AARHUS UNIVERSITET

Roles, Responsibilities and Behaviour

What is an object?

What is object-orientation?

A decade of OO teaching

Kristen Nygaard (1978)

Ole-Johan Dahl (1978)

I was deeply rooted in Scandinavian tradition of Object-Orientation:

- Kristen Nygaard & Ole-Johan Dahl: Simula
- Ole Lehrmann Madsen et al.: BETA

"A program execution is viewed as a physical model simulating the behaviour of either a real or imaginary part of the world."

But something was missing...

- Design patterns and frameworks are *not* simulating the world...
- My designs tended to cluster "Blobs" and was hard to maintain.
- How does it help me structure the GUI?

What do people say about objects?

(l) A class definition encapsulates its objects' data and actions. [Morelli, 2000, p. 61]

(l) An object is a program construction that has data (that is, information) associated with it and that can perform certain actions. [Savitch, 2001, p. 17]

(l) A class definition describes the behavior and attributes of typical instances of that class. [Barnes, 2000, p. 36]

(l) An object is defined by a class, which can be thought of as the data type of the object. [Lewis and Loftus, 2003, p. 62]

(l) An object is characterized by its state, behavior, and identity. [Hortsmann, 2004, p. 39]

(m) Model elements in Java programs are called objects. [Arnow and Weiss, 2000, p. 4]

(m) Java objects model objects from a problem domain. [Barnes and Kolling, 2005, p. 3]

(r) The best way to think about what an object is, is to think of it as something with responsibilities. [Shalloway and Trott, 2004, p. 16]

(r) An object-oriented program is structured as a community of interacting agents called objects. Each object has a role to play. Each object provides a service or performs an action that is used by other members of the community. [Budd, 2002, p. 9]

Perspectives

***Language centric* perspective:**

Object = Data + Actions

***Model centric* perspective:**

Object = Model element in domain

***Responsibility centric* perspective:**

Object = Responsible for providing service in community of interacting objects

Language-centric

Focus

Language centric focus

– object = fields + methods
– looking at the concrete “building blocks”
– language features are emphasized
– editor view / static

```
public class Foo {
  private int x;
  public static double y;

  public int double(int x) {
    return 2*x;
  }
}
```

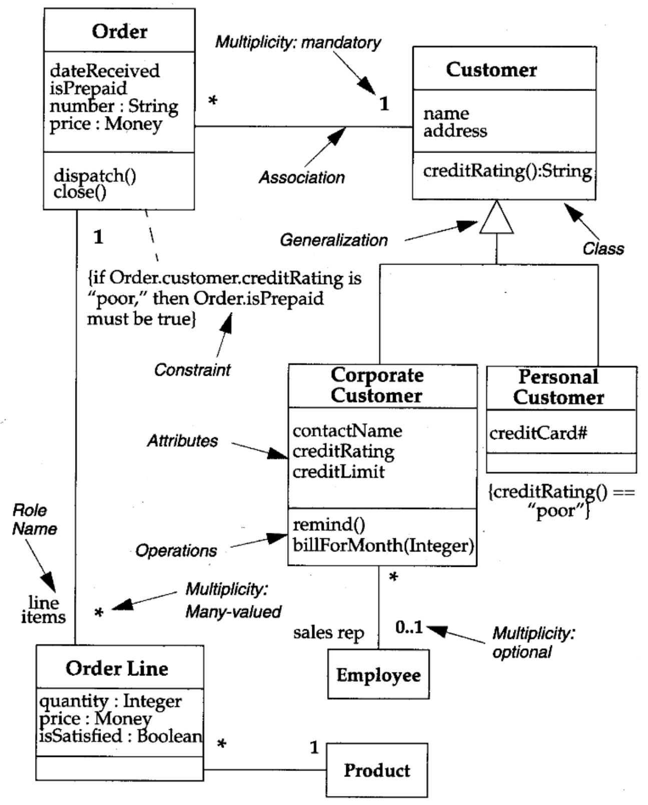
Model-centric

Focus

Model centric focus

- focus on concepts and relations (static!)
  - generalization, association, composition
- problem domain modeling
- object = part of model
- simulation (Kay/Nygaard)

```
public class Account {
  int balance;
  public Account() { balance = 0; }
  public void withdraw( int amount ) {
    balance -= amount;
  }
}
```



One definition of OO

Object Orientation

- *A program execution is viewed as a physical model simulating the behaviour of either a real or imaginary part of the world.*
- *[Madsen, Møller-Pedersen, Nygaard 1993]*

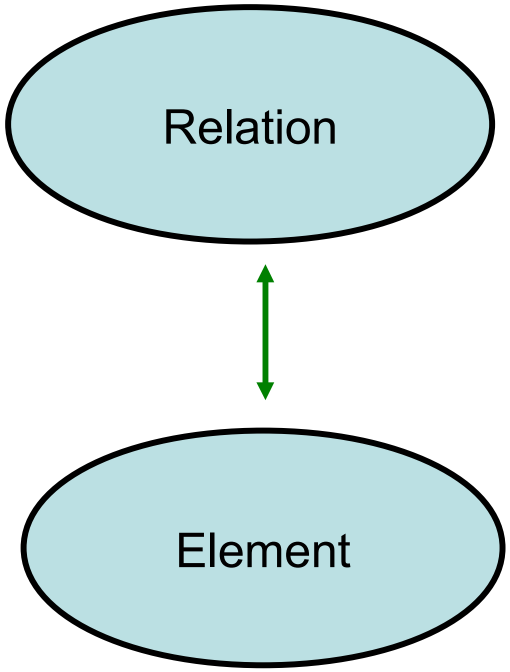
Computation as simulation.

