MUNI
FI

lasaris LAB OF SOFTWARE ARCHITECTURES AND INFORMATION SYSTEMS

Analytical Class Diagram

PB007 Software Engineering I

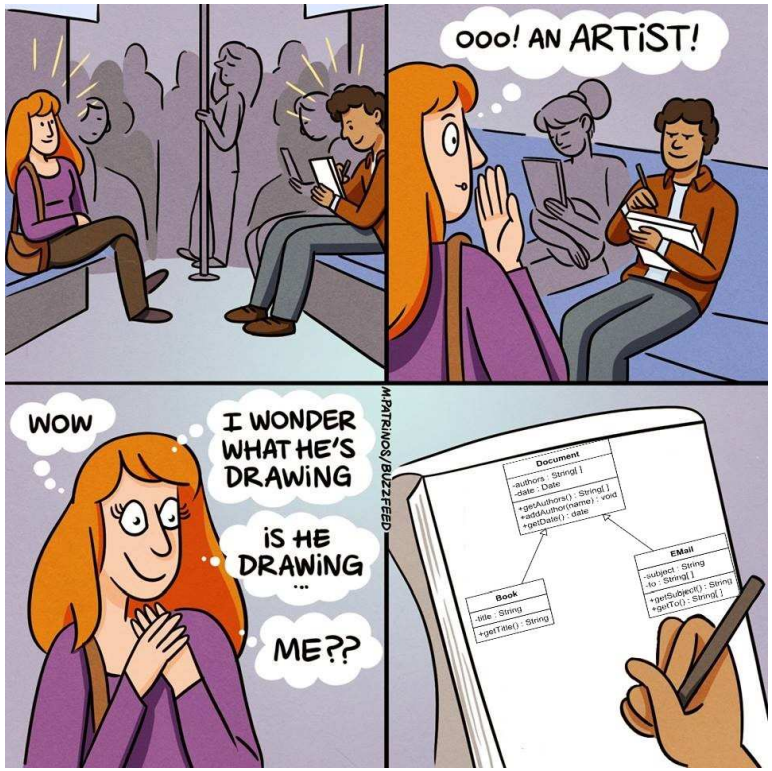
Lukáš Daubner
daubner@mail.muni.cz

Class Diagram

In general

- Static view

- Modelling of:
  - Classes
  - Attributes
  - Operations
  - Relationships



MUNI FI

Analytical Class Diagram

- Depicts **concepts**, abstractions, not pieces of code
  - Relationships
  - Attributes
- It helps us to grasp the domain, make a sense of it
- Specify terminology, relationships, dependances, …
- ADVANCED: Patterns (Analysis Patterns)
  - Reusable models – solution to a concrete repeating problems
  - See this book: **Martin Fowler: Analysis Patterns - Reusable Object Models**

Analytical Class Diagram

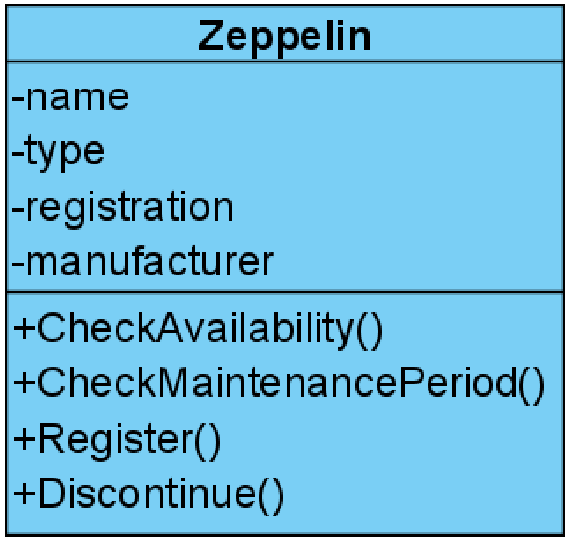
- Do not delve into implementation details. ~~Forget about programming~~
  - Types
  - Constructors
  - Boilerplate methods and classes
  - Properties (getters, setters)
  - Language-specific constructs
  - Etc.

MUNI FI

Analytical Class Diagram

How should it look like?

