AARHUS UNIVERSITET

Software Engineering and Architecture

UML

Motivation

- **Communication !!!**
  - An empirical investigation of software architects showed that about 25% of their time is spend **communicating**
    - Talking to customers, developers, testers, users, …
- For design we need a strong **language** to communicate our design ideas
- UML: Unified Modeling Language
  - A semi-precise language for *overview* of architecture and design

UML in SWEA

- We use the 20/80 rule
  - *20% let you get away with 80% of the job

- Until we get to distribution we will only use
  - Class diagrams   Static aspects of design
  - Sequence diagrams   Dynamic aspects of design

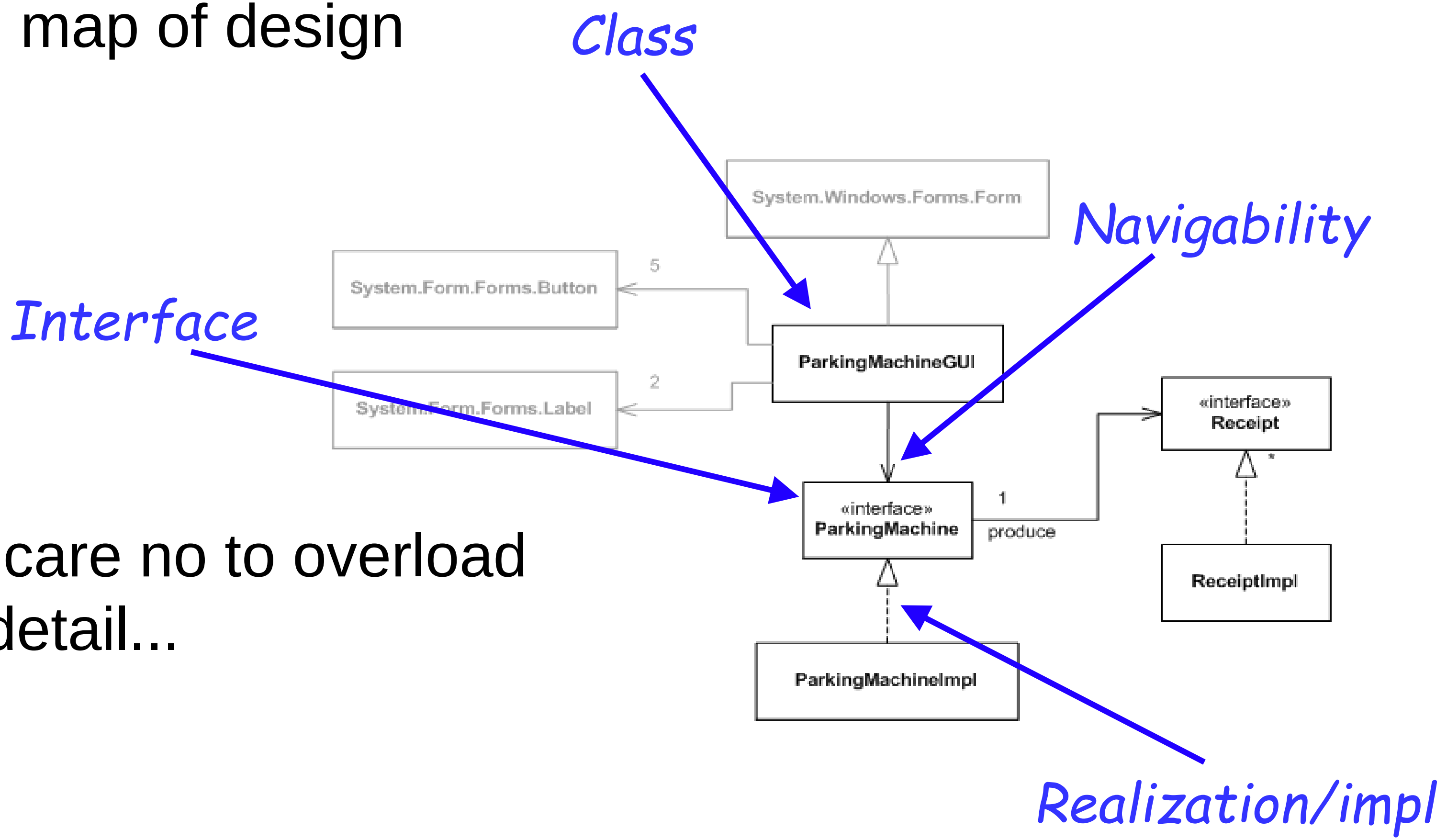
Class Diagram

Static aspect

”The code view”