Kristen Nygaard and Ove Johan-Dahl worked indeed in writing simulation software !

Consequences

This *model-centric* perspective results in

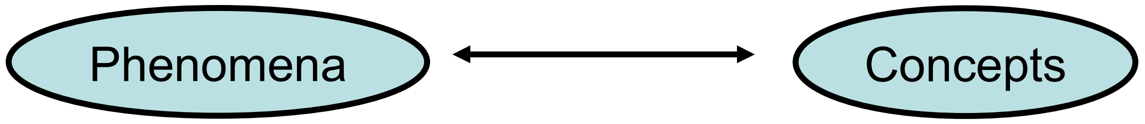
- Strong emphasis on “modelling real world”
- Strong emphasis on model elements and relations



Bass’ definition:
The software architecture of a computing system is the structures of the system, which comprise ***software elements****, the externally visible properties of those elements, and the* ***relationships*** *among them.*

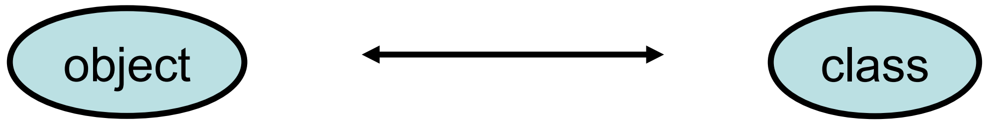
Examples

Real world



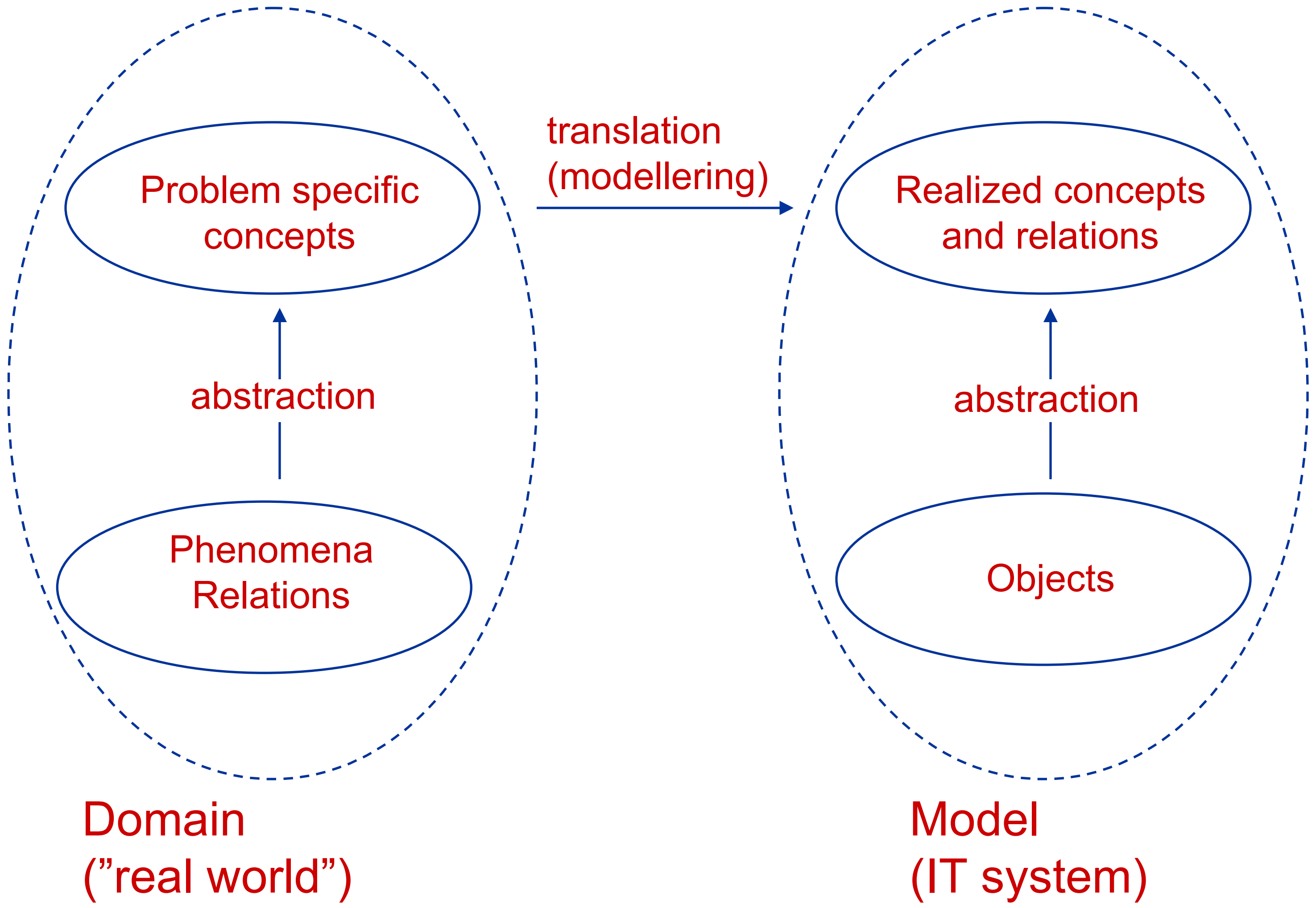
- Examples:
  - My particular Dell laptop computer (phenomena)
  - The concept “laptop”

