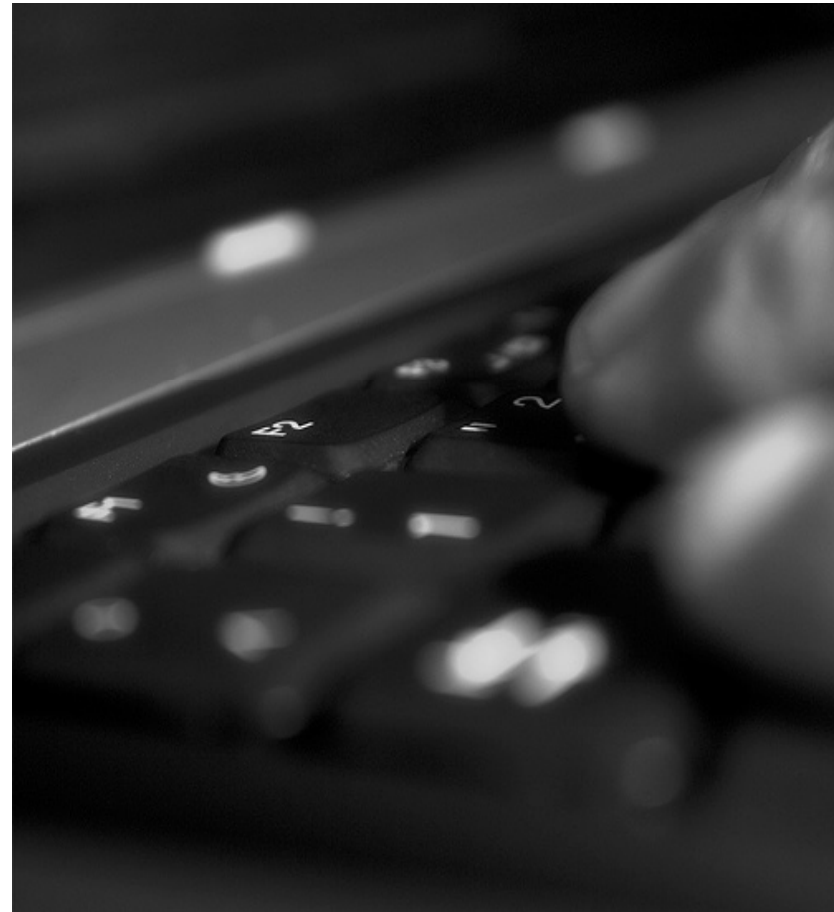


Exercise 3  
Business Informatics 2 (PWIN)

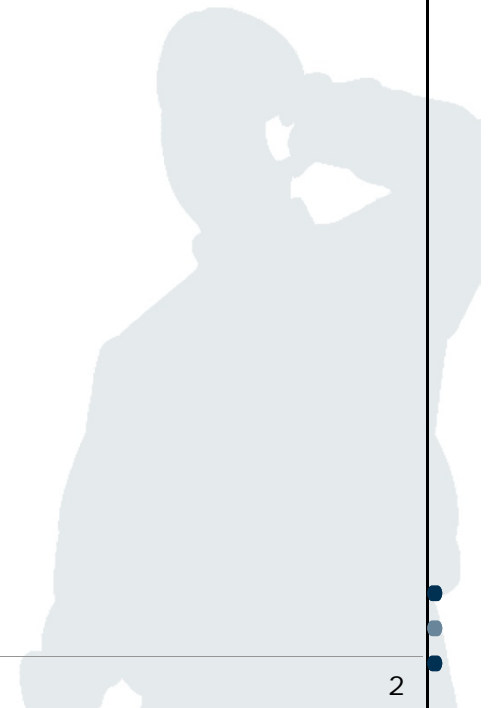
IT-Project Management &  
IC Software Development I+II  
SS 2011

Dr. Andreas Albers  
[www.m-chair.net](http://www.m-chair.net)

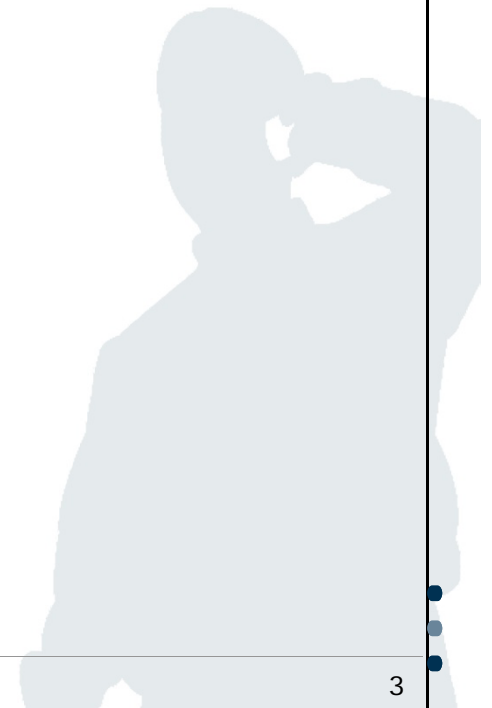


Jenser (Flickr.com)

- Exercise 1: IT-Projects
- Exercise 2: IT-Project Management
- Exercise 3: Quality Management
- Exercise 4: Modelling
- Exercise 5: Software Development Process Models
- Exercise 6: Object Orientation (OO)
- Exercise 7: UML

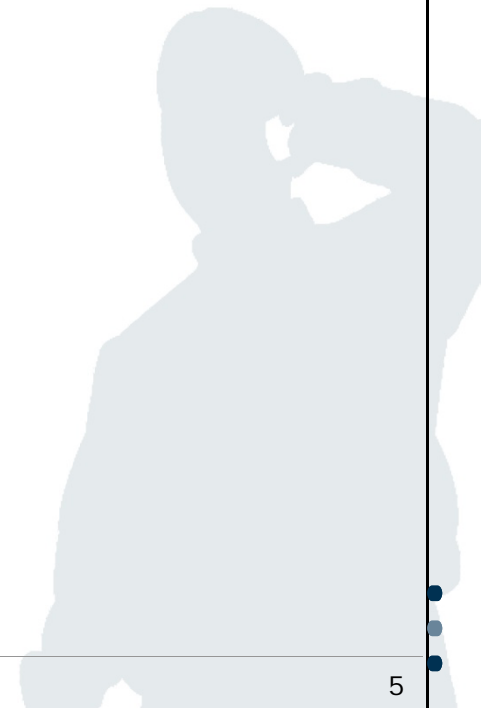


- Explain the term “Critical Success Factors” in relation to IT-Projects and give three examples for the InstantONS Service.



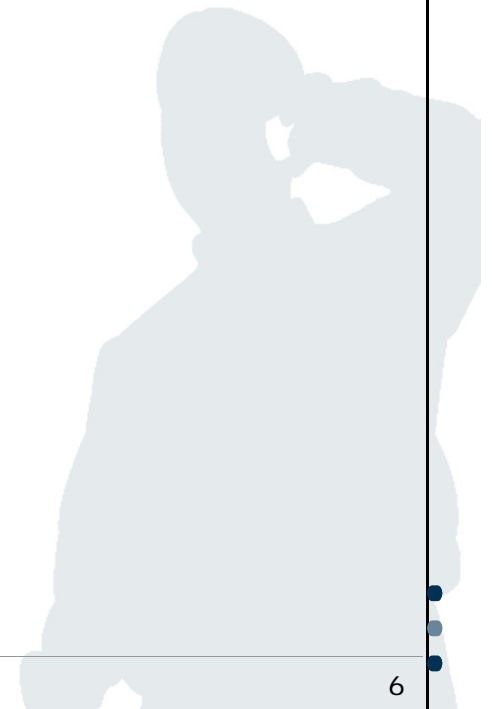
- Critical Success Factors
  - Decisively influence the success of a project
- Three Critical Success Factors for the InstantONS® System
  - Usability
  - Costs
  - Data protection
- Further questions
  - How can Critical Success Factors be identified?
  - How can Critical Success Factors be measurable and controlled?
  - What impact do Critical Success Factors have on the outcome of a project?

- Exercise 1: IT Projects
- Exercise 2: Project Management
- Exercise 3: Quality Management
- Exercise 4: Modelling
- Exercise 5: Software Development Process Models
- Exercise 6: Object Orientation (OO)
- Exercise 7: UML



## Exercise 2: Project Management

- What are the “SMART” project objectives? Explain them at the example of the InstantONS<sup>®</sup> Service.