- Representative name
- Small number of responsibilities (operations)
  - ~3-5
- It is not isolated – A part of a system
- Low coupling – No spaghetti classes
- High cohesion
  - Operations have **MUCH** in common
  - Zeppelin class has operations related to its operation:
    - CheckAvailability()
    - CheckMaintenacePeriod()
    - Discontinue()
    - Register()



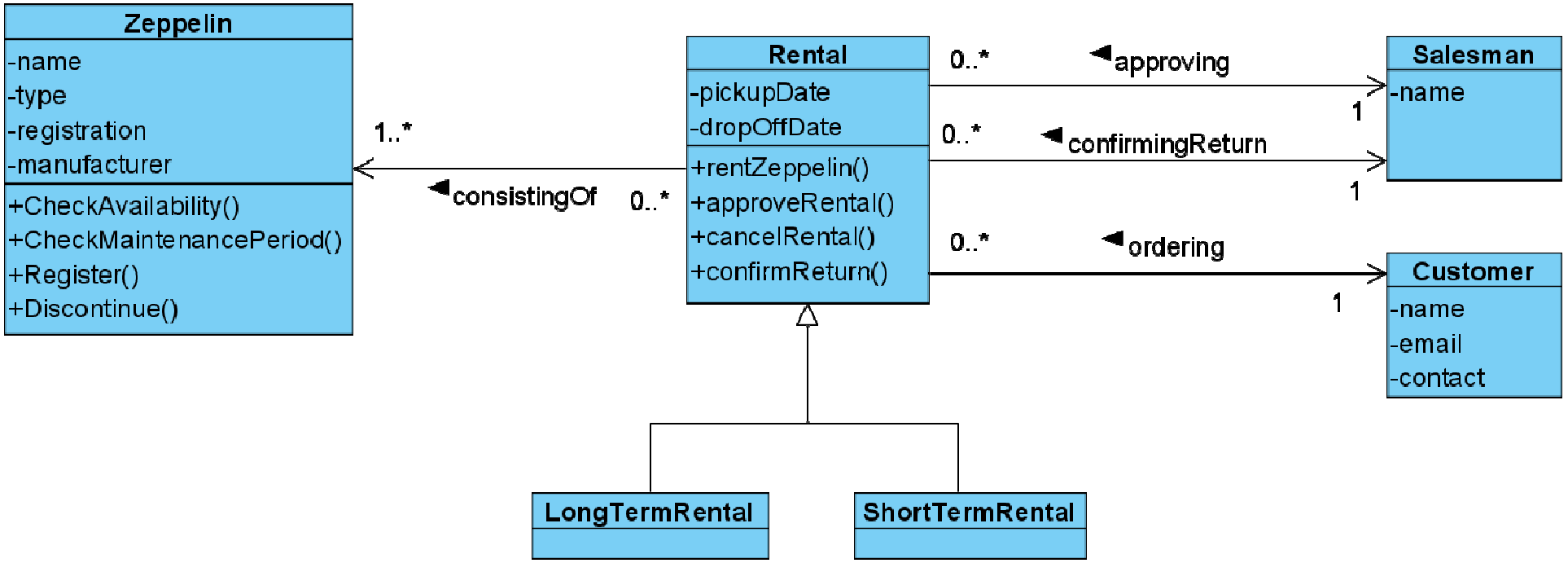
MUNI FI

Analytical Class Diagram

Do's and Dont's

- Be carefull about
  - Lots of small classes
  - Few big classes
  - Service/managers/builders/boilerplate classes
    Implementational details, not important for the concept
  - Complex inheritance hierarchy
  - Functoids
    Classes representing a function or procedure
- Try not to think so much about how would you code it
  - We are not there yet
- No interfaces
  - Again, implementational details

Analytical Class Diagram – Example



MUNI FI

How to find analytical classes

For example…

- Textual analysis
  - Use specification, use cases, or any other material available to you
  - Nouns (Podstaná jména) are often classes or attributes
  - Verbs (Slovesa) are often relationships or operations
  - Watch out for “hidden” classes – only implied in the text
- Brainstorming
  - Records candidate classes on sticky notes
  - Write down their responsibilities
  - Search for collaborations between them

Relationships between classes

- Association
  - Semantic relationship – from domain
  - It implies an attribute in one (or both) of the classes – usually
  - Long-term relationship
  - It contains:
    - Name, or name or roles
    - Multiplicity
    - Navigability