IT world



- Examples:
  - The PayStationImpl class (concept)
  - The PayStationImpl object that ran last week (phenomena)

System design as translation



Translation support

The strong point in OO is the language support for these aspects of human every day life.

- phenomena ➔ object
- concept ➔ class
- generalization ➔ superclass
- association ➔ object reference
- aggregation ➔ object reference (or inner class)

Simulation

The definition mentions *simulation*

- *A program execution is viewed as a physical model simulating the behaviour of either a real or imaginary part of the world.*

Simulation means “do something” ☺

- Objects have methods
- Objects interact by invoking methods on each other.

- But – where do the inspiration for methods come from?

Simulation

Most real world objects *do not* exhibit any interesting behaviour

- a seat in a plane *does not* know who reserved it
- a backgammon checker *does not* know how to move nor which position it has on the board

Here we usually assign extra meaning

- **Animistic:** *attribution of conscious life to objects in and phenomena of nature or to inanimate objects*

Example

The well worn example of an **Account** class.

Before IT an account was simply lines in a book and an understanding in the head of the banker.

An account *could not* add interest !

A human banker had to do that using his mathematical and business skills.

But – the IT account could be smarter ! The computerized account acts like in a Disney movie ☺

- *account.addInterest();*

Implications

(Language) Model perspective:
- A) Identify landscape of concepts
- B) Distribute behavior over this landscape

Guidelines:
- A) Problem domain concepts ⇨ Objects
- B) “Animate” objects / “Expert pattern”
  - Keep behavior with the data
    - But - an account that can add interest to itself ???

Analysis
- Problem domain concepts only fraction of full system
  - database issues, architectural issues, GUI, networking, ...
- Behavior that does not belong to a domain concept?

Translation process

In other words: *Who / What cycle*

- **Who**: the objects comes **first**
- **What**: the behaviour comes **second**

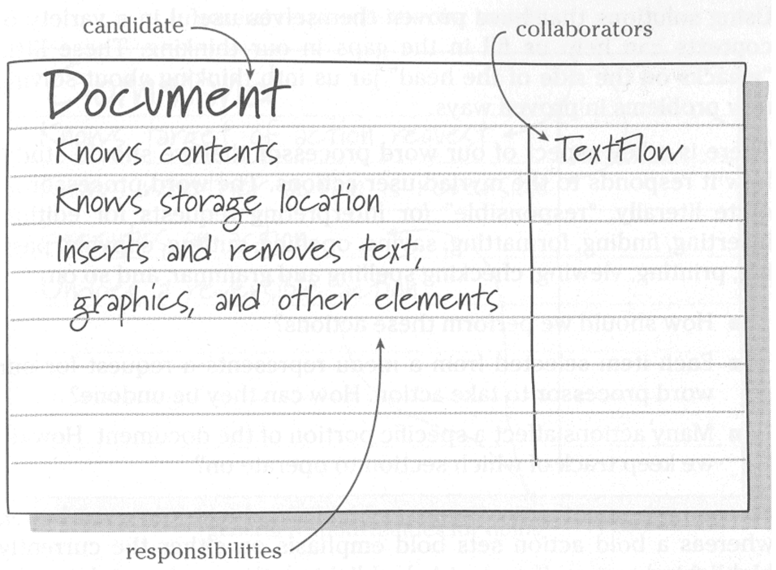
Define the classes, next define their methods

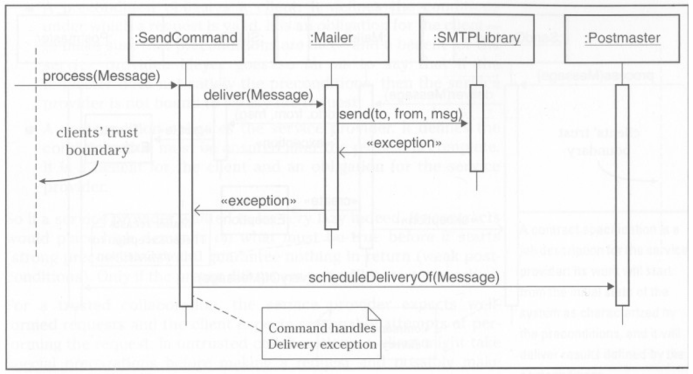
Responsibility-centric

Focus

Responsibility centric focus

- Role, responsibility, and collaboration
- Object = provider of service in community
- Leads to strong *behavioral* focus
- CRC cards (Beck, Wirfs-Brock)