Specific:

- Improve quality of the personal profile matching
- Reduce calculation time for personal profile matching down to 3ms

Measurable:

- Number of matching requests per performed date
- Calculation time for the personal profiles matching

Attainable:

- E.g. five matching requests per performed date
- E.g. Reduce calculation time for the personal profiles matching down to 3ms

Relevant:

- Personal profile matching constitutes the core functionality of the InstantONS system

Time-bound:

- Objective completion until end of 2012

- Exercise 1: IT-Projects
- Exercise 2: IT-Project Management
- Exercise 3: Quality Management
- Exercise 4: Modelling
- Exercise 5: Software Development Process Models
- Exercise 6: Object Orientation (OO)
- Exercise 7: UML

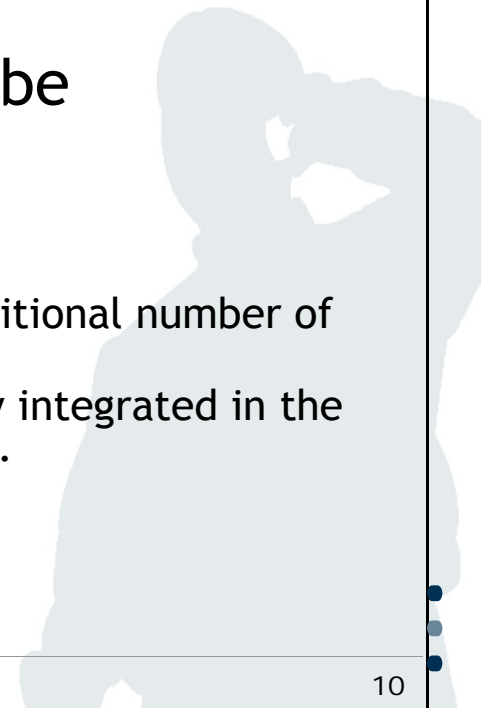


## Exercise 3: Quality Management

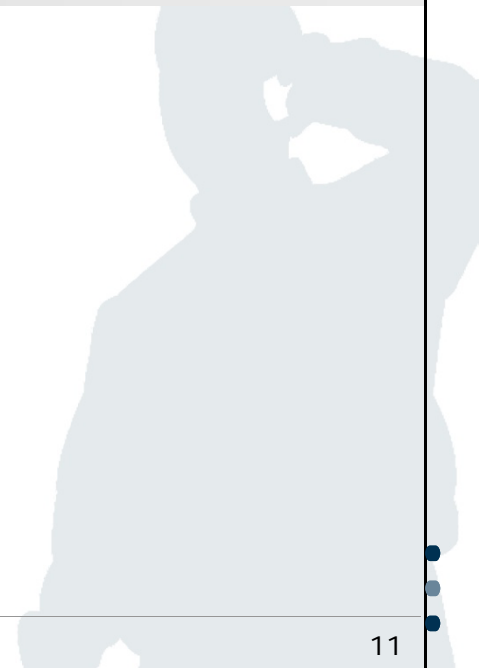
- *Scalability* is an attribute for software quality. What does scalability in relation to the InstantONS Service mean? What other software quality attributes exist?



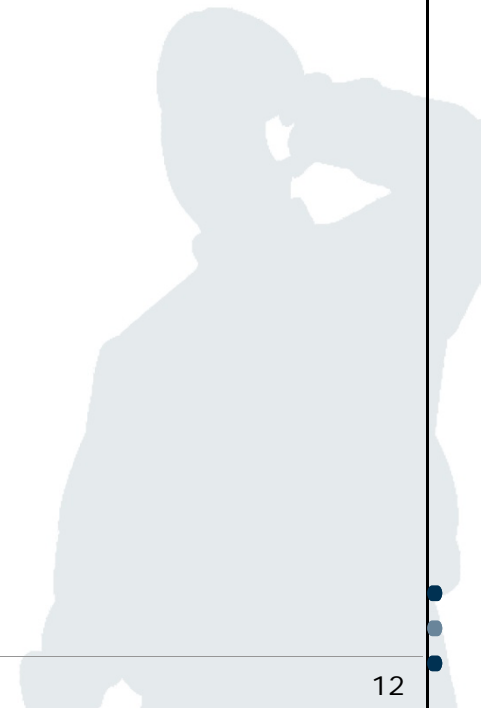
- Scalability
  - The software is easily adaptable and extendable to new needs.
- Scalability of InstantONS Service
  - Assumption: The InstantONS Service will be successful and gains a lot of new users.
  - Scalability:
    - The system has to be easily adaptable to handle the additional number of users without breaking down.
    - For instance, by adding new servers, which can be easily integrated in the system without having to adapt the InstantONS software.



- Exercise 1: IT-Projects
- Exercise 2: IT-Project Management
- Exercise 3: Quality Management
- Exercise 4: Model Development
- Exercise 5: Software Development Process Models
- Exercise 6: Object Orientation (OO)
- Exercise 7: UML



- Explain the difference between the development of a model for an IT-Project and an IT-Product.




- Two modelling types within IT-Projects

- IT-Project-Process

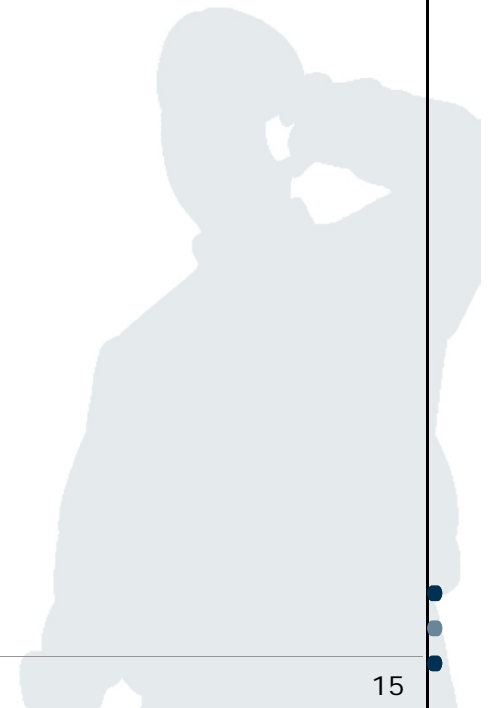
- Models describing the process of an IT-Project (e.g. starting with the project idea, project planning, implementation, project finalisation) -> Process models
    - E.g. sequential model, spiral model

- IT-Product

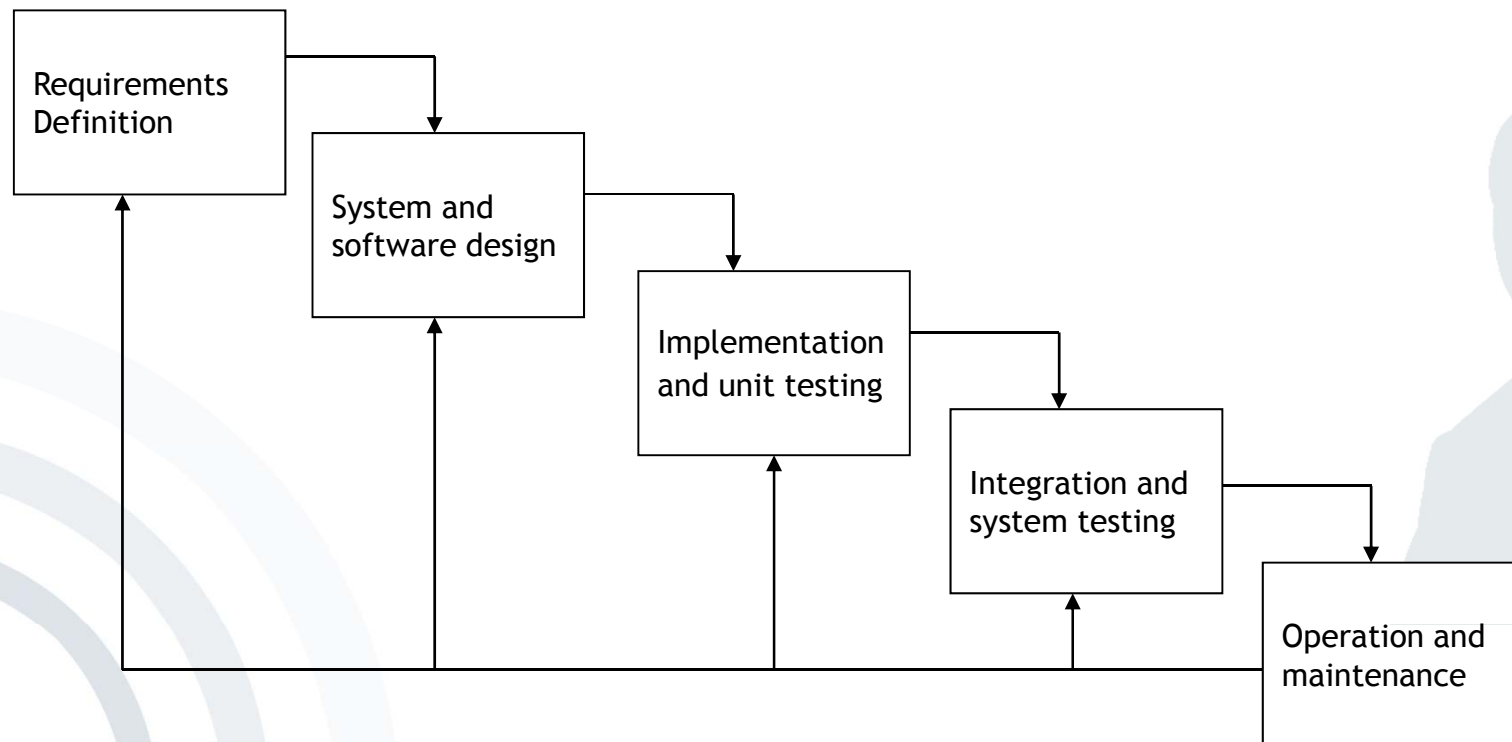
- Models for the architecture of a software (e.g. process models, data models, object models)
    - E.g. ERM, UML, ...

