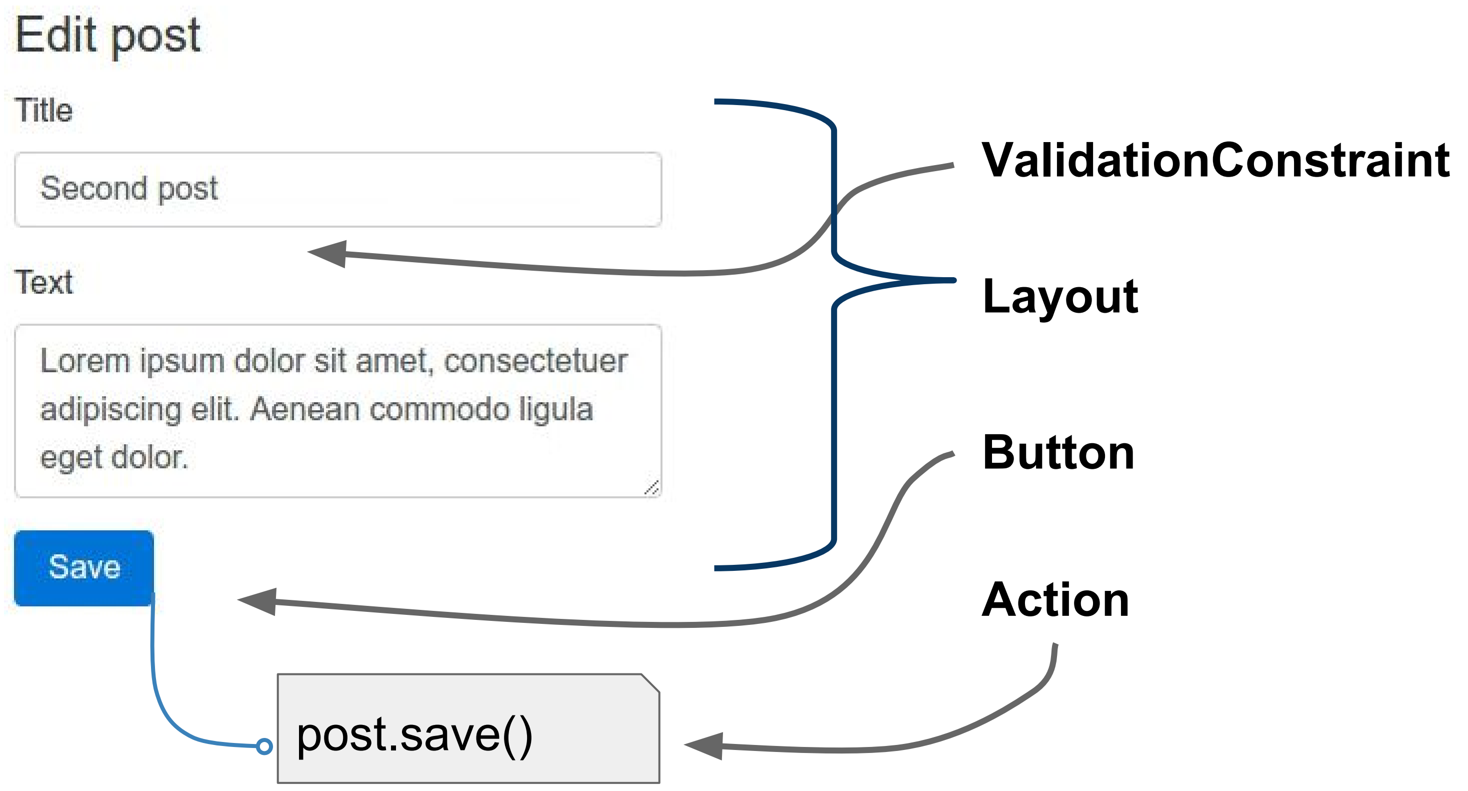
What’s the point of Object Orientation?

Iwan Vosloo

reahl™

Introduction

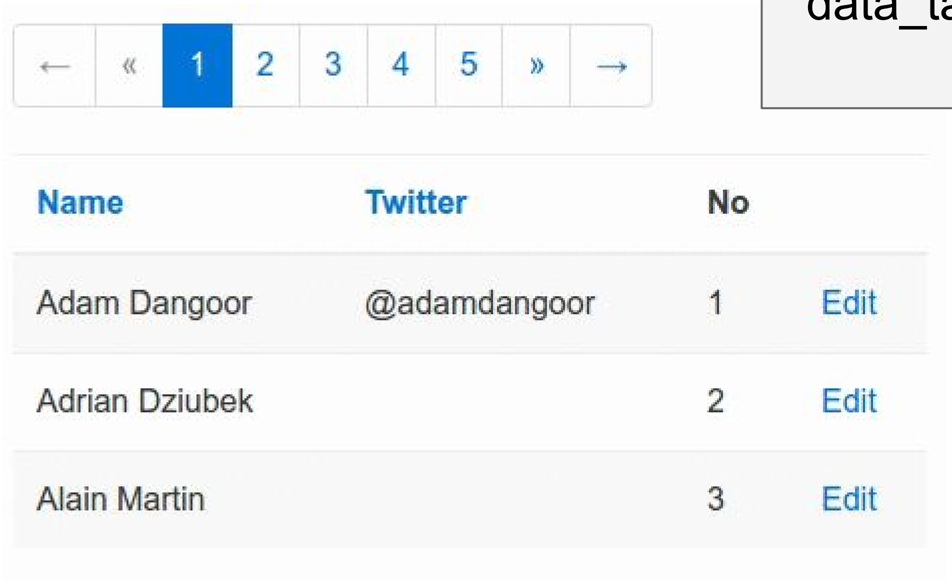
About


Edit post
Title
Second post
Text
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor.
Save
post.save()
ValidationConstraint
Layout
Button
Action

https://goo.gl/3DMHFH

Introduction

About



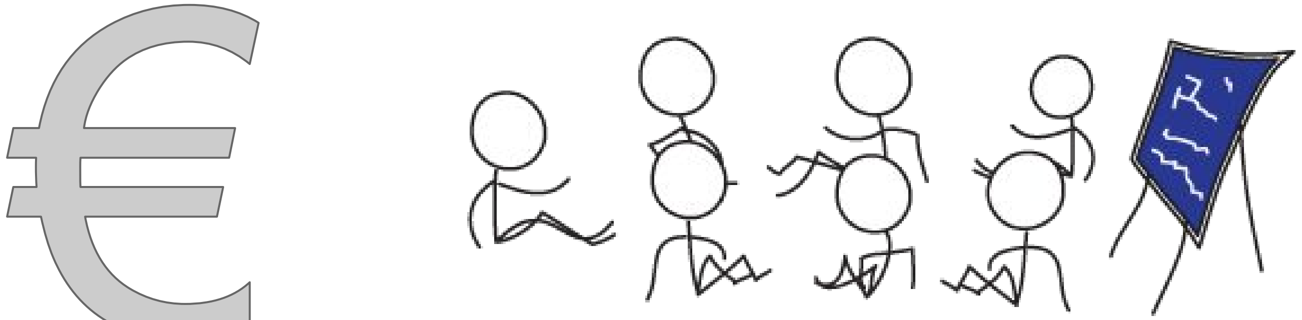
Name	Twitter	No	
Adam Dangoor	@adamdangoor	1	Edit
Adrian Dziubek		2	Edit
Alain Martin		3	Edit

```
data_table = DataTable(view,  ….. )
```

https://goo.gl/3DMHFH

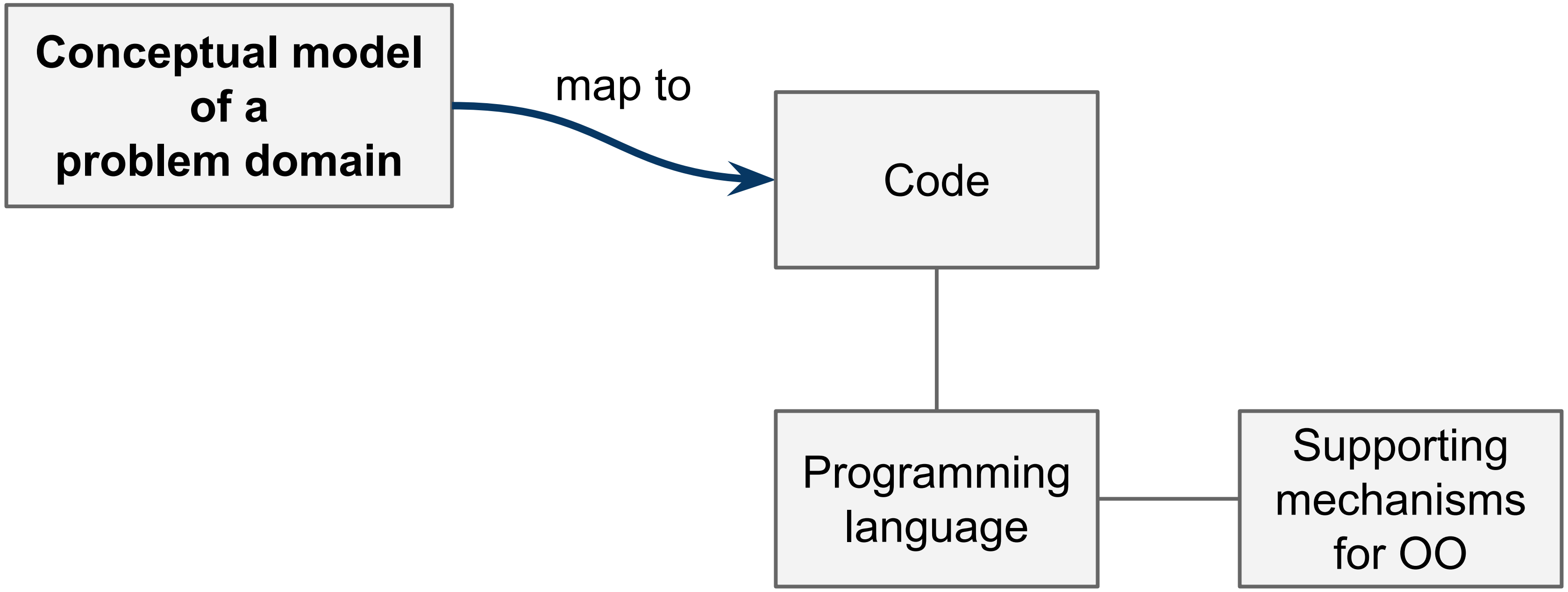
Introduction

About / 2



Introduction

OOP is not OO

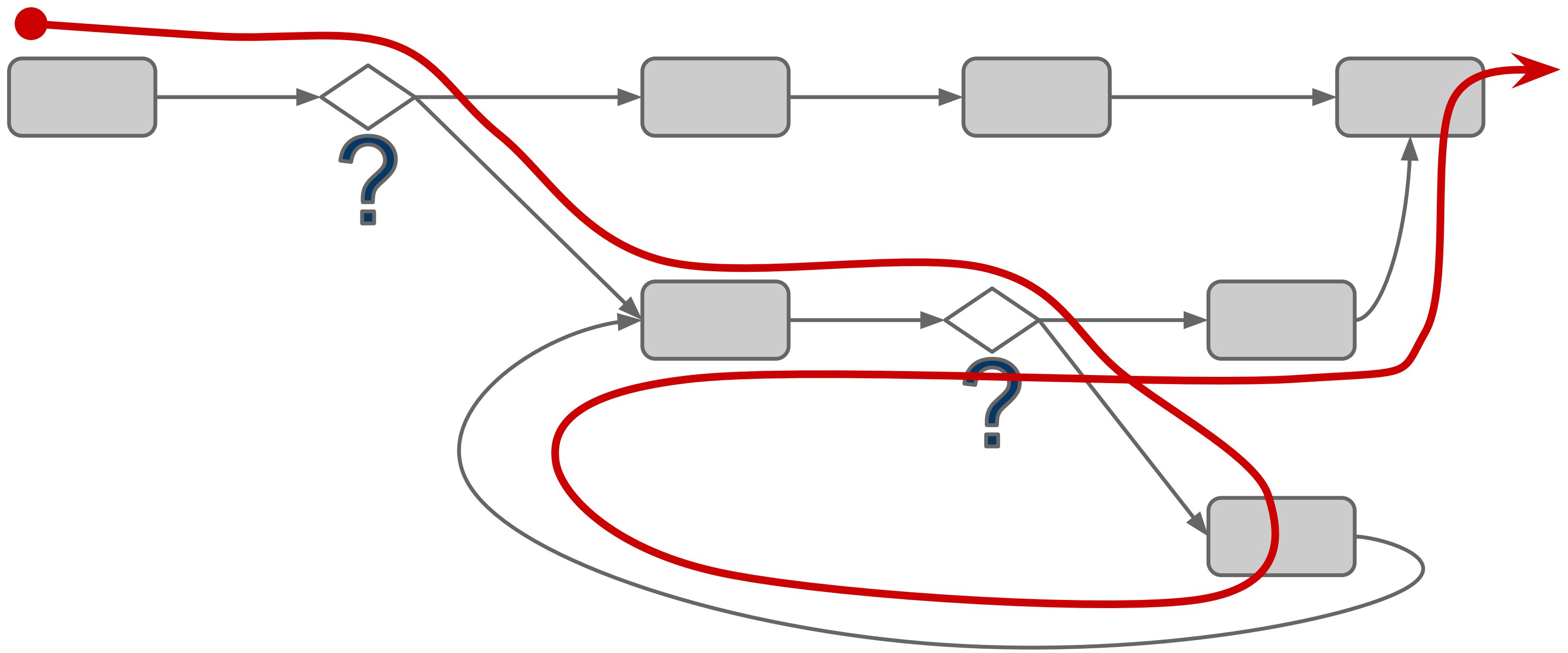

Conceptual model of a problem domain
map to
Code
Programming language
Supporting mechanisms for OO

Introduction

Cause and effect

Introduction

Understand a program?

