenario	Representative scenario of successful use case execution
Extensions	Alternative scenarios of success or failure

Extensions

Refers unambiguously to the step of the main success scenario being altered, or where failure might occur

If step 2 is altered, use 2.a, 2.b, . . . to refer to a variation of that step:

2.a *Condition for variant or failure condition*

2.a.1 ⟨step 1⟩ of variant/failure handling

2.a.2 ⟨step 2⟩ of variant/failure handling . . .

Software Engineering Group

Use Case Extension for CaSh Change Booking

TECHNISCHE UNIVERSITÄT DARMSTADT

Change Booking

When a new duration is supplied, the system checks availability and records the change. In case of no availability, nothing happens and an information message is displayed.

Before any change happens, incurred cost is displayed, and a confirmation action is requested.

Use Case Extension for CaSh Change Booking

TECHNISCHE UNIVERSITÄT DARMSTADT

Change Booking

When a new duration is supplied, the system checks availability and records the change. In case of no availability, nothing happens and an information message is displayed.

If the given duration is identical to the reserved period, then nothing happens.
Before any change happens, incurred cost is displayed, and a confirmation action is requested.

Use Case Extension for CaSh Change Booking

TECHNISCHE UNIVERSITÄT DARMSTADT

Change Booking

When a new duration is supplied, the system checks availability and records the change. In case of no availability, nothing happens and an information message is displayed.

If the given duration is identical to the reserved period, then nothing happens.
Before any change happens, incurred cost is displayed, and a confirmation action is requested.
If the session times out before confirmation is given, nothing happens.

Template for Fully Dressed Use Case (Part III) Scenarios

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Main Success Scenario	Representative scenario of successful use case execution
Extensions	Alternative scenarios of success or failure

Extensions

Refers unambiguously to the step of the main success scenario being altered, or where failure might occur

If step 2 is altered, use 2.a, 2.b, . . . to refer to a variation of that step:

2.a *Condition for variant or failure condition*

2.a.1 ⟨step 1⟩ of variant/failure handling

2.a.2 ⟨step 2⟩ of variant/failure handling . . .

Software Engineering Group

Template for Fully Dressed Use Case (Part III) Scenarios

TECHNISCHE UNIVERSITÄT DARMSTADT

Use Case Section	Purpose/Guidelines
Main Success Scenario	Representative scenario of successful use case execution
Extensions	Alternative scenarios of success or failure
Special Requirements	Related non-functional requirements
Technology and Data Variation List	Varying I/O methods and data formats
Frequency of Occurrence	Influences investigation, testing and timing of implementation
Miscellaneous	For example, open issues

Excerpt of A Fully Dressed Use Case (Part I)

TECHNISCHE UNIVERSITÄT DARMSTADT

Name	List Available Cars
Scope	CaSh Booking Module
Level	User goal
Primary Actor	CaSh Customer

Software Engineering Group

Excerpt of A Fully Dressed Use Case (Part I)

TECHNISCHE UNIVERSITÄT DARMSTADT

Name	List Available Cars
Scope	CaSh Booking Module
Level	User goal
Primary Actor	CaSh Customer
Stakeholders and Interests	Customer: wants to find available cars for a given duration and location CaSh Company: wants to make accurate offer

Software Engineering Group

Excerpt of A Fully Dressed Use Case (Part I)

TECHNISCHE UNIVERSITÄT DARMSTADT

Name	List Available Cars
Scope	CaSh Booking Module
Level	User goal
Primary Actor	CaSh Customer
Stakeholders and Interests	Customer: wants to find available cars for a given duration and location CaSh Company: wants to make accurate offer
Precondition	Customer is authenticated

Software Engineering Group

Excerpt of A Fully Dressed Use Case (Part I)

TECHNISCHE UNIVERSITÄT DARMSTADT

Name	List Available Cars
Scope	CaSh Booking Module
Level	User goal
Primary Actor	CaSh Customer
Stakeholders and Interests	Customer: wants to find available cars for a given duration and location CaSh Company: wants to make accurate offer
Precondition	Customer is authenticated
Minimal guarantee	No side effects

Software Engineering Group

Excerpt of A Fully Dressed Use Case (Part I)

TECHNISCHE UNIVERSITÄT DARMSTADT

Name	List Available Cars
Scope	CaSh Booking Module
Level	User goal
Primary Actor	CaSh Customer
Stakeholders and Interests	Customer: wants to find available cars for a given duration and location CaSh Company: wants to make accurate offer
Precondition	Customer is authenticated
Minimal guarantee	No side effects
Success guarantee	All available cars of requested class, specified duration, and location are displayed to customer