- Exercise 1: IT-Projects
  - Exercise 2: IT-Project Management
  - Exercise 3: Quality Management
  - Exercise 4: Model Development
  - Exercise 5: Software Development Process Models
  - Exercise 6: Object Orientation (OO)
  - Exercise 7: UML
- 

- Name and describe three software development process models and select one of them for the InstantONS Service. Motivate your choice.

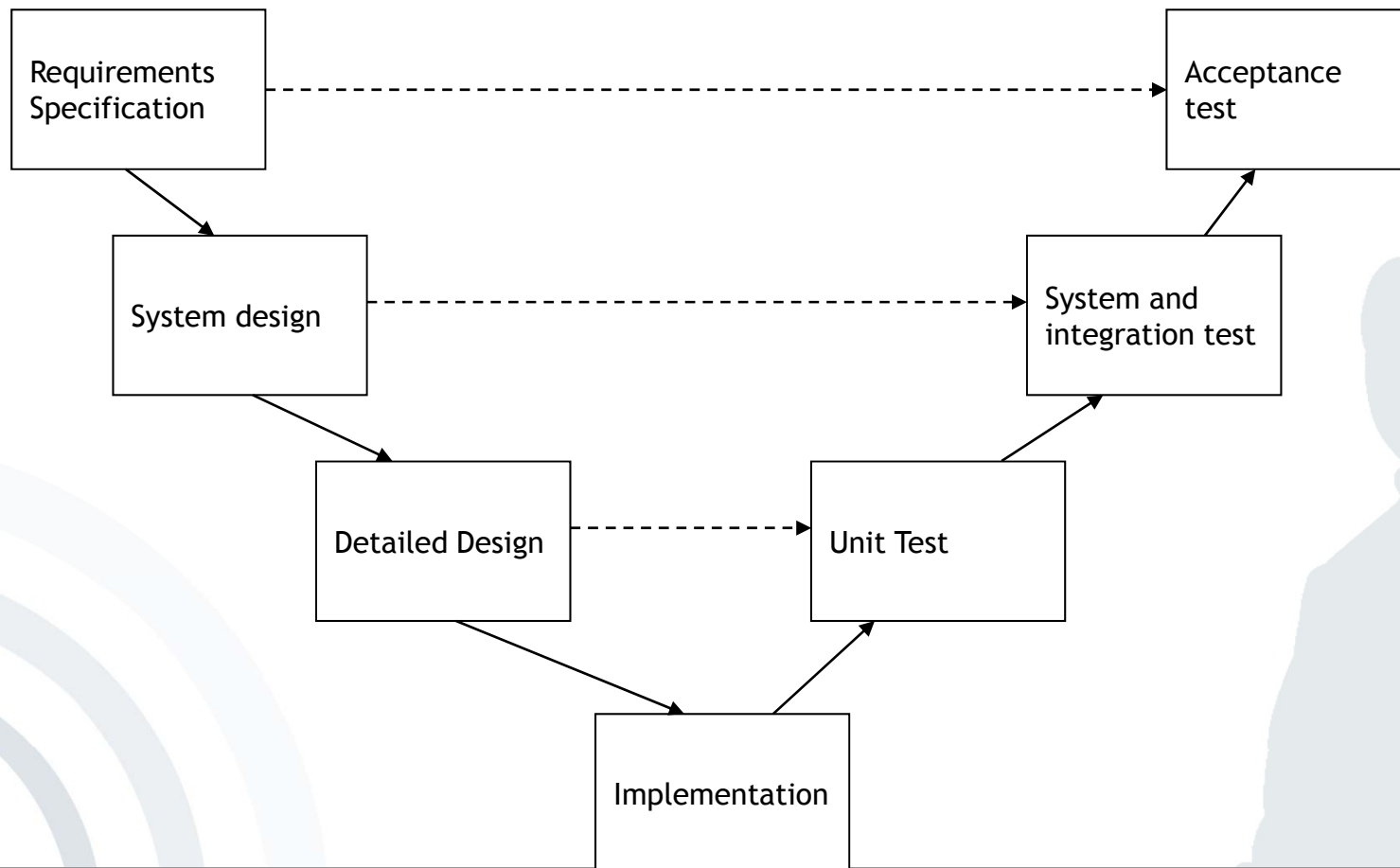


- The waterfall model
  - First described by Royce in 1970
- There seem to be at least as many versions as there are authorities - perhaps more



- One or more documents are produced after each phase and “signed off”.
- Points to note:
  - “Water does not flow up”.
    - it is difficult to change artifact produced in the previous phase.
  - This model should be used only when the requirements are well understood.
  - Reflects engineering practice.
  - Simple management model.

- Horizontal lines denote the information flow between activities at the same abstraction level.

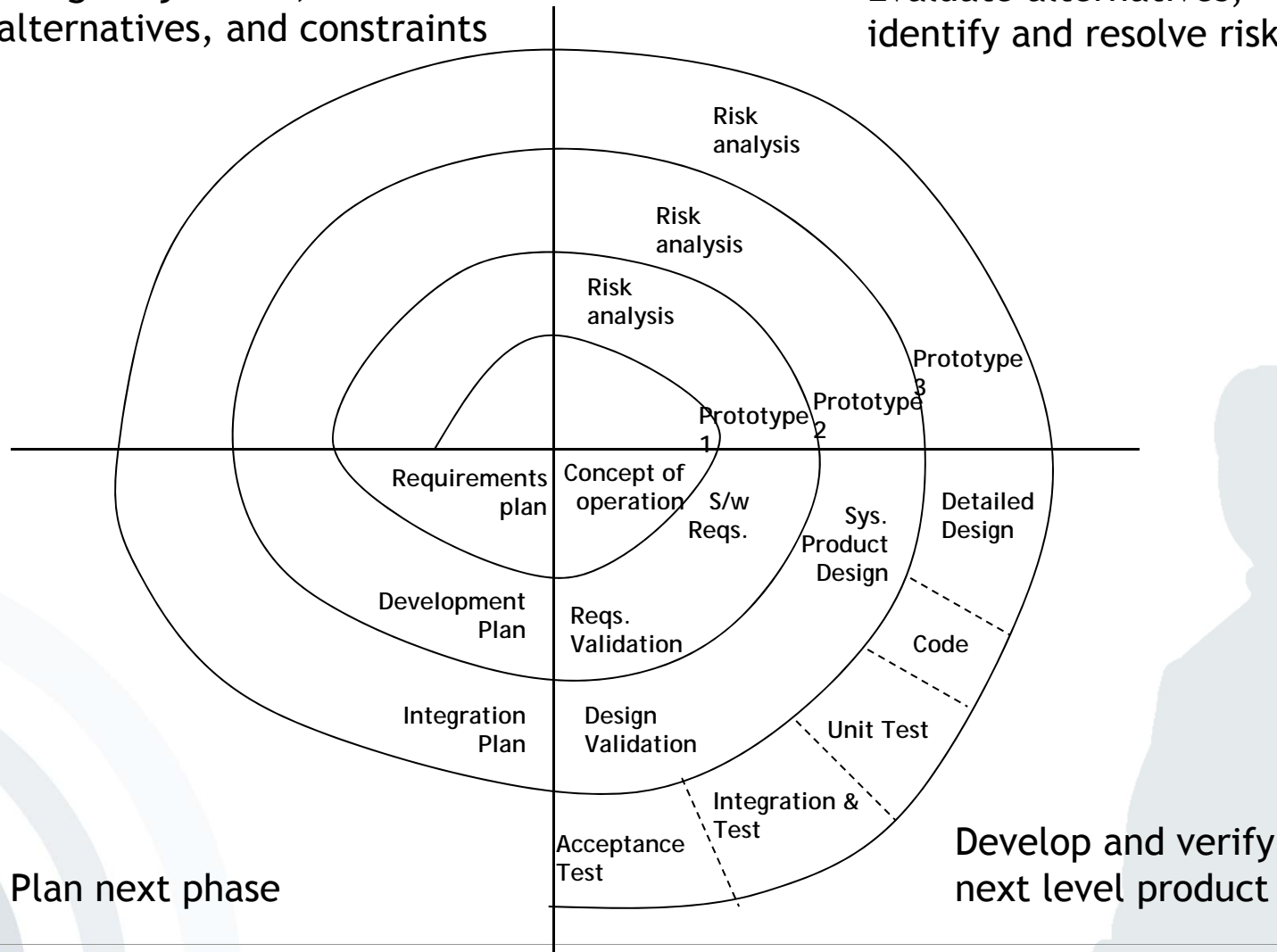


- Similar to pure waterfall model but makes explicit the dependency between development and verification activities.
- The left half of the V represents *development* and the right half system *validation*.
- Note the requirements specification includes requirements elicitation and analysis.