Another Definition

Another definition:

An object-oriented program is structured as a ***community*** *of* ***interacting agents*** *called objects. Each object has a* ***role*** *to play. Each object* ***provides a service*** *or performs an action that is used by other members of the community.*

– *Budd 2002*

Shifting focus

– away from “model of real world”
– towards “community”, “interaction”, and “service”

Service

Budd's definition is more skewed towards the functionality of the system.

At the end of the day, software pays the bill by providing *functionality* that the users need, not by being a nice model of the world!

Services are what developers get paid to create!

“Success lies in how clearly we invent a software reality that satisfies our application’s requirements—and not in how closely it resembles the real world.”

[Wirfs-Brock & McKean]

We can describe ‘services’ or ‘functionality’ from different levels of abstraction:

– Behaviour: *actually gets done*

– Responsibility: *ensure that it gets done*

Behaviour

Behaviour is defined as:

Behaviour:

Acting in a particular and observable way.

In OO, it is the objects that exhibit behaviour when we invoke a method on it:

– database.executeSQL(“SELECT * FROM ...”);

– ps.addPayment(5);

However...

Behaviour (algorithms in methods) is of course the fundamental level of an executing program.

However, the concept is *too low-level* to be really useful in design situations.

In design, a more abstract view of behaviour is needed that shields us from the nitty-gritty details of particular algorithms...

Responsibility

Responsibility is a higher level view:

Responsibility:

The state of being accountable and dependable to answer a request.

The difference is in *doing* versus *accountable for doing.*

Example

An example:

I am responsible for getting house garbage to a central garbage can.

Exercise:

– What kind of behaviours may be involved?
– Is there a one-to-one match between the responsibility and a particular behaviour?
– Am I interested in how exactly the garbage gets to the garbage can?

Discussion

The main point is:

- It is a **responsibility** to ensure the garbage gets to its right place
- I do not care what **exact behaviour** is used to fulfil the task.
- There may be a 100 different concrete behaviours associated with a single responsibility.

Responsibility decouple clients from the exact behaviour!

Responsibility abstracts concrete behaviour!

Describing responsibility

Responsibilities are best described in broad terms in the design phase:

```
Responsibilities:

1) Accept payment;
2) Calculate parking time based on payment;
3) Know earning, parking time bought;
4) Issue receipts;
5) Handle buy and cancel events.
```

No

- Algorithm specifications
- Programmatic details

Programming relations

Behaviour corresponds to concrete methods in Java, C#, etc.

Responsibilities are not first-class citizens in main stream languages ☹

But... The notion of an **interface's method** signatures can be used to partially express responsibility...

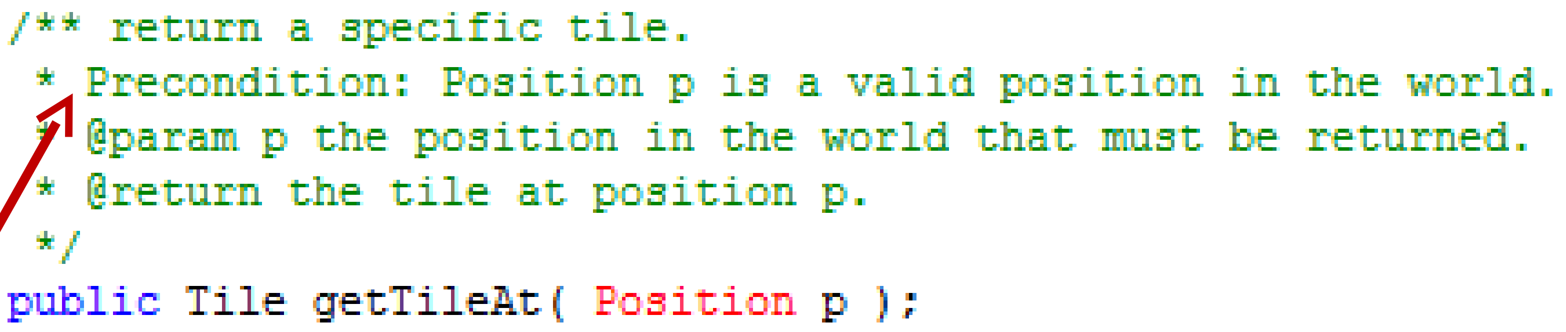
Example

public interface PayStation {…}

The method signatures declares a set of responsibilities but does not dictate the actual behaviour (= method body). Any class implementing the interface is free to make the appropriate implementation.

Discussion

An interface specification alone is a poor way to describe a responsibility as often there are commitments that cannot be seen only from the method definitions.



```
/** return a specific tile.
  * Precondition: Position p is a valid position in the world.
  * @param p the position in the world that must be returned.
  * @return the tile at position p.
  */
public Tile getTileAt( Position p );
```

Use JavaDoc to provide more detail!