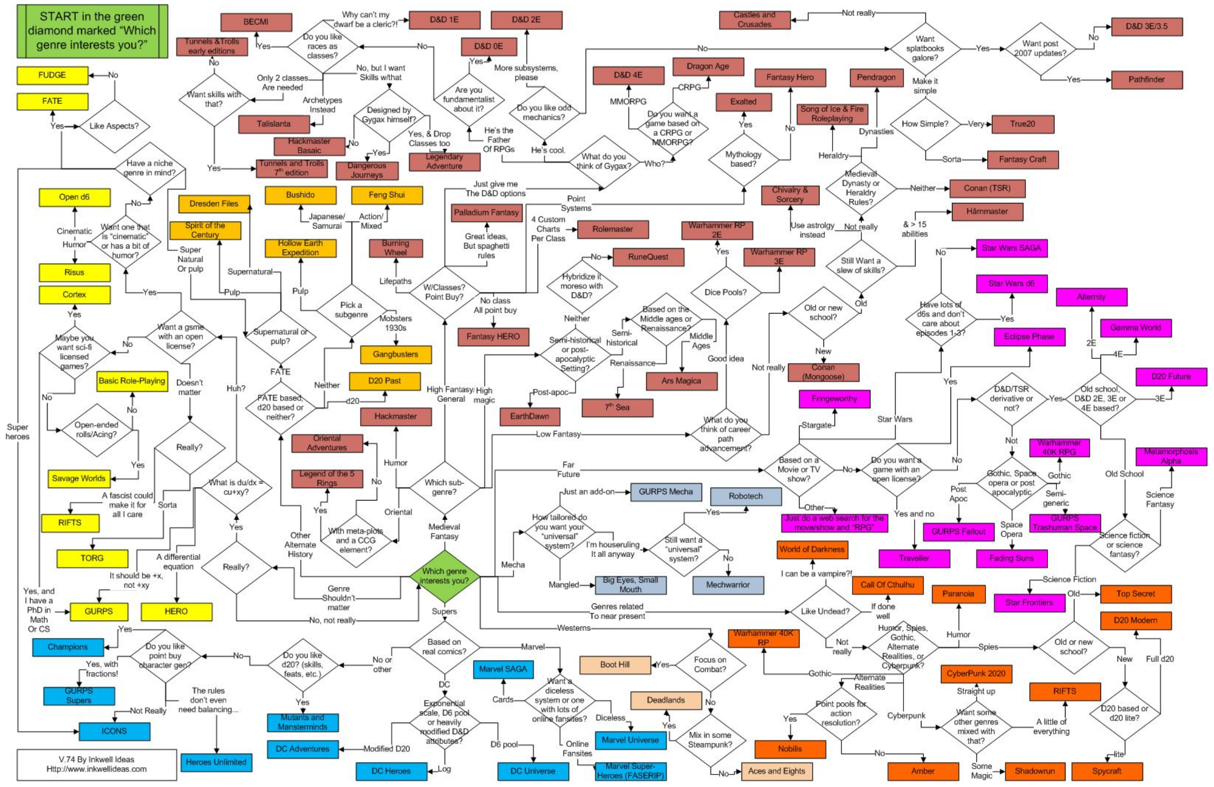


Introduction

Scaling up

reahl™

100
lines
of code?



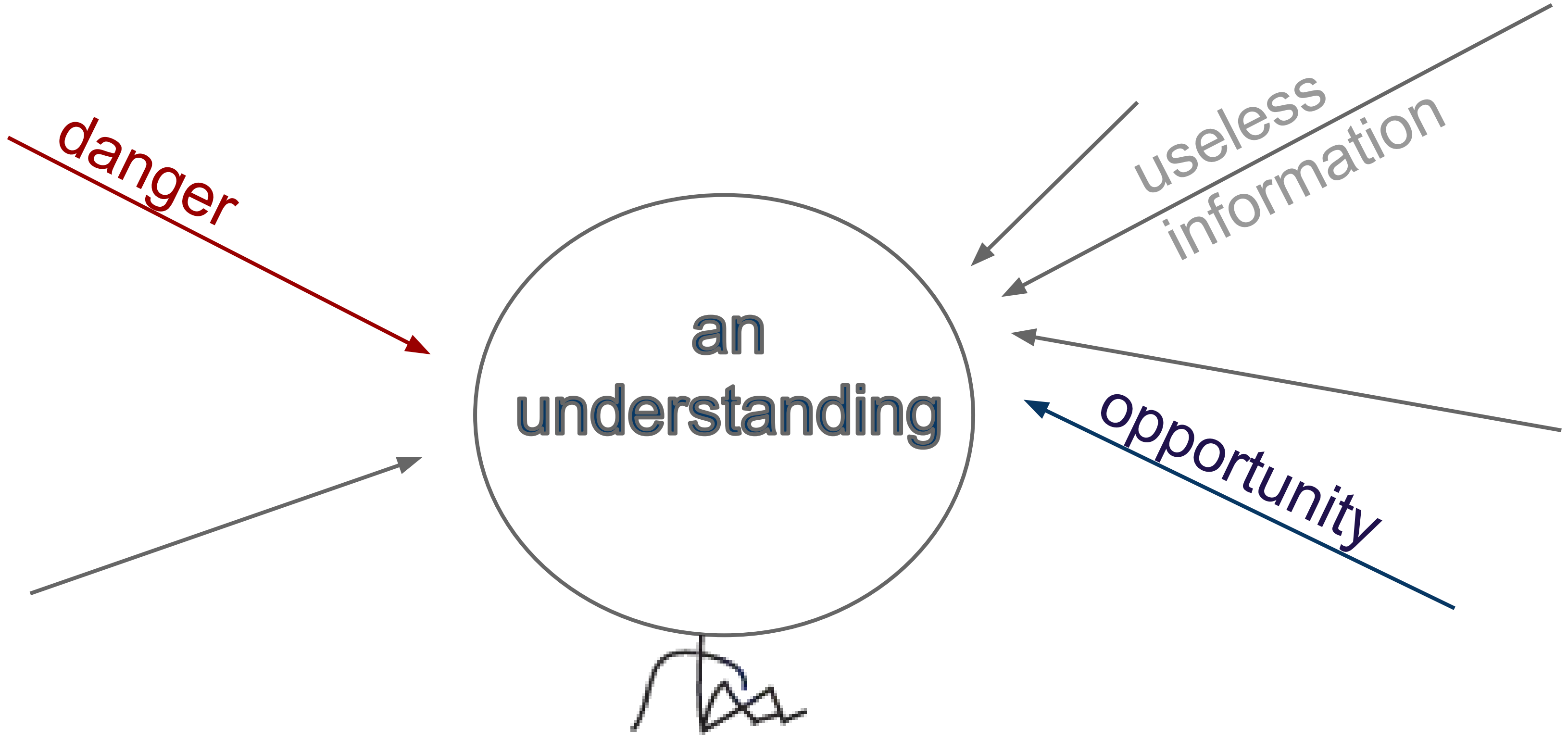
Courtesy: inkwellideas.com

How about...

180 000
lines
of code?