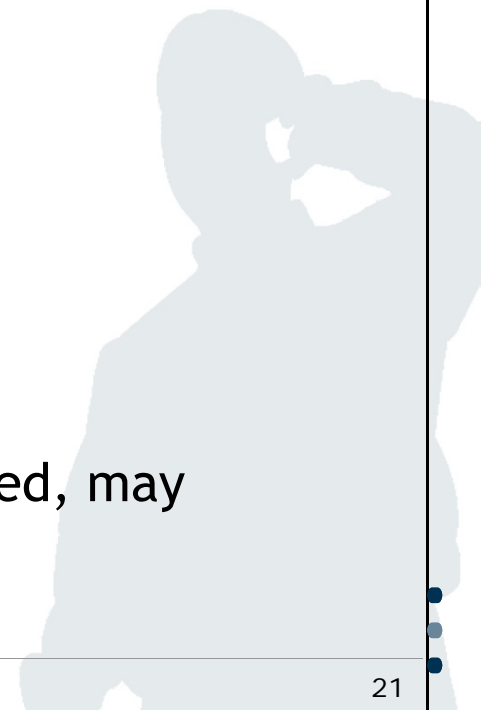
# Exercise 5: Solution

Design objectives,  
alternatives, and constraints


Evaluate alternatives,  
identify and resolve risks



- Basic Idea
  - develop initial implementation, expose it to user, and refine it until an adequate system is produced.
- Two types:
  - *Exploratory*
  - *Throw-away prototyping*
- Advantages
  - model used when problem is not clearly defined.
- Disadvantages
  - Process not visible, systems are poorly constructed, may require special tools and techniques.

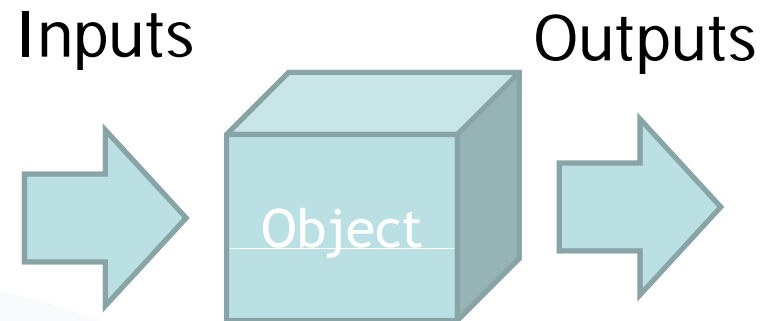


- Process model proposal for the InstantONS<sup>®</sup>  
Service: V-Model
- Motivation
  - The InstantONS<sup>®</sup> system is a complex system. The V-model was designed for complex systems.
  - The V-model makes explicit the dependency between development and validation and allows to jump back to earlier development phases.

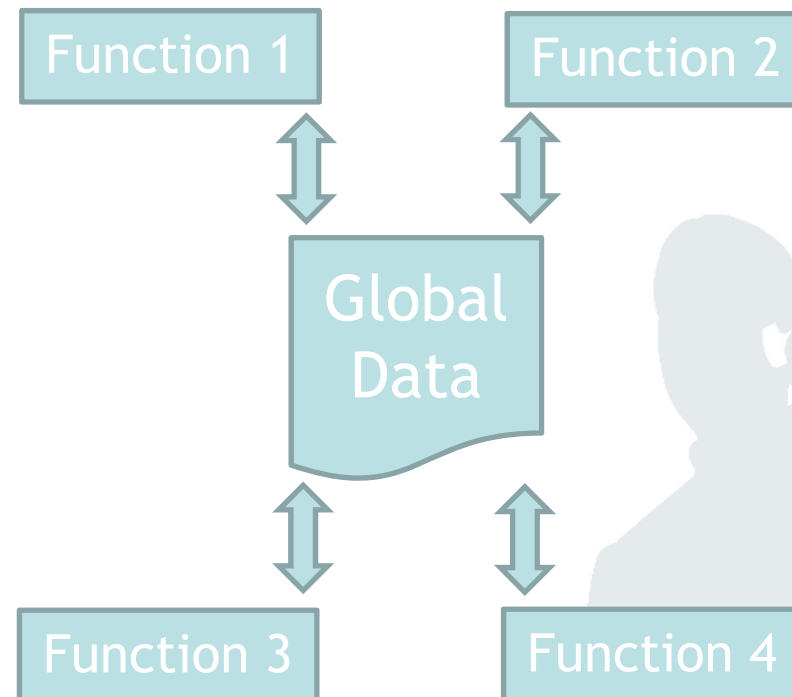
- Exercise 1: IT-Projects
  - Exercise 2: IT-Project Management
  - Exercise 3: Quality Management
  - Exercise 4: Model Development
  - Exercise 5: Software Development Process Models
  - Exercise 6: Object Orientation (OO)
  - Exercise 7: UML
- 

# Object Orientation vs. Procedural Paradigm

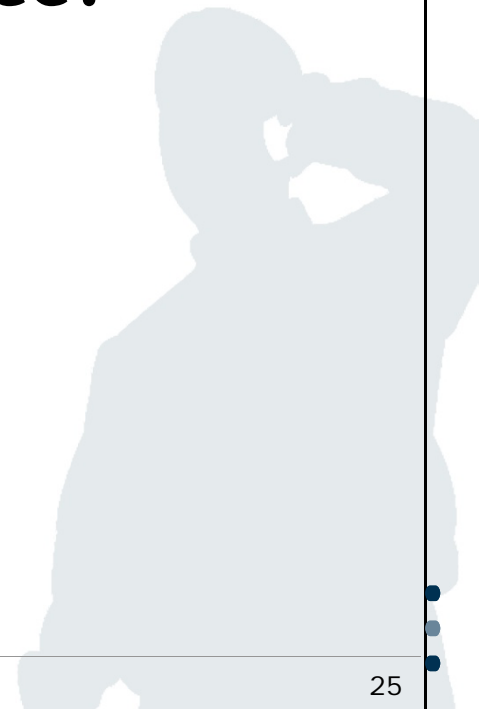
## Object Orientation



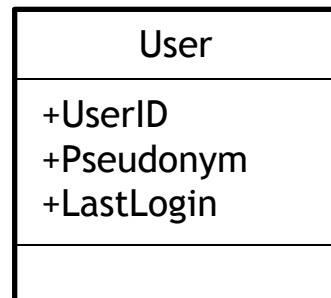
## Procedural Paradigm



- a) One central concept of OO constitutes “Encapsulation”. Explain the idea behind this concept at the example of a class “User” for the InstantONS Service.

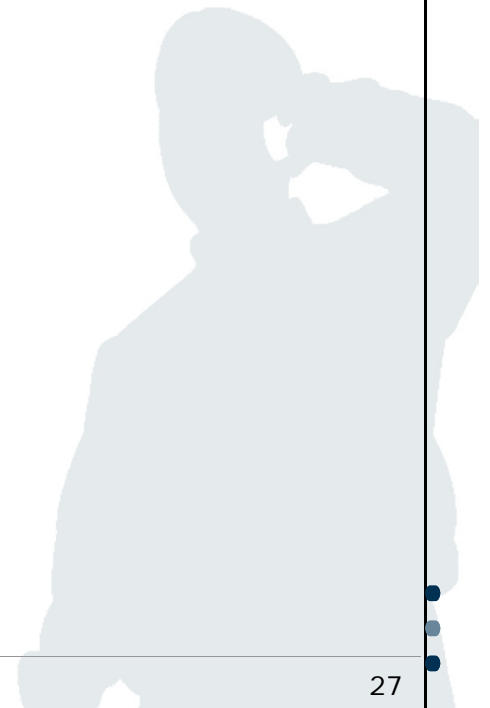
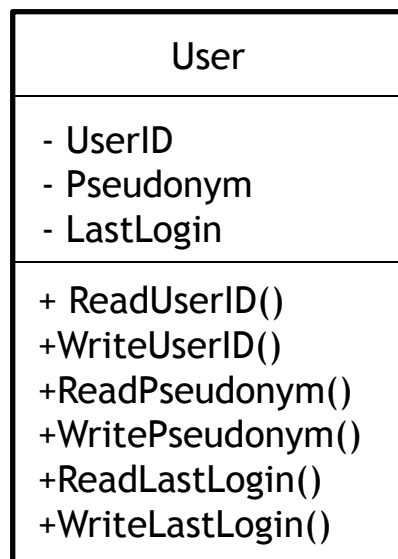