Roles

Responsibility describes a commitment to someone else...

– (I’ve got three kids, I know what I am talking about ☺)

Thus responsibility is only interesting in relation to some collaboration and mutual commitment with other objects...

We also want a more abstract (design oriented) way to express collaboration patterns and mutual obligation: **Roles.**

Role

Role (General):

A function or part performed especially in a particular operation or process

The role concepts

- focus both on “function” as well as “in a particular process”
- that is: both *responsibilities* and *collaboration patterns*

- *decouples function from performer*; maybe actor X does Hamlet better but still actor Y can do it.

Discussion

We use roles all the time!

- I would like to talk to the doctor, I am sick ☹
- Quick, call a police man
- We need to hire more programmers!

Are these specific persons?

A role defines a set of responsibilities and defines the way it collaborates with associated object (or rather ”roles”).

Protocol

The proper sequence of actions or events is important

- Hamlet must perform speech and movement in the proper sequence
- Doctor and I must interact properly
- Lecturer assume students are silent during lectures; students assume lecturer will tell something boring ☺

Protocol:

A convention detailing the expected sequence of interactions or actions expected by a set of roles.

Protocol

Warstory

- A programmer from India did not follow Danish protocol when it came to food
  - Is it polite to share your candy with collegues?
  - Is it polite to share your lunch with collegues?

Roles again

Role (Software):

A set of responsibilities and associated protocol with associated roles

Roles for organising work

Roles are central for how humans organize work.

- Software shop: Roles are project leader, programmer, tester. In small companies a single person may play both project leader and programmer role; a programmer often is also tester.
- I play the role as teacher, researcher, colleague, father and husband – each with its own set of responsibilities and commitments.
- Roles are invented by need. A pre-school kindergarten invented a *Flyer* roles whose responsibility it was to ‘catch’ all interruptions.

Implications

Responsibility perspective:

– A) Analyze behavior (what?)

– B) Assign objects (who?)

Guidelines:

– A) Behavior abstracted ⇨ landscape of *responsibilities*

– B) Implement responsibilities in objects

Analysis

– Resemble human organizations – “Flyer” role

– Still need to define the objects ☺

What/Who

Timothy Budd:

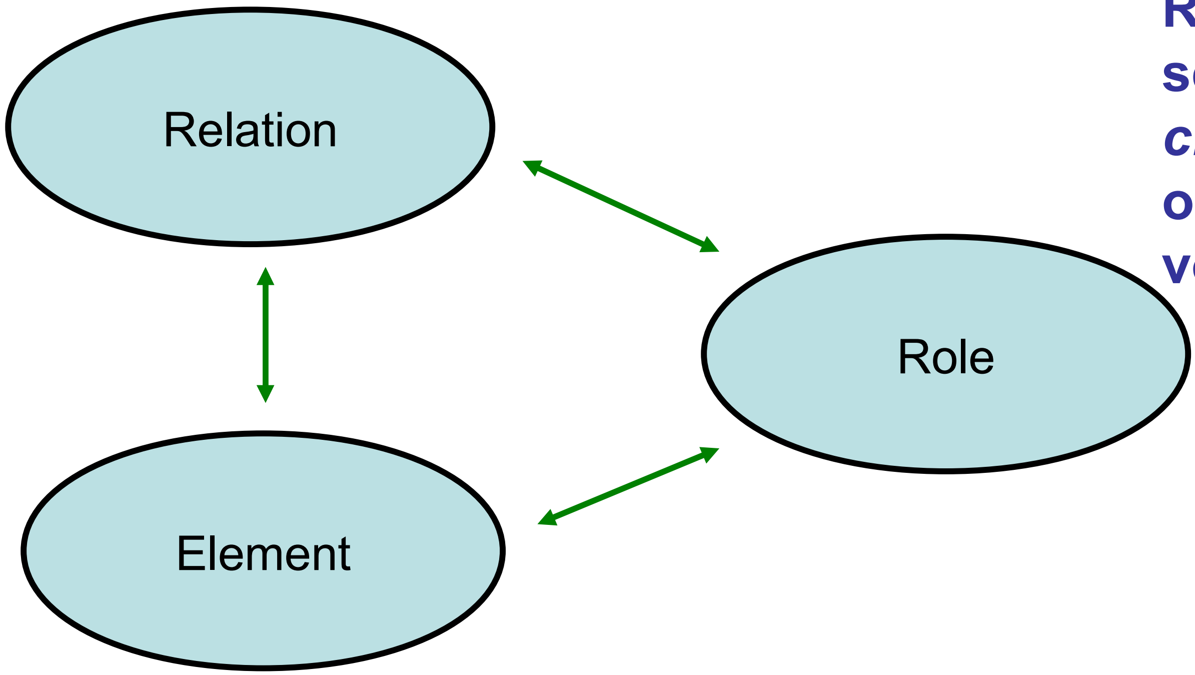
“Why begin the design process with an analysis of behaviour? The simple answer is that the behaviour of a system is usually understood long before any other aspects.”