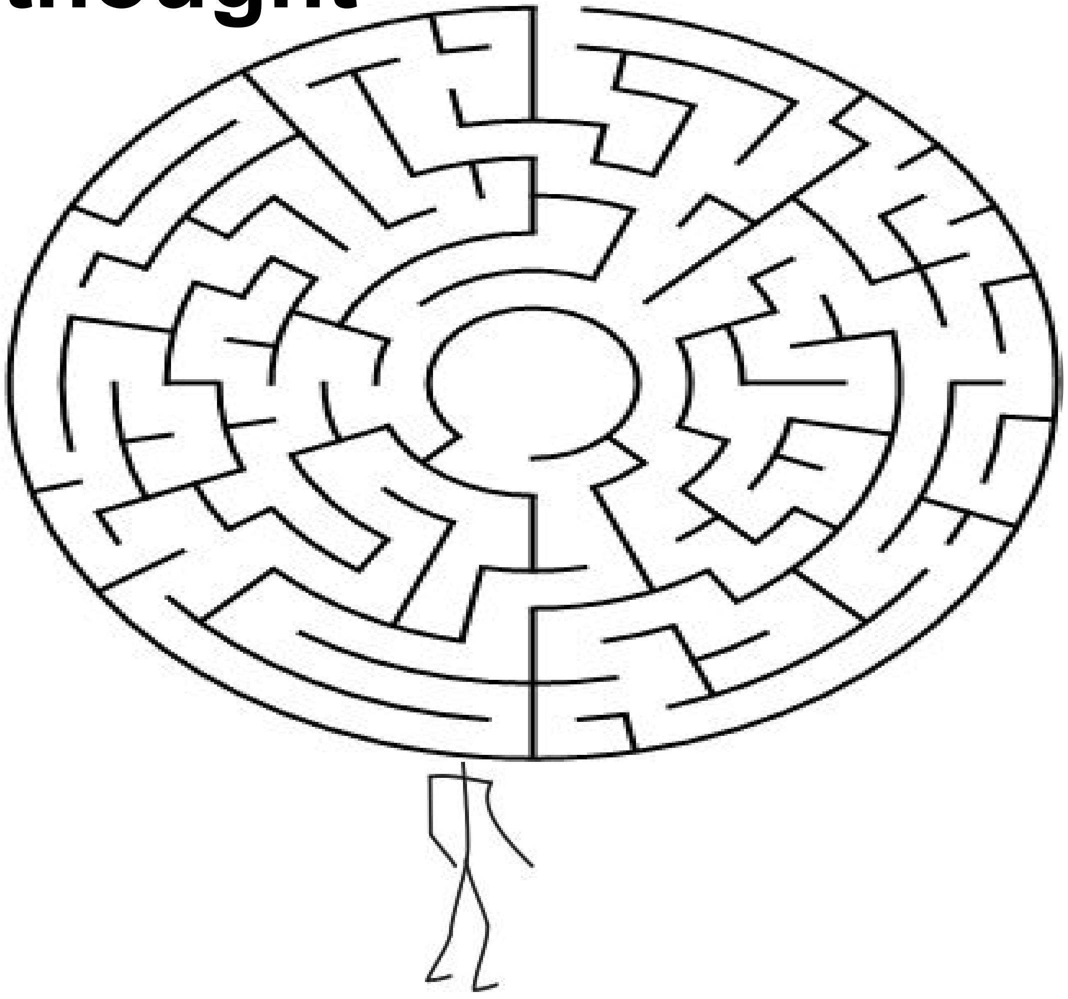
Introduction

Understanding



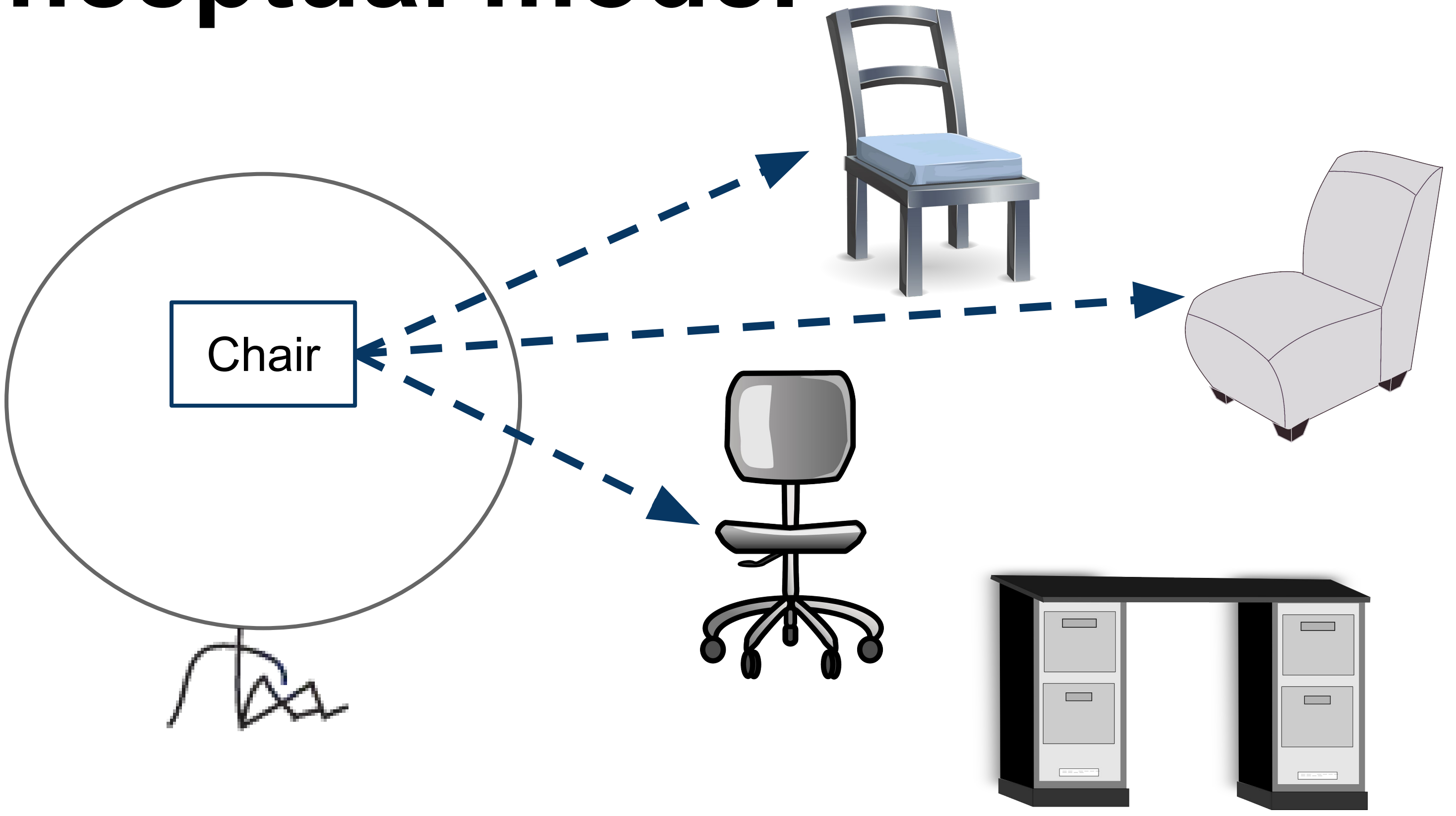
Introduction

A scary thought

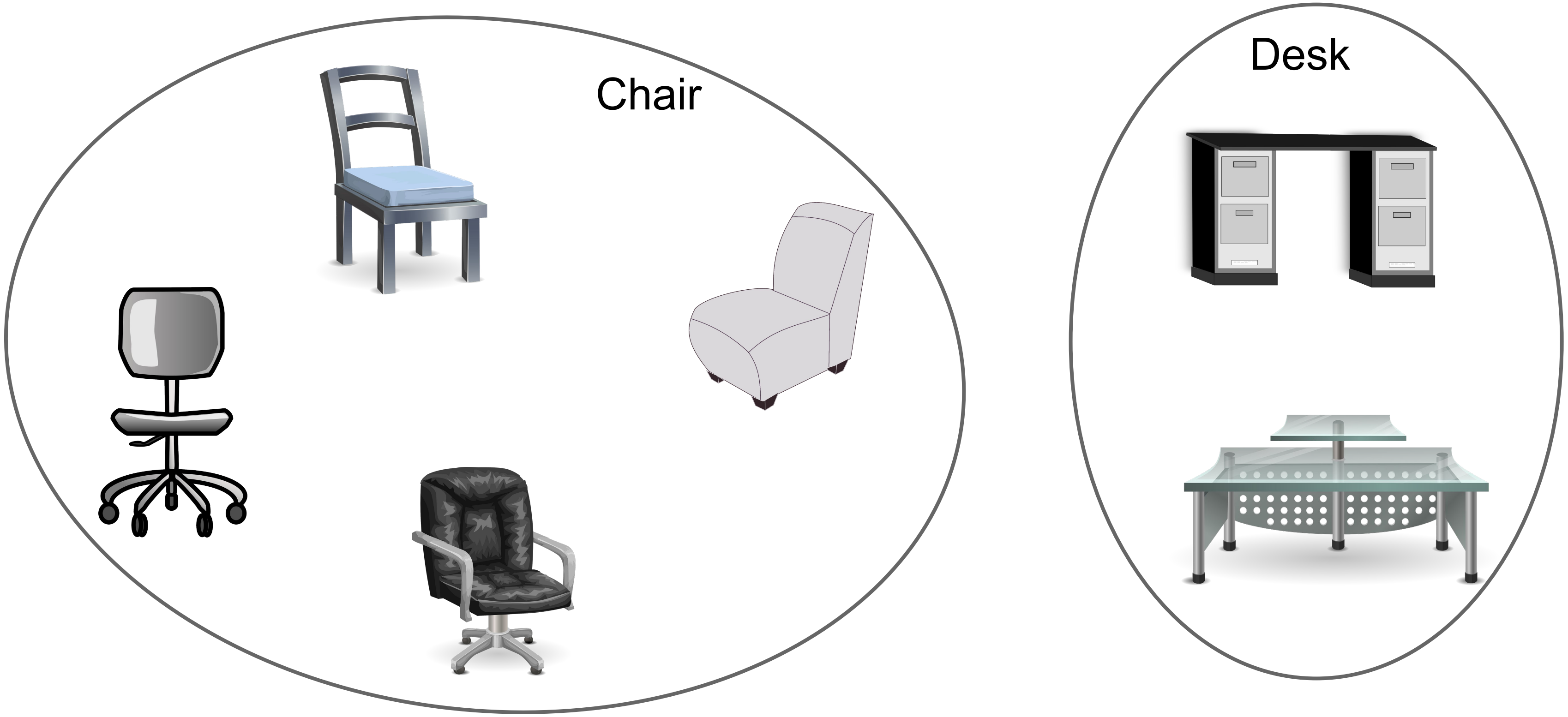


Concepts

A conceptual model

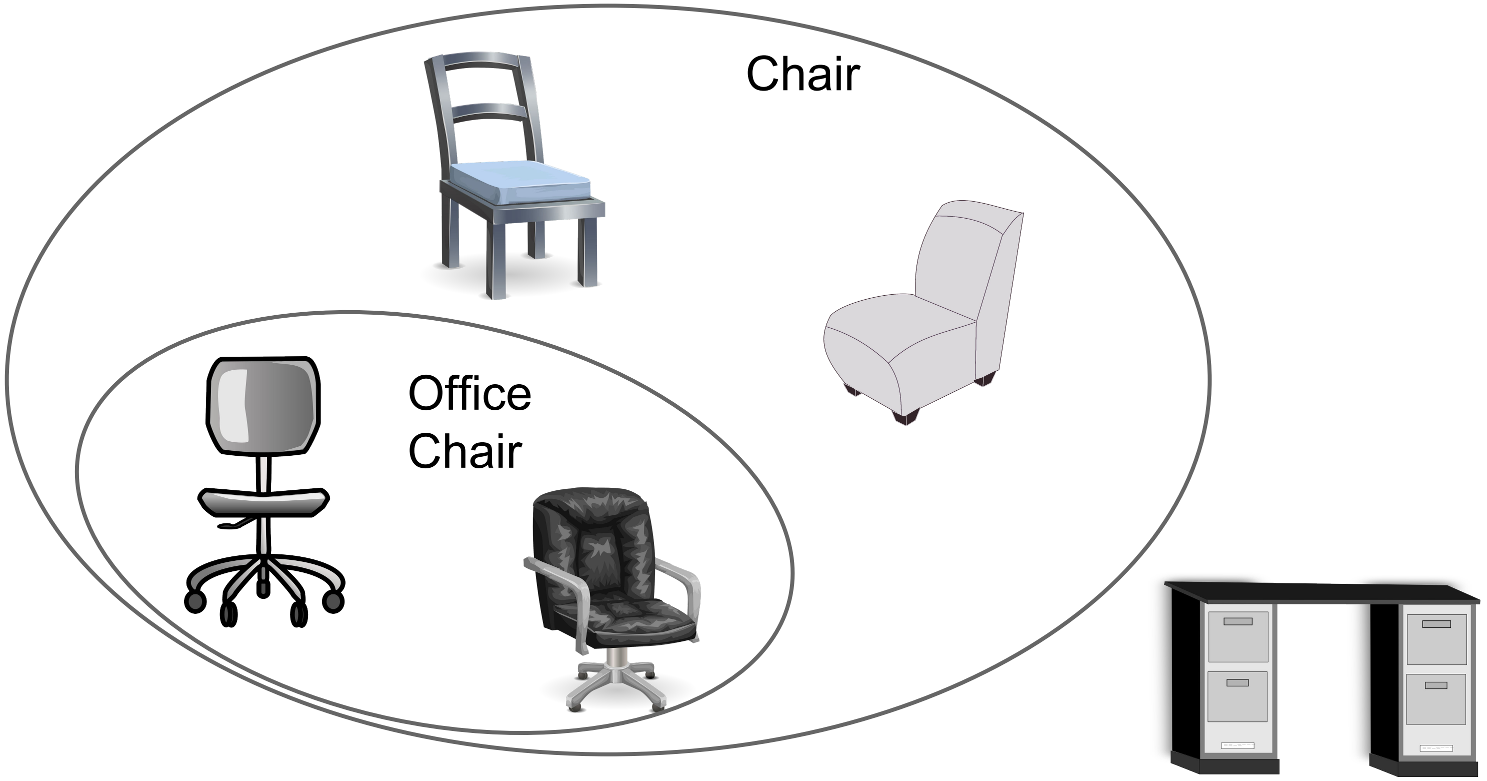


Concepts as sets



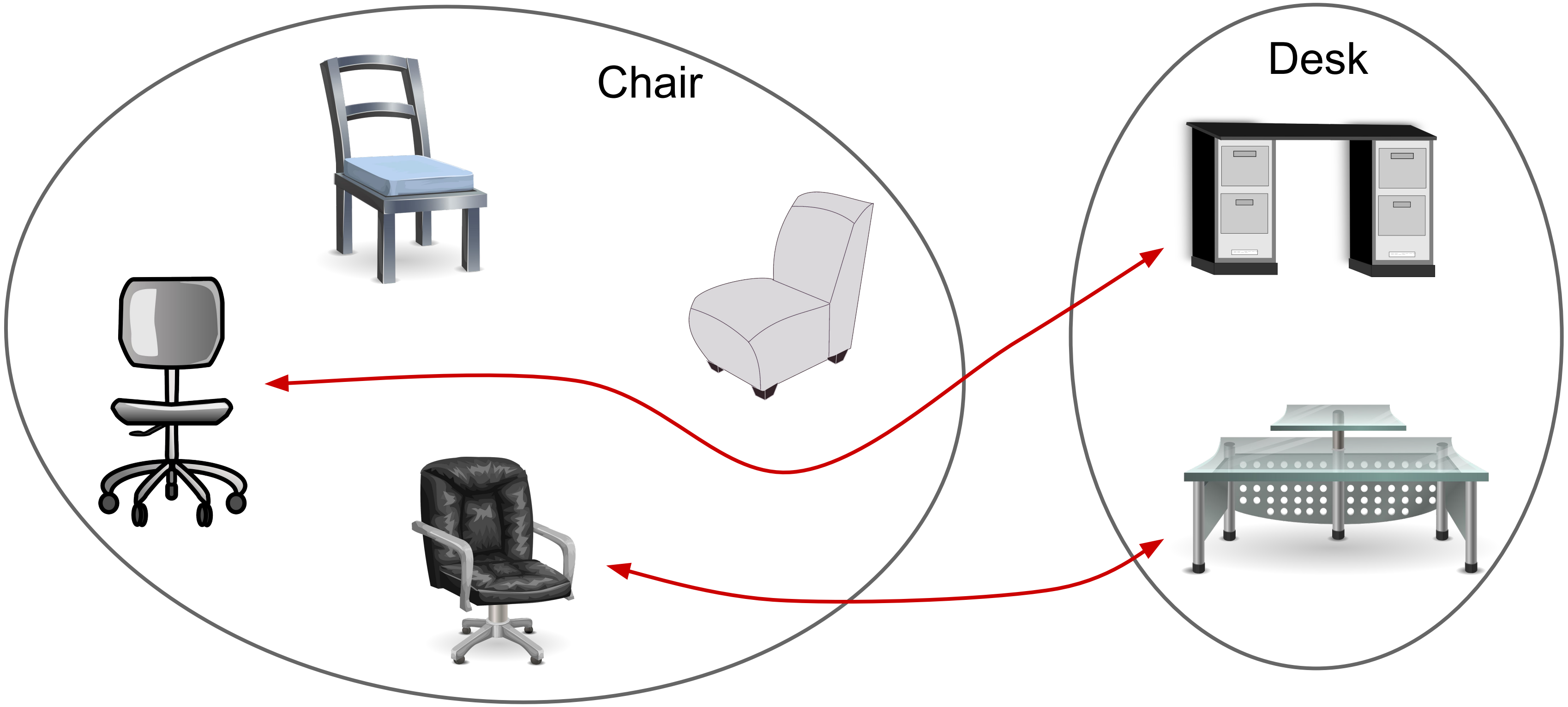