- Example class „User“ of the InstantONS<sup>®</sup> systems

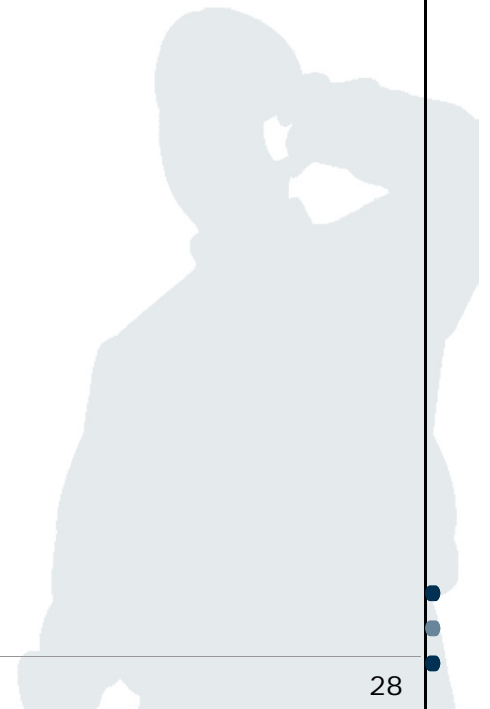


- Attributes of the user can be manipulated freely.

- Therefore Encapsulation is used. Public access to attributes of a class only via operations!
- Attributes are declared “private”.

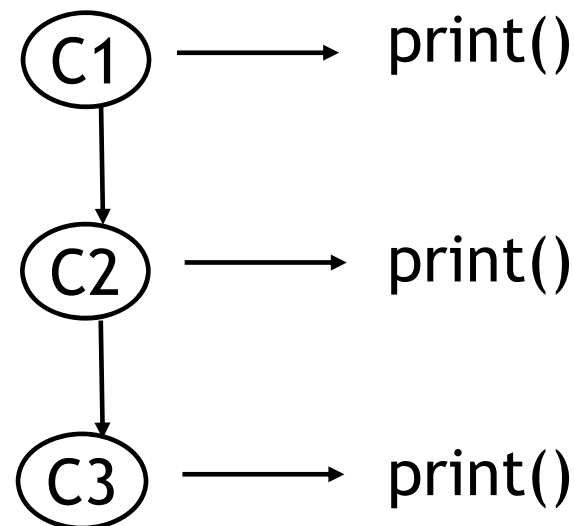


b) Give an example for “Polymorphism” in relation to the InstantONS Service.



### ■ Polymorphism

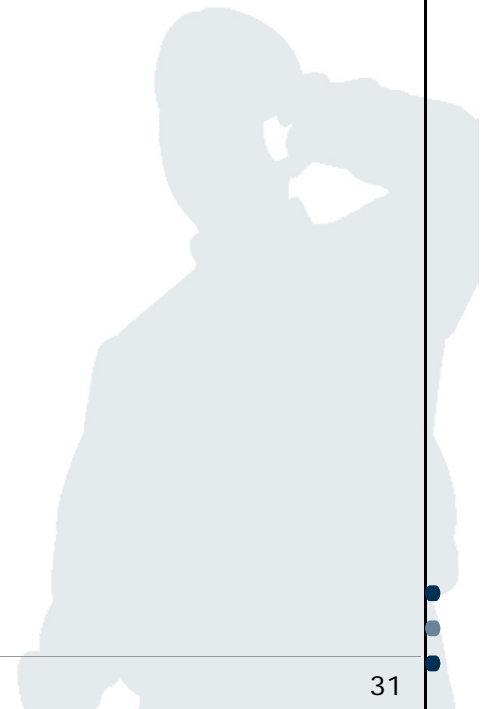
If a message is sent to objects of different classes, these objects return different results, as the called method can be implemented differently for each object.



### Print\_name()

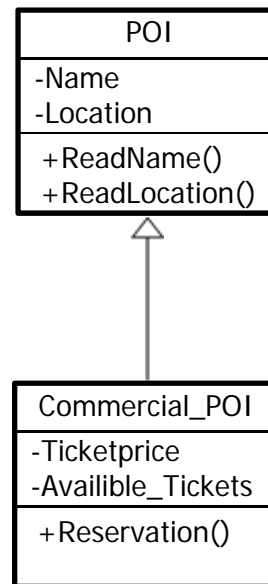
- Mobile operator->Print\_Name()
  - Returns the name of the mobile operator (e.g. Deutsche Telekom)
- Meeting point-> Print\_Name()
  - Returns the name of the meeting point (e.g. Bockenheimer Warte)
- User-> Print\_Name()
  - Returns the Pseudonym of the user (e.g. Harry4711)

c) What is idea of “Inheritance” for OO classes? Name one further OO concept.



### ■ Inheritance

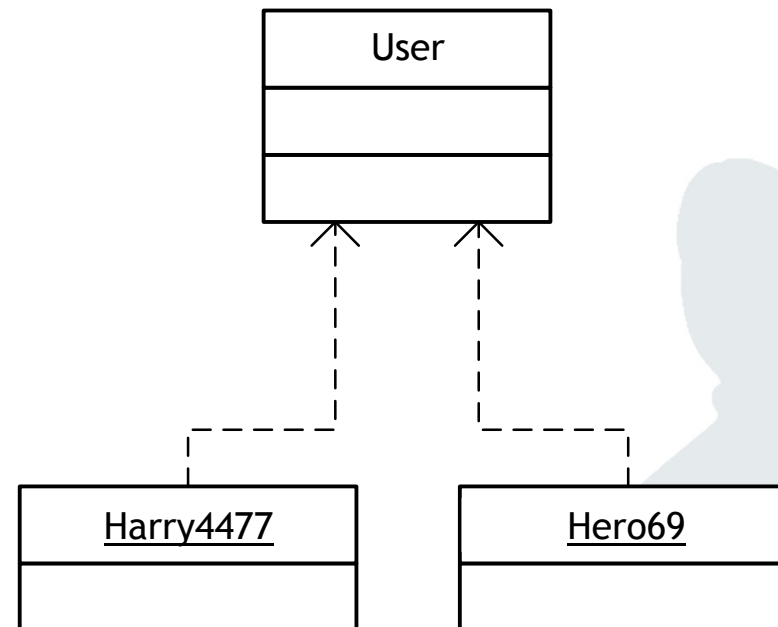
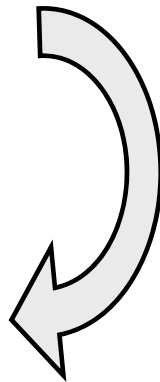
- Classes can inherit attributes or methods to other classes. The inheriting class is called super class or parent class. The new class is called sub class.




## Instantiation

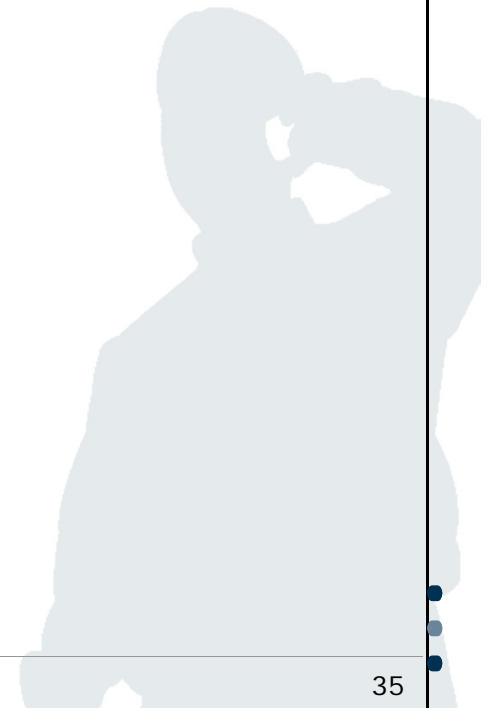
- Representation of the relation “class-object“
- An object is an instance of a class.