Class Diagram

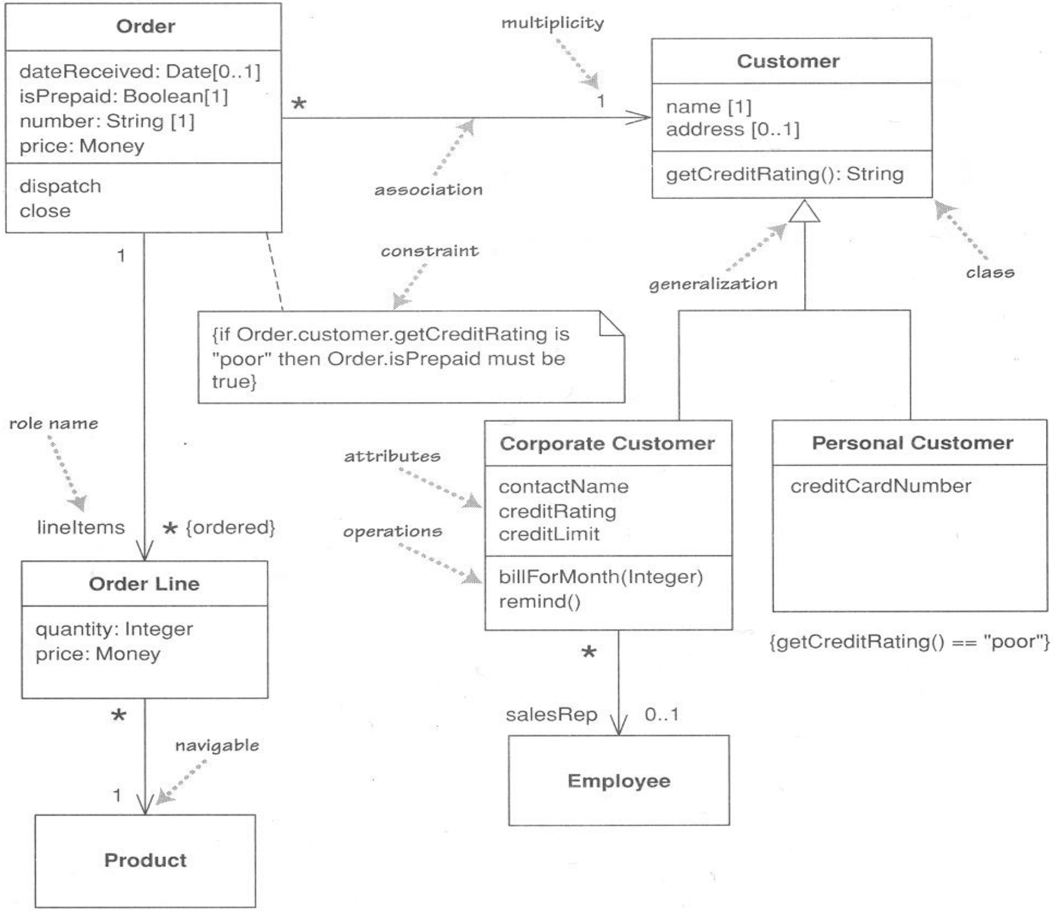
- Road map of design



- Take care no to overload with detail...

