2IW80 Software specification and architecture

Structural specification

Alexander Serebrenik

TU/e Technische Universiteit Eindhoven University of Technology

Where innovation starts

Before we start

- Time-related signal in an activity diagram is represented as

a)  *b)*  *c)* 

Before we start

- Time-related signal in an activity diagram is represented as

a) 

b) 

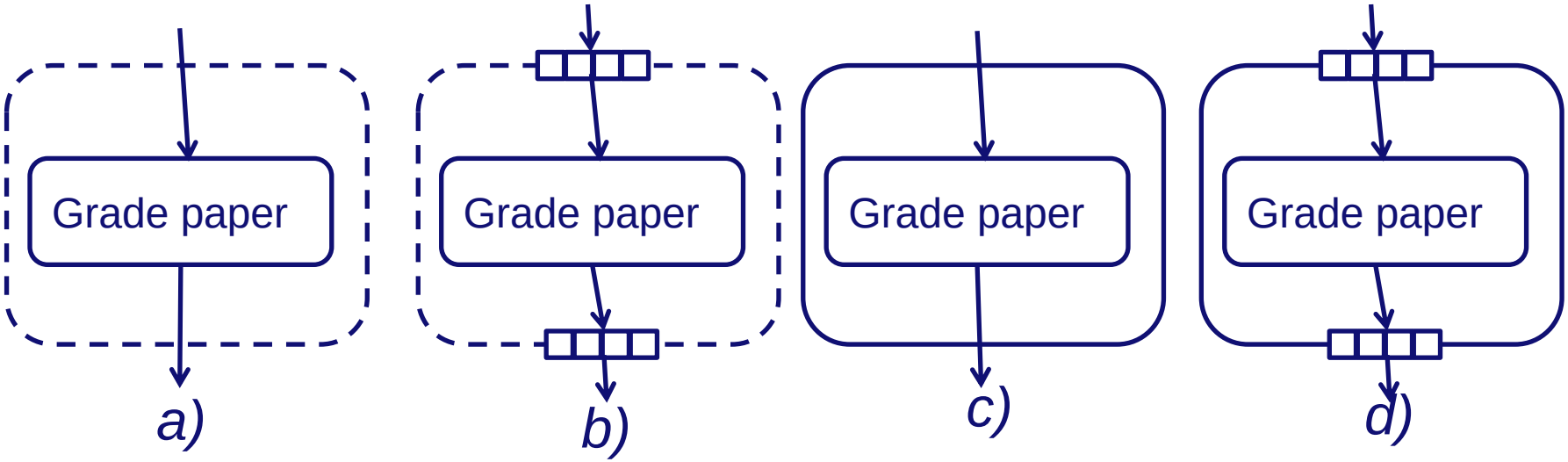
c)  ✓

Before we start

- Time-related signal in an activity diagram is represented as

a)  b)  c) 

- Dr. Smith is grading students' exam papers. How would you model her grading process?

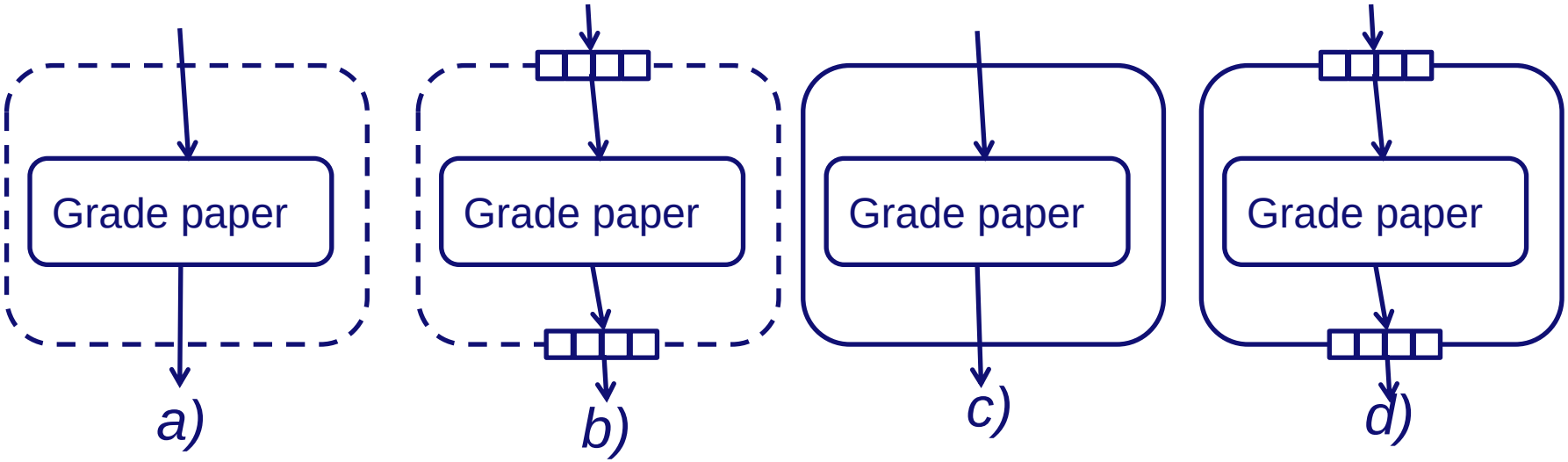


Before we start

- Time-related signal in an activity diagram is represented as

a)  b)  c) 

- Dr. Smith is grading students' exam papers. How would you model her grading process?



interruptible activity region

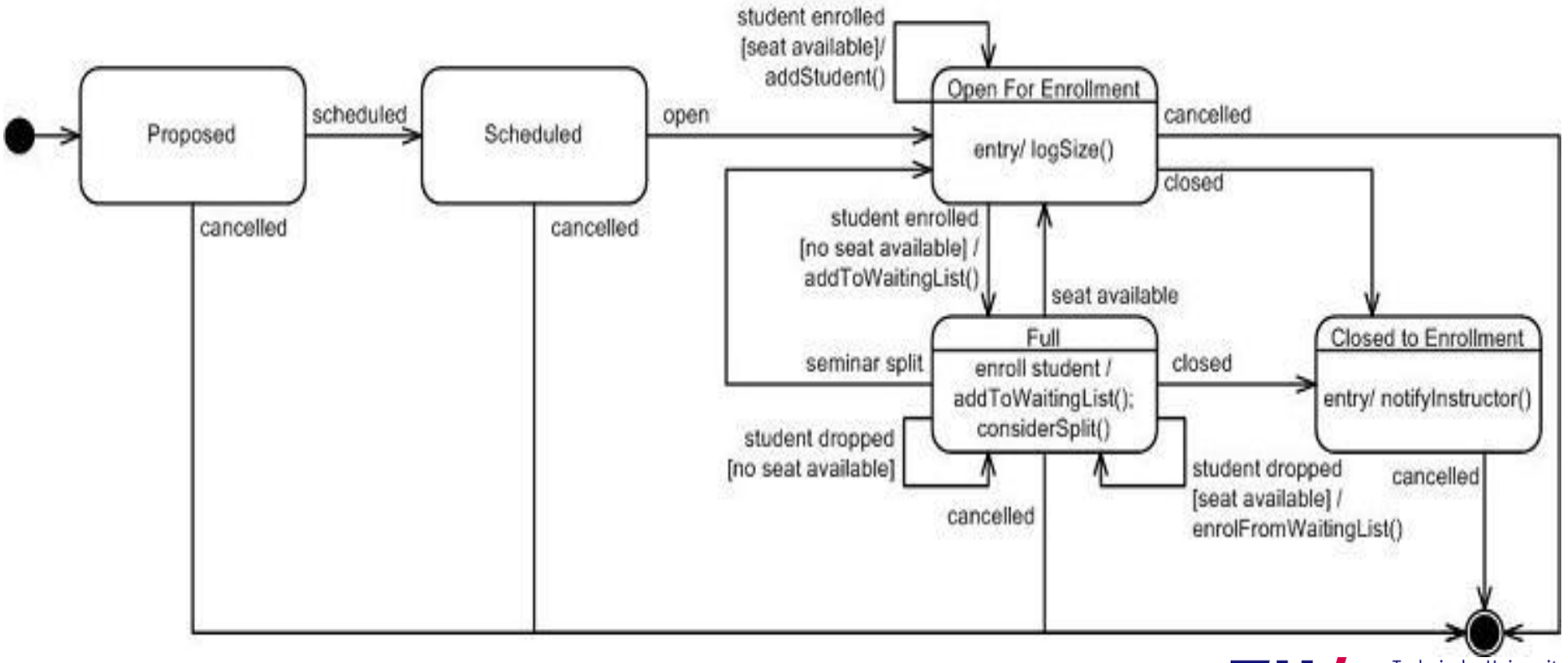
CORRECT: expansion region

subactivity (name missing)

does not exist

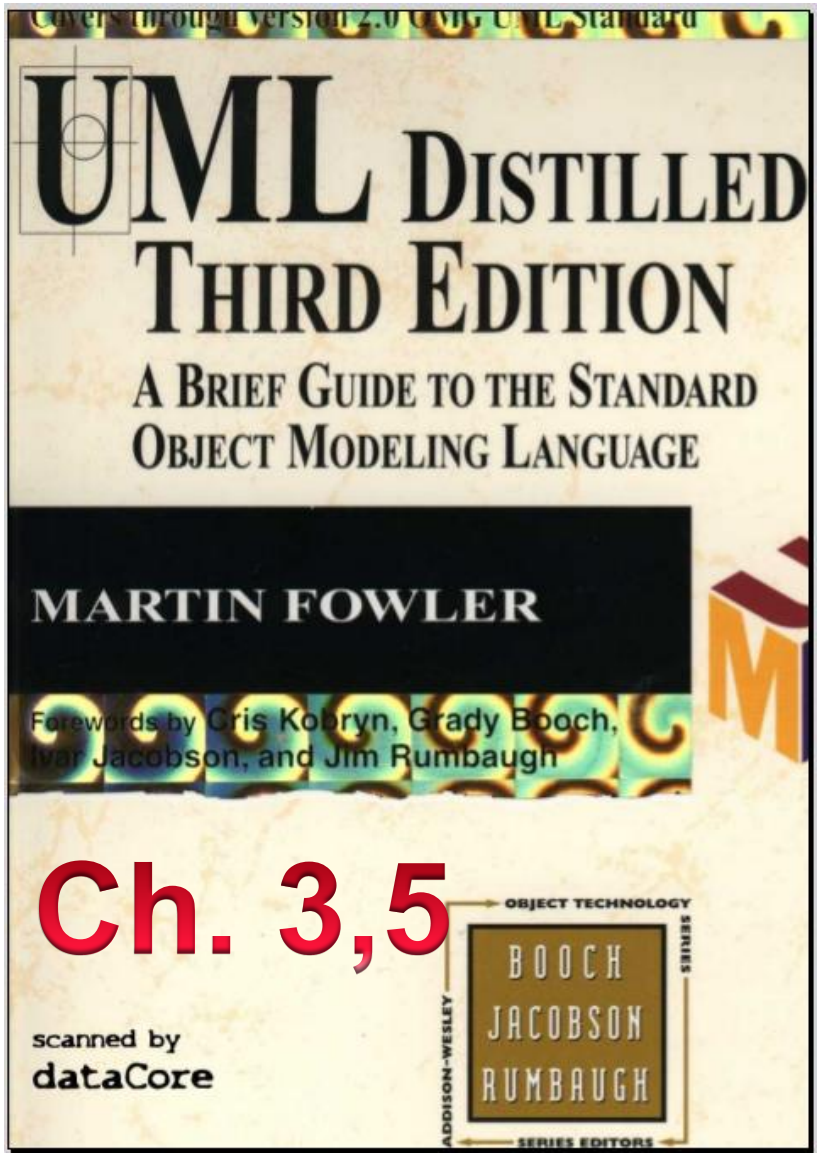
Before we start

- Seminar-registration state machine. What happens when a student is enrolled?



http://www.agilemodeling.com/artifacts/stateMachineDiagram.htm

This week sources



Slides of Mohammad Mousavi

What is structure?

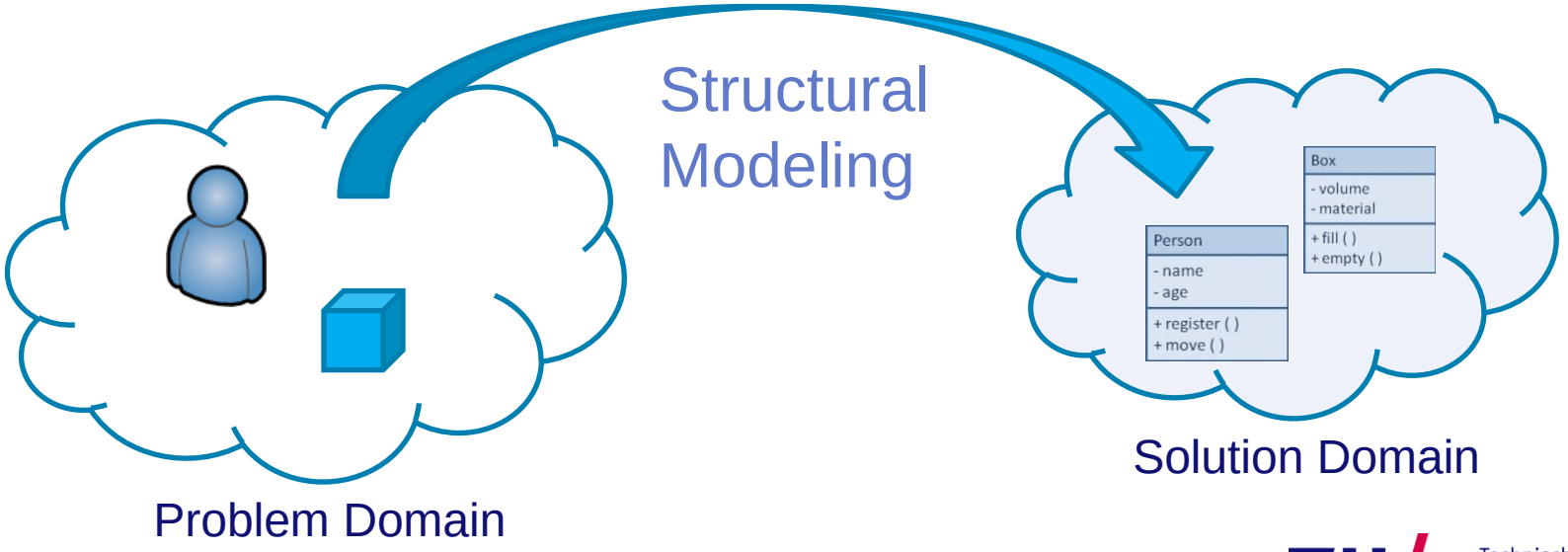
- **Structure:** the aggregate of elements of an entity in their relationships to each other. *Merriam-Webster*

- **Structural model**

What is structure?