Software Engineering Group

Excerpt of A Fully Dressed Use Case (Part II)

TECHNISCHE UNIVERSITÄT DARMSTADT

Name	List Available Car
⋮	⋮
Main Success Scenario	1. Customer wants to know whether there is a suitable car to book 2. Customer enters address or uses location service; enters maximal location distance or use default 3. Customer enters desired reservation period 4. System validates time period 5. Customer completes search details by entering desired class of car 6. System determines matching options 7. System displays all available cars matching selection criteria

Excerpt of A Fully Dressed Use Case (Part III)

TECHNISCHE UNIVERSITÄT DARMSTADT

Name	List Available Car
⋮	⋮
Extensions	3.a End date is equal to or before start date 3.a.1 Ask customer to specify non-empty period 5.a Session times out before search details completed 5.a.1 Customer is logged out 7.a Search criteria give no result 7.a.1 System includes cars one class lower and higher 7.a.2 System includes cars with partial overlap of requested duration 7.a.3 System suggests to customer to increase location distance ⋮

Software Engineering Group

Guidelines for Developing Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

In General: Proceed Incrementally, Top-Down

1. Identify all currently relevant use cases accurately at an abstract level: User goal
2. Add precision gradually, work out the details

Guidelines for Developing Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

Recommended Workflow

1. List supported actors & their goals
   Review list for accuracy and completeness
   Outcome: first level of precision of functional requirements

Guidelines for Developing Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

Recommended Workflow

1. List supported actors & their goals
   Review list for accuracy and completeness
   Outcome: first level of precision of functional requirements
2. Write stakeholders, trigger, and main success scenario for each use case
   Validate that the system delivers to important stakeholders
   Outcome: second level of precision for functional requirements

Guidelines for Developing Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

Recommended Workflow

1. List supported actors & their goals
   Review list for accuracy and completeness
   Outcome: first level of precision of functional requirements
2. Write stakeholders, trigger, and main success scenario for each use case
   Validate that the system delivers to important stakeholders
   Outcome: second level of precision for functional requirements
3. Identify and list all failure conditions

Guidelines for Developing Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

Recommended Workflow

1. List supported actors & their goals
   Review list for accuracy and completeness
   Outcome: first level of precision of functional requirements
2. Write stakeholders, trigger, and main success scenario for each use case
   Validate that the system delivers to important stakeholders
   Outcome: second level of precision for functional requirements
3. Identify and list all failure conditions
4. Write failure handling
   Do not interleave this step with the previous one:
   Danger of not completing list of all failures

Further Recommendations for Writing Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

- During early requirements analysis: Keep the user interface out (Focus on intent)
- Write terse use cases
- Write black box use cases:

  Describe what the system must do and not how

  "The system records the booking." (✓)
  "The system writes the booking to a database." (×)
- Take an actor or actor-goal perspective:
  Focus on the users or actors of a system:
  - Ask about their goals and their typical situations
  - Focus on understanding what the actor considers a valuable outcome

Further Recommendations for Writing Use Cases

TECHNISCHE UNIVERSITÄT DARMSTADT

- During early requirements analysis: Keep the user interface out (Focus on intent)
- Write terse use cases
- Write black box use cases:

  Describe what the system must do and not how

  "The system records the booking." (✓)
  "The system writes the booking to a database." (×)
- Take an actor or actor-goal perspective:
  Focus on the users or actors of a system:
  - Ask about their goals and their typical situations
  - Focus on understanding what the actor considers a valuable outcome

Identifying and writing good use cases may take weeks

Use Cases during Initial Requirements Analysis: Examples

TECHNISCHE UNIVERSITÄT DARMSTADT

Which of these phrases is a valid use case?

- Negotiate a supplier contract
- Handle returned sale
- Log in
- Move piece on game board

Think about those, we will come back to it . . .

Use Cases during Initial Requirements Analysis: Checklist

TECHNISCHE UNIVERSITÄT DARMSTADT

1. Is it a well-defined task . . .
   - performed by one stakeholder in one place at one time,
   - to model a business event,
   - which adds measurable business value and
   - leaves the system's data in a consistent state?

   This is called Elementary Business Process (EBP)
2. Is it not merely a single step of a scenario (size test)?

Use Cases during Initial Requirements Analysis: Examples Revisited

TECHNISCHE UNIVERSITÄT DARMSTADT

Which of these phrases is a valid use case?

- Negotiate a supplier contract
- Handle returned sale
- Log in
- Move piece on game board

Use Cases during Initial Requirements Analysis: Examples Revisited

TECHNISCHE UNIVERSITÄT DARMSTADT

Which of these phrases is a valid use case?