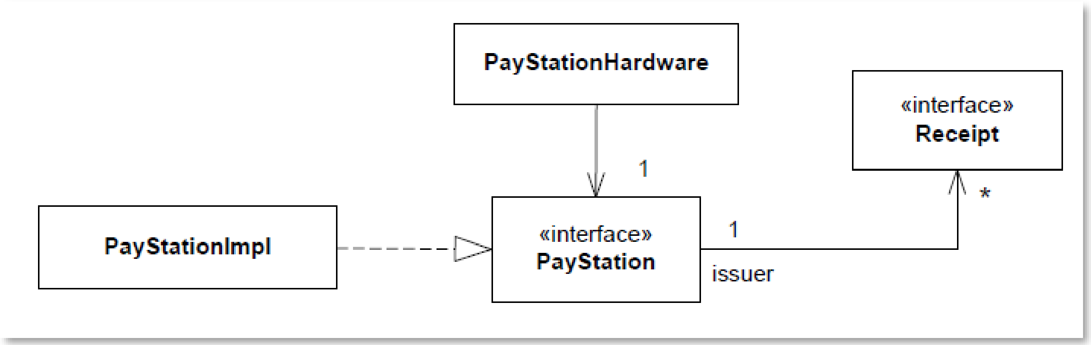
Class diagram

- All those details...
- Focus: Classes and their relations.
  - classes
    - attributes
    - methods
  - relations
    - generalization
    - association
  - multiplicity
  - roles
  - navigability