- **Structure:** the aggregate of elements of an entity in their relationships to each other. *Merriam-Webster*

- **Structural modelling:** discover the key data contained in the problem domain and to build a structural model of the objects



So what is a structural model/diagram?

A diagram that identifies **modules, activities, or other entities** in a system or computer program and **shows how larger or more general entities break down into smaller**, more specific entities.

IEEE Standard Glossary of Software Engineering Terminology 610.12 1990

UML structure diagrams

Class diagram

Object diagram

Packages diagram

Component diagram

Deployment diagram

Composite structure diagram

Quote

If someone were to come up to you in a dark alley and say,

“Psst, wanna see a **UML diagram**?”

that diagram would probably be a **class diagram**.

[Martin Fowler, UML Distilled, Chapter 3]

Class diagrams

- most **common** and most **useful**
- describe the **types of objects** in the System and the various kinds of **static relationships** that exist among them
- show the **properties** and **operations** of a Class and the **constraints** that apply to the way objects are connected

[Martin Fowler, UML Distilled, 2003]

- kind of extended ER diagrams
  - NB: differences in notation

Objects and classes

- describe the **types of objects** in the System and the various kinds of static relationships that exist among them

- Do you remember?
  - **objects**: units of encapsulations of data (state) and functionality with an identity
  - **classes**: collections of objects [types of the objects]

Properties, operations and constraints

- show the **properties** and **operations** of a Class and the **constraints** that apply to the way objects are connected

- Do you remember?
  - **properties** (**attributes** and associations): units of data (responsibilities for knowing)
  - **operations**: units of functionality (responsibilities for doing)
  - **constraints**: conditions on attributes, pre- and post-conditions of operations

Example

- e-Commerce site (Amazon, bol.com, …)
- Class: **Order**

ORDER PLACED
2 Sep 2013
Order Details | Printable Order Summary

ORDER #026-4747744-1109156
RECIPIENT Alexander Serebrenik
TOTAL £210.97

Delivered On: Saturday 7 Sep 2013
Delivered

Qualitative Organizational Research: Core Methods and Current Challenges
Symon, Gillian
Sold by Amazon EU S.a.r.L.

Return or Replace Items
Write a Product Review

Experimentation in Software Engineering
Wohlin, Claes
Sold by Amazon EU S.a.r.L.

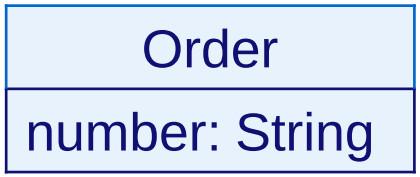
Guide to Advanced Empirical Software Engineering
Shull, Forrest
Sold by Amazon EU S.a.r.L.

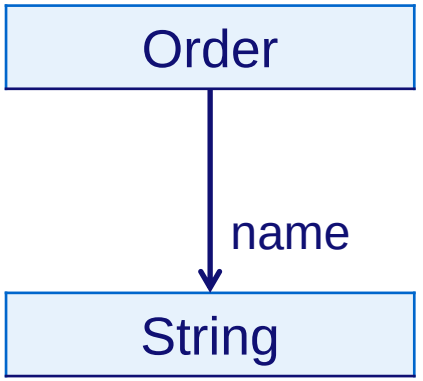
Making Software: What Really Works, and Why We Believe It
Andy Oram
Sold by Amazon EU S.a.r.L.

Give examples of **properties**

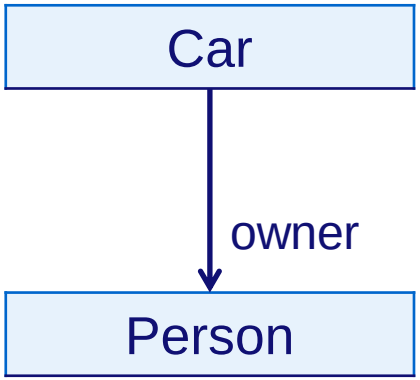
Attributes and associations

- Two different ways **properties** can be presented
- **Attribute** – property as a line of text within the class box.
- **Association** – property as a solid line between two classes, directed from the source class to the target class. Name is written on the line close to the target class.

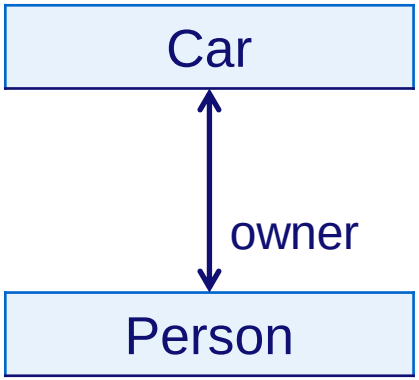




Navigability



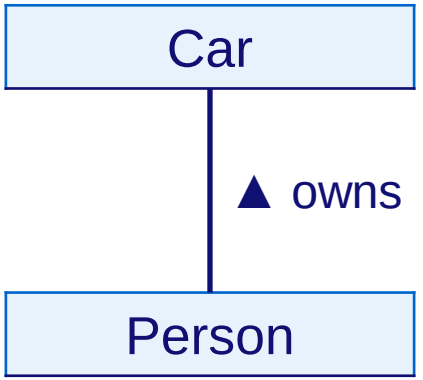
We can query Car about its owner but not Person about her cars.



We can query Car about its owner, and Person about her cars. Moreover,

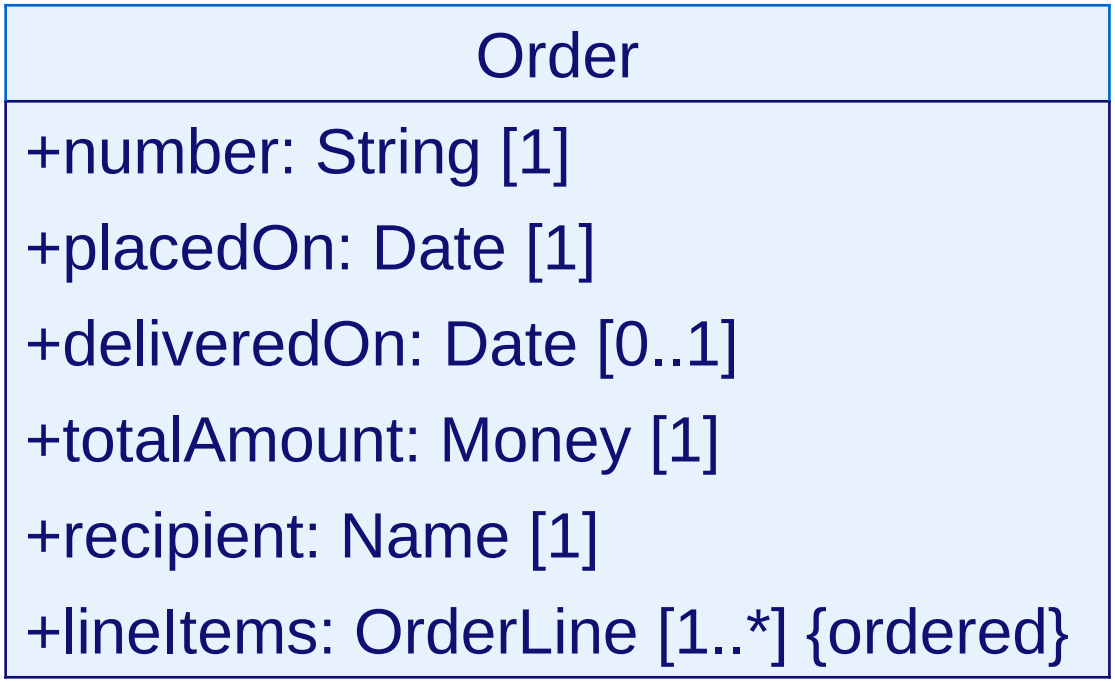
x in cars(owner(x)),

y in owner(cars(y))

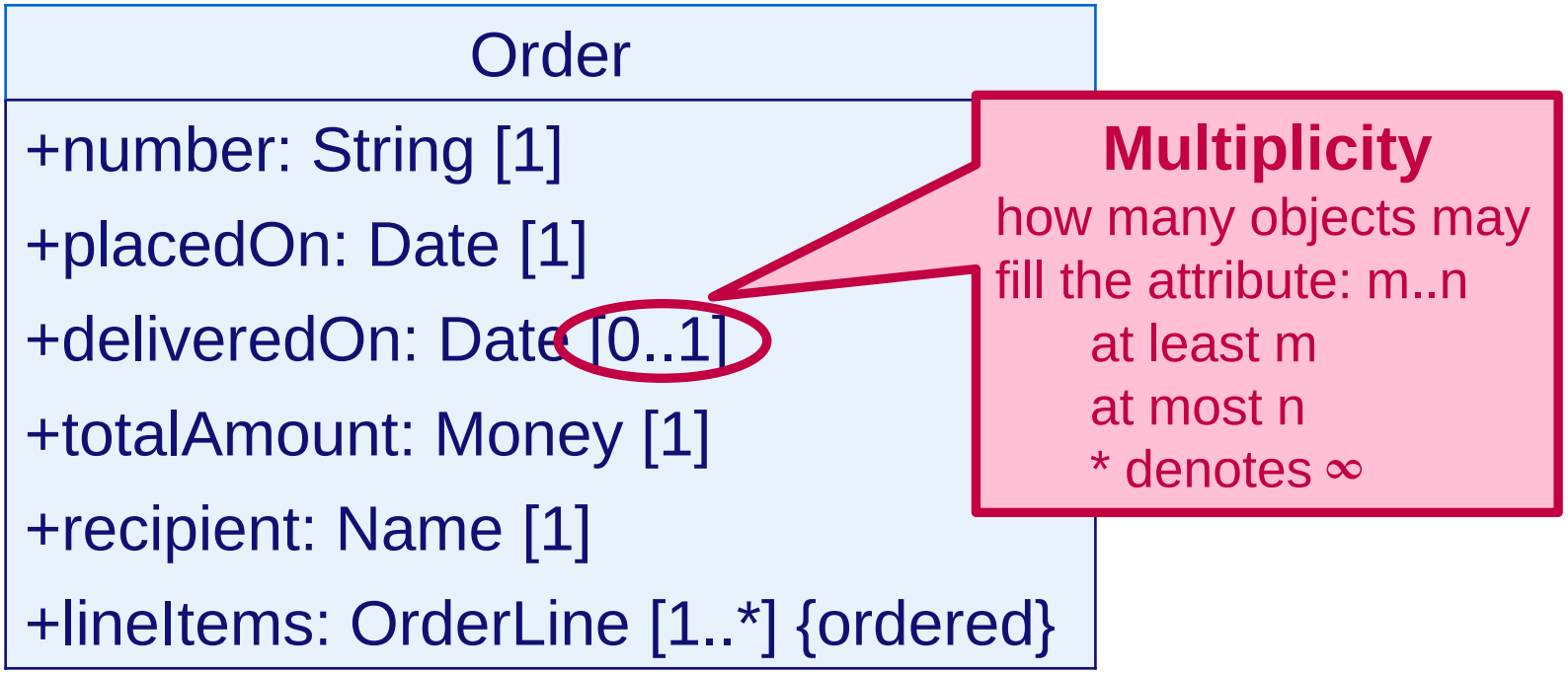


No navigability information.

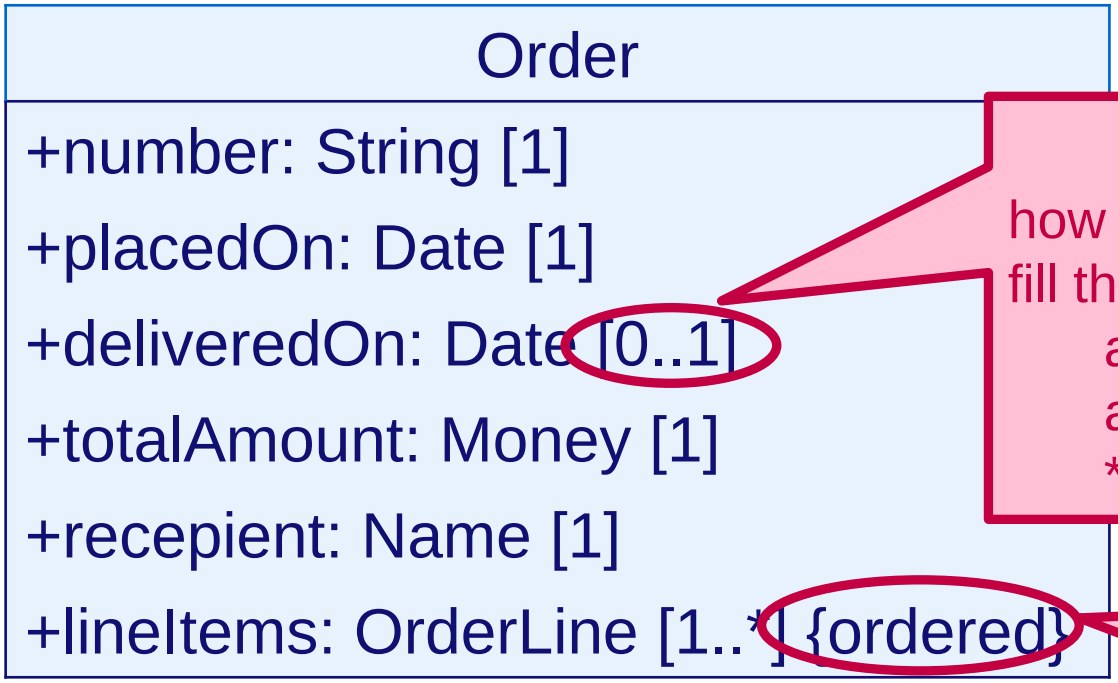
Closer look at the attributes



How would we write this in UML?