What / Who cycle

- **What:** identify behaviour / responsibility ⇨ roles
- **Who:** identify objects that may play the roles
  - or even invent objects to serve roles only
    - Larman “Pure fabrication”; example: Flyer role

Role concept

The role concept allows us to use *either* approach (who/what or what/who) because "what" can be expressed as roles.



Role makes service a *first-class citizen* of our design vocabulary

Roles versus objects

The role – object relation is many-to-many.

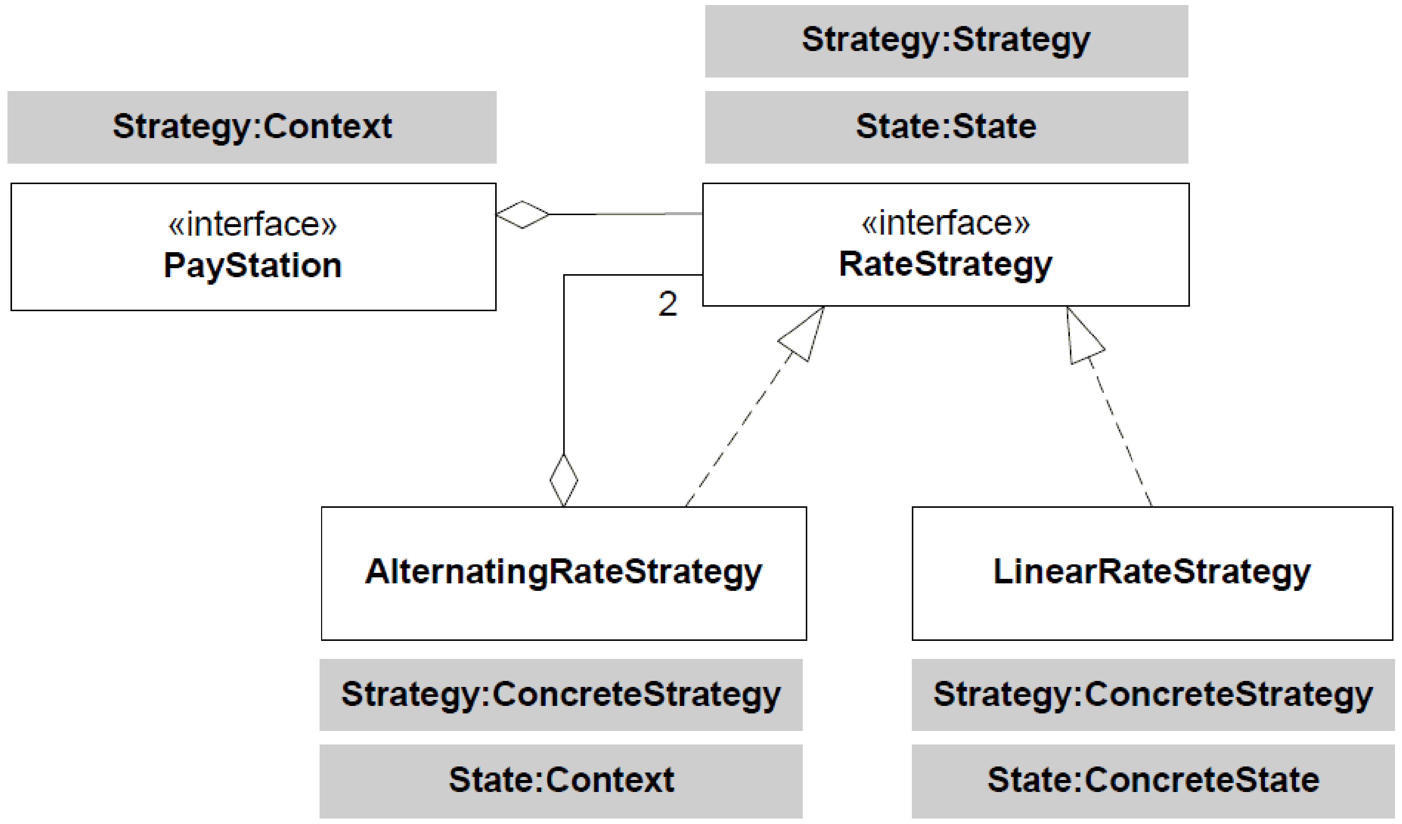
One Role - many Objects

- Interface Comparable in Java
  - Allow any object to be sorted by Collections.sort

One Object – many Roles

- Henrik implements Teacher, Researcher, Writer, Student, Father, Husband, Company-Owner, Tax-payer, Citizen, Danish, ...
  - but not at the same time...

Roles & Patterns



Role – Object
Why difference does it make?

Where is the difference?

What is the point?

- I associate responsibilities with roles?
- I associate responsibilities with concepts (classes)?

At first sight there seems to be none...

Shalloway and Trott has a nice example of the difference...

The difference

Umbrella **concept**

– metal rod aggregate spikes associated with linen

?