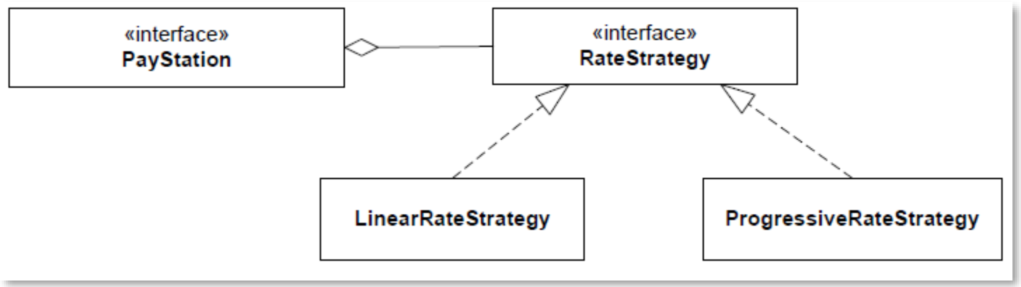


Association/Aggregation

- Association
  - Casual, temporary
  - Independent

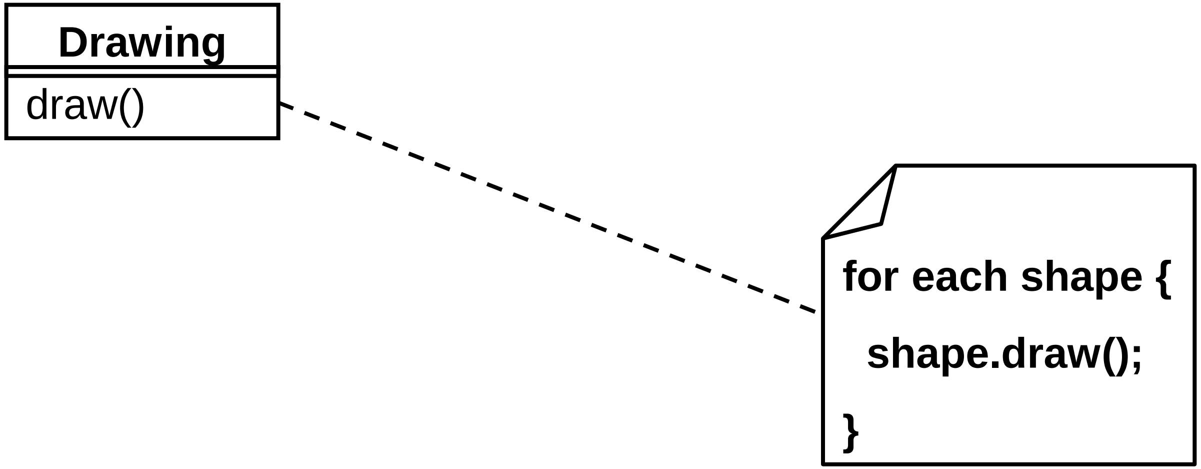


- Aggregation
  - Whole/Part
  - Dependent on



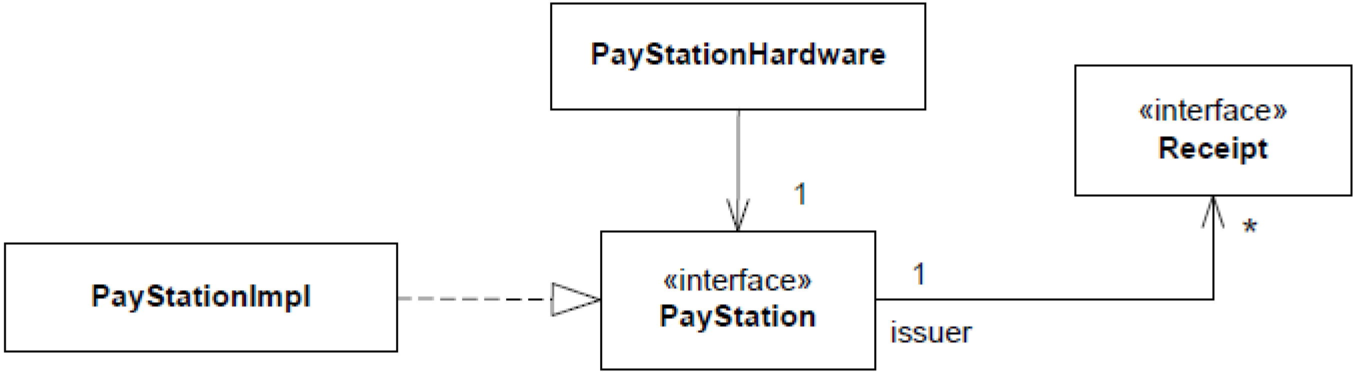
Additional notation

- **Notes** allow us to add useful but non-semantic information anywhere.



Architecture Vrs Code

- UML should be used to express **architecture/design**



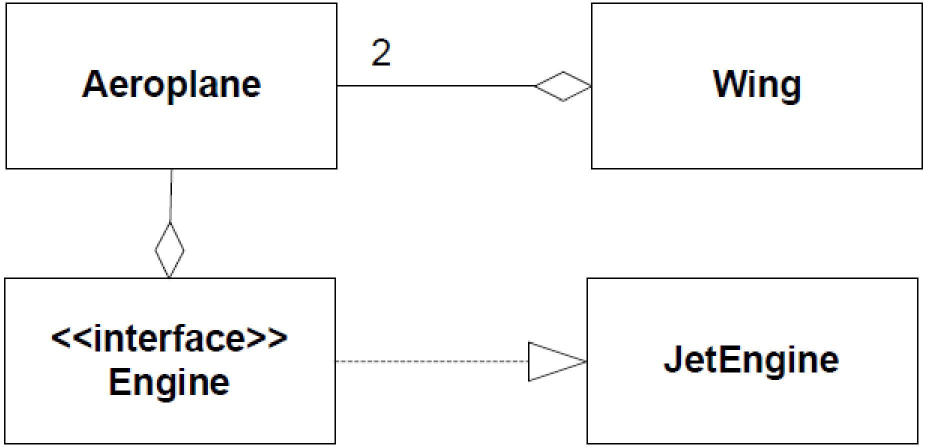
- *”You cannot draw that association because an interface cannot contain an object reference to a receipt object.”*
- ***Yes I can, I am the architect, god damn it!***
  – *I dictate that all PayStation realizations must have it!*

Exam Note

- Many of you will have to draw UML on the whiteboard!
- **Arrows and direction matter!** It is a language.
  - “I have a yellow banana in my hand.”
  - “Hand have yellow I in my banana.”
  - *I see quite a lot of the last type of ‘sentences’ at exam ☹*

These are different sentences!

- Exercise:
  - *What does this aero plane design express?*
  - *Why is it nonsense*