Concepts

Refinement via subsets



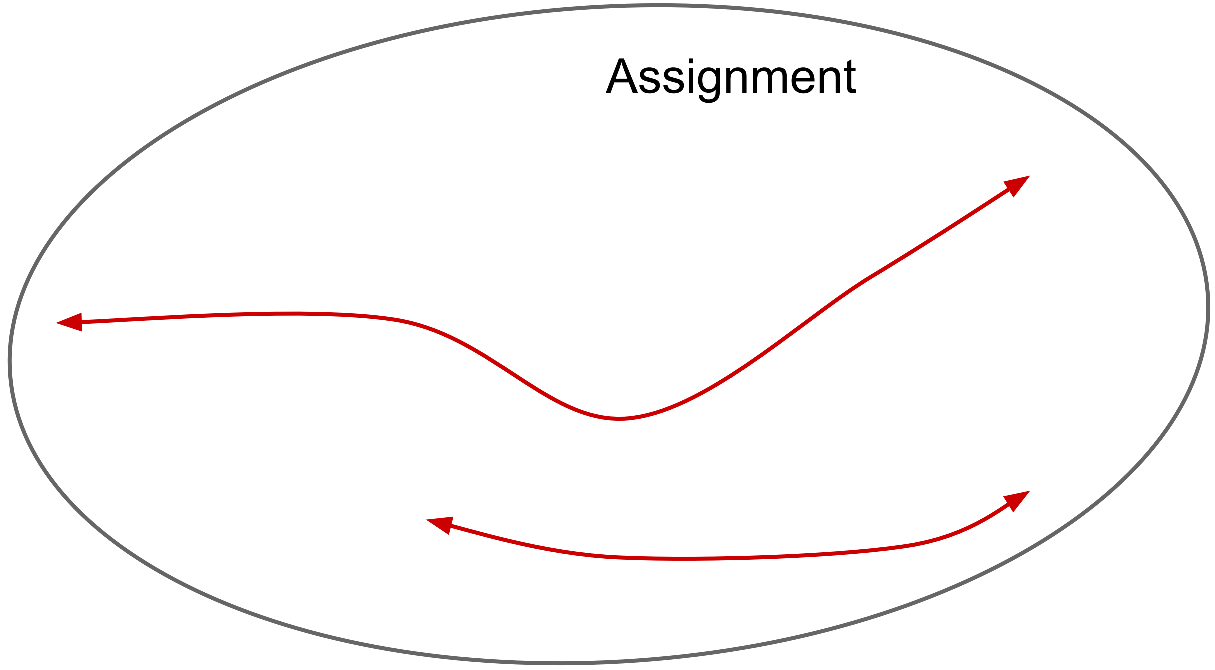
Relations

Connecting objects



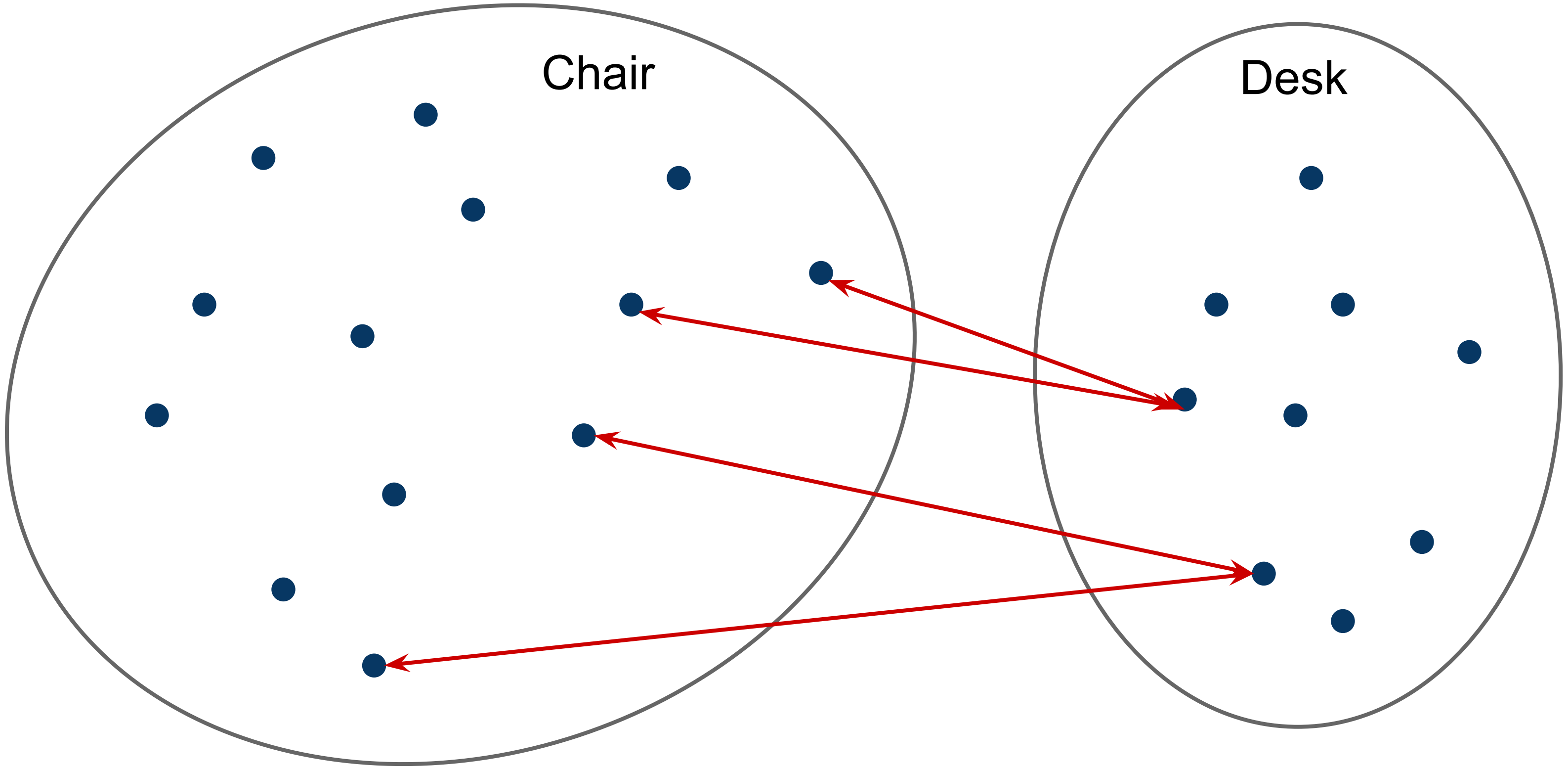
Relations

Relation as a concept



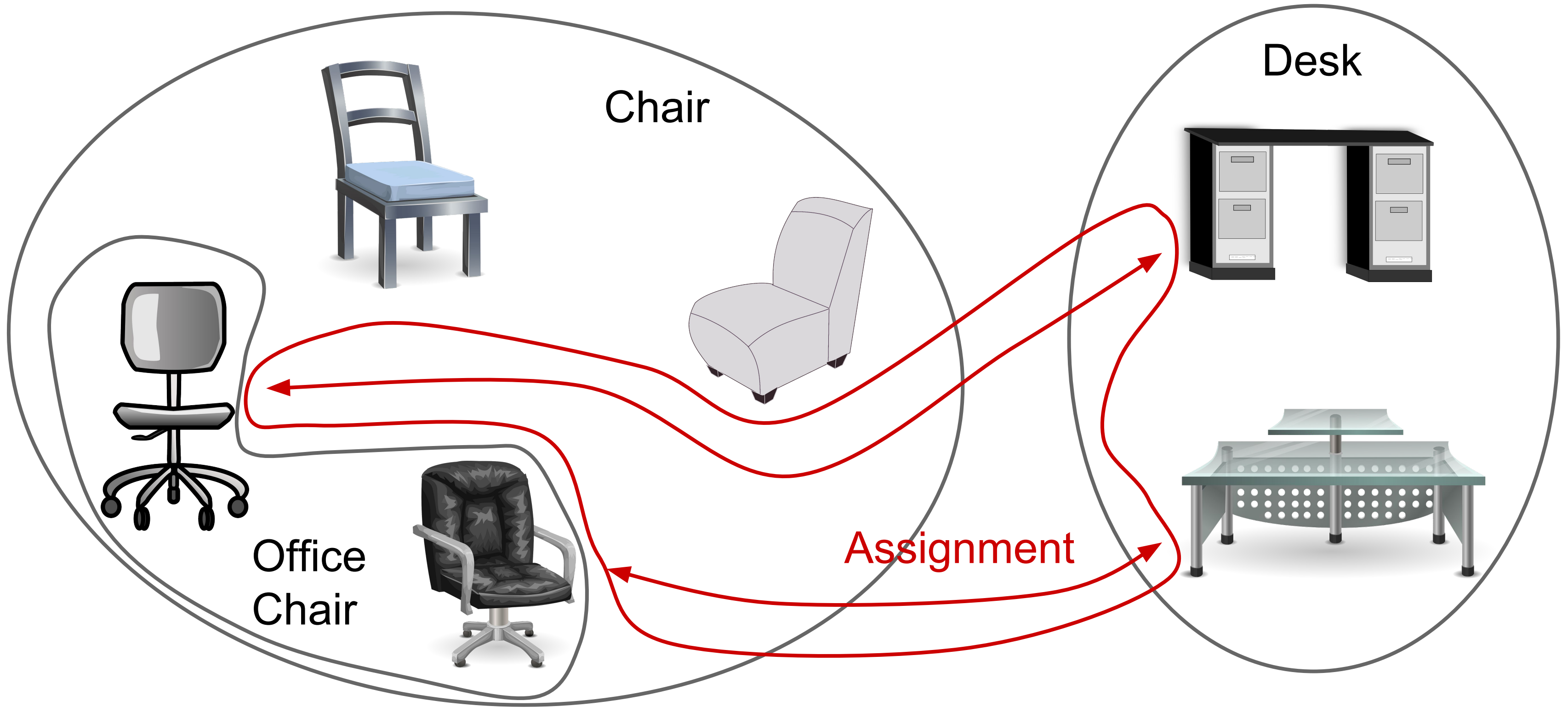
Relations

Multiplicity

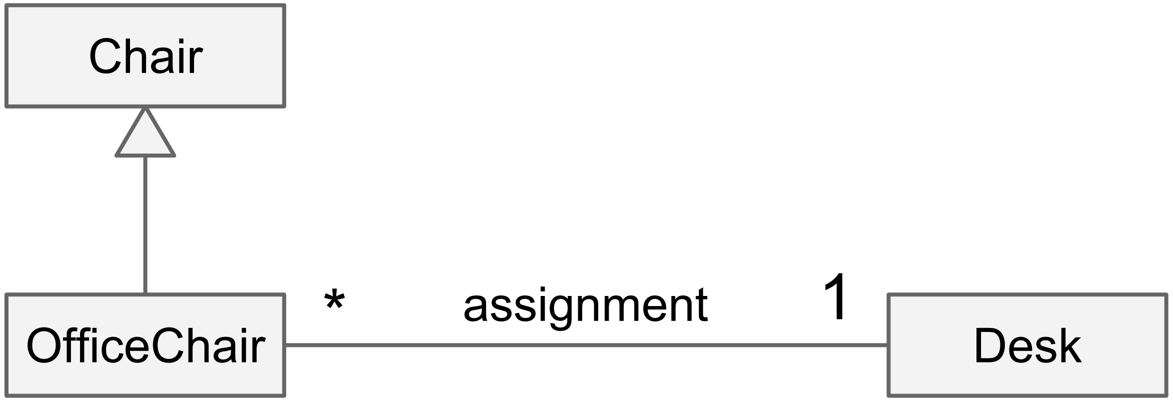


Notation

It gets complicated...