- Negotiate a supplier contract
  X too complex—no elementary business case
- Handle returned sale
- Log in
- Move piece on game board

Software Engineering Group

Use Cases during Initial Requirements Analysis: Examples Revisited

TECHNISCHE UNIVERSITÄT DARMSTADT

Which of these phrases is a valid use case?

- Negotiate a supplier contract
  X too complex—no elementary business case
- Handle returned sale
  ✓
- Log in
- Move piece on game board

Software Engineering Group

Use Cases during Initial Requirements Analysis: Examples Revisited

TECHNISCHE UNIVERSITÄT DARMSTADT

Which of these phrases is a valid use case?

- Negotiate a supplier contract
  X too complex—no elementary business case
- Handle returned sale
  ✓
- Log in
  (X) Depends on context, usually does not add value to business case
  could be sub-use case
- Move piece on game board

Software Engineering Group

Use Cases during Initial Requirements Analysis: Examples Revisited

TECHNISCHE UNIVERSITÄT DARMSTADT

Which of these phrases is a valid use case?

- Negotiate a supplier contract
  X too complex—no elementary business case
- Handle returned sale
  ✓
- Log in
  (X) Depends on context, usually does not add value to business case
  could be sub-use case
- Move piece on game board
  X Single step, fails size test

Software Engineering Group

Part II

UML Use Case Diagrams

The Unified Modeling Language (UML)

TECHNISCHE UNIVERSITÄT DARMSTADT

Some Facts

- The Unified Modeling Language (UML) is a visual, yet precise, design notation for software development
- Originated as merger of the three OO modelling approaches: Booch, OMT, OOSE — as well as best practices
- Maintained and developed by the Object Modeling Group (OMG)
- Original motivation: support object modelling tool interoperability by agreeing on syntax and semantics of underlying modelling language

The Unified Modeling Language (UML)

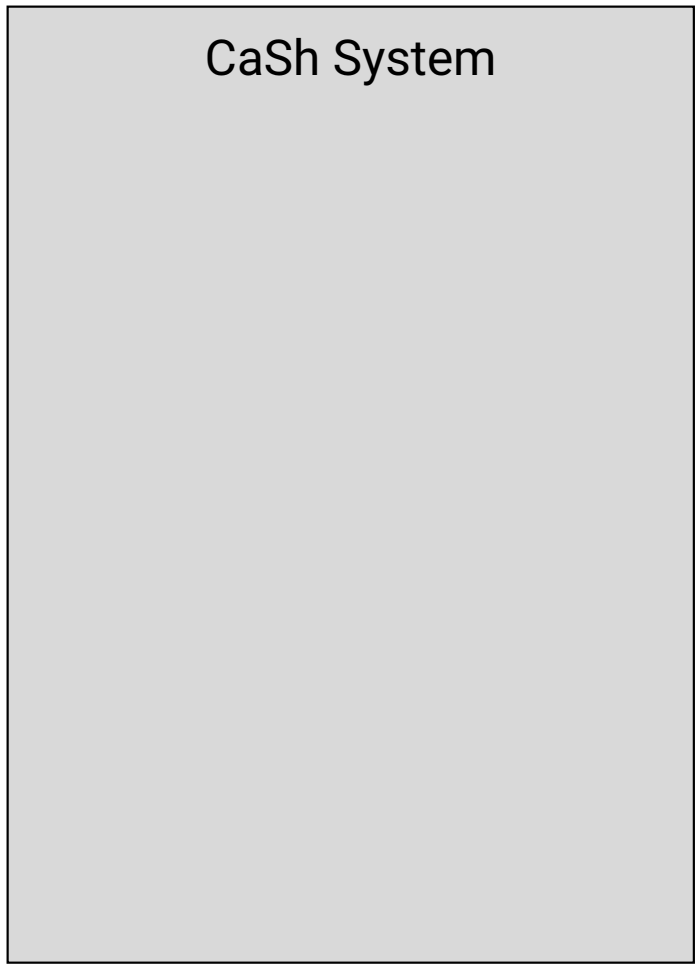
TECHNISCHE UNIVERSITÄT DARMSTADT

Some Facts

- The Unified Modeling Language (UML) is a visual, yet precise, design notation for software development
- Originated as merger of the three OO modelling approaches: Booch, OMT, OOSE — as well as best practices
- Maintained and developed by the Object Modeling Group (OMG)
- Original motivation: support object modelling tool interoperability by agreeing on syntax and semantics of underlying modelling language
- Consists of collection of diagrammatic modelling notations, including:
  - Use case diagram
  - Class, object, package diagram
  - Sequence, collaboration, activity diagram
  - State diagram, . . . + ca. 10 more