Sequence Diagram

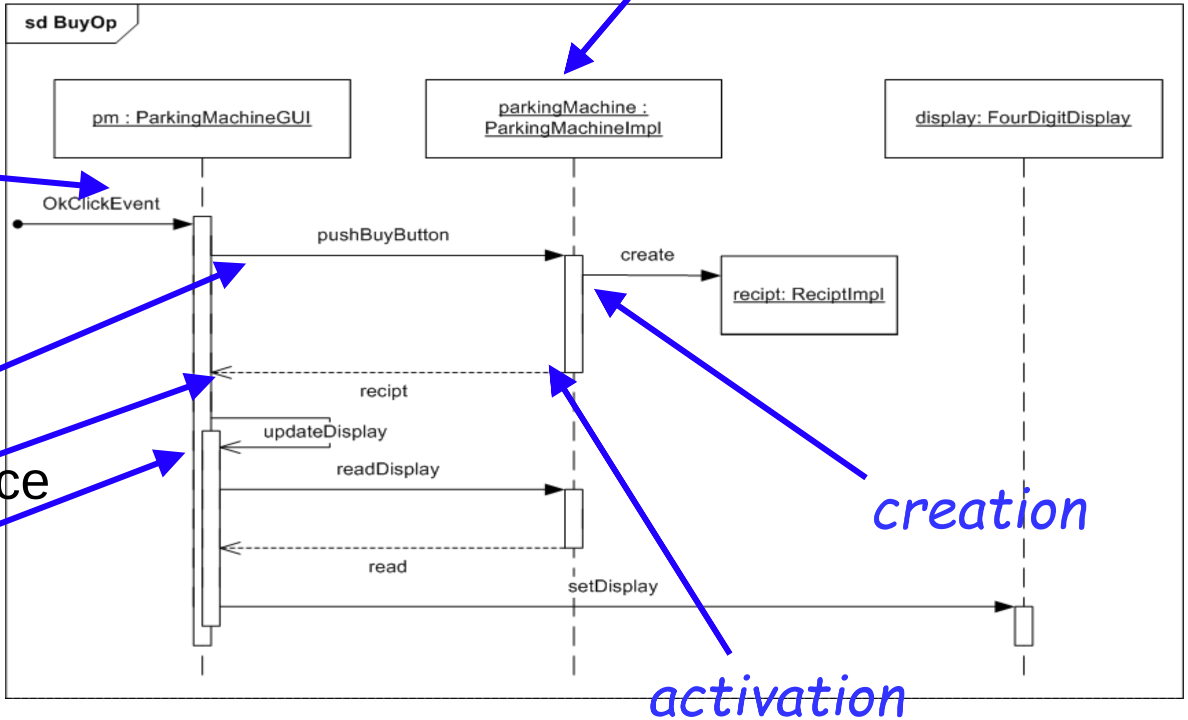
Dynamic aspect

”The execution view”

AARHUS UNIVERSITET

Behavior

- Sequence diagrams
  - Describe a single scenario
- Example
  - Buy ticket sequence



Participant (object)