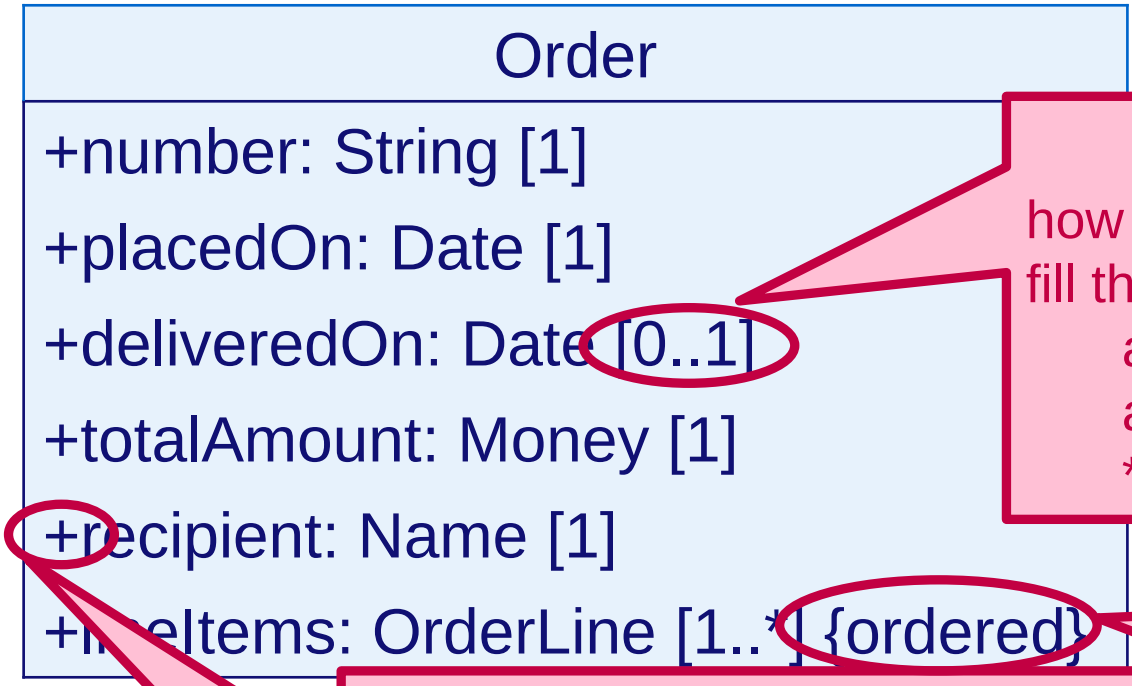
How would we write this in UML?



Multiplicity
how many objects may fill the attribute: m..n
at least m
at most n
* denotes ∞

Additional
Any kind of extra information needed about the attribute

How would we write this in UML?



Multiplicity
how many objects may fill the attribute: m..n
- at least m
- at most n
- * denotes ∞

Additional
Any kind of extra information needed about the attribute

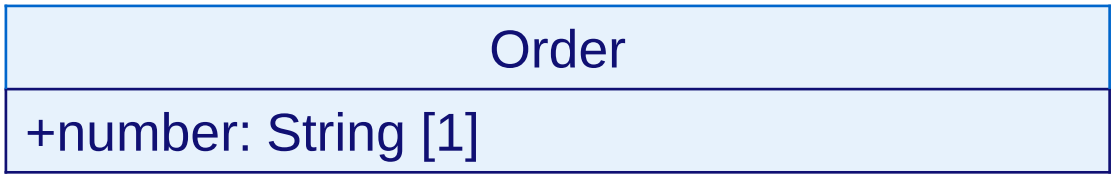
Visibility
Which other classes can access this attribute?
- + (public) every other class
- - (private) no other class
- # (protected) only classes that inherit from Order
- ~ (package)

Beware: visibility in Java ≠ visibility in UML

- Different programming languages use the **same names** for visibility levels but subtly **different meanings**

Beware: visibility in Java ≠ visibility in UML

- Different programming languages use the **same names** for visibility levels but subtly **different meanings**
- **Translation** from UML to a programming language depends on the programming language



Java

```
public class Order {
    private String number;
    public String getNumber() {
        return number;
    }
    public void setNumber(String number) {
        this.number = number;
    }
}
```

C#

```
public class Order
{
    public String number;
}
```

Recall our example

- e-Commerce site (Amazon, bol.com, …)
- Class: **Order**

ORDER PLACED
2 Sep 2013
Order Details | Printable Order Summary

ORDER #026-4747744-1109156
RECIPIENT Alexander Serebrenik
TOTAL £210.97

Delivered On: Saturday 7 Sep 2013
Delivered

Qualitative Organizational Research: Core Methods and Current Challenges
Symon, Gillian
Sold by Amazon EU S.a.r.L.

Return or Replace Items
Write a Product Review

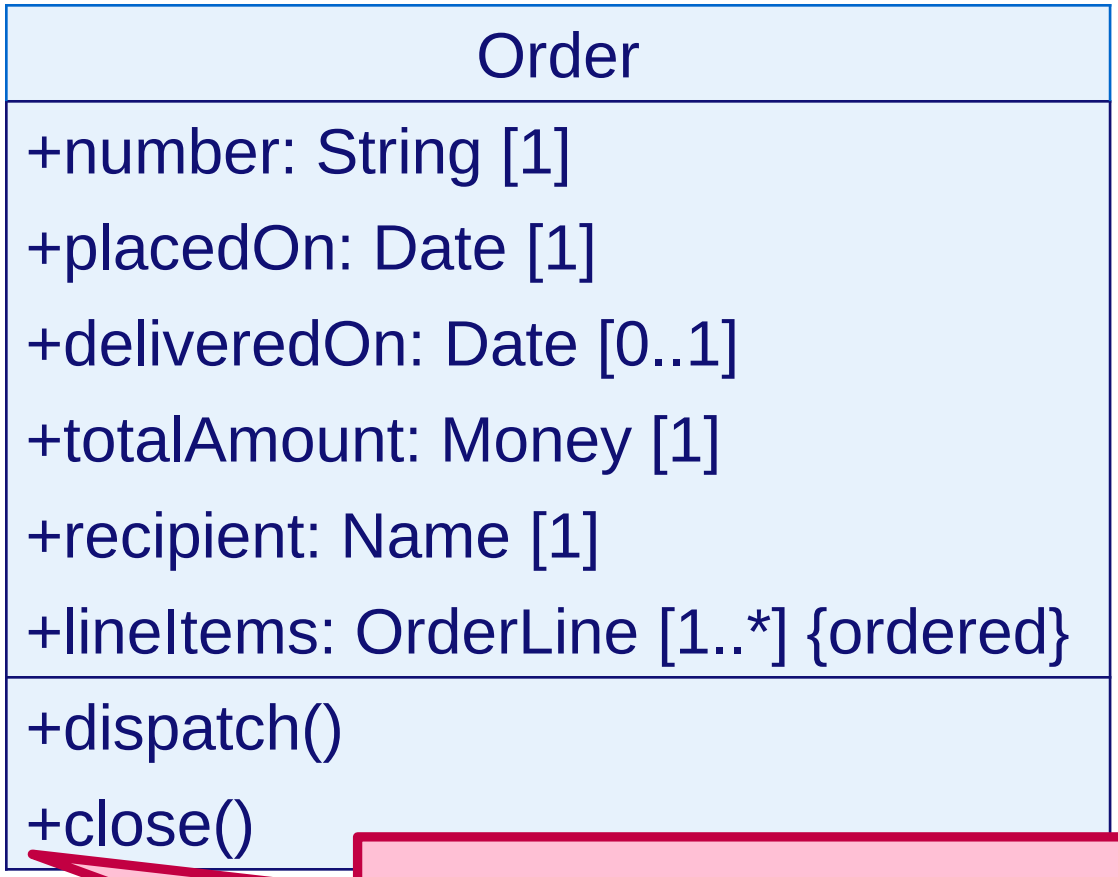
Experimentation in Software Engineering
Wohlin, Claes
Sold by Amazon EU S.a.r.L.

Guide to Advanced Empirical Software Engineering
Shull, Forrest
Sold by Amazon EU S.a.r.L.

Making Software: What Really Works, and Why We Believe It
Andy Oram
Sold by Amazon EU S.a.r.L.

Give examples of **operations**

How would we write this in UML?



Visibility

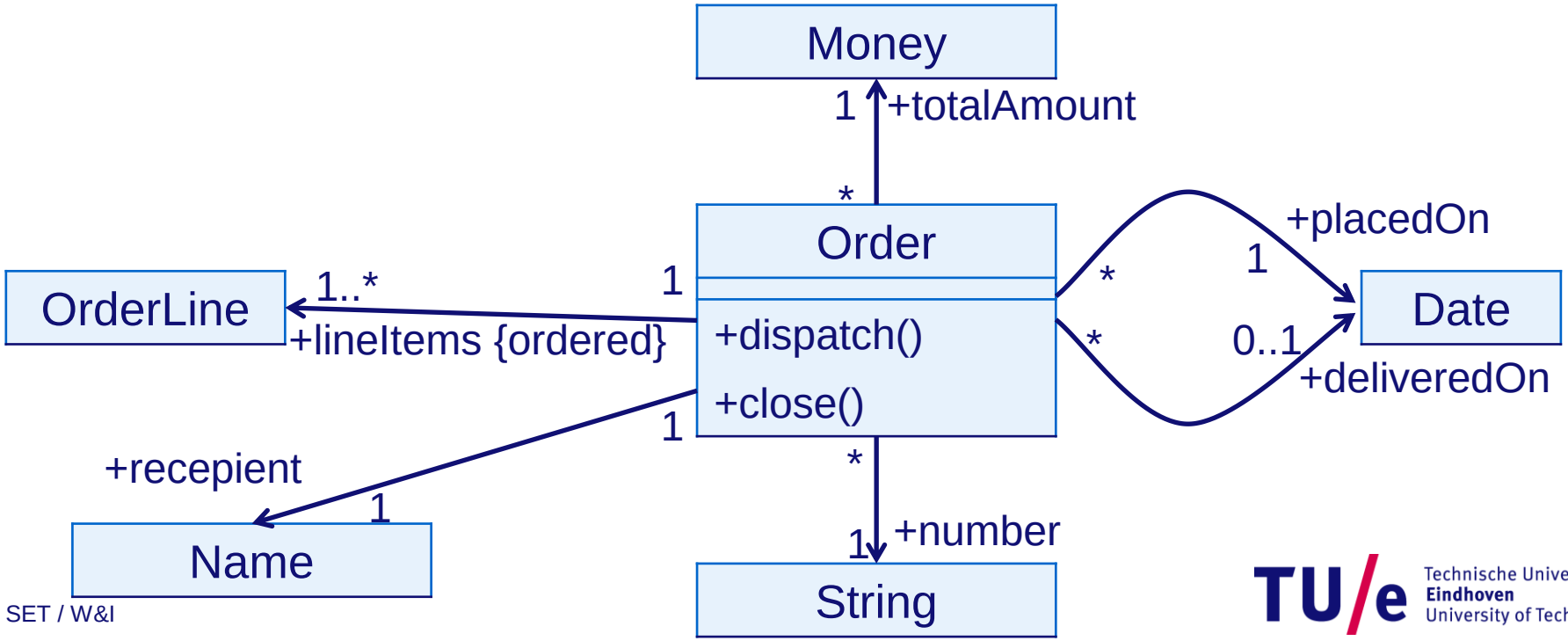
Which other classes can access this operation?

The same idea as with attributes

Attributes or associations?

Order
+number: String [1] +placedOn: Date [1] +deliveredOn: Date [0..1] +totalAmount: Money [1] +recipient: Name [1] +lineItems: OrderLine [1..*] {ordered}
+dispatch() +close()

Differences?
Advantages?
Disadvantages?
Combinations?



Attribute or association?

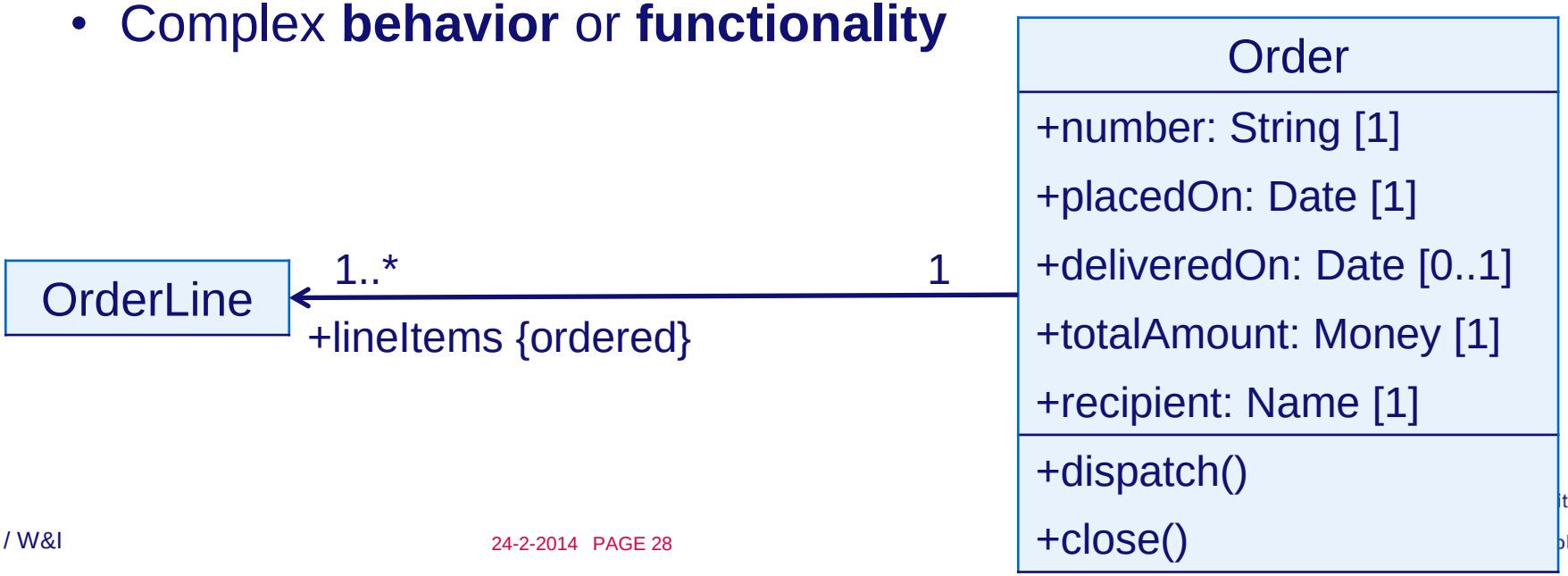
- **Attribute**
  - **Value** is important, not **identity**: Date, Number, String
  - Little **behavior** and **functionality**

- **Association** (i.e., a separate class)
  - **Identity** is important: Customer, Order, Student, Book
  - Complex **behavior** or **functionality**

Attribute or association?