UML Use Case Diagram

TECHNISCHE UNIVERSITÄT DARMSTADT

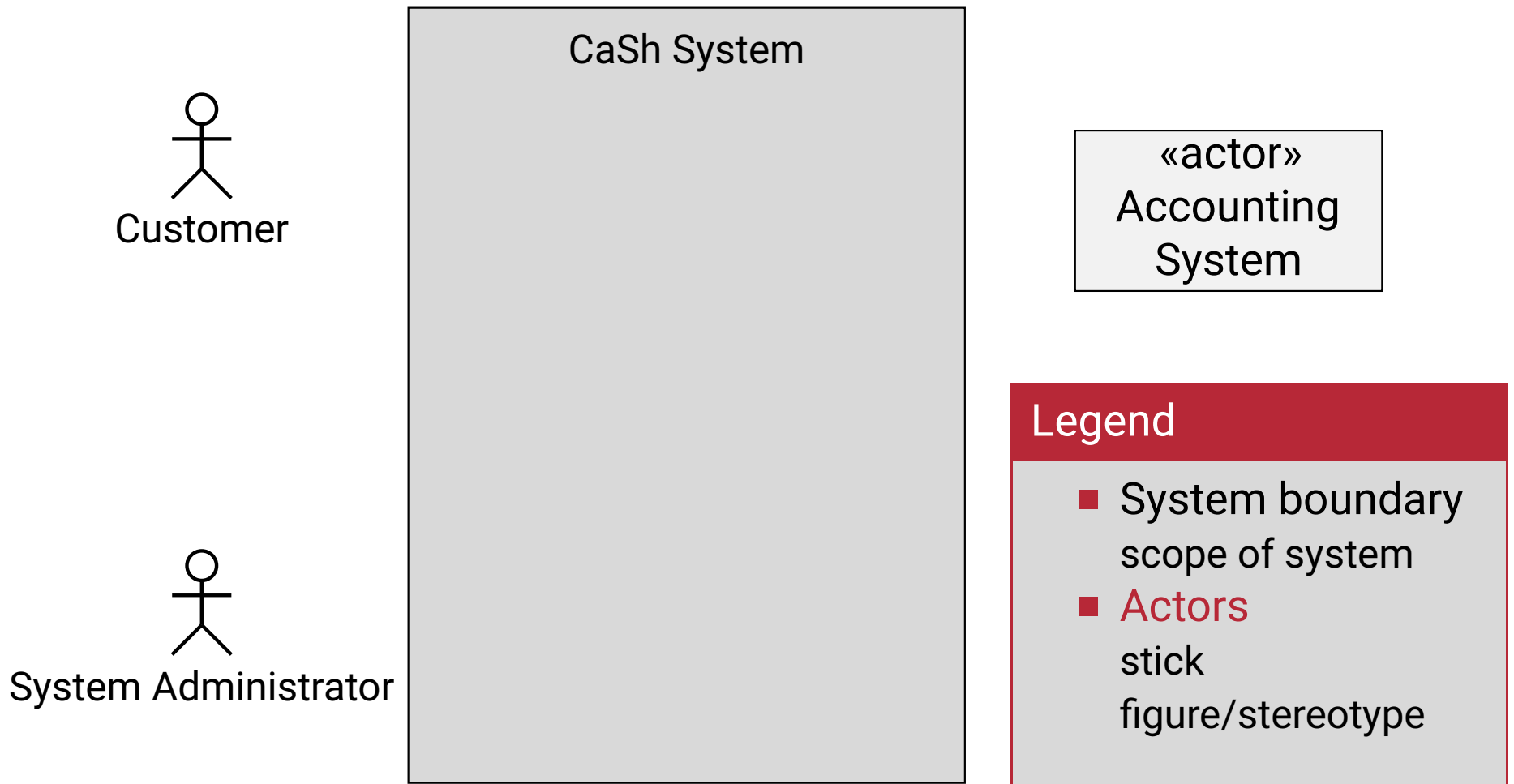


Legend

- System boundary
  scope of system

Software Engineering Group

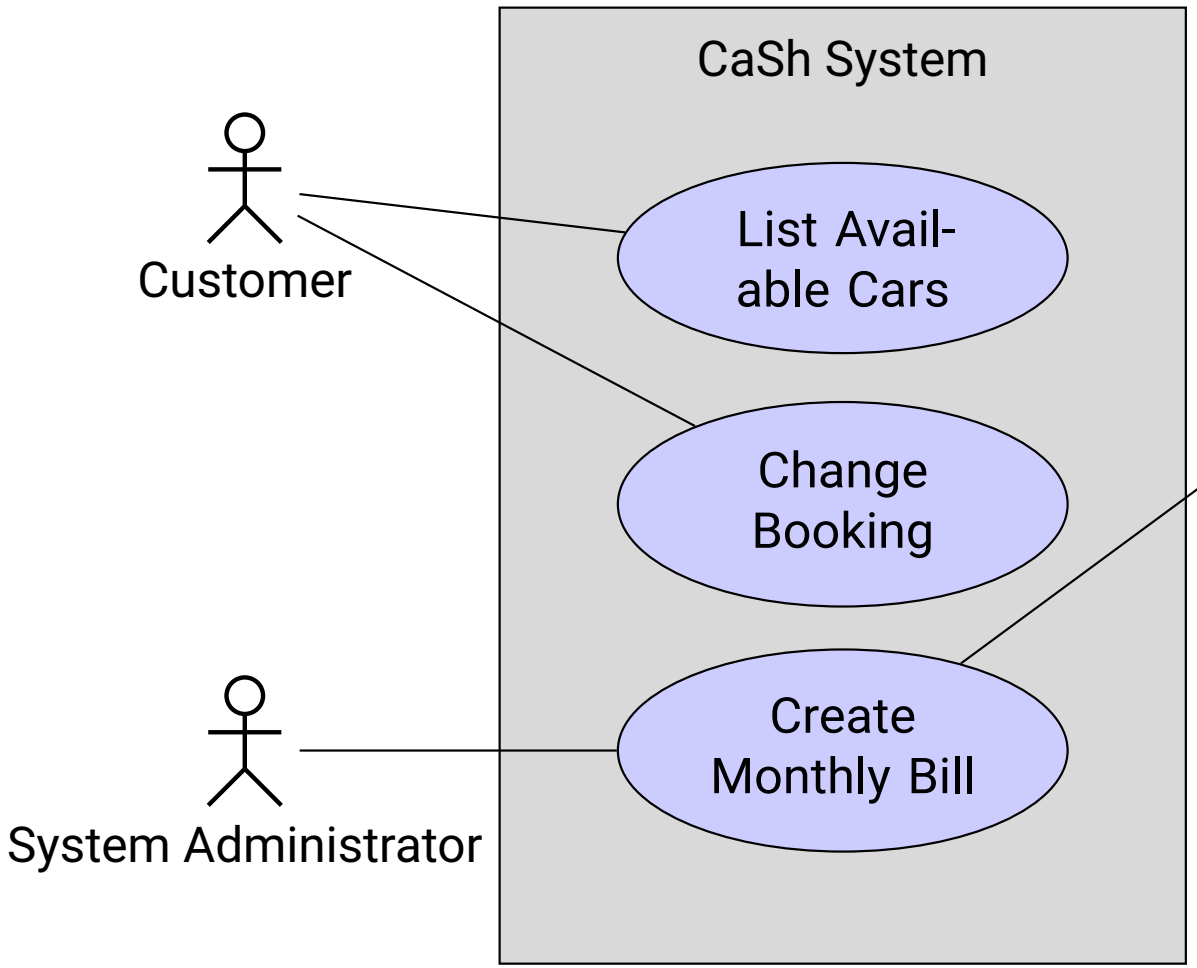
UML Use Case Diagram



Legend

- System boundary
  scope of system
- Actors
  stick
  figure/stereotype

UML Use Case Diagram



Legend

- System boundary
  scope of system
- Actors
  stick
  figure/stereotype
- Use cases

Remarks On UML Use Case Diagrams

TECHNISCHE UNIVERSITÄT DARMSTADT

- UML use case diagram notation is intentionally minimalist

Remarks On UML Use Case Diagrams

TECHNISCHE UNIVERSITÄT DARMSTADT

- UML use case diagram notation is intentionally minimalist
- UML use case diagrams are organizational method to improve
  - communication and comprehension of use cases
  - to reduce text duplication

Organizing use cases into relationships
has no impact on the behavior or requirements of a system

Remarks On UML Use Case Diagrams

TECHNISCHE UNIVERSITÄT DARMSTADT

- UML use case diagram notation is intentionally minimalist
- UML use case diagrams are organizational method to improve
  - communication and comprehension of use cases
  - to reduce text duplication

Organizing use cases into relationships
has no impact on the behavior or requirements of a system