- Class
  - Attributes
  - Methods
- Object
  - Attribute values
  - Messages



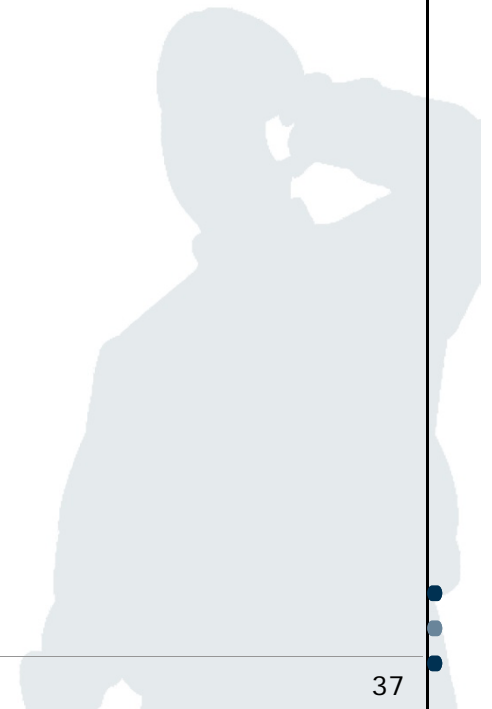
- Exercise 1: IT Projects
  - Exercise 2: Project Management
  - Exercise 3: Quality Management
  - Exercise 4: Modelling
  - Exercise 5: Software Development Process Models
  - Exercise 6: Object Orientation (OO)
  - Exercise 7: UML
- 

- a) What is the benefit of using UML for the modelling of Information Systems? What activities of software development does UML not support?



- Supports analysis and design of object-oriented software systems
- UML includes multiple Views on a system
  - Each View specifies and documents a system from a different perspective.
  - Each View is supported by one or more diagrams.
- UML is not a process model → UML does not define a process for creating UML models.

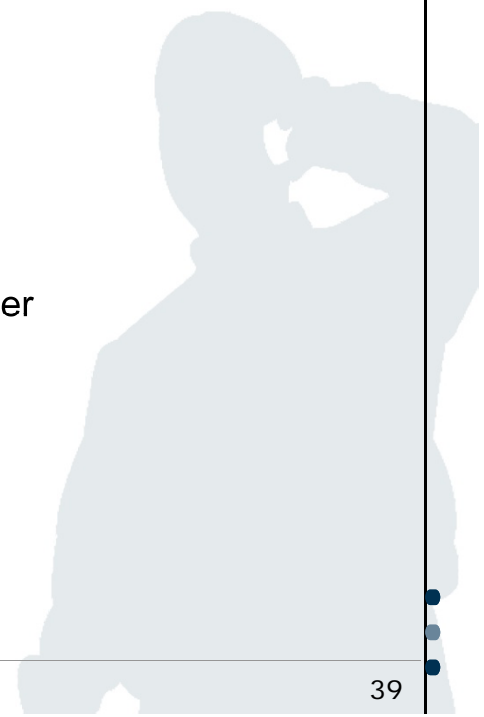
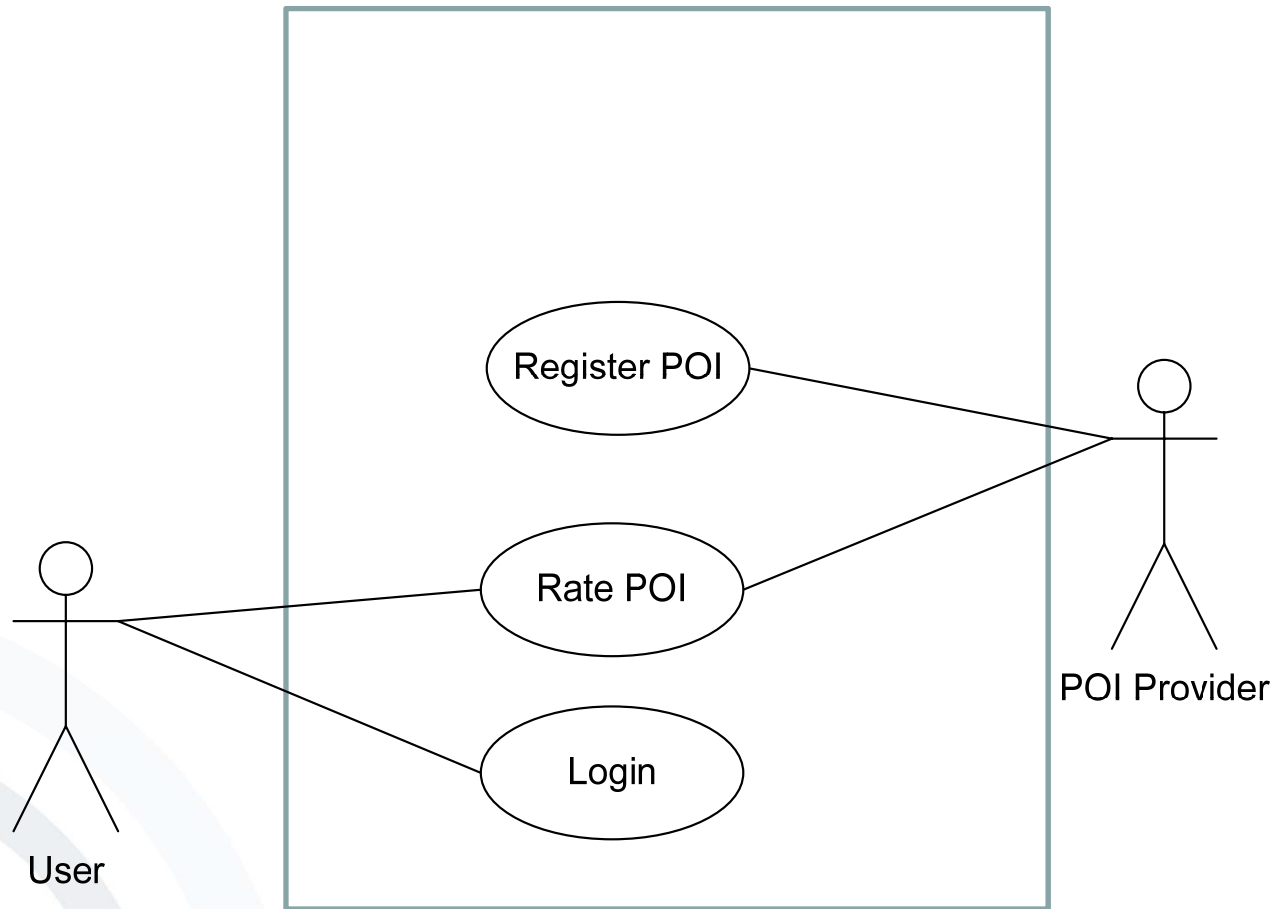
b) What are the differences between Use Case, Activity and Class diagrams?



### Use Case diagrams

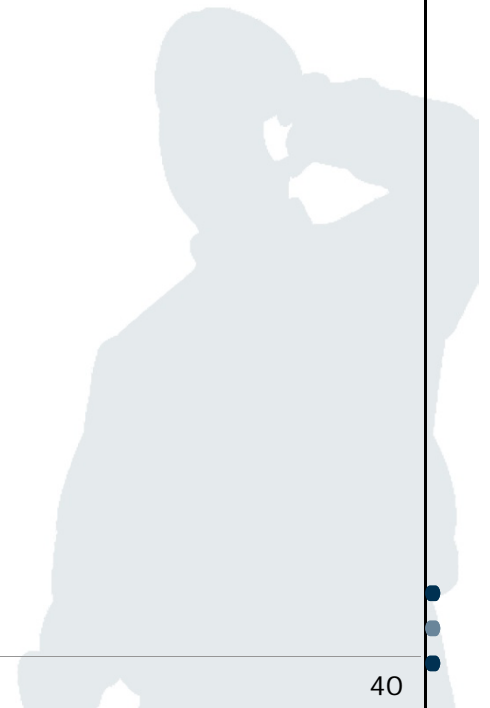
- Use cases describe the functionality, which a system has to provide
- The sum of all “Use cases” comprises the technical requirements of a system.
- Use cases define the interfaces between a user and the system
- Specification is developed together with the client/customer



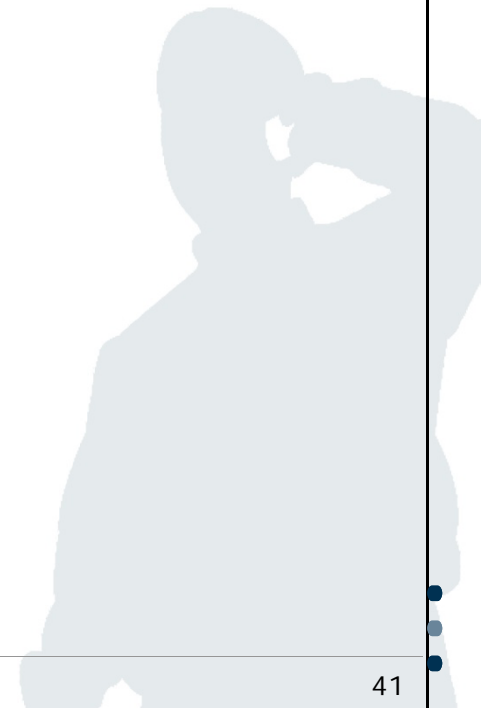


## Activity diagrams

- Activity diagrams are used to model workflows in a system.
- Central element: Activity  
An activity is an “action” within a process.
- Activities are structured by responsibilities.
- Different views
  - Conceptual View
    - e.g. business processes
  - Implementation View
    - e.g. methods of objects