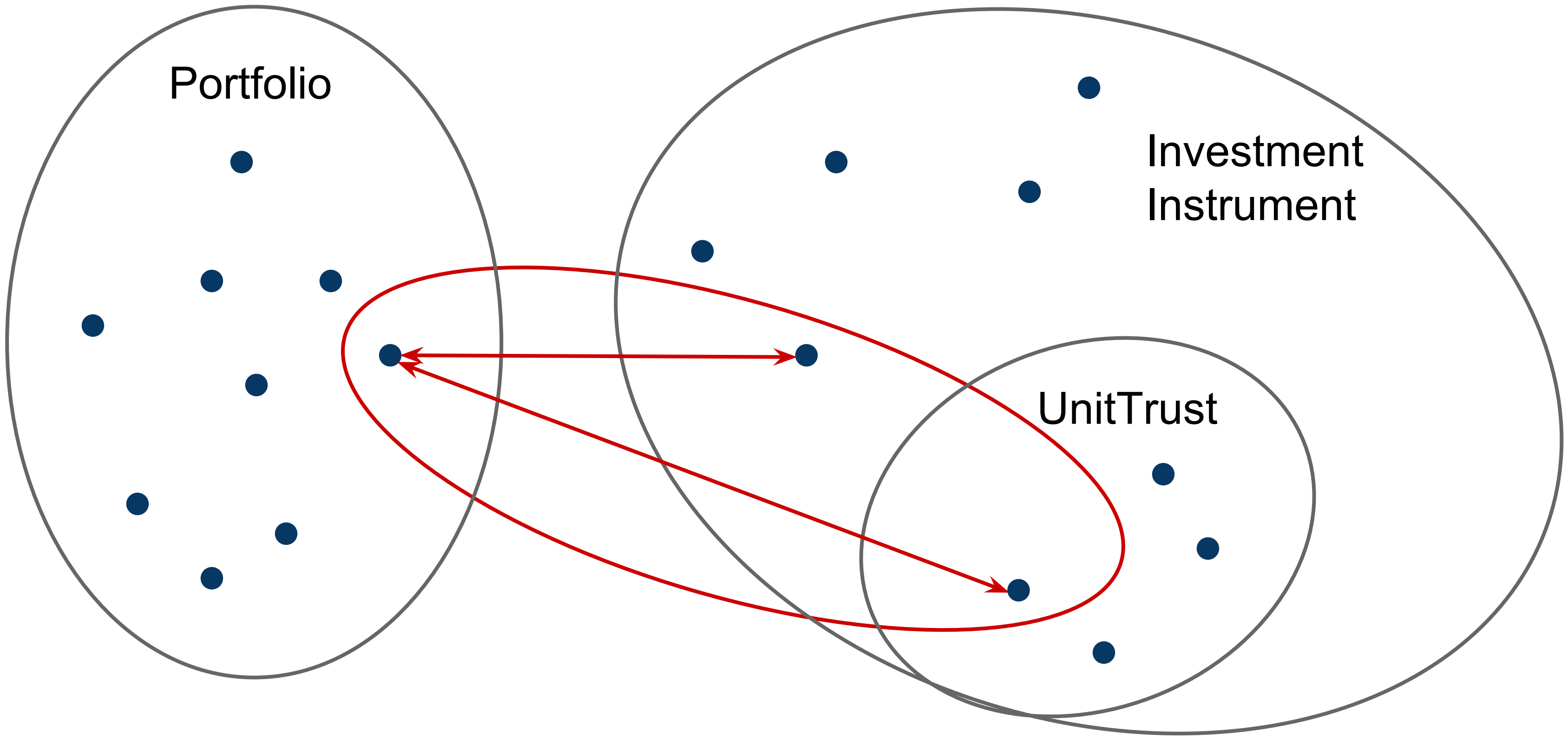


Notation
UML



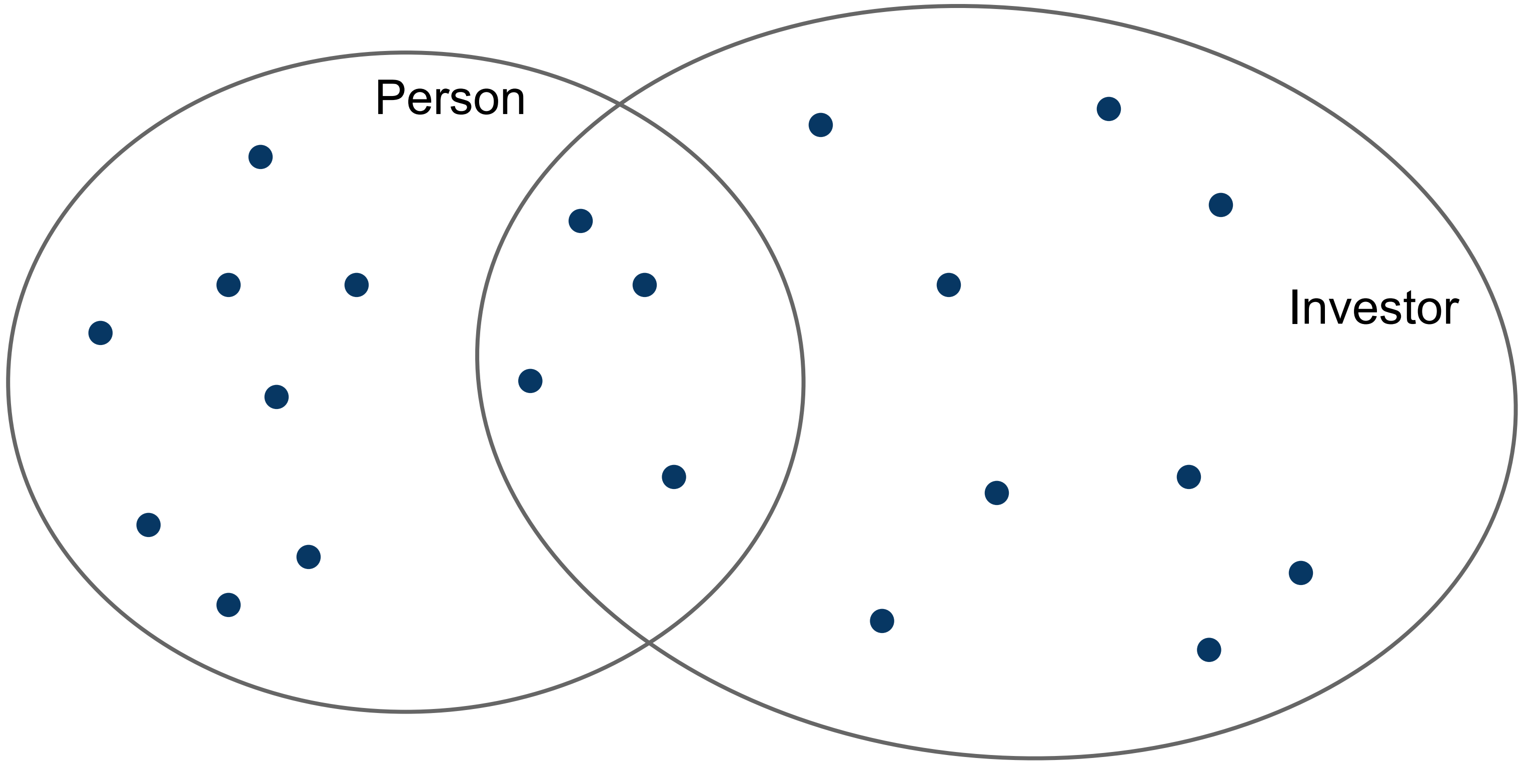
More on concepts

Intangible concepts



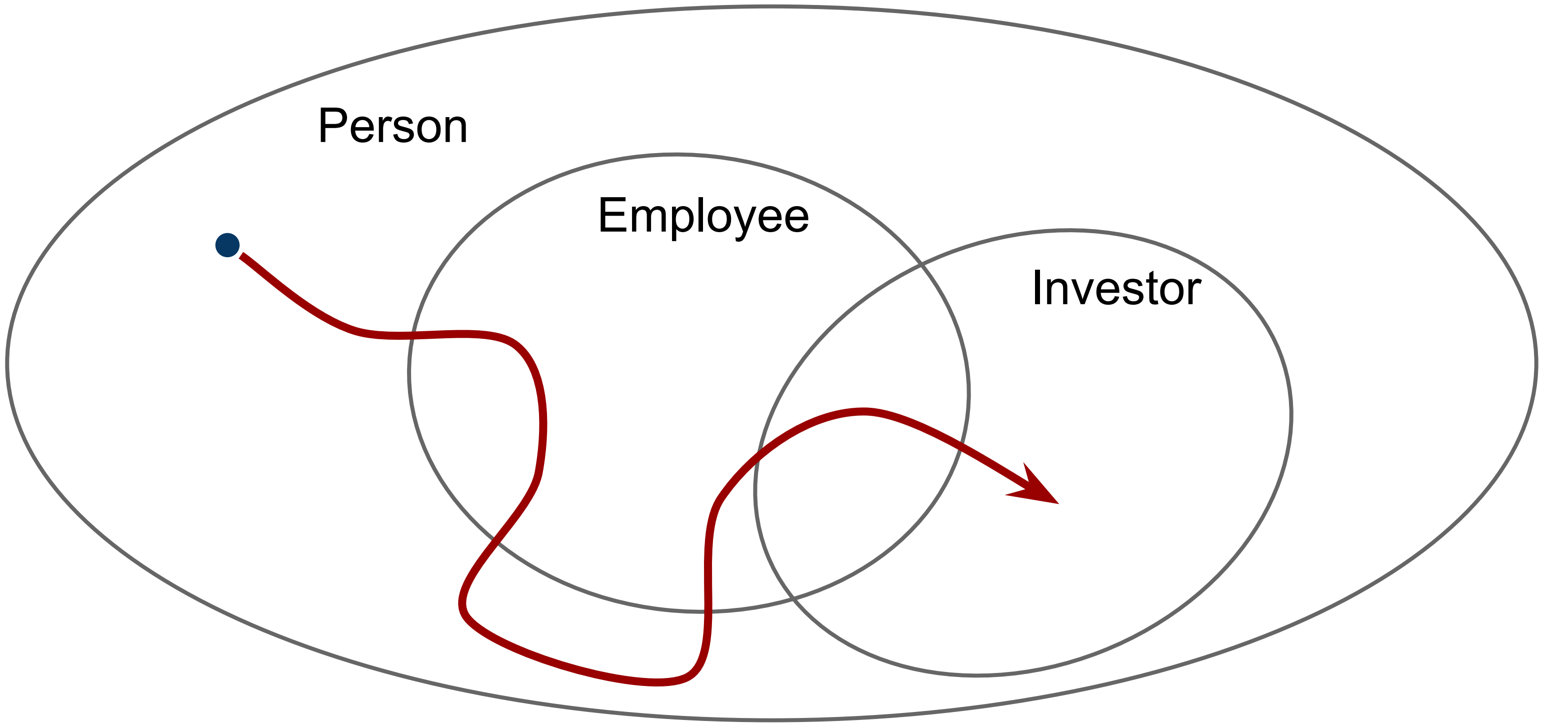
More on concepts

Overlapping concepts



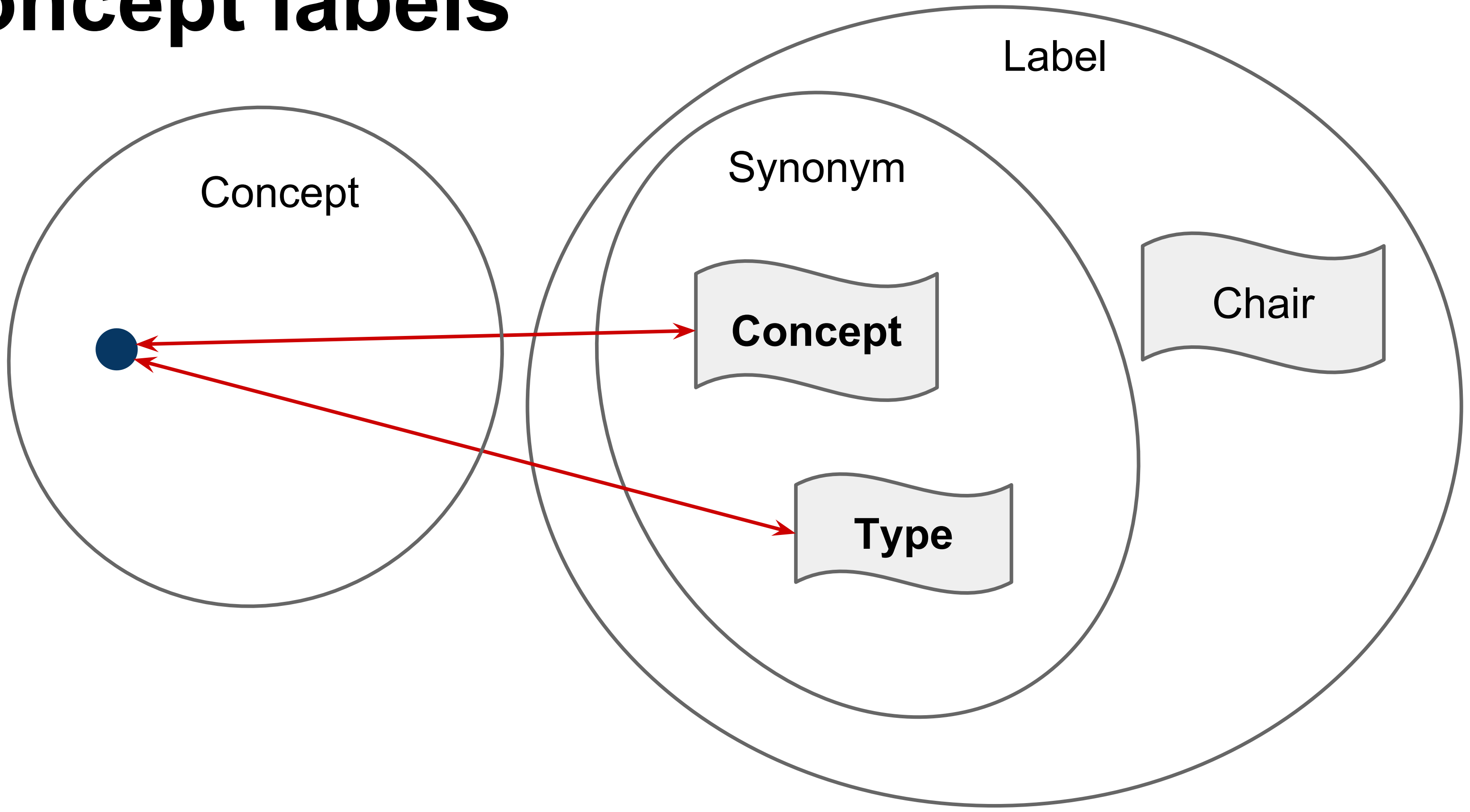
More on concepts

Changing classification



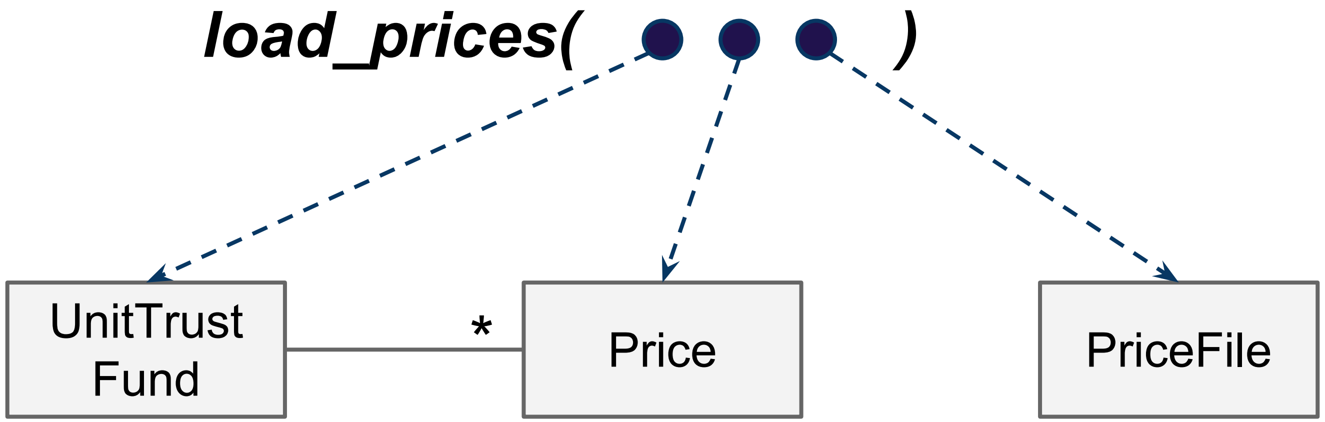
More on concepts

Concept labels



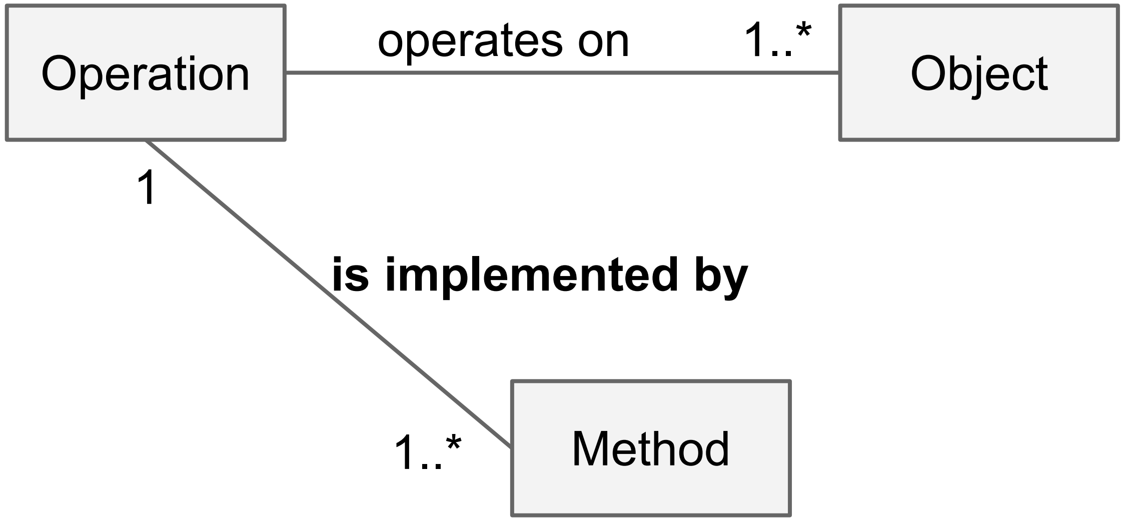
Affecting objects

Operations



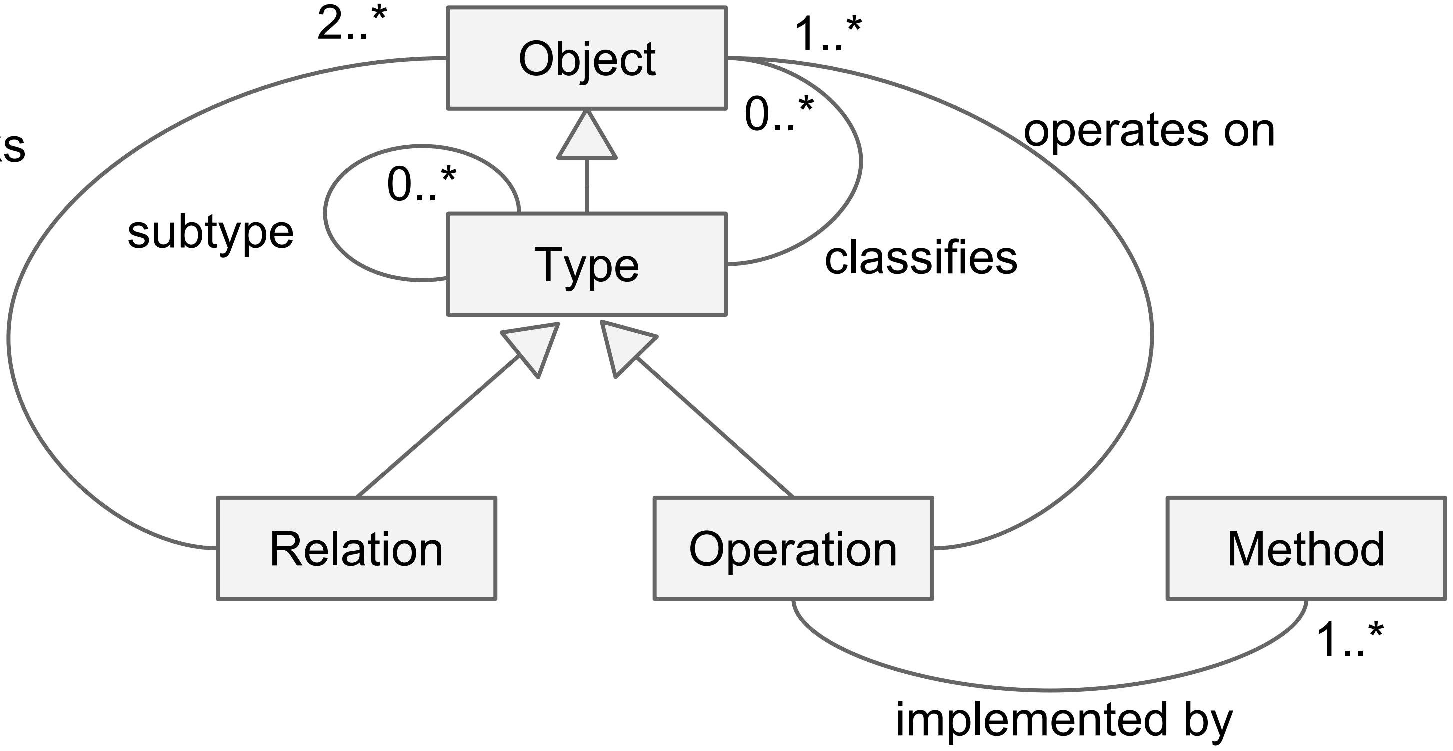
Summary

Operations vs methods



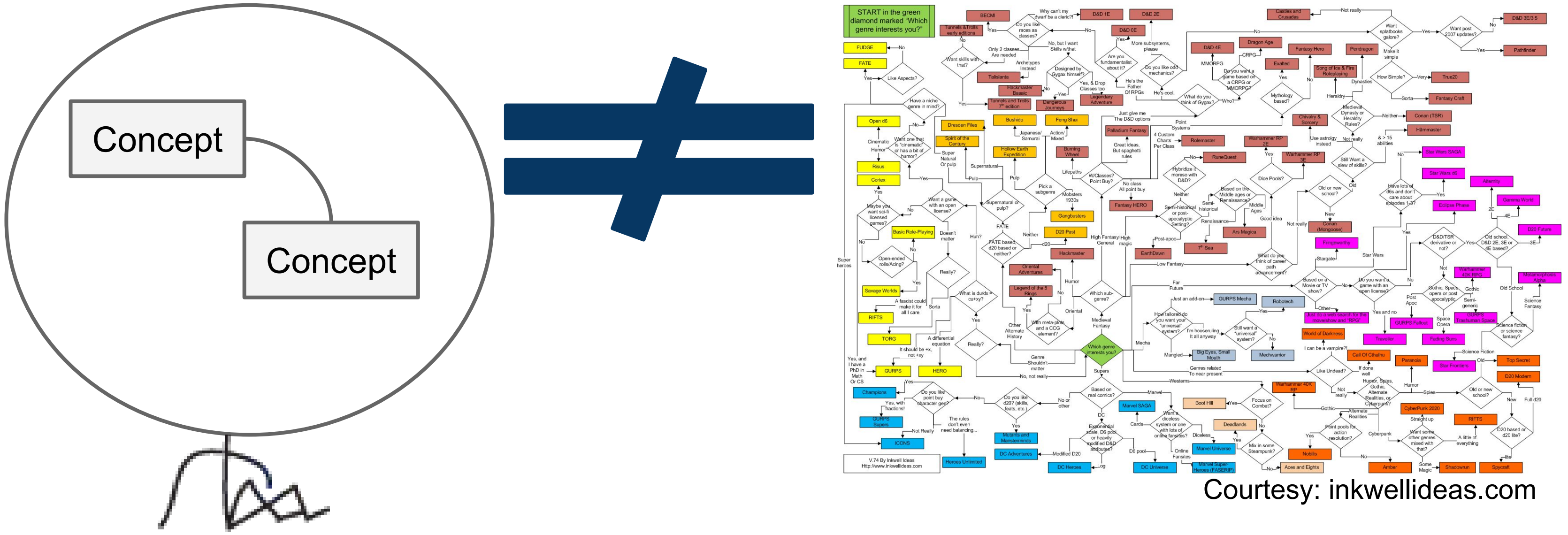
Summary

OO concepts



How can this help?

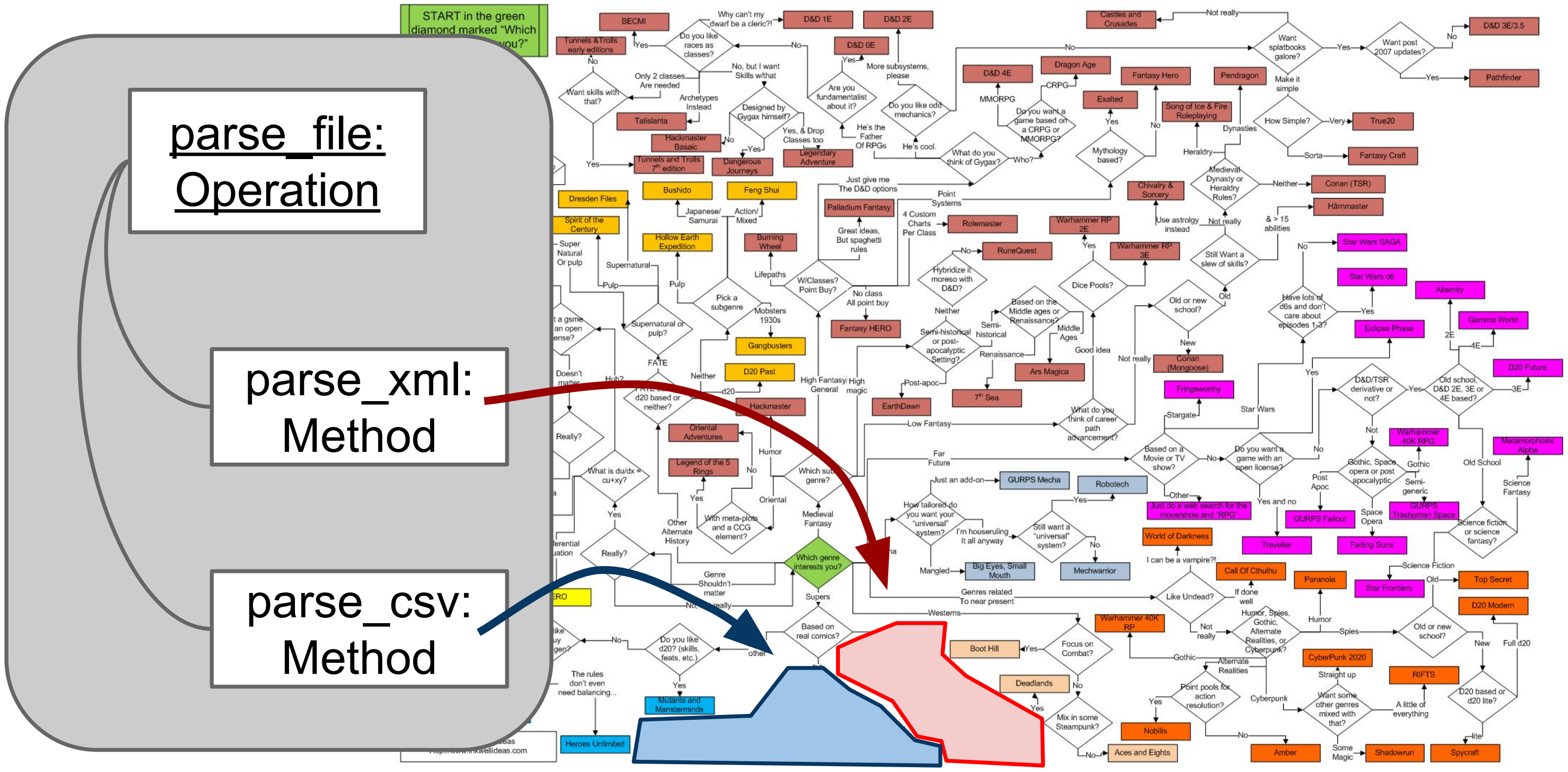