- **Attribute**
  - **Value** is important, not **identity**: Date, Number, String
  - Little **behavior** and **functionality**

- **Association** (i.e., a separate class)
  - **Identity** is important: Customer, Order, Student, Book
  - Complex **behavior** or **functionality**

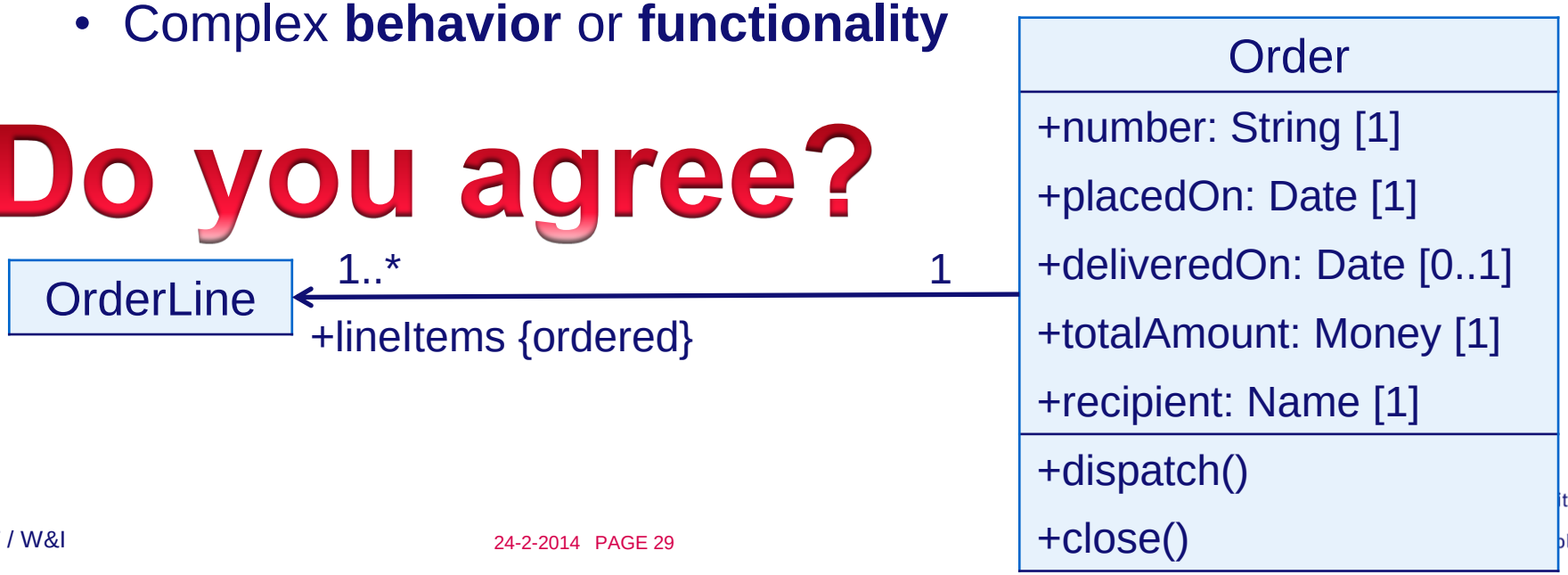


Attribute or association?

- **Attribute**
  - **Value** is important, not **identity**: Date, Number, String
  - Little **behavior** and **functionality**

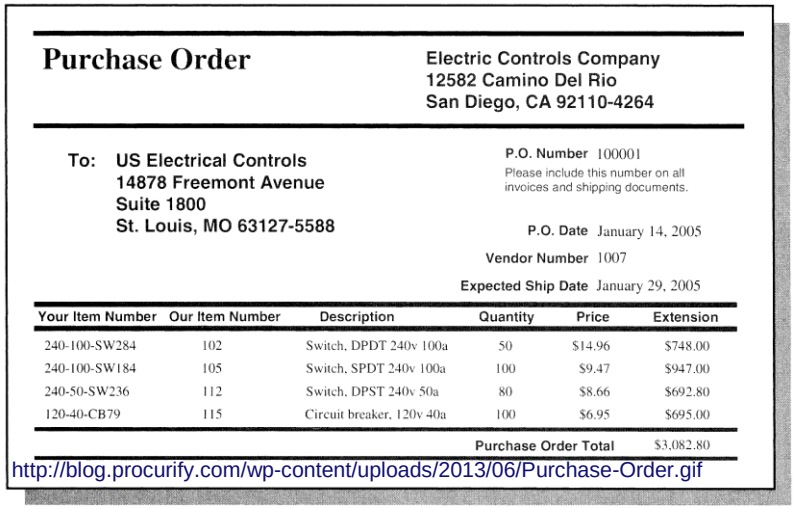
- **Association** (i.e., a separate class)
  - **Identity** is important: Customer, Order, Student, Book
  - Complex **behavior** or **functionality**

Do you agree?



Special kind of association

- Order lines do not really have identity
- Two orders cannot share an order line
- When an order is deleted, order lines do not “survive”…



Purchase Order

Electric Controls Company
12582 Camino Del Rio
San Diego, CA 92110-4264

To: US Electrical Controls
14878 Freemont Avenue
Suite 1800
St. Louis, MO 63127-5588

P.O. Number 100001
Please include this number on all invoices and shipping documents.

P.O. Date January 14, 2005

Vendor Number 1007

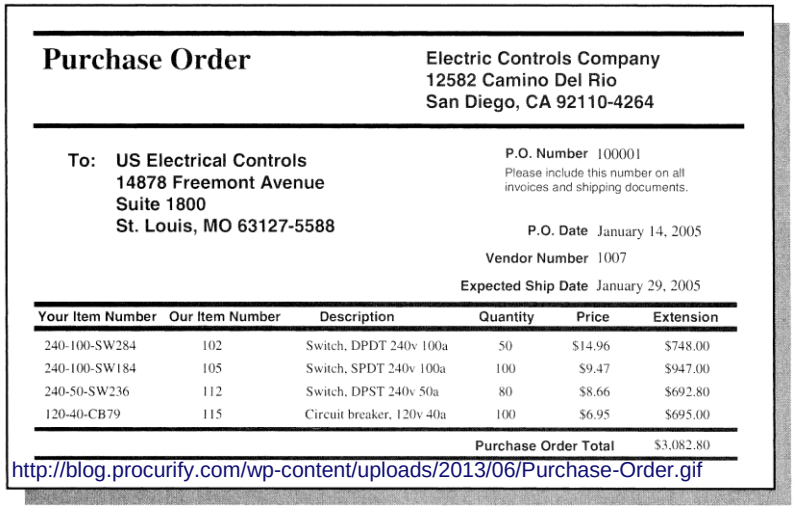
Expected Ship Date January 29, 2005

Your Item Number	Our Item Number	Description	Quantity	Price	Extension
240-100-SW284	102	Switch, DPDT 240v 100a	50	$14.96	$748.00
240-100-SW184	105	Switch, SPDT 240v 100a	100	$9.47	$947.00
240-50-SW236	112	Switch, DPST 240v 50a	80	$8.66	$692.80
120-40-CB79	115	Circuit breaker, 120v 40a	100	$6.95	$695.00
				Purchase Order Total	$3,082.80

http://blog.procurify.com/wp-content/uploads/2013/06/Purchase-Order.gif

Special kind of association

- Order lines do not really have identity
- Two orders cannot share an order line
- When an order is deleted, order lines do not “survive”…



Purchase Order

Electric Controls Company
12582 Camino Del Rio
San Diego, CA 92110-4264

To: US Electrical Controls
14878 Freemont Avenue
Suite 1800
St. Louis, MO 63127-5588

P.O. Number 100001
Please include this number on all invoices and shipping documents.
P.O. Date January 14, 2005
Vendor Number 1007
Expected Ship Date January 29, 2005

Your Item Number	Our Item Number	Description	Quantity	Price	Extension
240-100-SW284	102	Switch, DPDT 240v 100a	50	$14.96	$748.00
240-100-SW184	105	Switch, SPDT 240v 100a	100	$9.47	$947.00
240-50-SW236	112	Switch, DPST 240v 50a	80	$8.66	$692.80
120-40-CB79	115	Circuit breaker, 120v 40a	100	$6.95	$695.00
				Purchase Order Total	$3,082.80

http://blog.procurify.com/wp-content/uploads/2013/06/Purchase-Order.gif

Composition

Does this make sense?

Does this make sense?

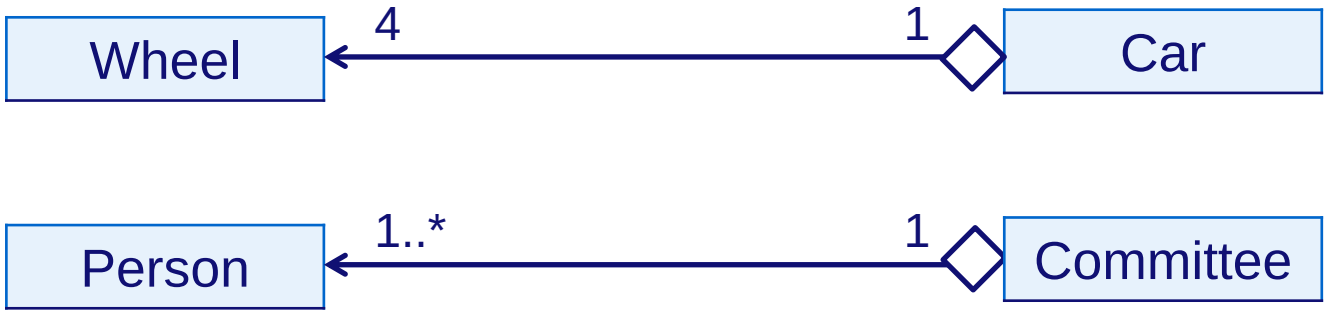
No, we can have spare parts!

Does this make sense?

Does this make sense?

No, persons might be members of multiple committees!

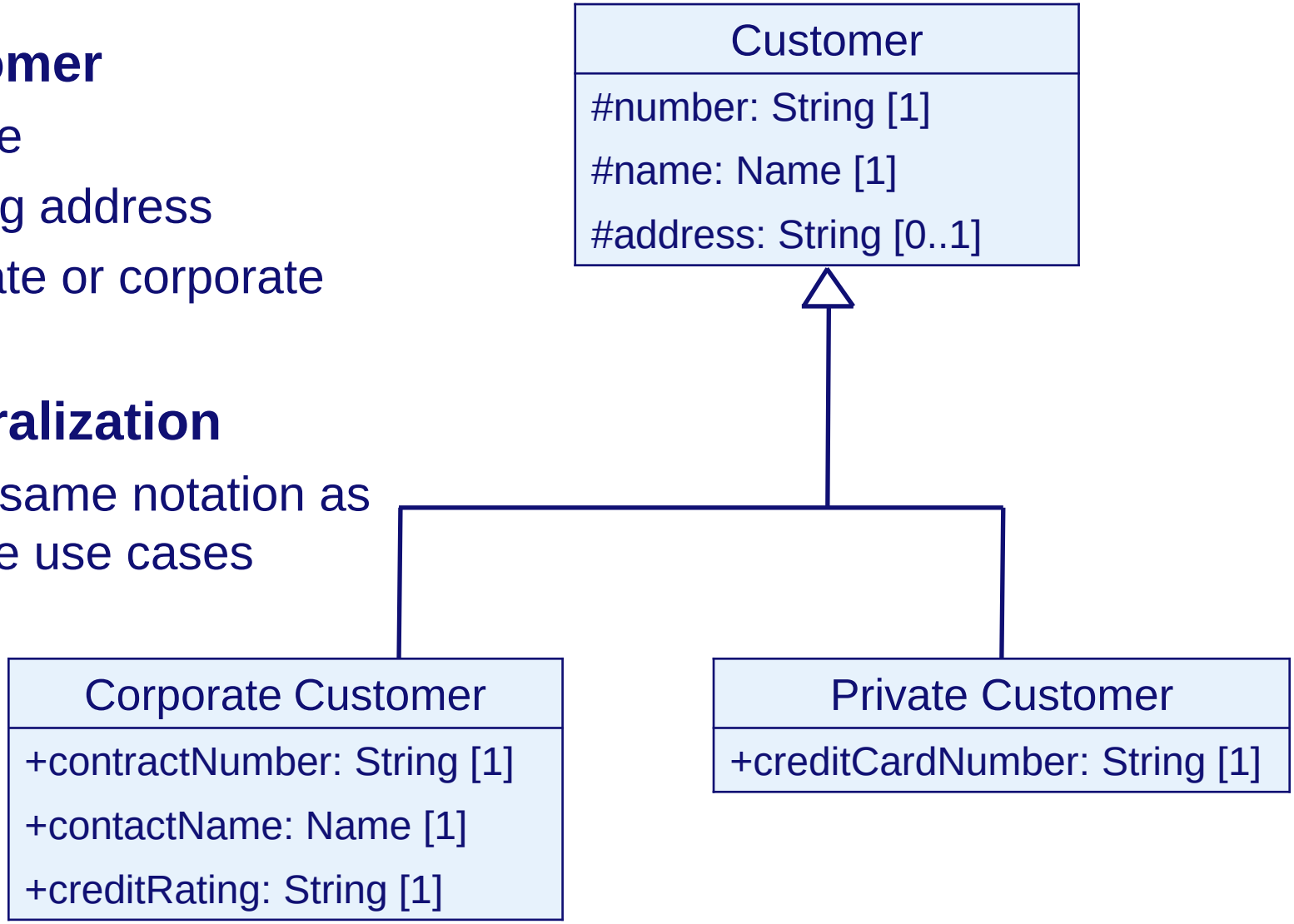
Weaker form: aggregation



- **Aggregation**: “part of” relation
  - Parts can exist without the whole
  - Parts can be shared by multiple “wholes”
  - “Whole” can exist without its parts