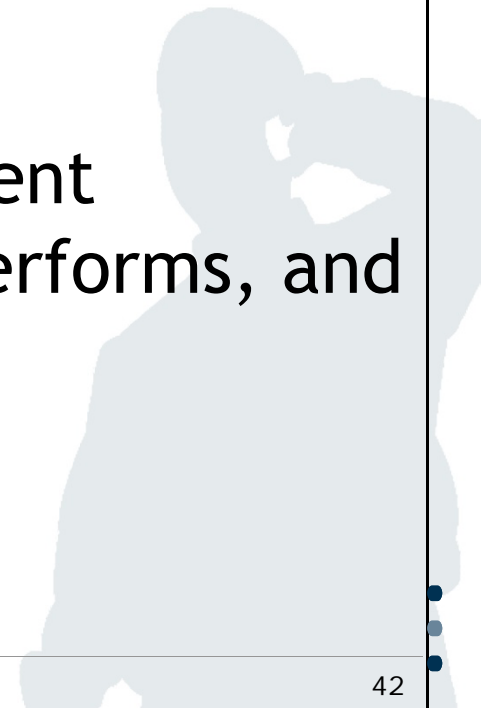


- Class diagrams
  - Representation of the static structure of a software system
  - Description of logical relations between structural elements
  - No activity or control logic
- Object diagrams
  - Instances of a class diagram
  - „Snapshot“ of a system during runtime

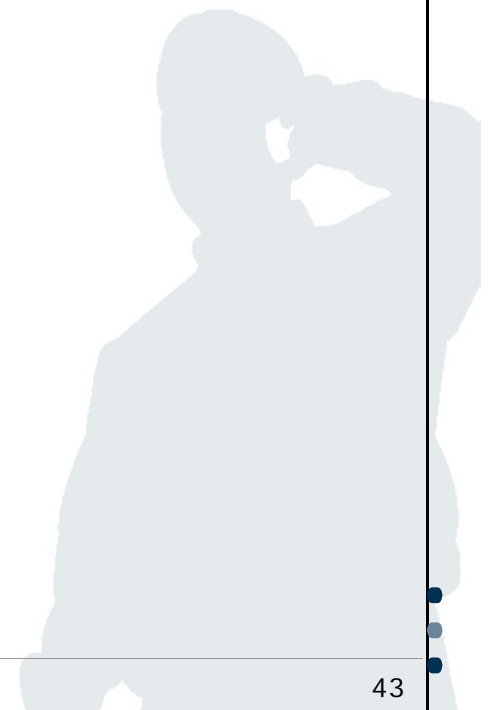


### Static vs. Dynamic models

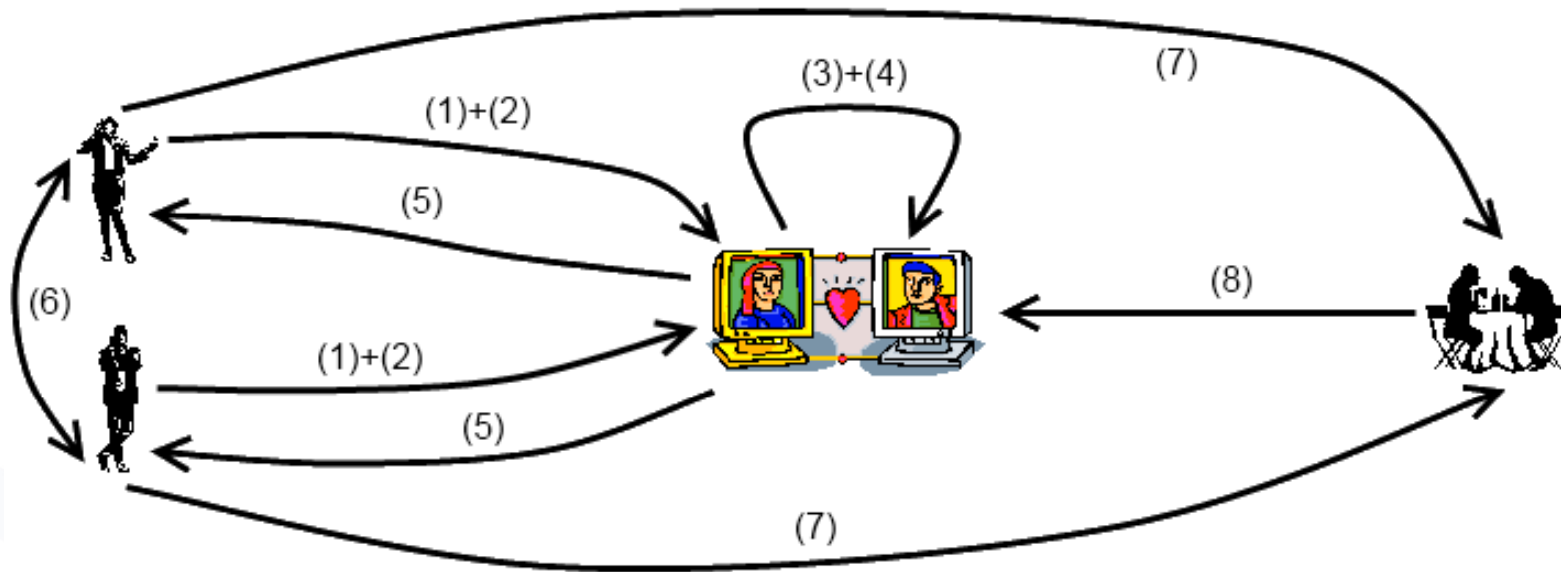
- A central aspect of the static models is to describe the structure of a system and the relations between its components.
- Dynamic models describe the different activities, process steps a system performs, and the interaction of its components.



- c) Develop a Use Case and Activity diagram for the InstantONS<sup>®</sup> Service on based on Figure 1 of the InstantONS scenario.