Umbrella **role**

– keeps me dry when it is raining

Umbrella again...

class Car extends Umbrella ?

class Umbrella extends Car ?

NONSENSE!

class Car implements Umbrella

More sensible

Another Example

AARHUS UNIVERSITET

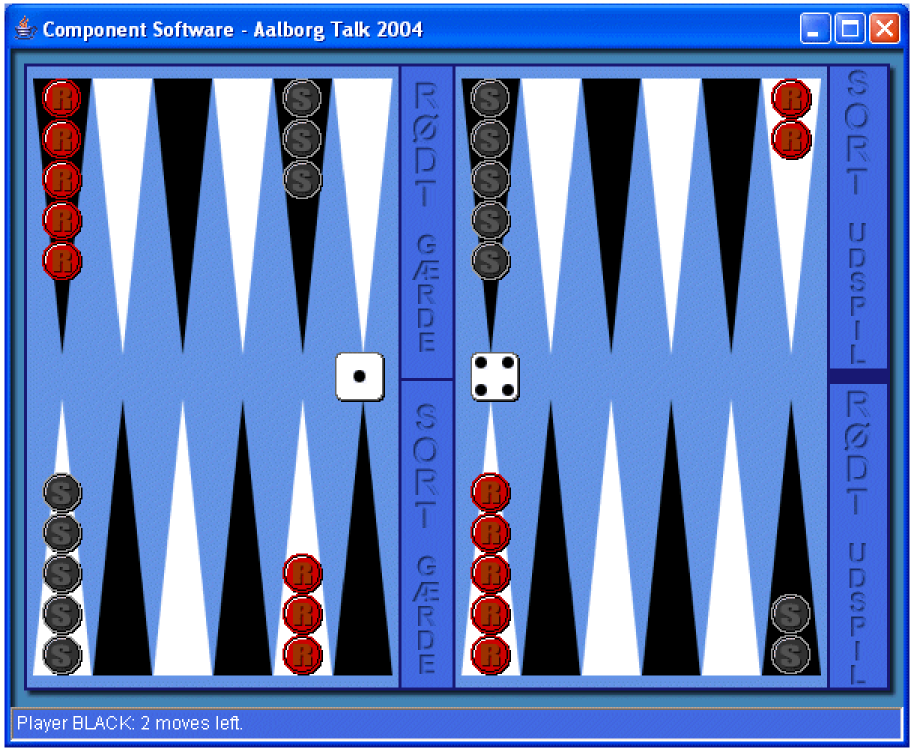
A Case Study

Backgammon requirements:

- Offer GUI for two players
- Guaranty proper play

Variants

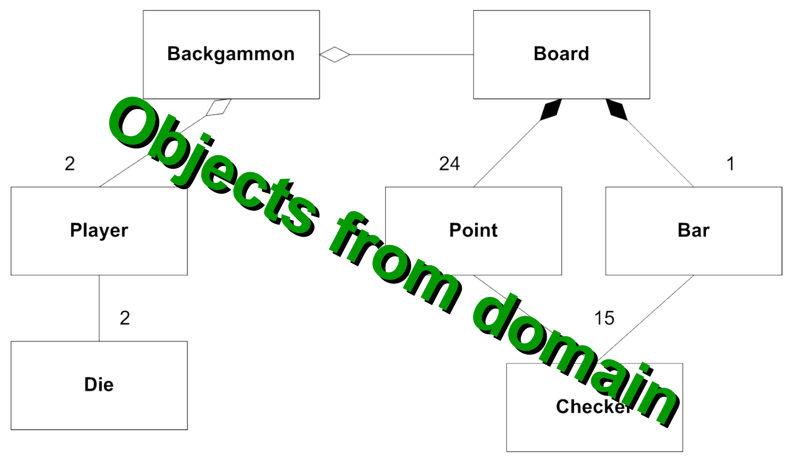
- *new rules* for which moves are legal
- how many moves you can make per turn
- how the board is initially set up



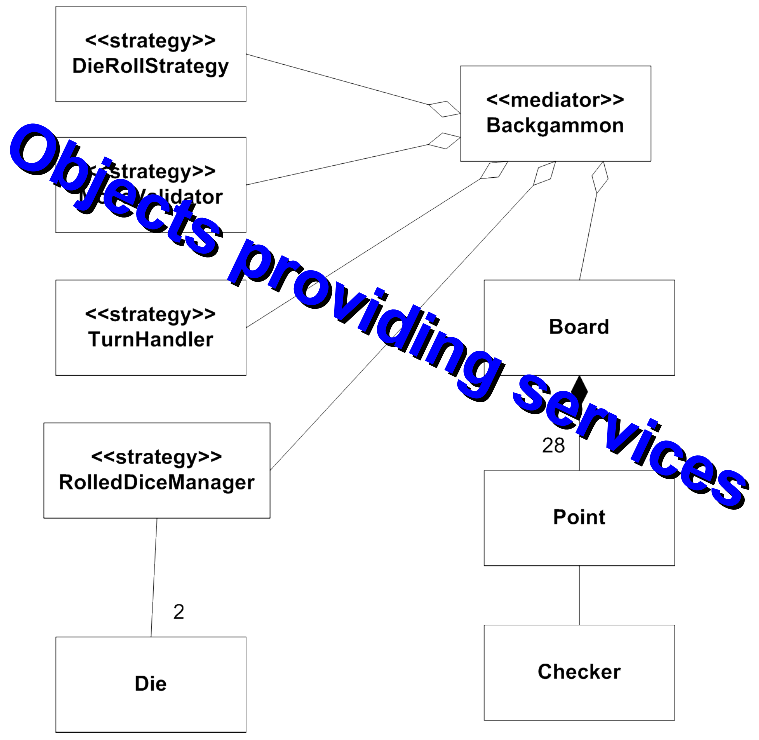
Player BLACK: 2 moves left.