Recipient – we need more than a name

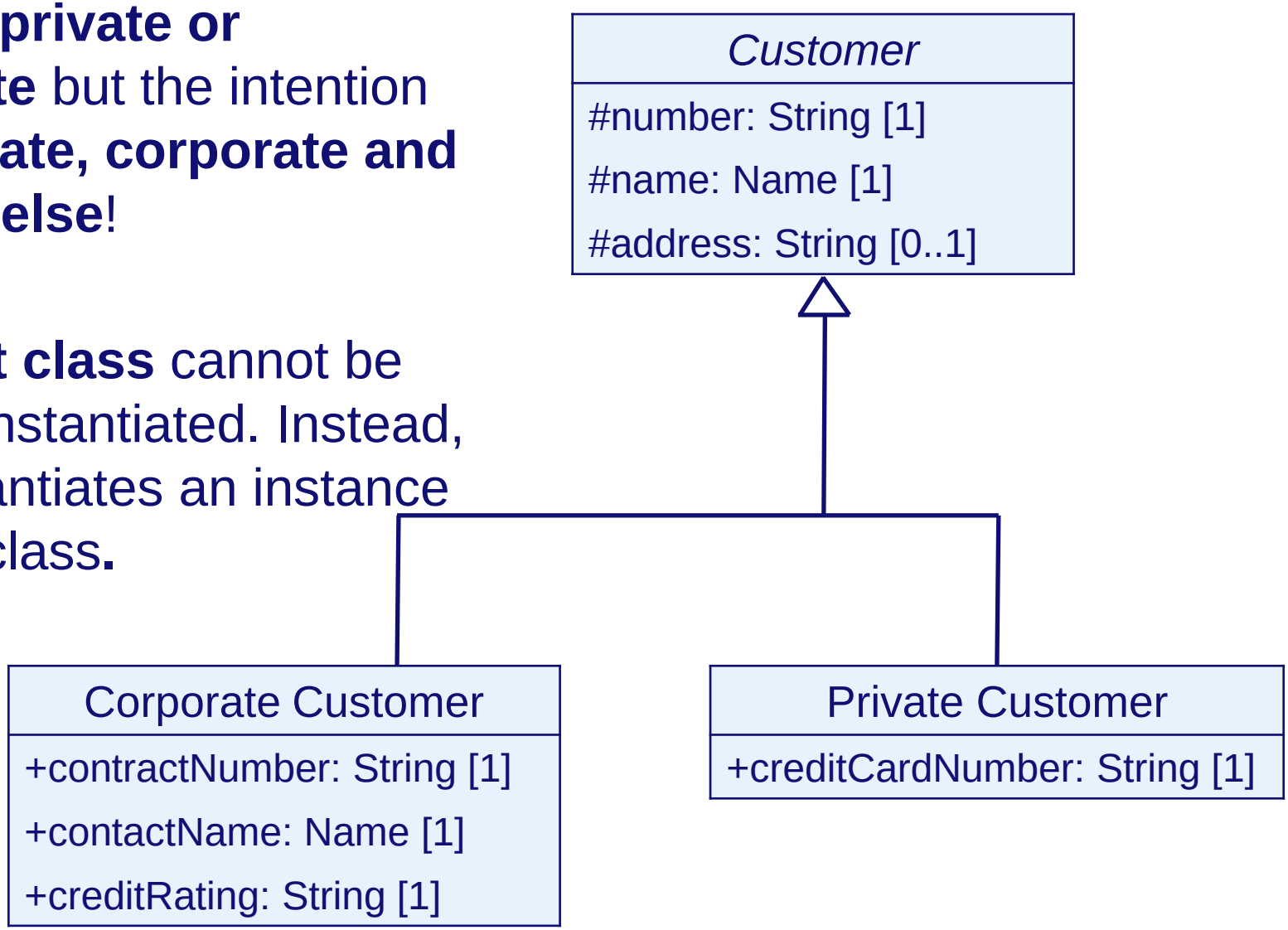
- **Customer**
  - name
  - billing address
  - private or corporate
- **Generalization**
  - The same notation as in the use cases



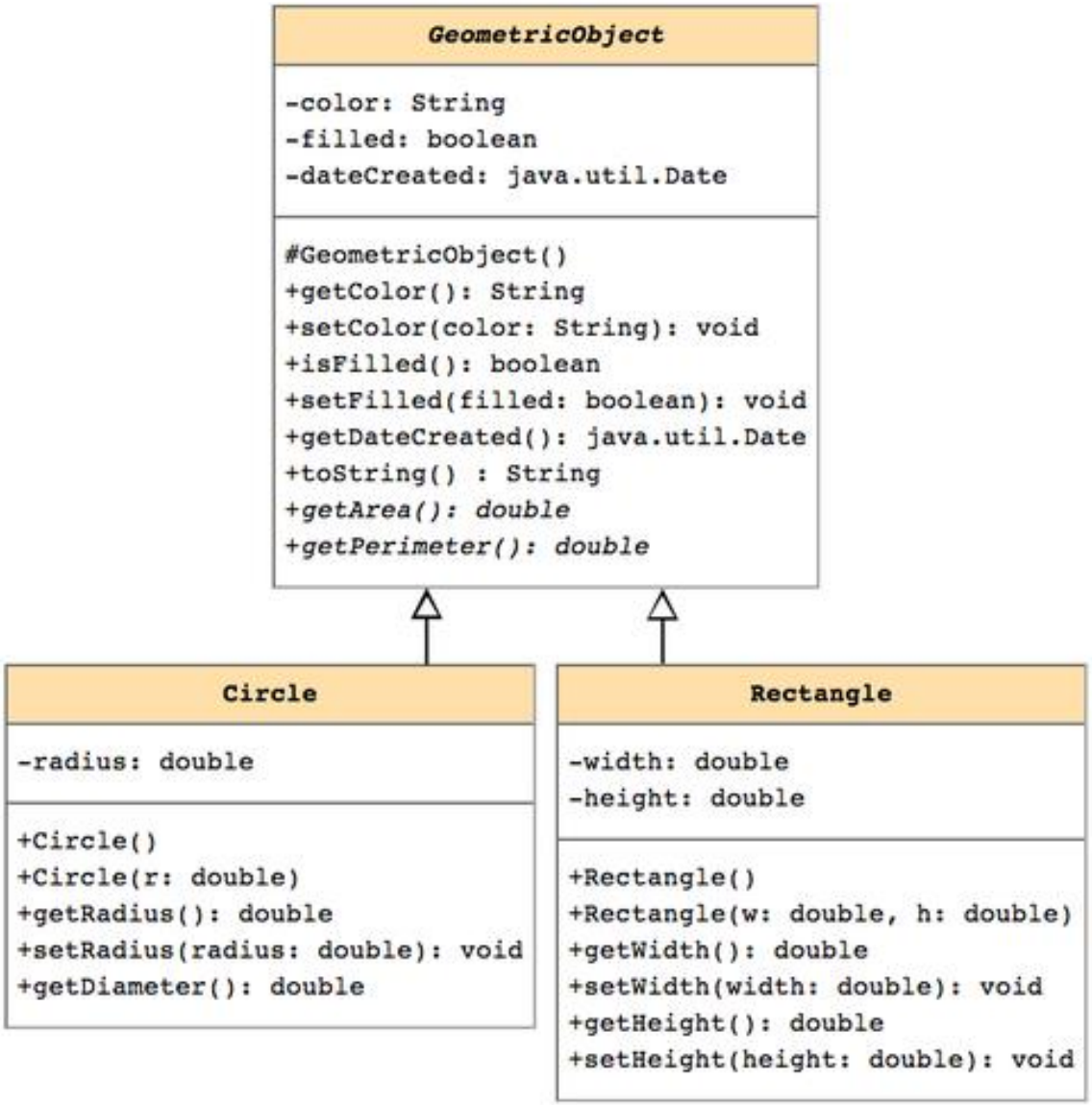
Abstract class

- We said **private or corporate** but the intention was **private, corporate and nothing else**!

- **Abstract class** cannot be directly instantiated. Instead, one instantiates an instance of a subclass**.**



Abstract class: Another example



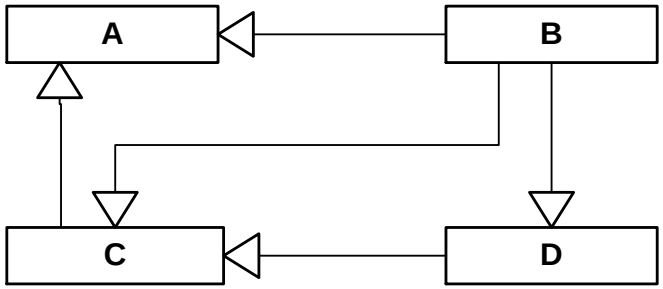
- ***GeometricObject*** is an abstract class
- ***getArea()*** and ***getPerimeter()*** are abstract methods
  - Not implemented in *GeometricObject* but in its subclasses
  - Why?

http://www.oxfordmathcenter.com/drupal7/node/35

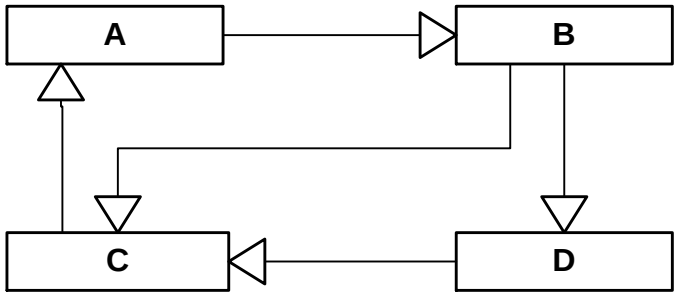
Exercise

For classes A, B, C, D which class diagram(s) is/are illegal?

- **A)**

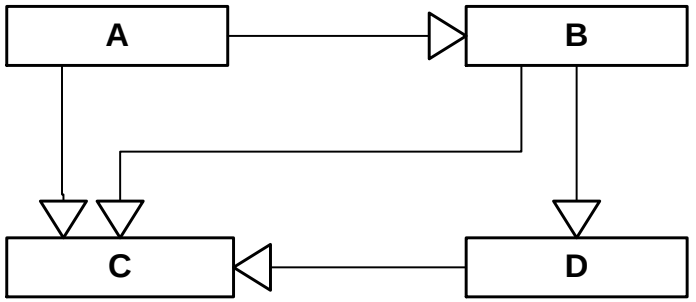


- **B)**

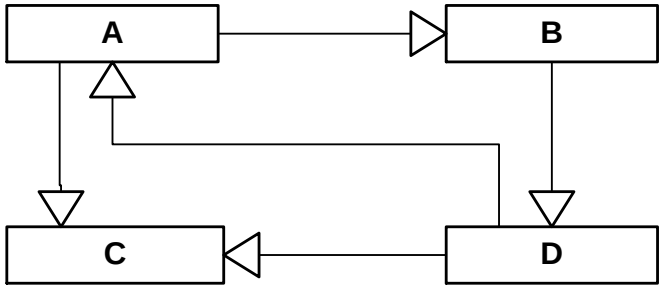


illegal (cycle)

- **C)**



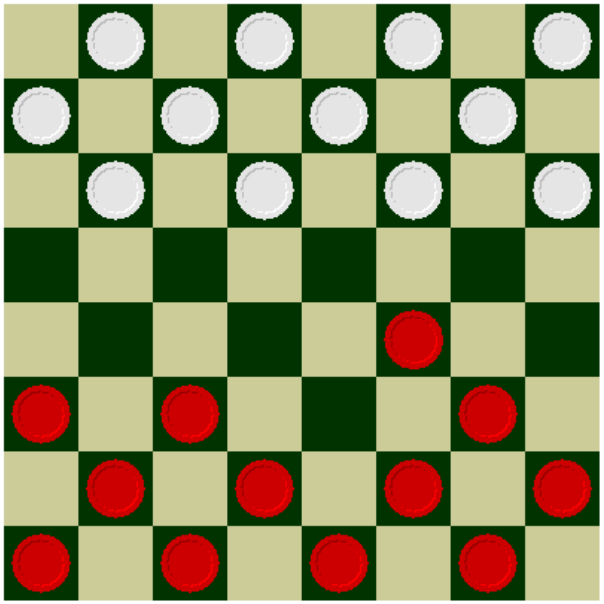
- **D)**



illegal (cycle)

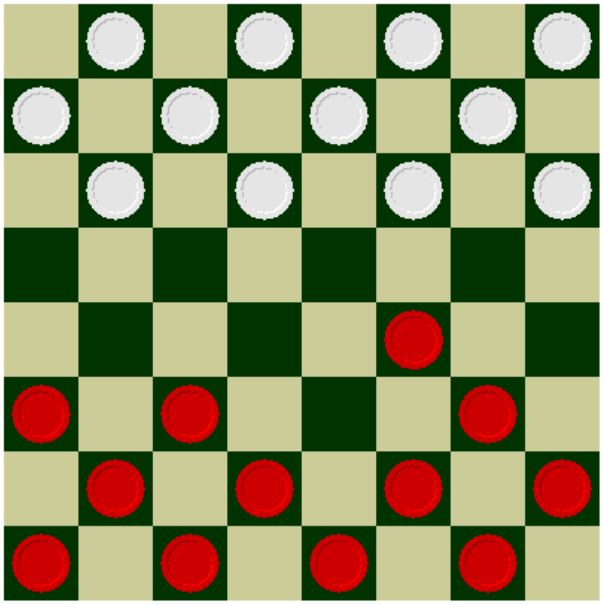
Exercise (from Northumbria University)

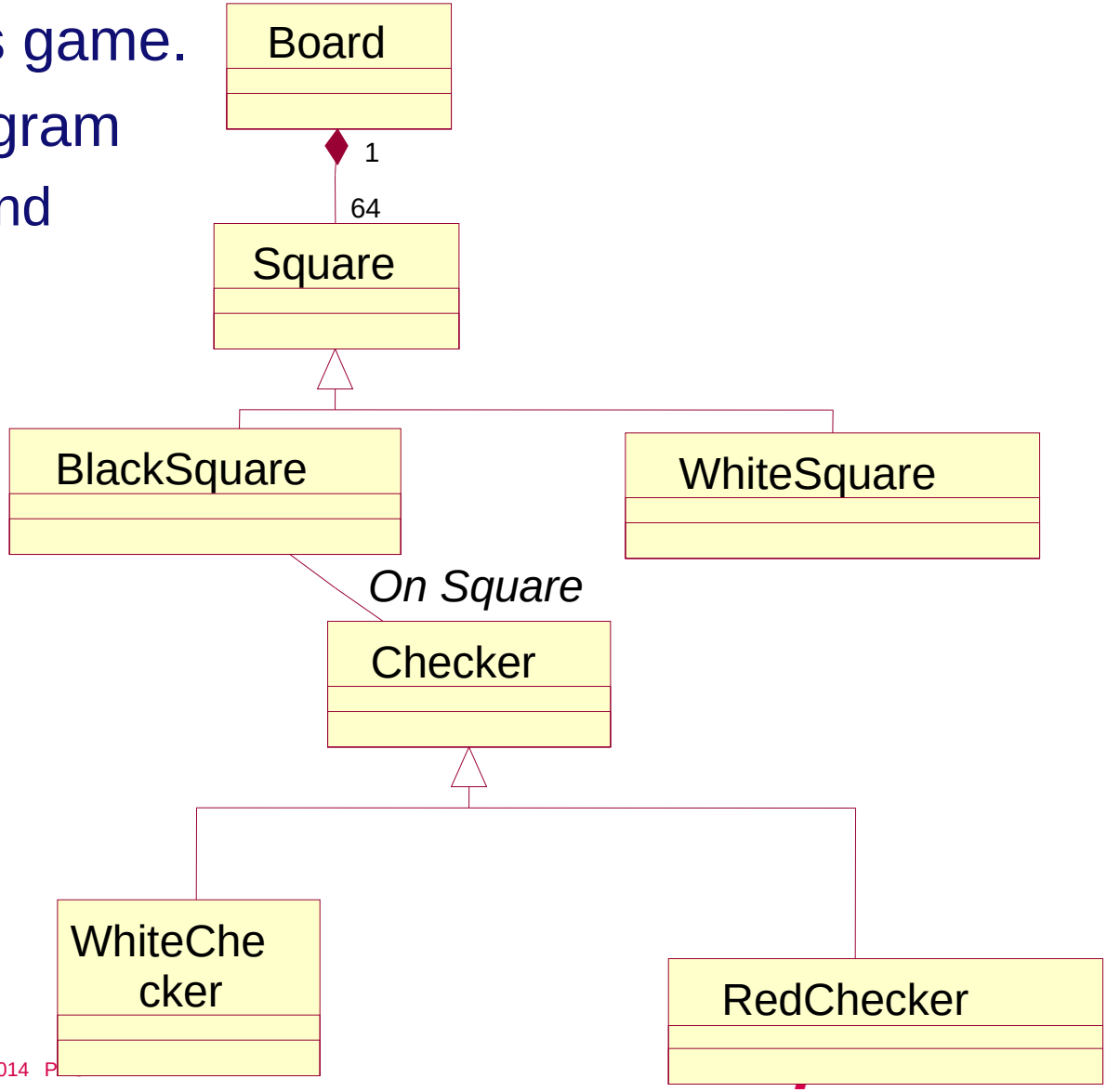
- Consider Checkers game.
- Draw the class diagram
  - Ignore attributes and operations