- Use case diagrams provide a black-box view on a software system

Software Engineering Group

Remarks On UML Use Case Diagrams

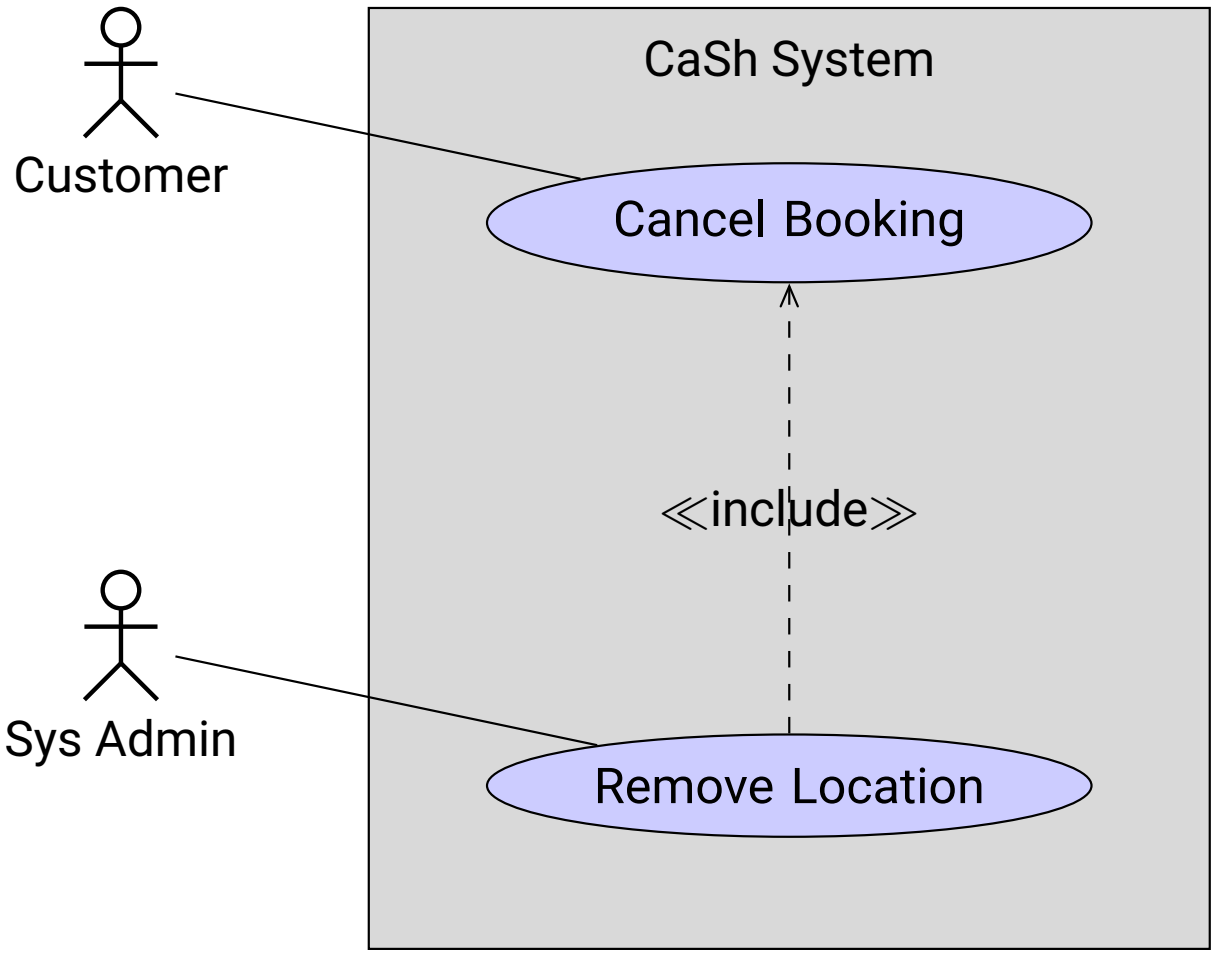
TECHNISCHE UNIVERSITÄT DARMSTADT

- UML use case diagram notation is intentionally minimalist
- UML use case diagrams are organizational method to improve
  - communication and comprehension of use cases
  - to reduce text duplication

Organizing use cases into relationships
has no impact on the behavior or requirements of a system

- Use case diagrams provide a black-box view on a software system
- Use case diagrams are helpful during early phase of use case analysis
  - Unsuitable to represent fully dressed use cases