Programs are programs


Concept
Concept
≠
START in the green diamond marked "Which genre interests you?"
Which genre interests you?
V.74 By Inkwell Ideas Http://www.inkwellideas.com

Courtesy: inkwellideas.com

How can this help?

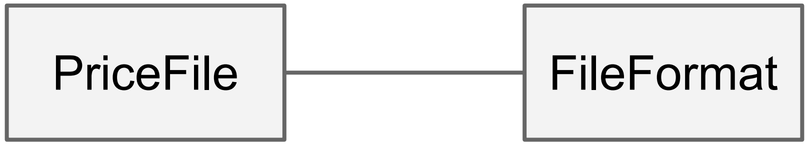
Understanding-structured



Courtesy: inkwellideas.com

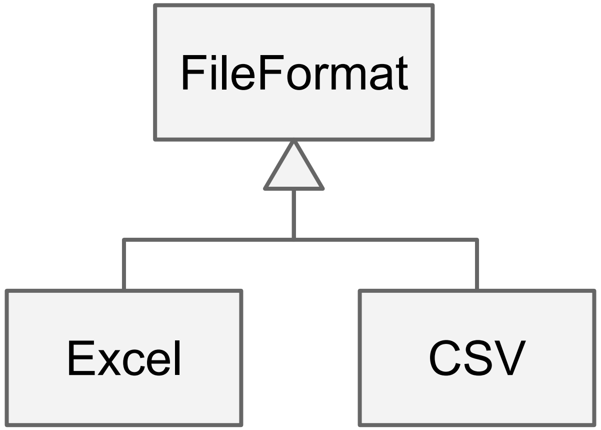
How can this help?

Focus on one thing



How can this help?

Zoom in



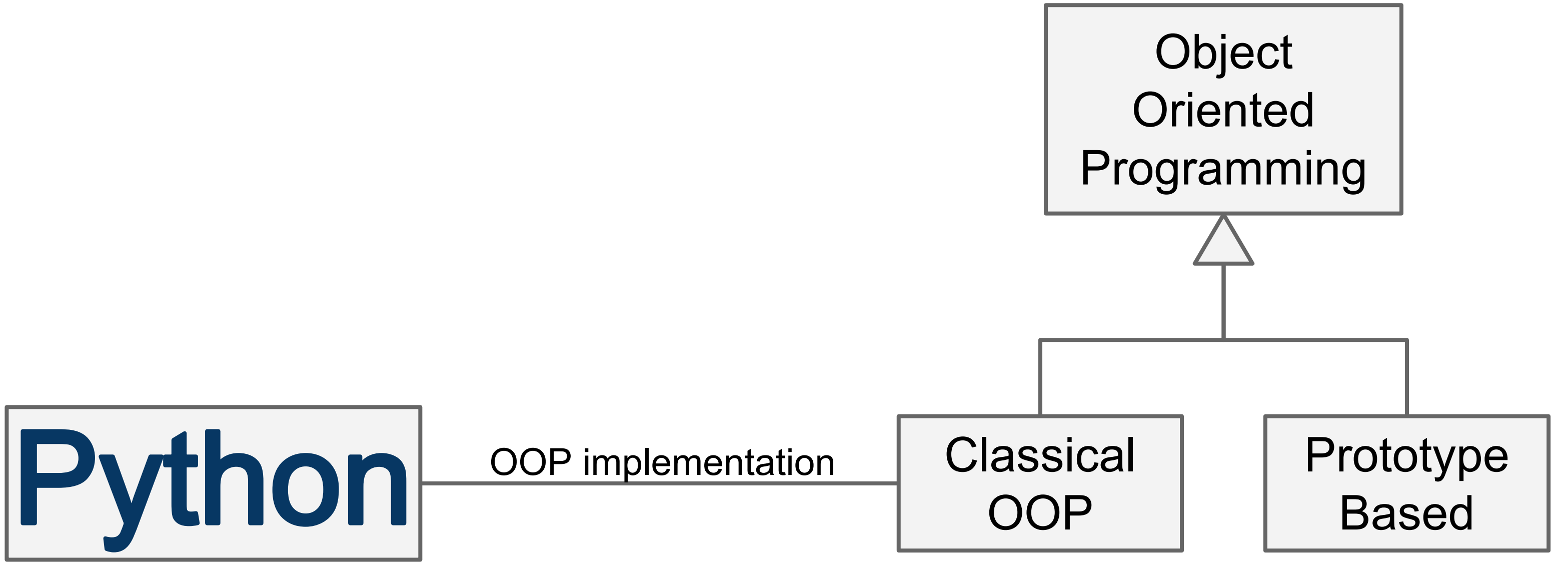
How can this help?

Zoom in more

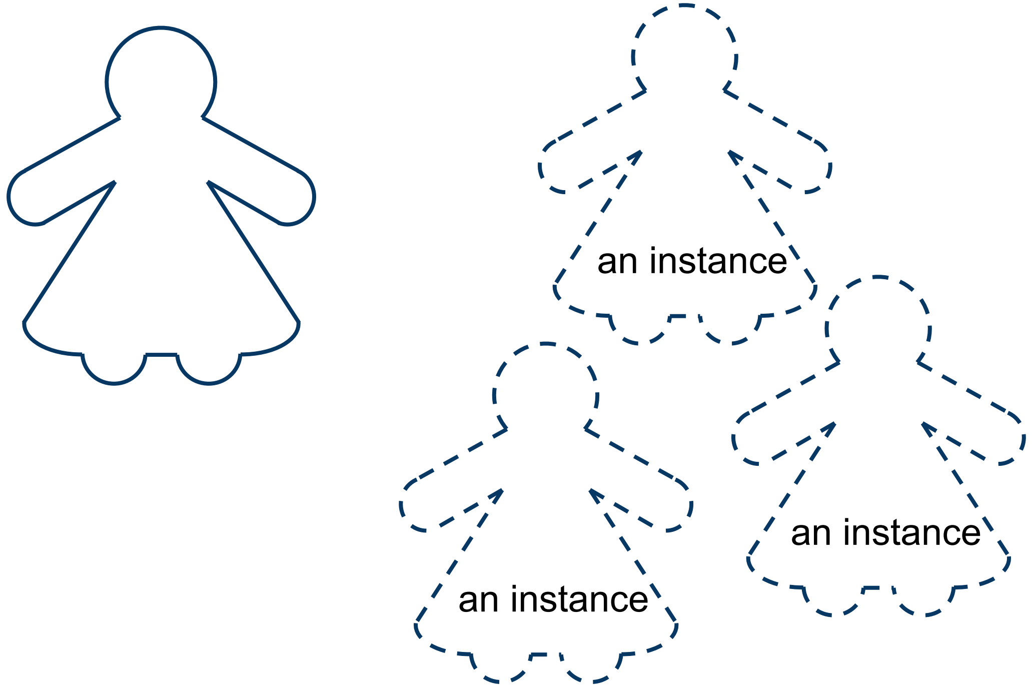
```
def parse_file(price_file):
    for row in price_file:
        yield row.split(‘,’)
```