Exercise (from Northumbria University)

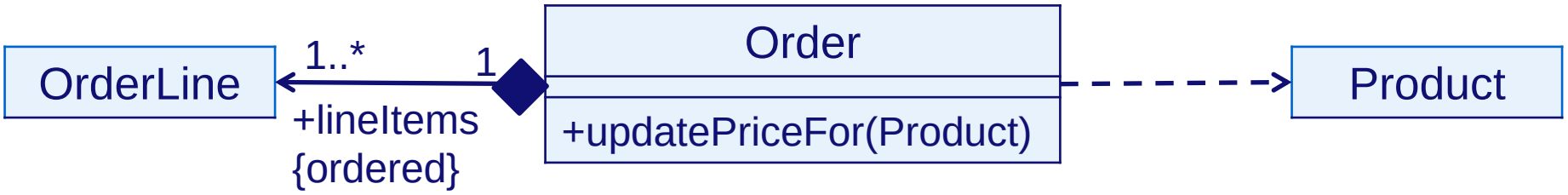
- Consider Checkers game.
- Draw the class diagram
  - Ignore attributes and operations





Back to Orders: another kind of relation

- **Dependency**: the source uses the target in order to realize its functionality (but does not include an instance of it)
  - Lots of dependencies clutter the diagram.
    - Less frequently shown than association, generalization, composition and aggregation
    - UML distinguishes more than 10 kinds of dependencies
      - Stereotypes are frequently omitted
  - *Usually*, target appears a parameter in source's operations.



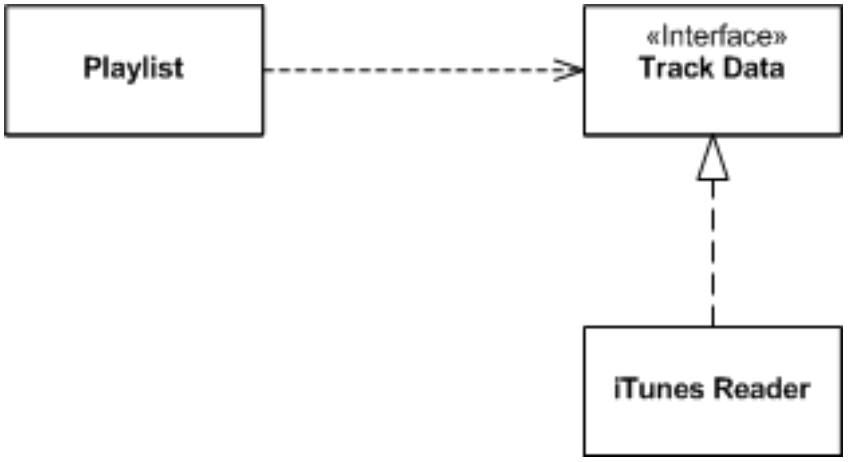
Examples of dependencies

- **<<create>>**
  - Order creates entries in the log book
- **<<use>>**
  - Order uses information about products
- **<<call>>**
  - Order calls a System operation to obtain the current time
- …

Cousin of an Abstract Class

- **Interface**: a class-like construct that contains only constants and abstract methods.
  - Directly correspond to interfaces in Java and C#
  - Subject to all relations we have seen
  - Classes **require** (dependency) interfaces or **provide** (implementation) interfaces

UML 1

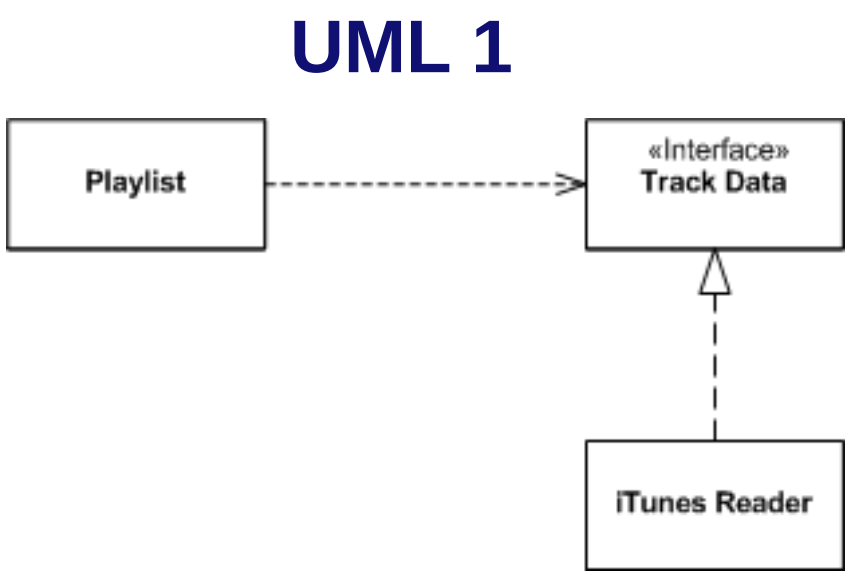


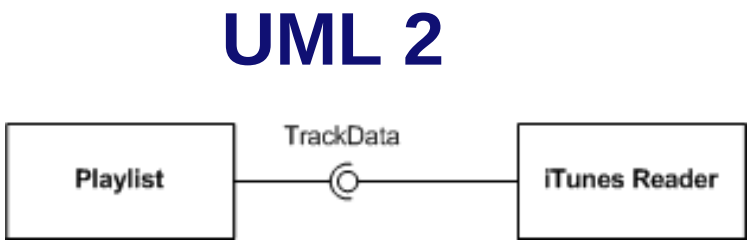
UML 2

http://martinfowler.com/bliki/BallAndSocket.html

Cousin of an Abstract Class

- What are **advantages** and **disadvantages** of the UML 2 “ball and socket” notation for interfaces?



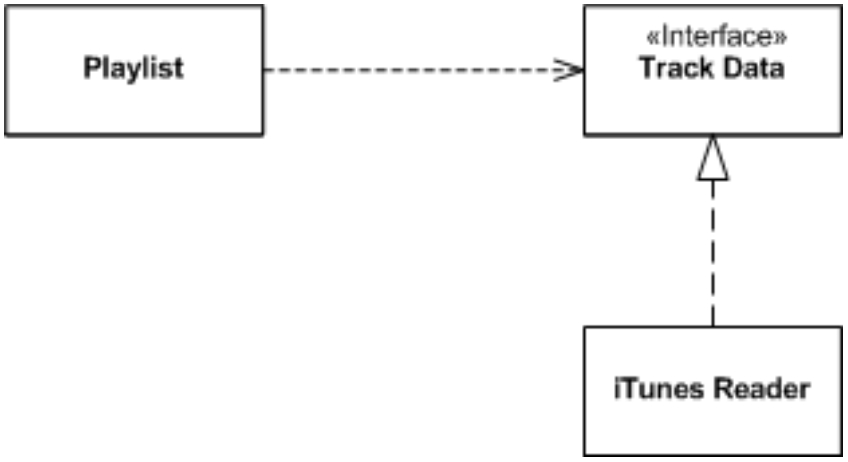


http://martinfowler.com/bliki/BallAndSocket.html

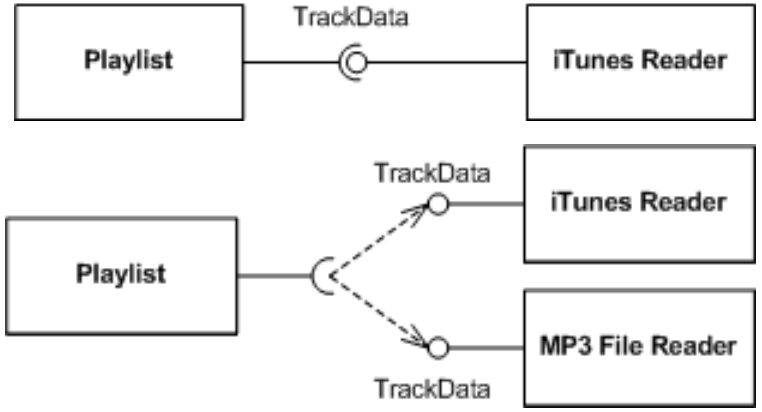
Cousin of an Abstract Class

- What are **advantages** and **disadvantages** of the UML 2 “ball and socket” notation for interfaces?
  - How to represent multiple classes providing the same interface?
    - Solution due to Jim Rumbaugh and Martin Fowler

UML 1



UML 2



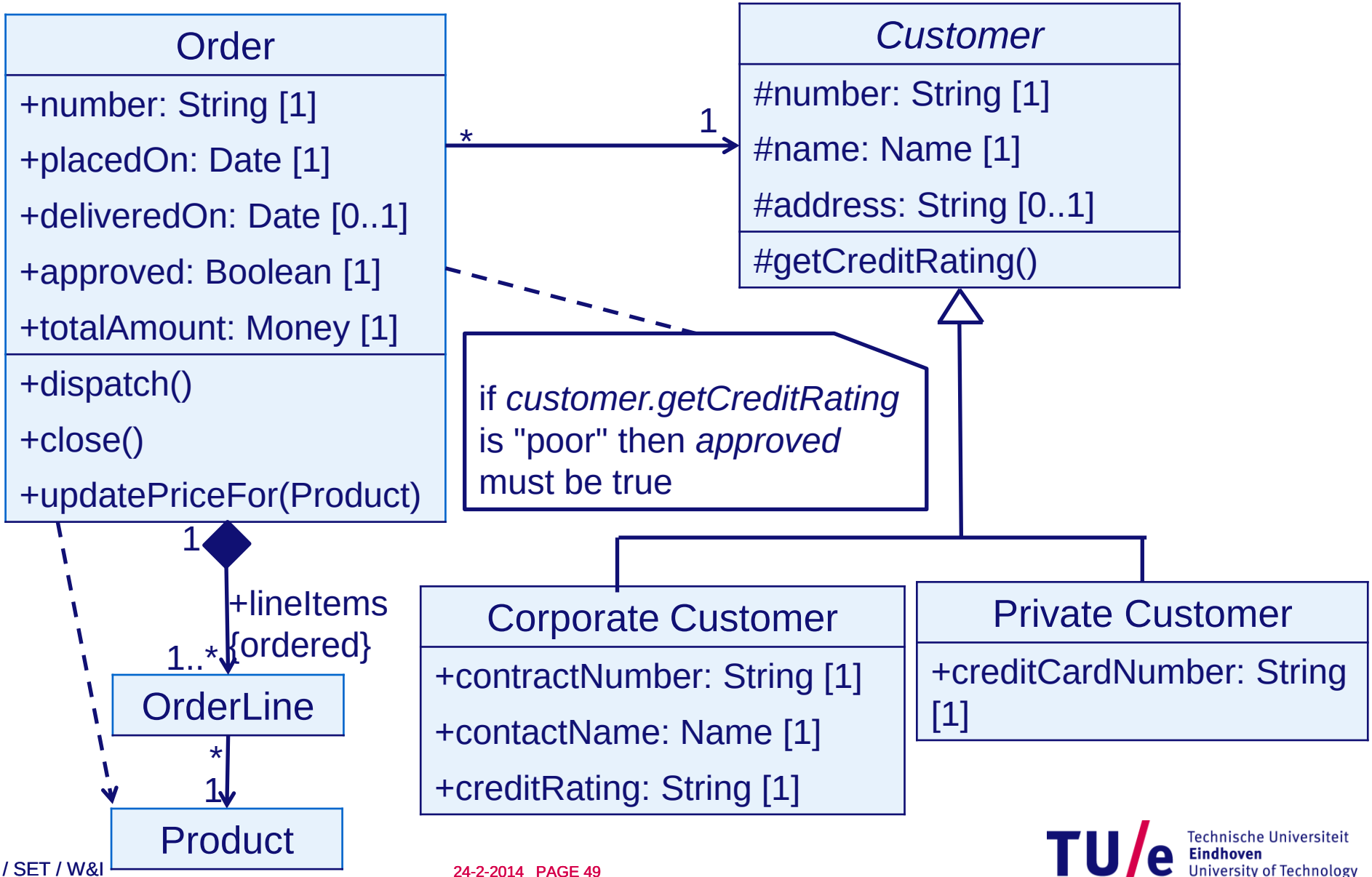
http://martinfowler.com/bliki/BallAndSocket.html

Balls, sockets and usability

When I first saw the mated ball and socket notation I rather liked it. Since then, however, I've found no great inclination to use it. For simple cases the UML 1 style with sockets works well, but when things get more complicated I prefer to have explicit class boxes for the interfaces.