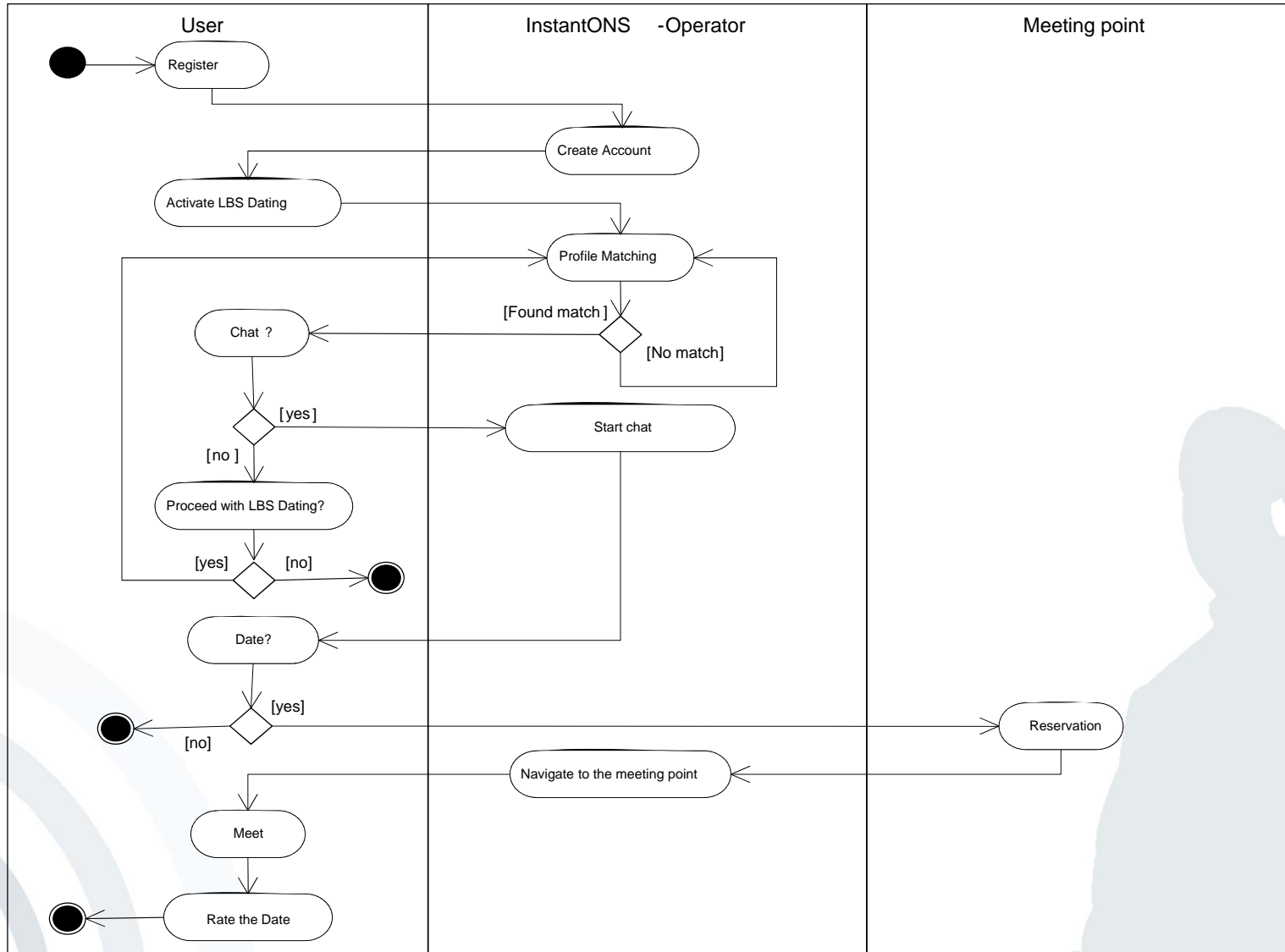


# Exercise 7c): Solution



## Use Case diagram



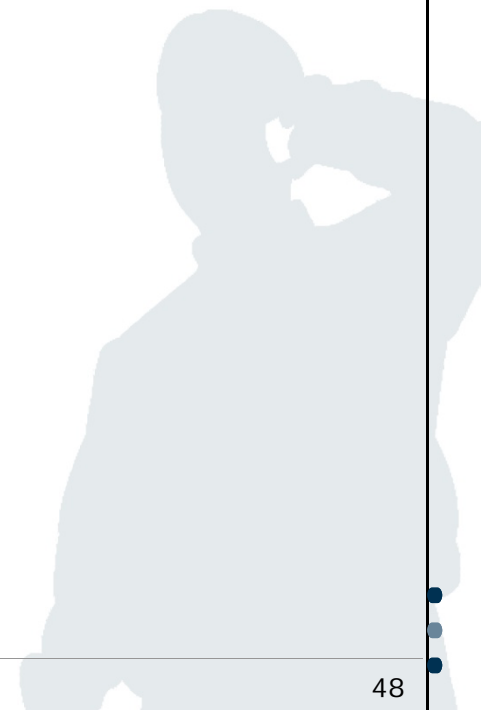


- d) A meeting point in the InstantONS scenario has the following characteristics:
- It is described by its type (e.g. hotel, cinema, etc.),
  - it has a location specified by its address (i.e. street, postal code, etc.)
  - it has specific opening hours,
  - it has a counter on how many visitors are currently at the place and,
  - it has an indicator if the place is full and no more visitors are allowed to enter

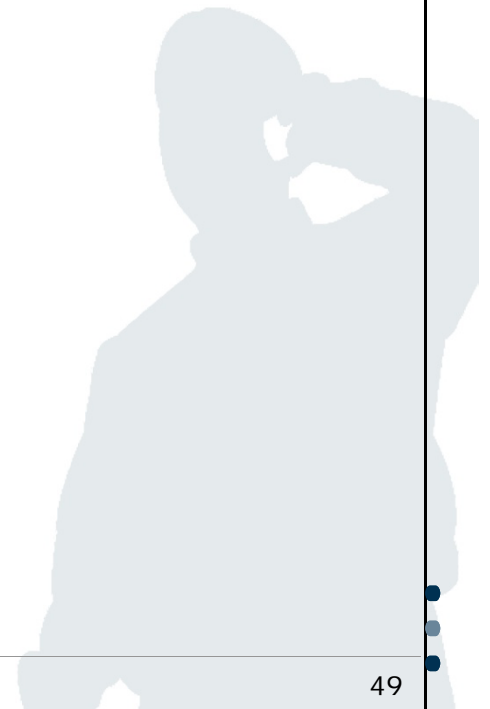
Specify the UML class “Meeting\_Point” corresponding to the description above – and especially with the OO concept “Encapsulation” in mind.

## Exercise 7d): Solution

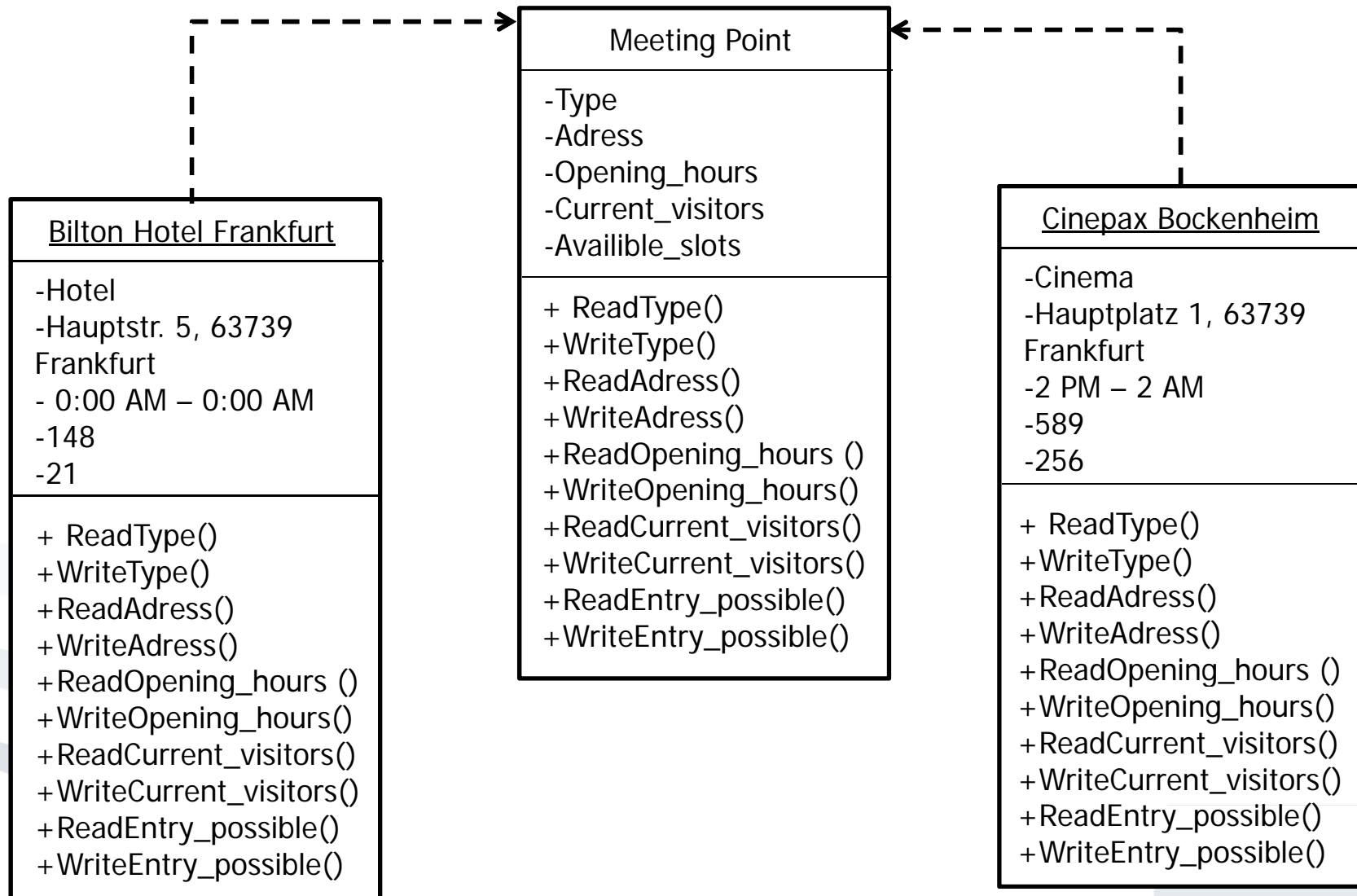
Meeting Point
-Type -Adress -Opening_hours -Current_visitors -Availible_slots
+ ReadType() + WriteType() + ReadAdress() + WriteAdress() + ReadOpening_hours () + WriteOpening_hours() + ReadCurrent_visitors() + WriteCurrent_visitors() + ReadEntry_possible() + WriteEntry_possible()



e) Develop two example *instances* of the “Meeting\_Point” class as specified in d).



# Exercise 7e): Solution



# Open Questions?

A light gray silhouette of a person is positioned on the right side of the slide. The person is shown from the waist up, facing left, with their right hand raised to their chin in a classic 'thinking' or 'pondering' pose.