Implementing OO

OOP and Python



Classical OOP & Python
Types as classes

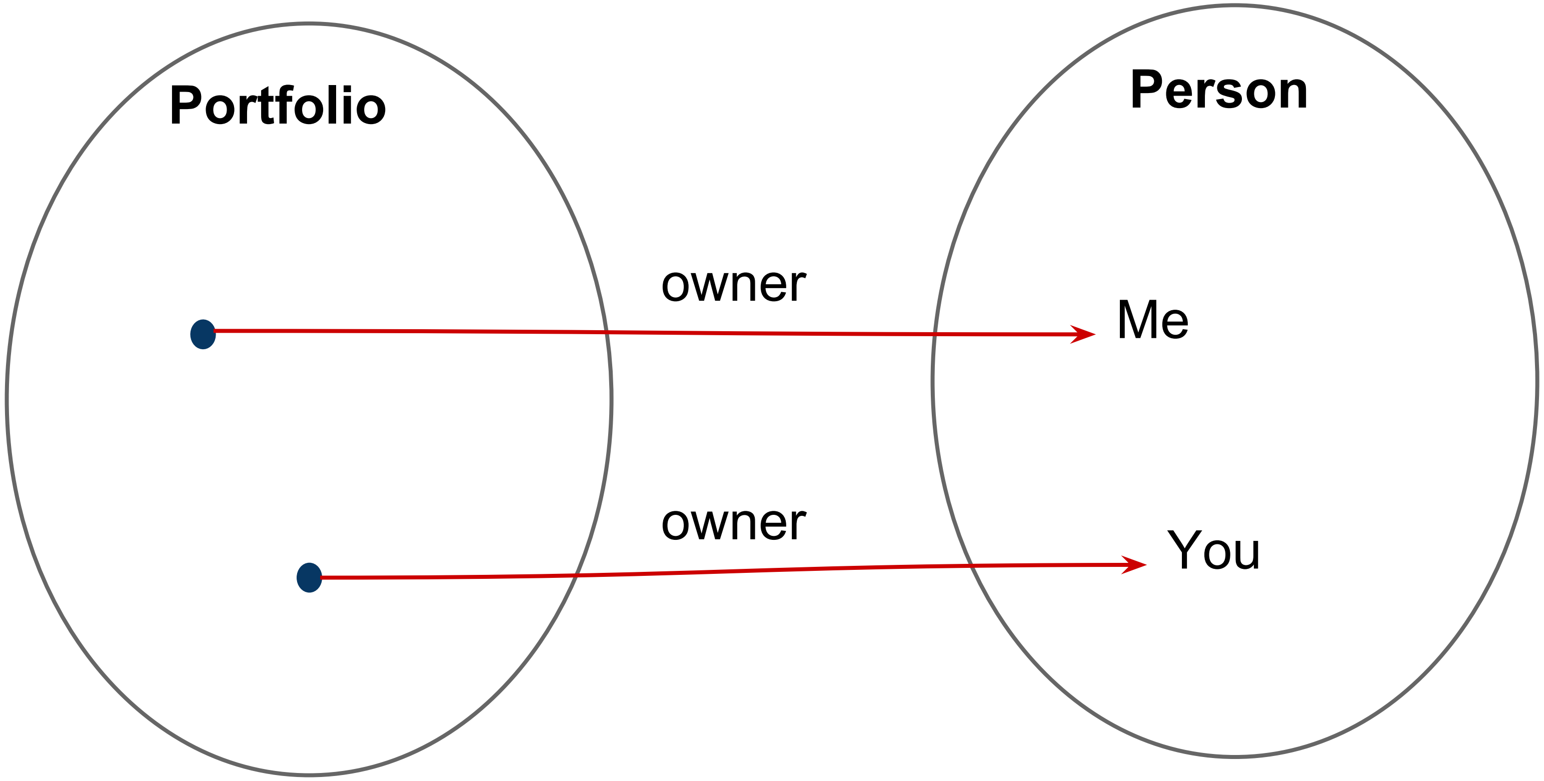


Classical OOP & Python
Python cookies

```
>>> class InvestmentInstrument:
...     pass

>>> fund = InvestmentInstrument()
>>> fund
<InvestmentInstrument object at 0x7f6815559fd0>

>>> isinstance(fund, InvestmentInstrument)
True

>>> type(fund)
<class ‘InvestmentInstrument’>
```

Classical OOP & Python

Relationships as attributes

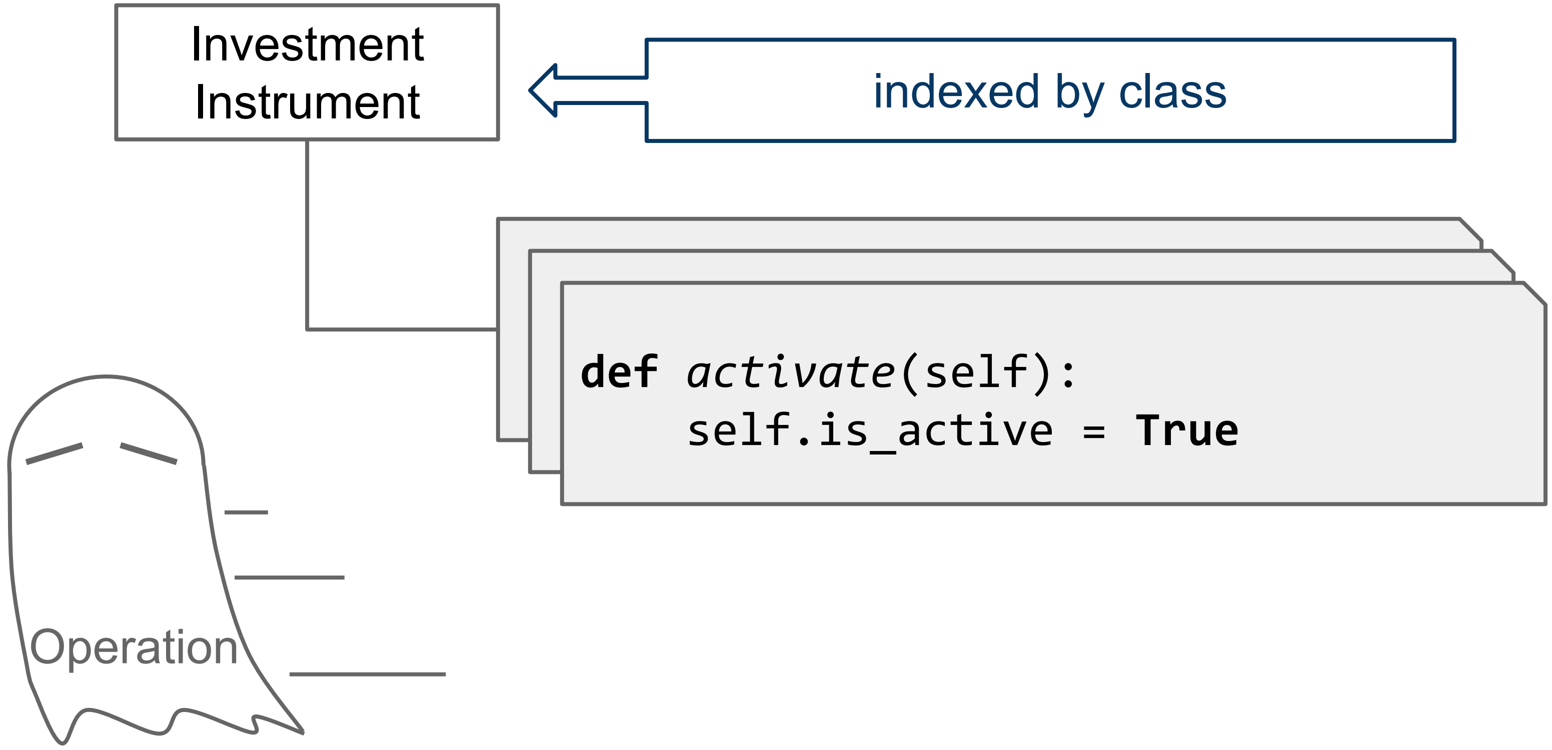


Classical OOP & Python

Attributes

```
>>> portfolio = Portfolio()
>>> someone = Person()
>>> portfolio.owner = someone

>>> portfolio.owner
<Person instance at 0x7fd6d88100e0>
```

Classical OOP & Python
Methods as (Python)methods



```
def activate(self):
    self.is_active = True
```

Classical OOP & Python

A Python method

```
class InvestmentInstrument:

    def activate(self):
        self.is_active = True
```

```
>>> fund = InvestmentInstrument()
>>> InvestmentInstrument.activate(fund)

>>> fund.is_active
True
```

```
>>> fund.activate()
```

Classical OOP & Python

Initialising instances

```
class InvestmentInstrument:
    def __init__(self):
        self.is_active = False

    def activate(self):
        self.is_active = True
```

```
>>> fund = InvestmentInstrument()
>>> fund.is_active
False
>>> fund.activate()
```

Classical OOP & Python

Subtyping as inheritance

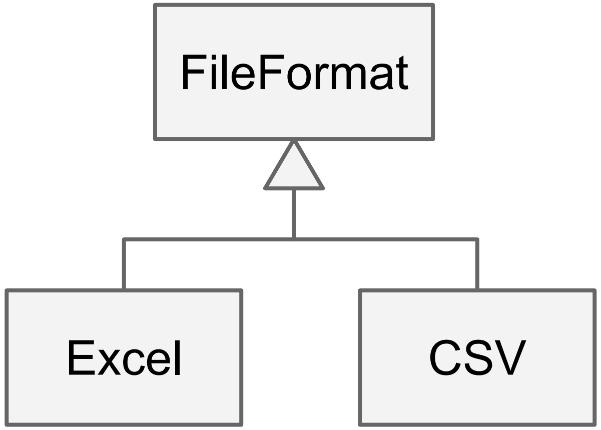
```
class InvestmentInstrument:
    def __init__(self):
        self.is_active = False

    def activate(self):
        self.is_active = True

class UnitTrustFund(InvestmentInstrument):
    def value_of(self, units):
         return units * self.unit_price
```

Classical OOP & Python
Subtyping as inheritance

```
>>> fund = UnitTrustFund()
>>> isinstance(fund, InvestmentInstrument)
True

>>> fund.activate()
>>> fund.is_active
True
```

Classical OOP & Python

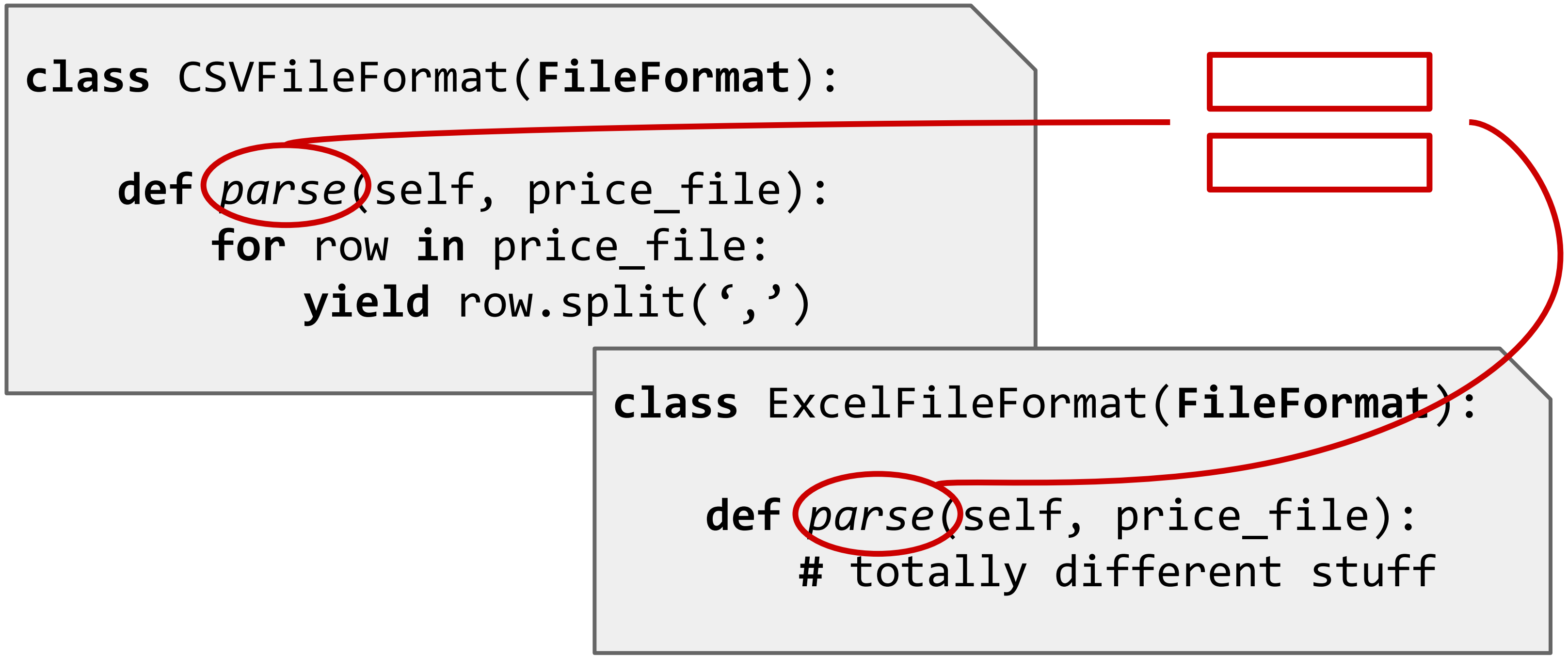
The ghost of operations



parse(   )

Classical OOP & Python

The ghost of operations

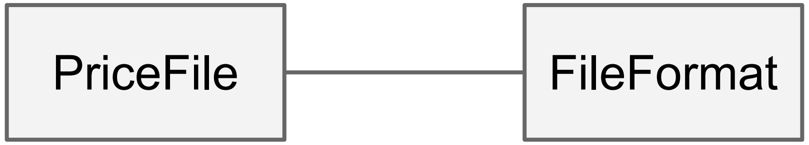


```
class CSVFileFormat(FileFormat):

    def parse(self, price_file):
        for row in price_file:
            yield row.split(‘,’)
```

```
class ExcelFileFormat(FileFormat):

    def parse(self, price_file):
        # totally different stuff
```