[Martin Fowler, http://martinfowler.com/bliki/BallAndSocket.html]

Putting it all together



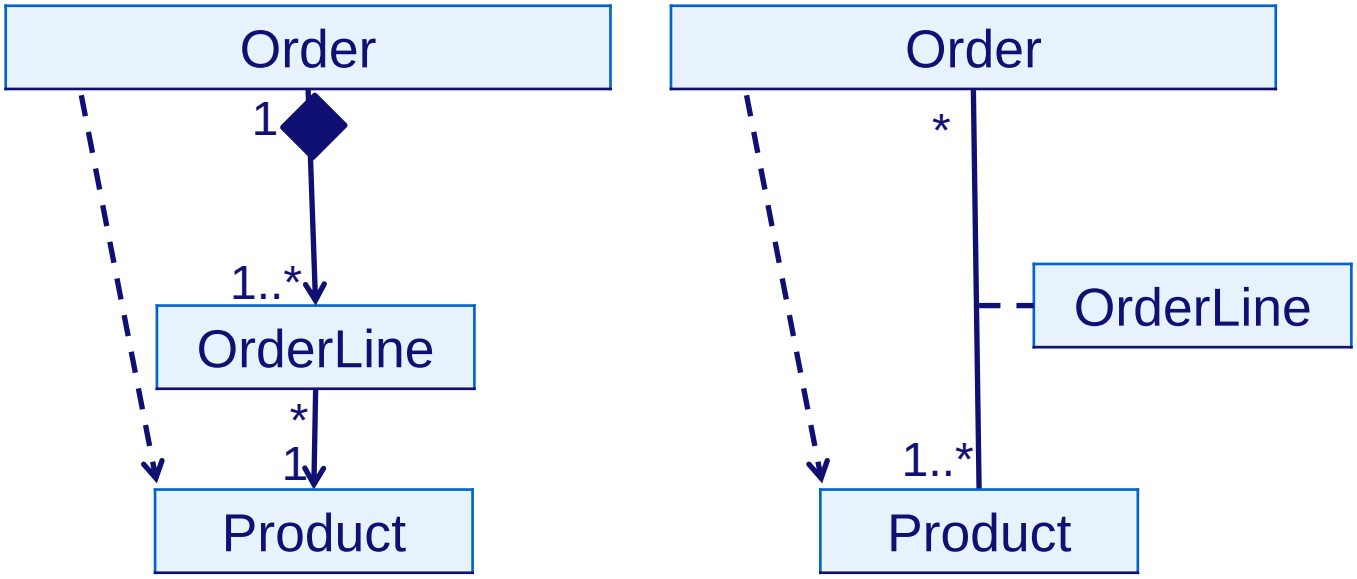
Advantages / disadvantages

- Looks nice but cluttered?
  - 6 classes only, 2 without attributes/operations

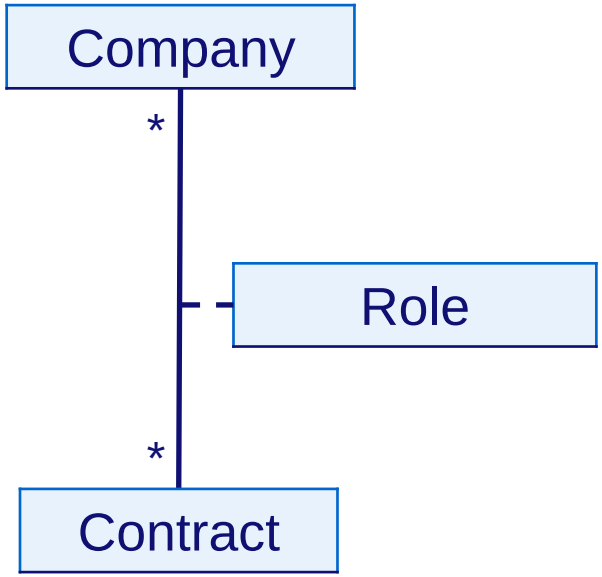
- How can we reduce the amount of information?
  - Reduce the number of classes
  - Reduce the number of associations / generalizations / compositions / aggregations
    - Association classes [see next slide]
  - Reduce information per association
    - Drop names of associations/roles unless meaningful
  - Reduce the number of attributes
  - Reduce information per attribute
    - Multiplicity of [1] can be assumed

Association class

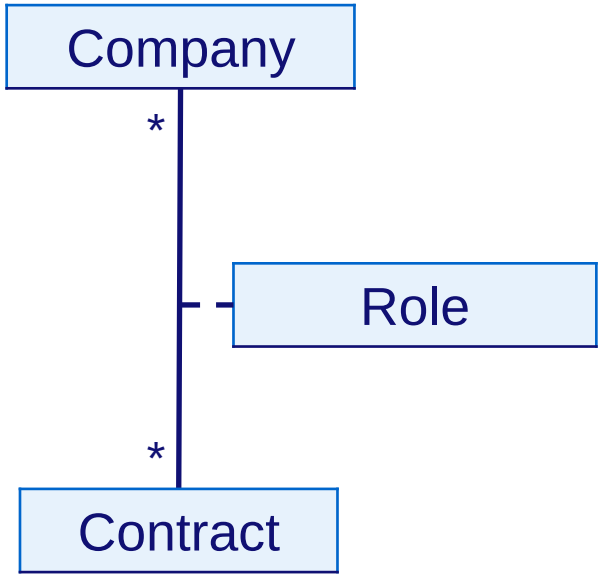
- OrderLine represents an association between Order and Product
  - No multiplicity: one instance per association
  - No additional associations



Correct or incorrect?



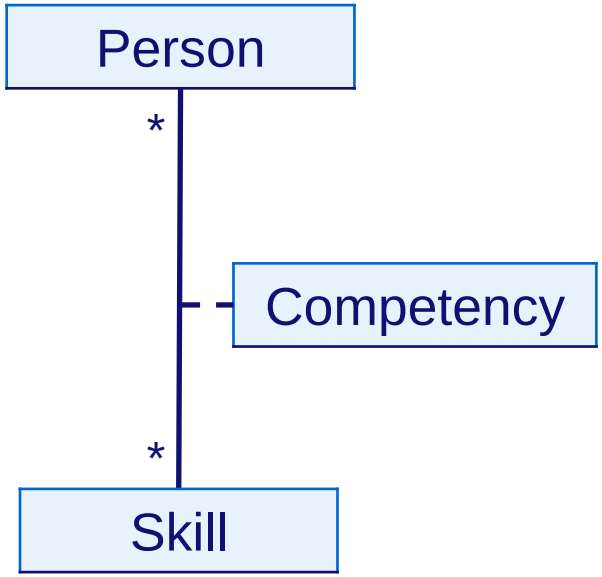
Correct or incorrect?



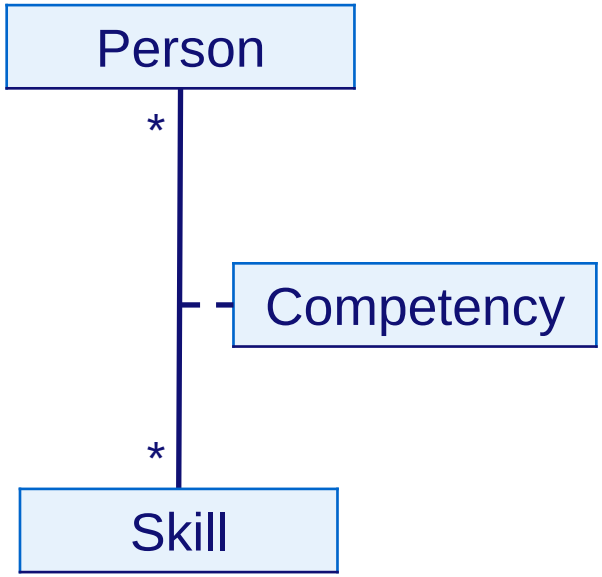
No, companies can play multiple roles in the same contract:

e.g., supplier and maintenance provider

Correct or incorrect?



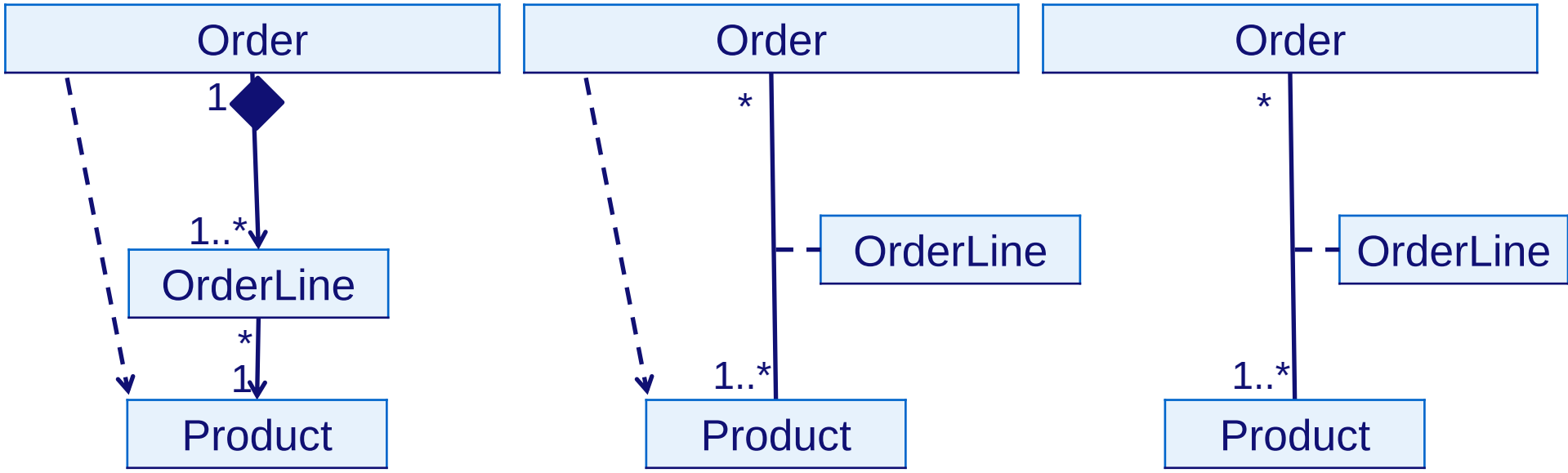
Correct or incorrect?



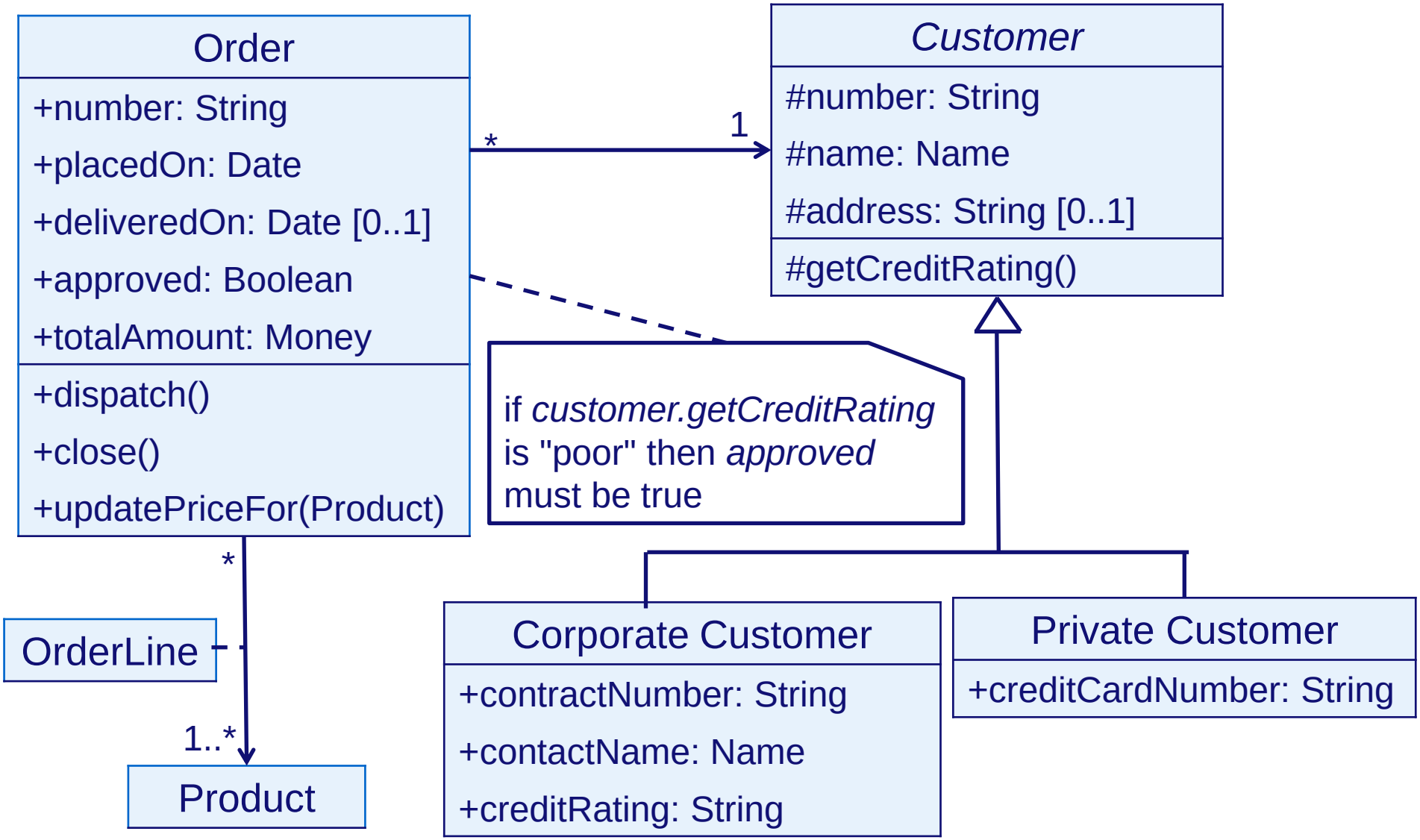
Yes, for each skill a person would typically have only one competency level.

Further simplification

- Association is **stronger** than dependency
- We do not need to show both!



Simplifying the diagram



Example: EasyShop system

Ellen and Portia live in Los Angeles.

Ellen is a TV-host and stand-up comedian; Portia is an actress and both have a very busy schedule.

To schedule their shopping and household activities: EasyShop.

EasyShop: business needs

Ellen and Portia should be able

- to enter their presence for meals for each day
- to register invitations for guests for each meal
- to fix a menu for cold meals with given ingredients
- to choose whether they will cook in their agenda, and if so, fix a recipe for each warm meal
- to prepare a weekly shopping list, and fax the shopping list to the supermarket for delivery
- to keep track of ingredients in the kitchen; give a report when needed
- to show recipes (e.g., when cooking a dish)

What do we model?