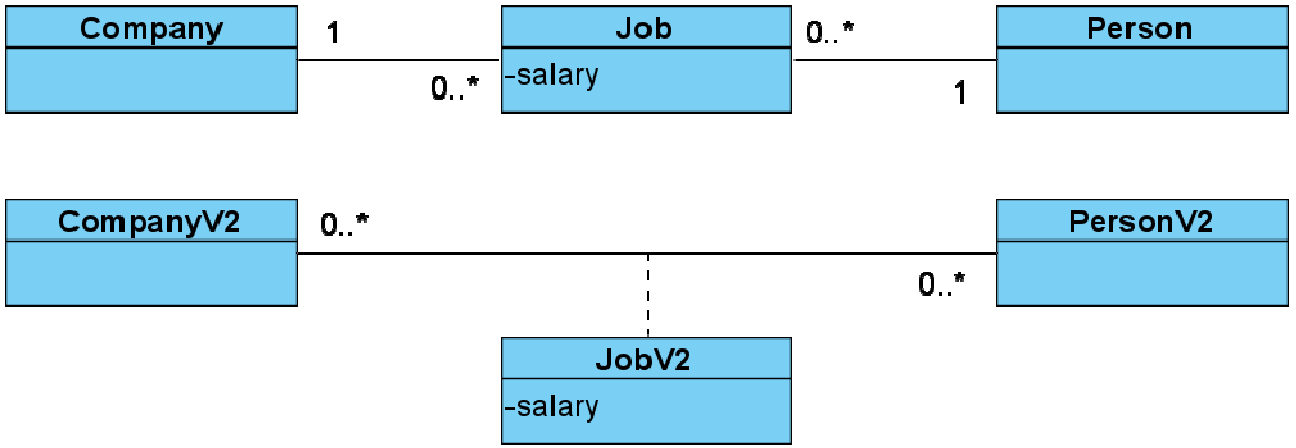
- Multiplicity
  - How many “partners” can the class have (1:1, 1:N, M:N)
- Navigability
  - Can we effectively “get” from one class to another

MUNI FI

Relationships between classes

- Association M:N
  - All fine in analytical class diagram, it is decomposed later in design class diagram
- But, if the relationship is complex, you need to decompose it now
  - Or use association class, there is at **most one relationship** between two instances

These two diagrams **are not equivalent**



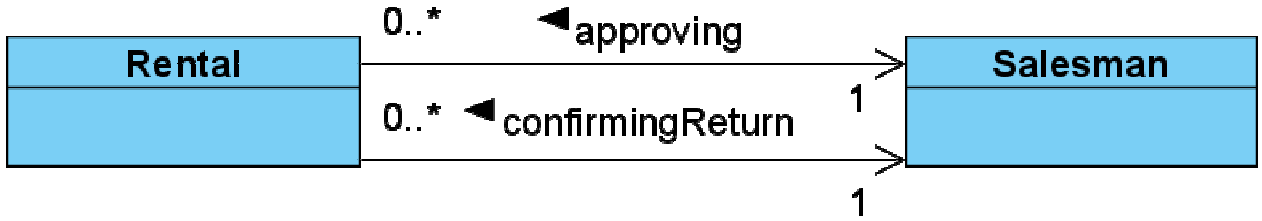
Person p can have multiple Jobs in a Company c