Classical OOP & Python

The ghost of operations



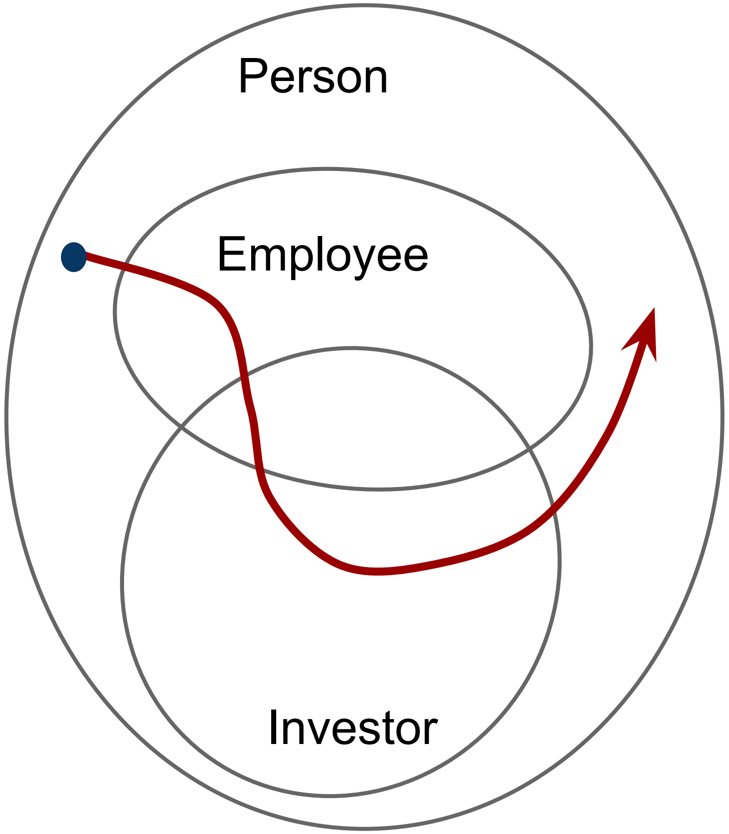
```
for line in price_file.format.parse(price_file):
        # do stuff with line
```

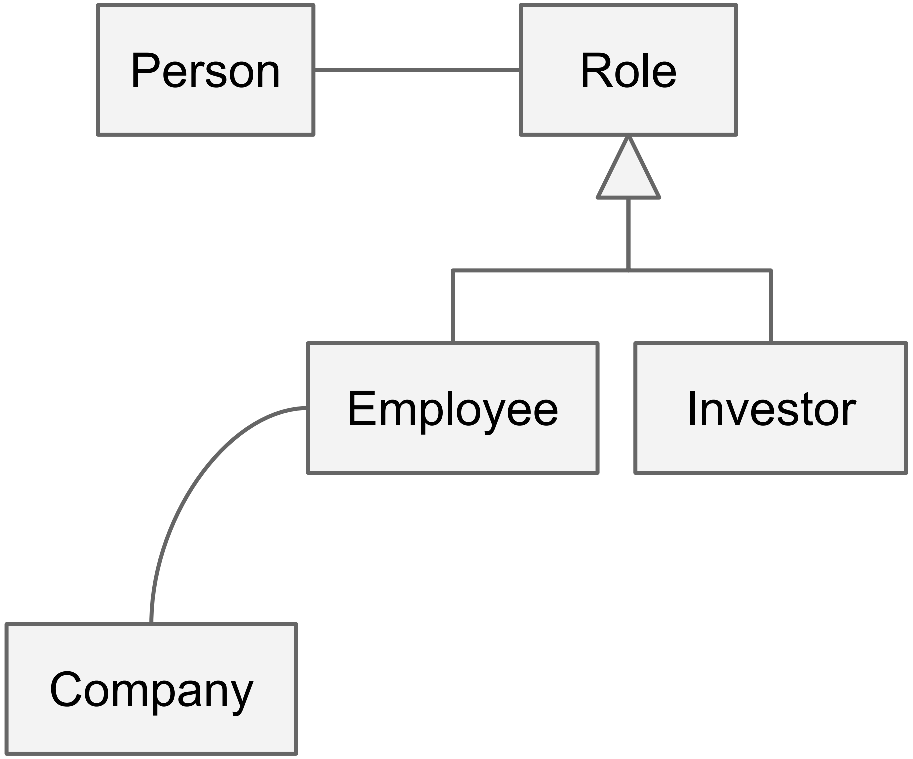
Classical OOP & Python

The ghost of operations

```
if price_file.format.is_csv:
    lines = parse_csv(price_file)
elif price_file.format.is_excel:
    lines = parse_excel(price_file)
else:
    raise Exception('not supposed to get here')
for line in lines:
    # do stuff with line
```

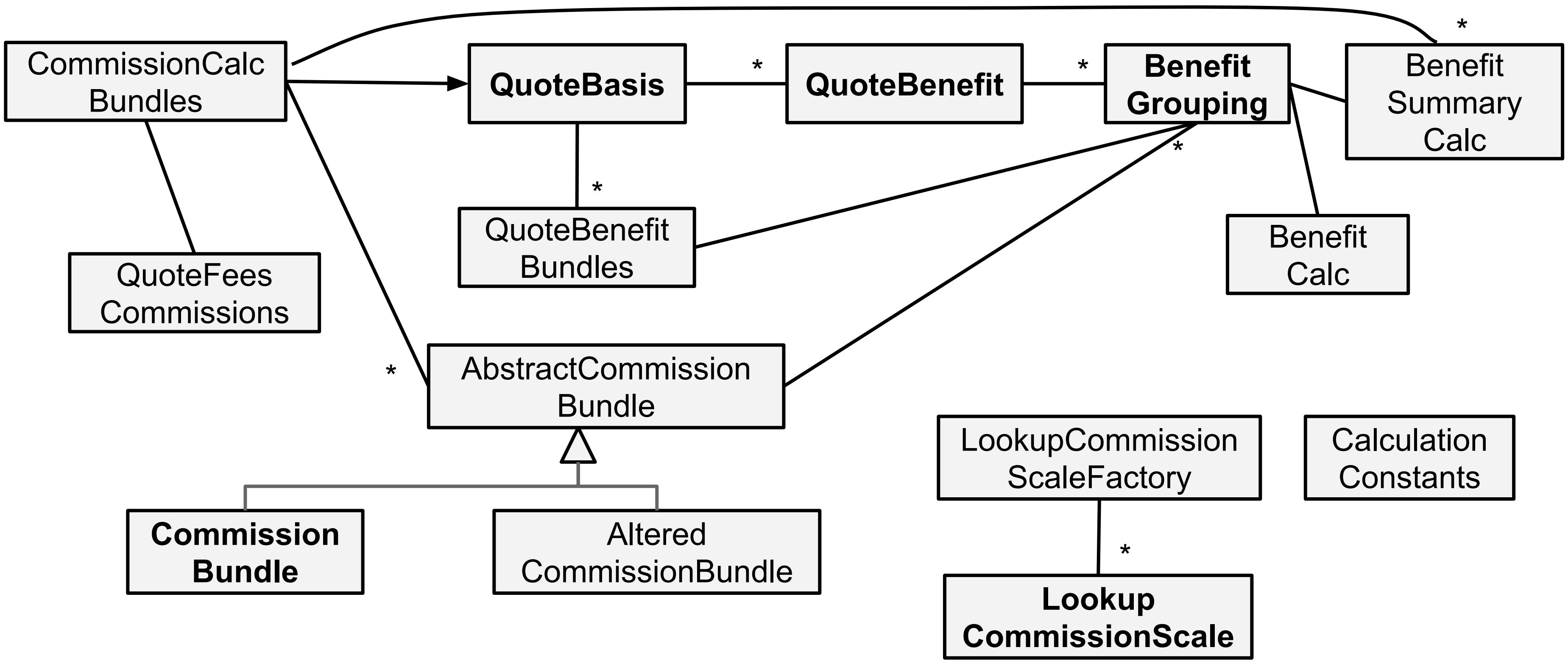
What's design?





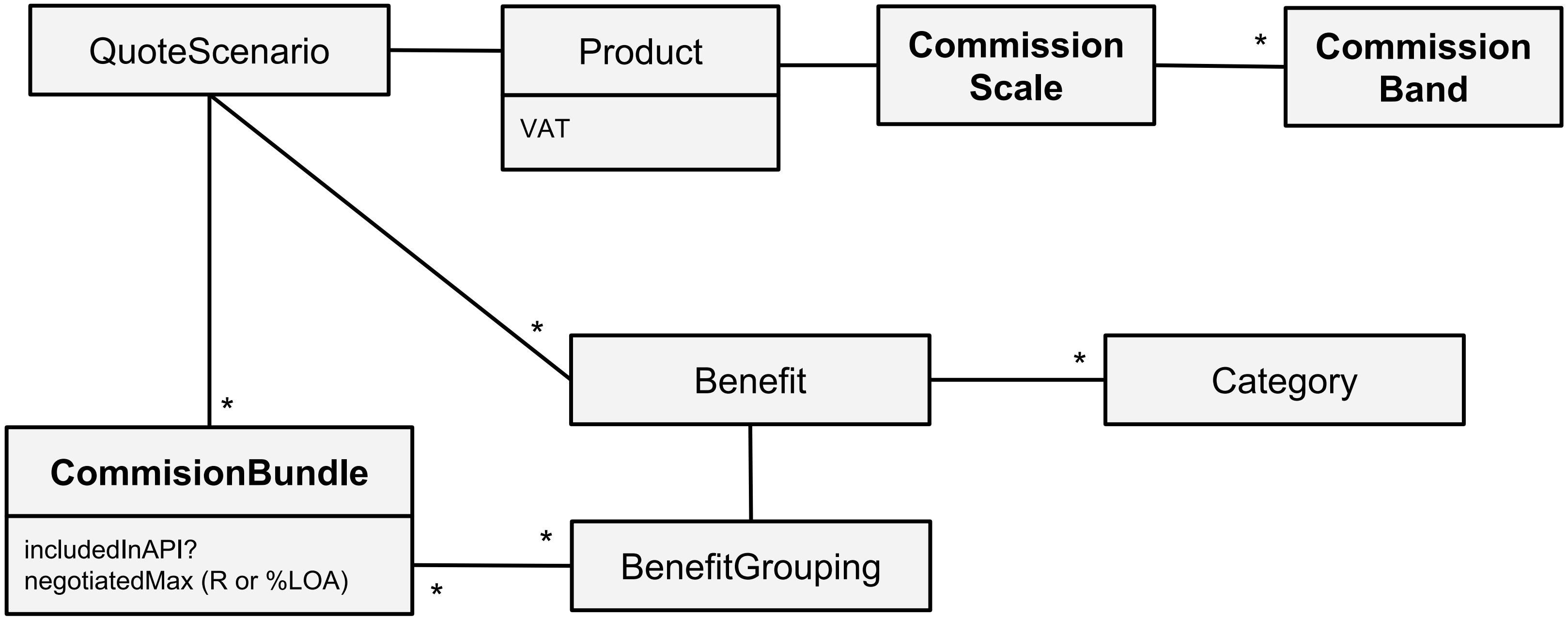
Design

Worse design



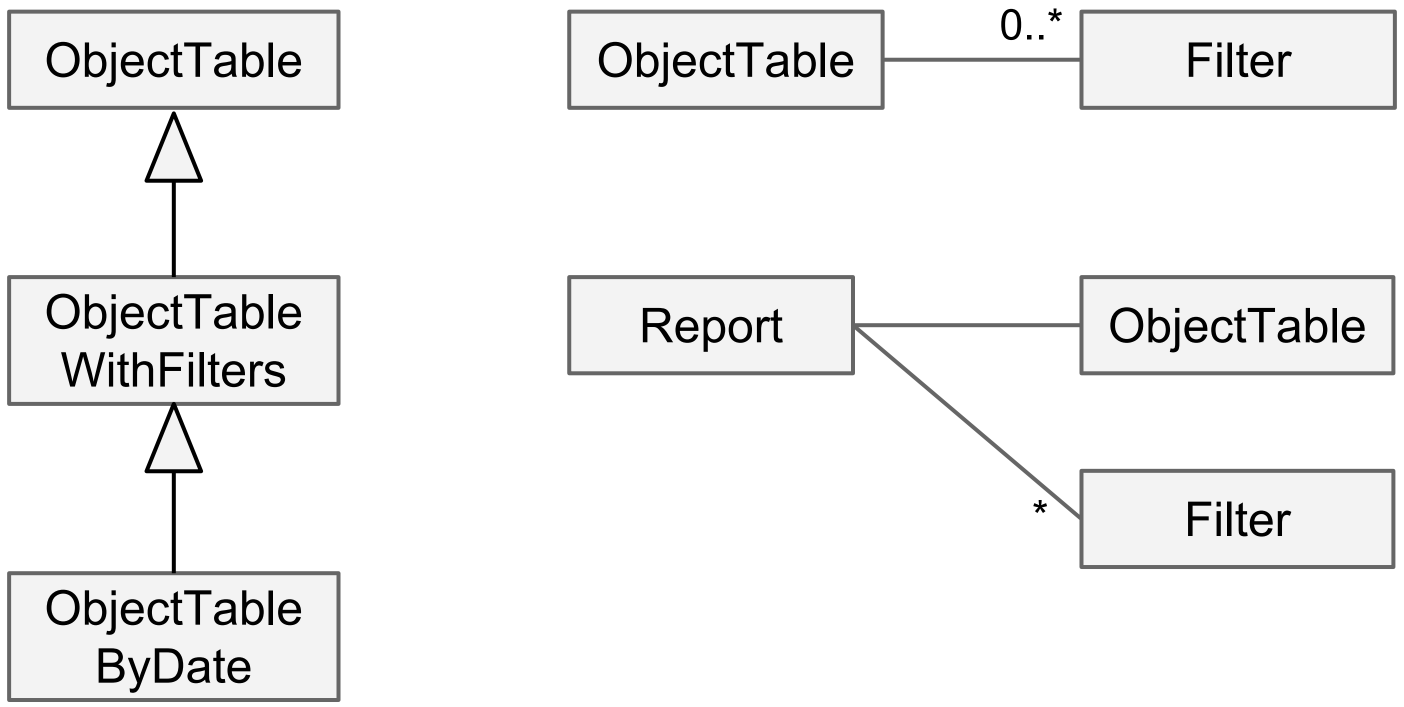
Design

Better design



Design

Inheritance vs subtyping



Thanks

James Martin & James Odell:
Object Oriented Methods: A Foundation

Martin Fowler: *Refactoring*

iwan@reahl.org

Slides:
https://goo.gl/NPLnCf

On **groups.google.com**:
- reahl-discuss