UML Use Case Notation I
The «include» Relation



Purpose of the «include» Relation

TECHNISCHE UNIVERSITÄT DARMSTADT

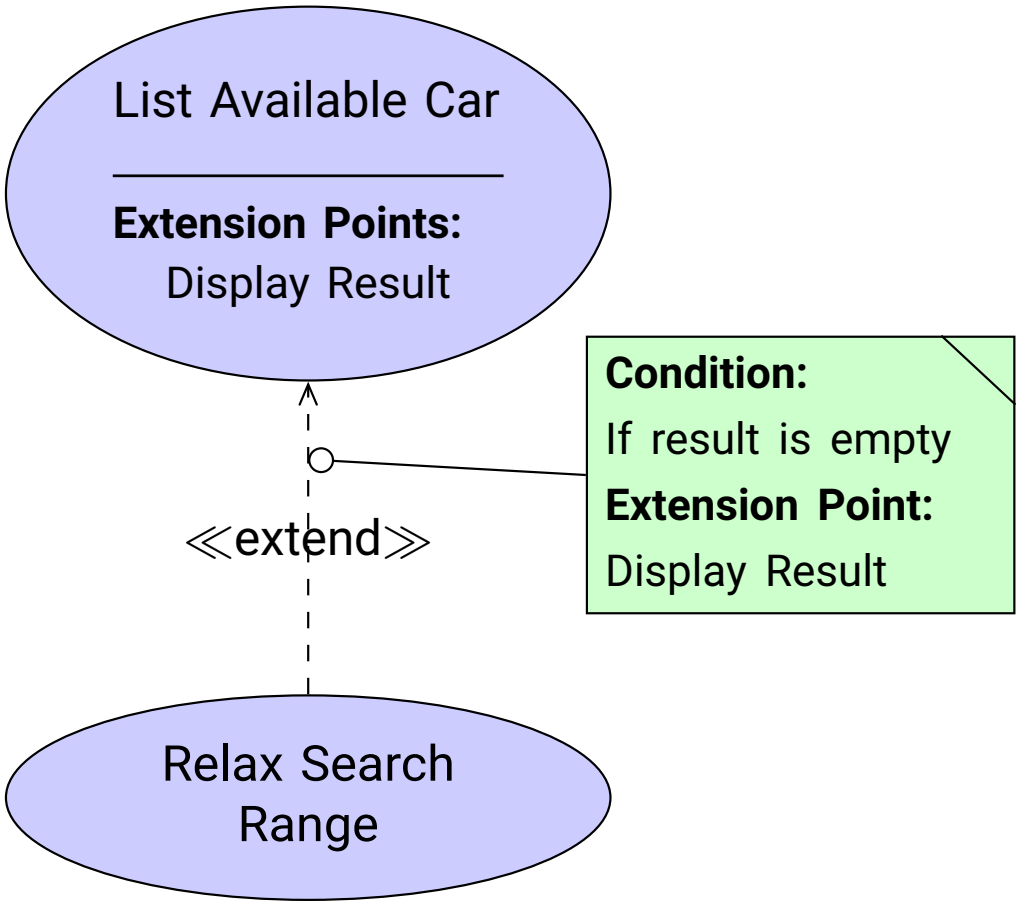
Include Relation

Factor out common behaviour across several use cases into its own sub-function use case and indicate inclusion

Facilitates decomposition of large use cases and enables reuse

Included use cases are always executed

UML Use Case Notation II: The «extend» Relation



Extend relation

Used to describe where and under what condition an extending or additional use case extends the behavior of a base use case

Remarks on the «extend» Relation

TECHNISCHE UNIVERSITÄT DARMSTADT

Original Motivation

Extension use cases allow to “modularly” extend existing use cases
But modularity violated by need for explicit use case extension points!