Timeline

creation

message

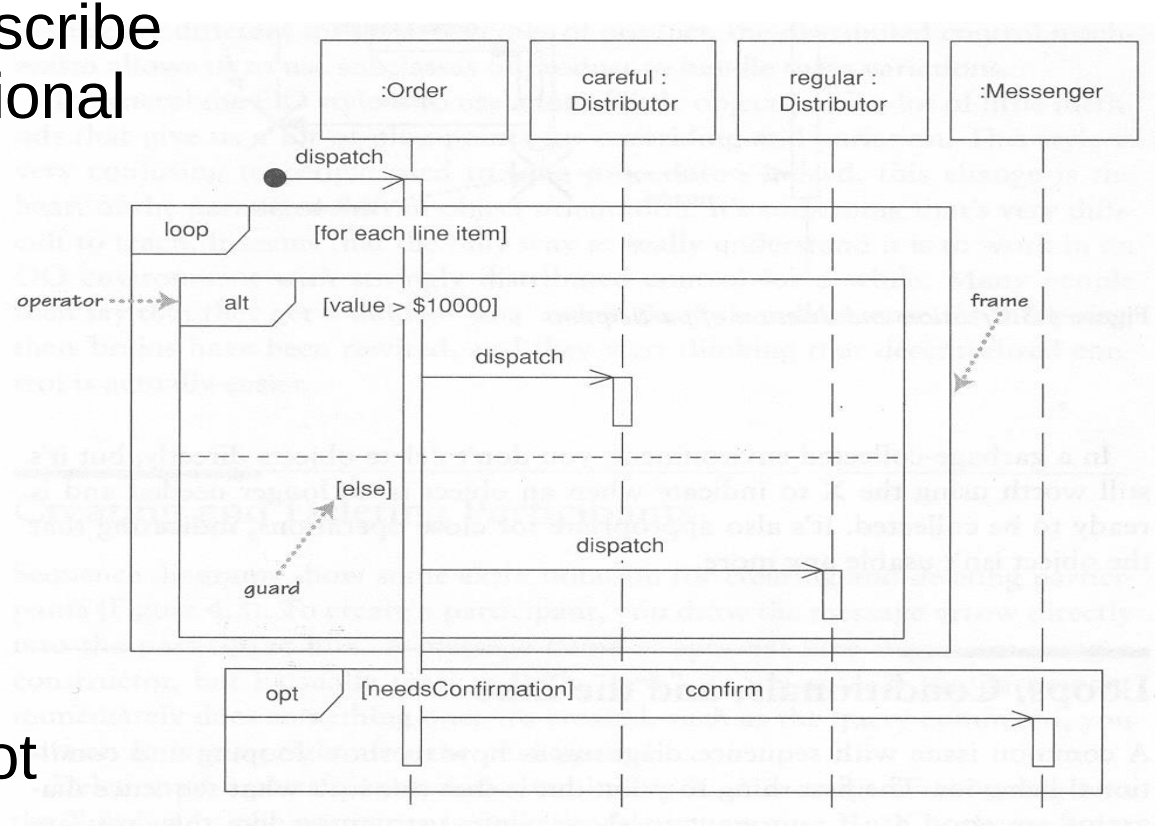
return

self-call

activation

Interaction frames

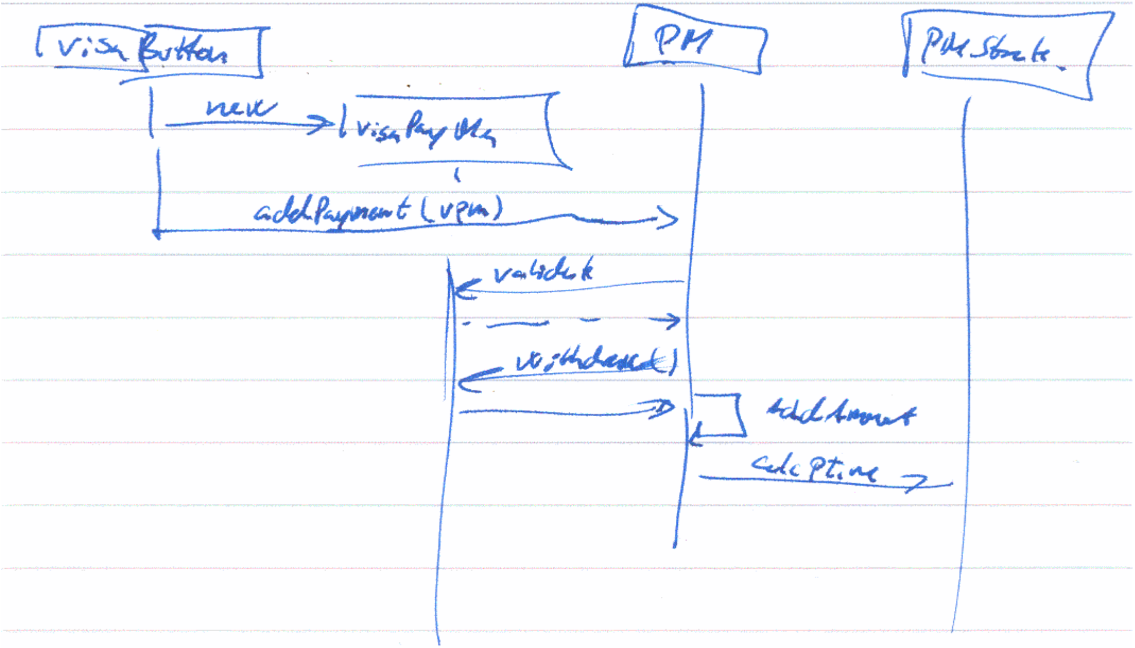
- UML 2.0 may describe loops and conditional statements.



- My advice: Do not overuse!!!

Playing around during design

- Dynamics is the most important aspect of computing!
  - Nice to play around



Summary

- UML A visual language for architecture and design

- Class diagram Interfaces and classes / static
- Sequence diagram Method calls / dynamic

- Train the syntax and semantics
  - Will be required in mandatory for communication
  - Will be required in **oral exam for communication**