Person p can have only one Job in a Company c

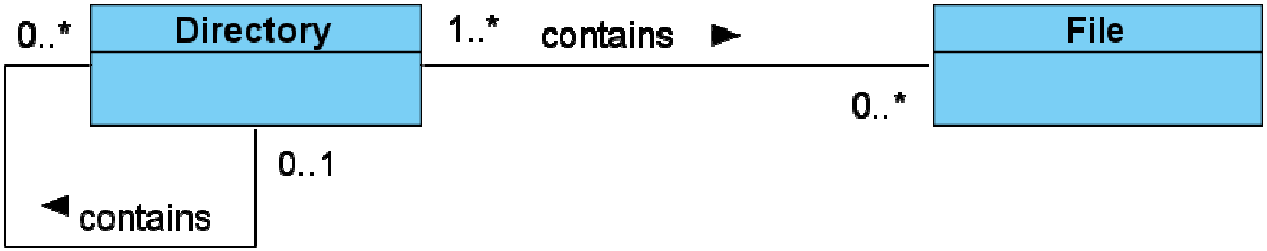
MUNI FI

Relationships between classes

- Multiple associations
  - Different relationships



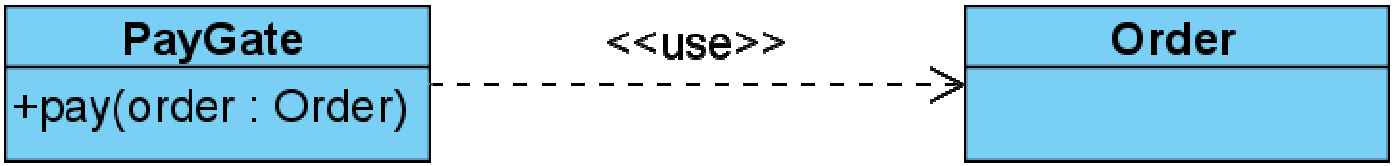
- Reflexive association



MUNI FI

Relationships between classes

- Dependency
  - „Weak association“
  - Represents a relationship where a change on one class might affect the other
  - One class *somehow* depends on the other
- The exact meaning is specified by stereotypes
  - The most common is «use», meaning that some operation uses the other object as attribute or return value. But it does not have it as attribute.



MUNI FI

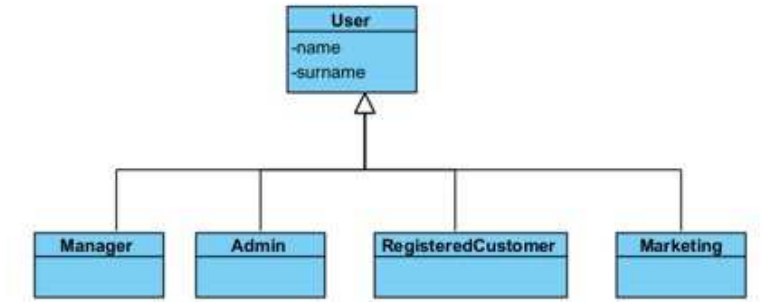
Task for this week

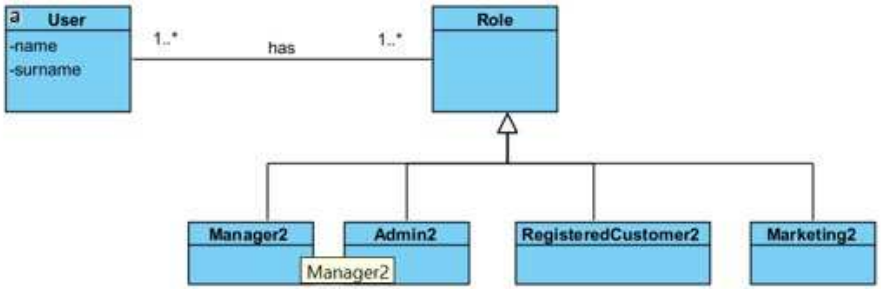
You gotta do what you gotta do

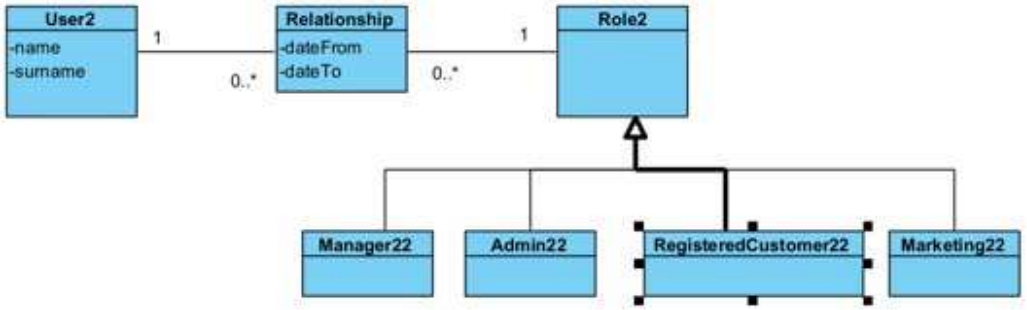
- Process the feedback
- Model entities, their attributes and associations
- Don’t forget inheritance and multiplicity
- BONUS: Navigability
  - You will have to do it someday

User roles

How can we deal with accounts in our project? Do we need unregistered customers?







MUNI FI

Task for next week

You gotta do what you gotta do

- Process the feedback
- Finish diagram – methods, …
- Update the Use Case Diagram. They must describe the same system.
- Consider the interaction between objects when fulfilling use cases. Are they represented by your Analytical Class Diagram?