- Identify the **actors**

Candidate classes

1. Identify noun phrases
2. Ignore actors

Ellen and Portia should be able

- to enter their presence for meals for each day
- to register invitations for guests for each meal
- to fix a menu for cold meals with given ingredients
- to choose whether they will cook in their agenda, and if so, fix a recipe for each warm meal
- to prepare a weekly shopping list, and fax the shopping list to the supermarket for delivery
- to keep track of ingredients in the kitchen; give a report when needed
- to show recipes (e.g., when cooking a dish)

Candidate classes

1. Identify nouns (noun phrases)
2. Ignore *actors*
3. Omit repetitions

Ellen and *Portia* should be able

- to enter **their presence** for **meal**s for each **day**
- to register **invitation**s for **guest**s for each meal
- to fix a **menu** for **cold meal**s with given **ingredient**s
- to choose whether they will cook in their **agenda**, and if so, fix a **recipe** for each **warm meal**
- to prepare a weekly **shopping list**, and fax the shopping list to the *supermarket* for **delivery**
- to keep track of ingredients in the **kitchen**; give a **report** when needed
- to show recipes (e.g., when cooking a dish)

Candidate classes

1. Identify nouns (noun phrases)
2. Ignore *actors*
3. Omit repetitions

Ellen and *Portia* should be able

- to enter **their presence** for **meal**s for each **day**
- to r[illegible]**vitations** for **guests** for each meal
- to [illegible] given **ingredient**s
- to choose whether they will cook in their **agenda**, and if so, fix a **recipe** for each **warm meal**
- to prepare a weekly **shopping list**, and fax the shopping list to the *supermarket* for **delivery**
- to keep track of ingredients in the **kitchen**; give a **report** when needed
- to show recipes (e.g., when cooking a dish)

refers to Ellen and Portia, ignore

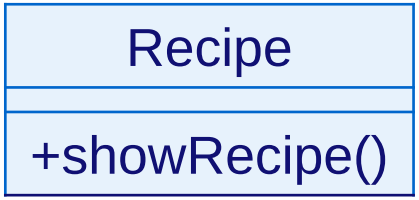
Sketching the class diagram step by step

- **meal**s for each **day**

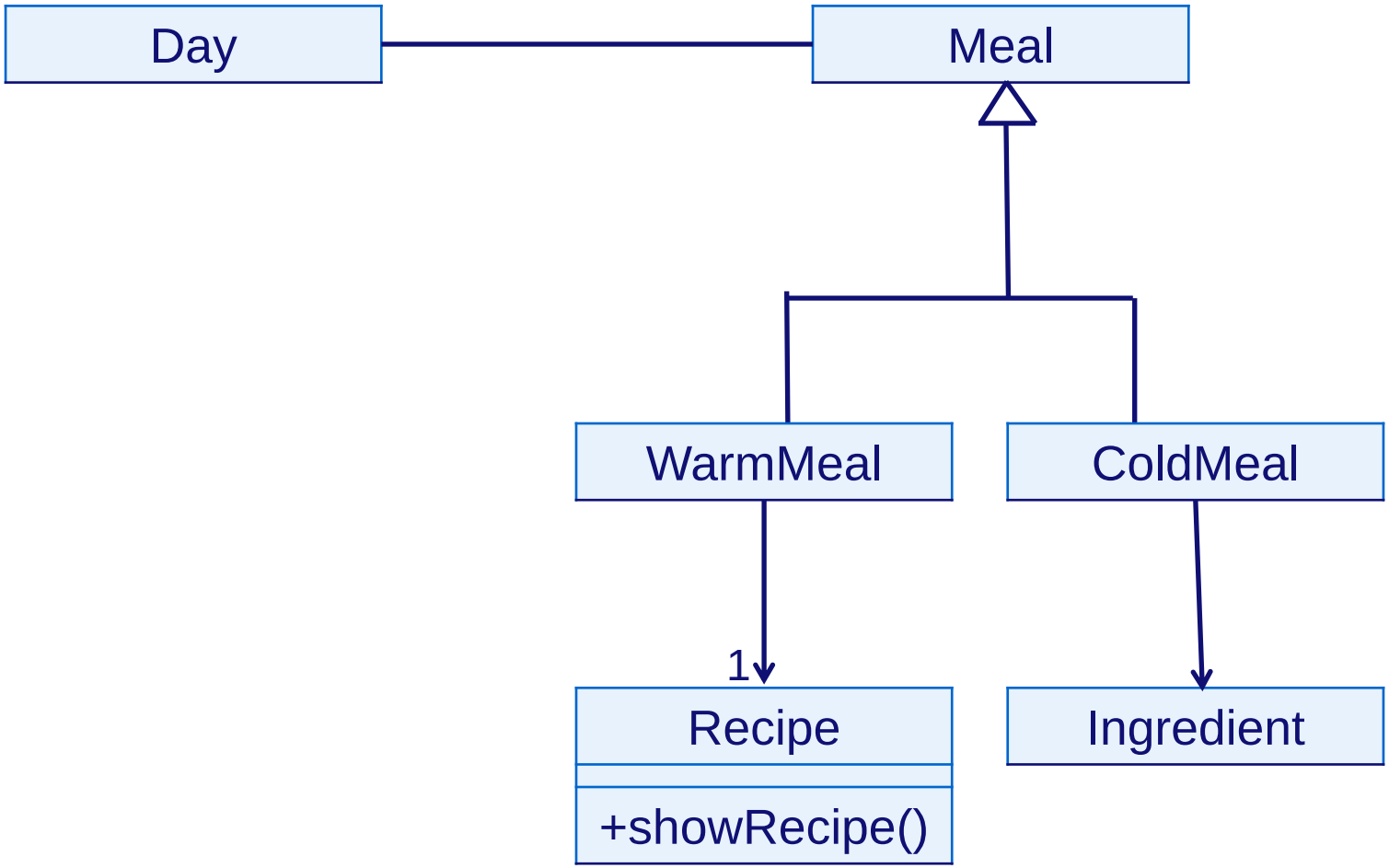
- **menu** for **cold meal**s with given **ingredient**s
  - menu should be recorded in ColdMeal

- fix a **recipe** for each **warm meal**

- to show recipes

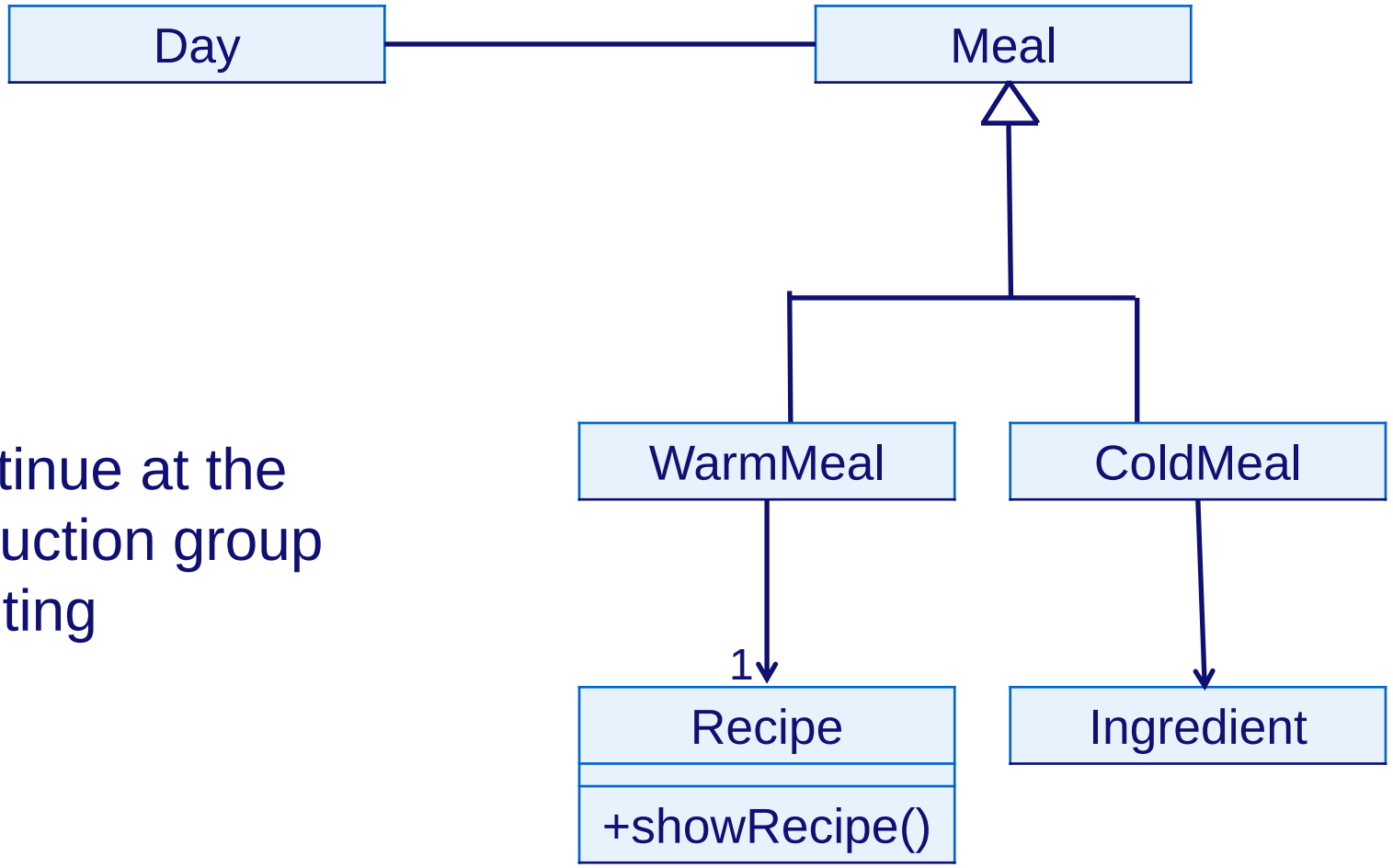


So far


Day
Meal
WarmMeal
ColdMeal
1
Recipe
+showRecipe()
Ingredient

So far

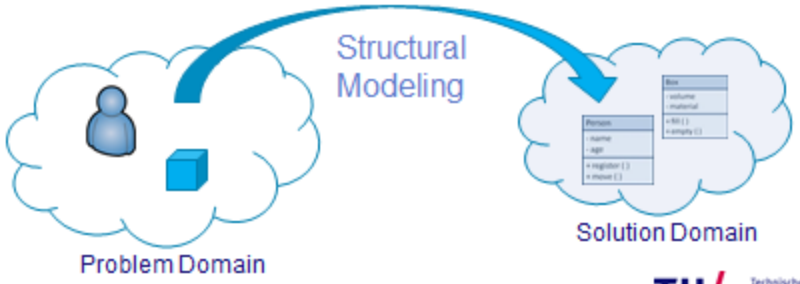
Continue at the instruction group meeting



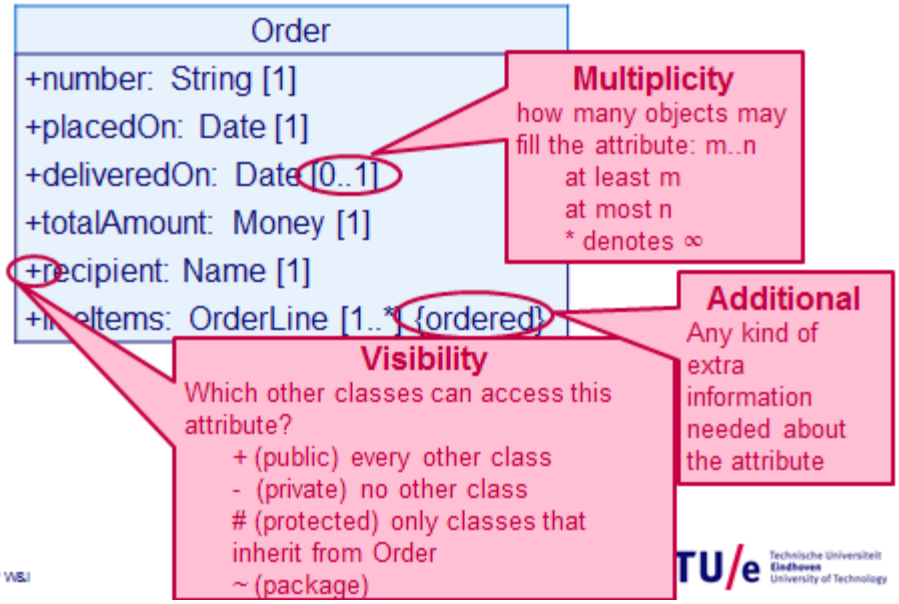
What is structure?

- **Structure:** the aggregate of elements of an entity in their relationships to each other. *Merriam-Webster*

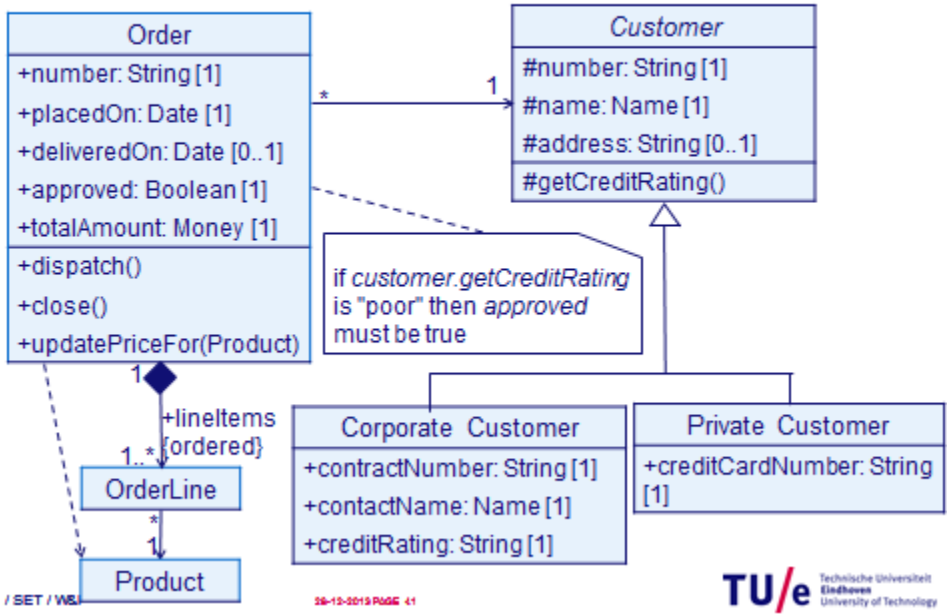
- **Structural modelling:** discover the key data contained in the problem domain and to build a structural model of the objects



How would we write this in UML?



Putting it all together



Technische Universiteit Eindhoven University of Technology