Same challenge – different designs

Model perspective:



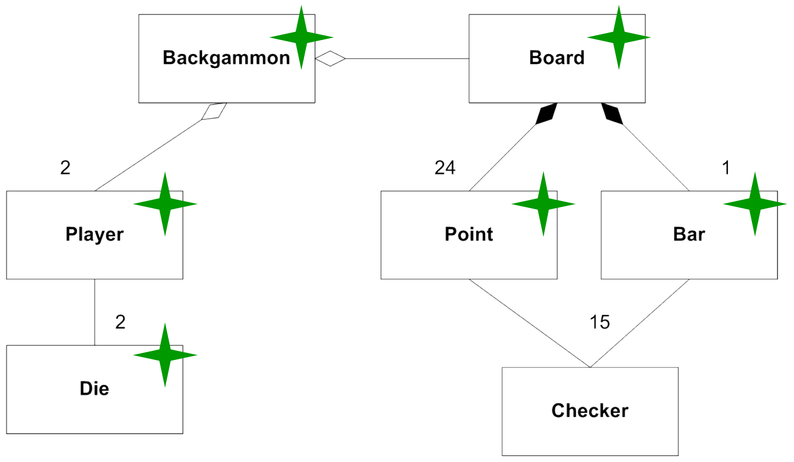
Responsibility perspective:


<<strategy>> DieRollStrategy
<<mediator>> Backgammon
<<strategy>> MoveValidator
<<strategy>> TurnHandler
Board
<<strategy>> RolledDiceManager
28
Point
2
Die
Checker
Objects providing services

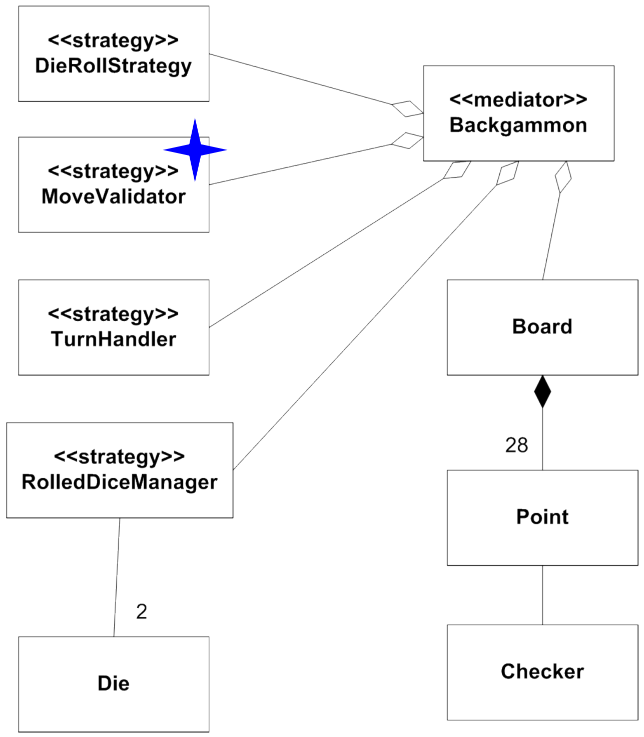
Who is responsible for validating moves?

AARHUS UNIVERSITET

Model perspective:



Responsibility perspective:


<<strategy>> DieRollStrategy
<<mediator>> Backgammon
<<strategy>> MoveValidator
<<strategy>> TurnHandler
Board
<<strategy>> RolledDiceManager
28
Point
2
Die
Checker

What is the cost of altering validation strategy?
How to change it at run-time?

Summary

Summary

Behaviour:

- The basic, concrete, functionality provided by an object

Responsibility:

- Design level concept of accountability on answering a request

Role:

- Design level concept to express a well defined set of responsibilities and collaboration pattern

Perspectives

Three different perspectives on OO

- Language: Important because code is basically only understandable in this perspective
- Model: Important because it gives us good inspiration for organizing the domain code
- Responsibility: Important because it allows us to build highly flexible software with low coupling and high cohesion

They do not have to be in conflict –they build upon each other...

Summary

Design in terms of what roles and responsibilities there are in a system.

Express these as interfaces with appropriate additional documentation.

Implement the roles by concrete classes.

Roles may define points of variability in programs…