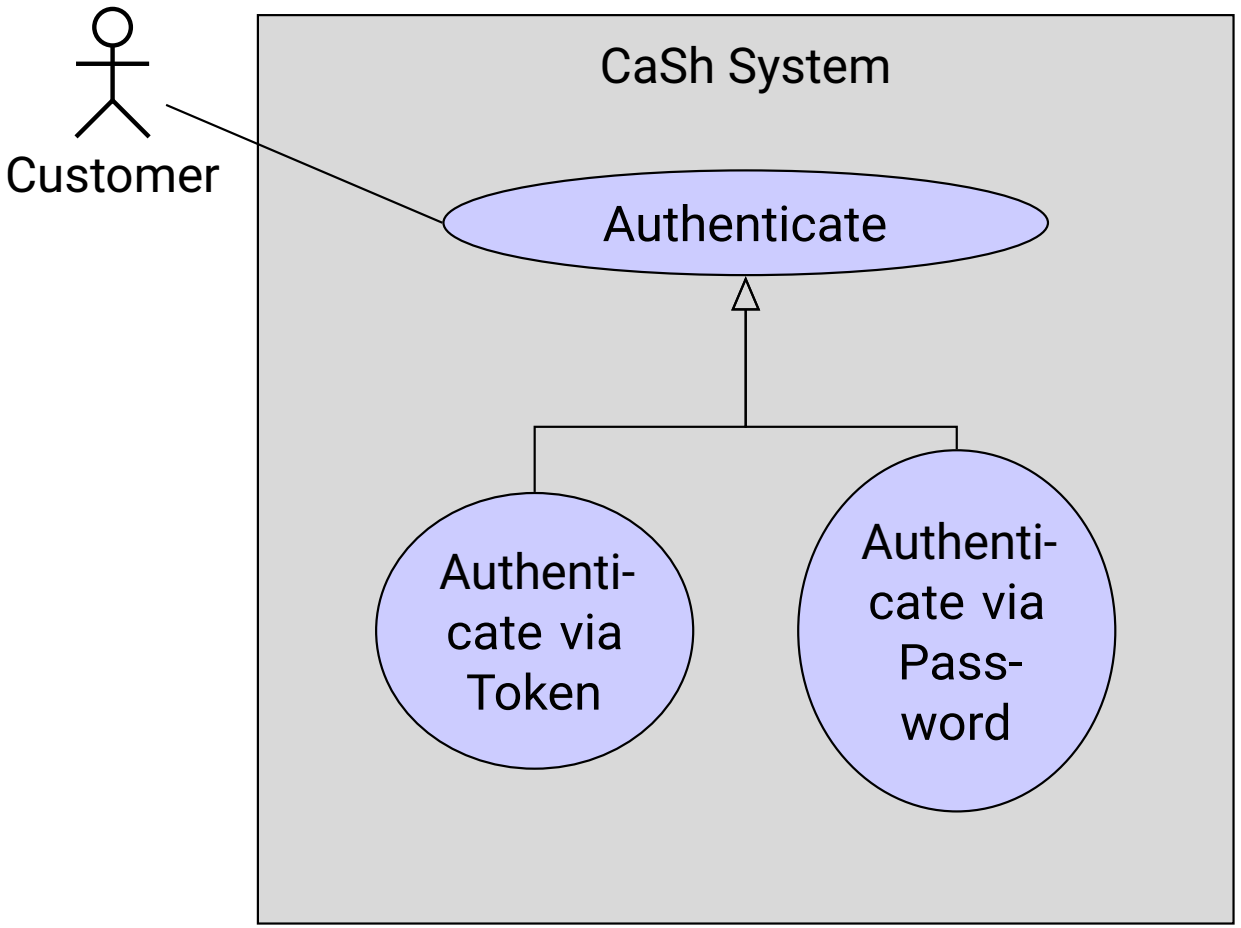
- Most extensions in fully dressed use cases
  do not qualify as separate use case (no Elementary Business Process)
- In general (not only UML) extension use cases
  expose internals of base use case

  (in UML even enforced by extension points)

The «extend» relation in use case diagrams must be well justified

Software Engineering Group

UML Use Case Notation III: Inheritance (Merely for Completeness, Do Not Use!)



Use Case Inheritance

Inheriting use case overrides behaviour of the inherited use case:

It replaces one or more of the courses of action of the inherited use case

UML Use Case Notation IV: Summary

TECHNISCHE UNIVERSITÄT DARMSTADT

UML Use Case Diagram notation is intentionally simple

Basics	System boundary, scope, actors, use cases, relations
«include» Relation	Useful to factor out common behavior
«extend» Relation	breaks modularity, must be justified
Use Cases Inheritance	Clutter, premature design decision — avoid

Use Cases: Final Remarks

TECHNISCHE UNIVERSITÄT DARMSTADT

Biggest danger: Abusing use case diagrams for system design

Use Cases: Final Remarks

TECHNISCHE UNIVERSITÄT DARMSTADT

Biggest danger: Abusing use case diagrams for system design

- Use cases are in the problem space
  They document behavioral requirements
- User involvement essential
- Adequate granularity is important:
  Elementary business process, not just a step
- UML use diagrams most helpful in early use case analysis

Use Cases: Final Remarks

TECHNISCHE UNIVERSITÄT DARMSTADT

Biggest danger: Abusing use case diagrams for system design

- Use cases are in the problem space
  They document behavioral requirements
- User involvement essential
- Adequate granularity is important:
  Elementary business process, not just a step
- UML use diagrams most helpful in early use case analysis
  - Stay simple, black box preferred
  - Use template, not diagrams, for fully dressed use case

Literature

TECHNISCHE UNIVERSITÄT DARMSTADT

- Alistair Cockburn, Writing Effective Use Cases, Addison-Wesley, 2016

  (Available as hardcover in TUDa ULB (English edition))
- Ian Sommerville, Software Engineering, 10th edition, Chapter 4.4, Pearson Education, 2015

  TUDa ULB eBook (German edition)
- Ulrike Hammerschall and Gerd Beneken, Software Requirements, Chapter 5, Pearson, 2013

  TUDa ULB eBook (German edition)