Introduction to Modelling and UML

Comp 47480: Contemporary Software Development

Modelling

Models are used heavily in engineering.

Models may be physical, like a scale model of a house, road or bridge.

We're more interested in abstract models, as this is what we use in software.

We use models so much software, so we often don't see them anymore e.g.

```
def foo(i: Int): String
```

is a model of a function that takes an integer argument and returns a string. Details are omitted.

What is a Model?

A model of X captures some properties of X and omits other properties.

Here's a model of this room:

COMP B106	
Seating Capacity: 96	AV Equipment:
Seating type: Tables and Chairs (ALE)	Data Projection
	Computer
	Plasma Screen x8
	HDMI Cable Input Supplied
	Whiteboard
	Wireless Microphone
	Lecture Capture

What does it capture? What does it omit?

Models and Correctness

Models never complete, but may be useful.

> *All models are wrong, but some models are useful. So the question you need to ask is not "Is the model true?" (it never is) but "Is the model good enough for this particular application?"* – George Box

We'll be looking at models in software, but keep this aspect of models in mind.

Models and Software Development

Modelling is a huge part of the Waterfall process.

The Requirements, Analysis and Design phases are all about building more and more detailed models of the software to be built.

Modelling is also part of Agile, but in a different was as we'll see.

What is UML?

UML stands for **U**nified **M**odeling **L**anguage.

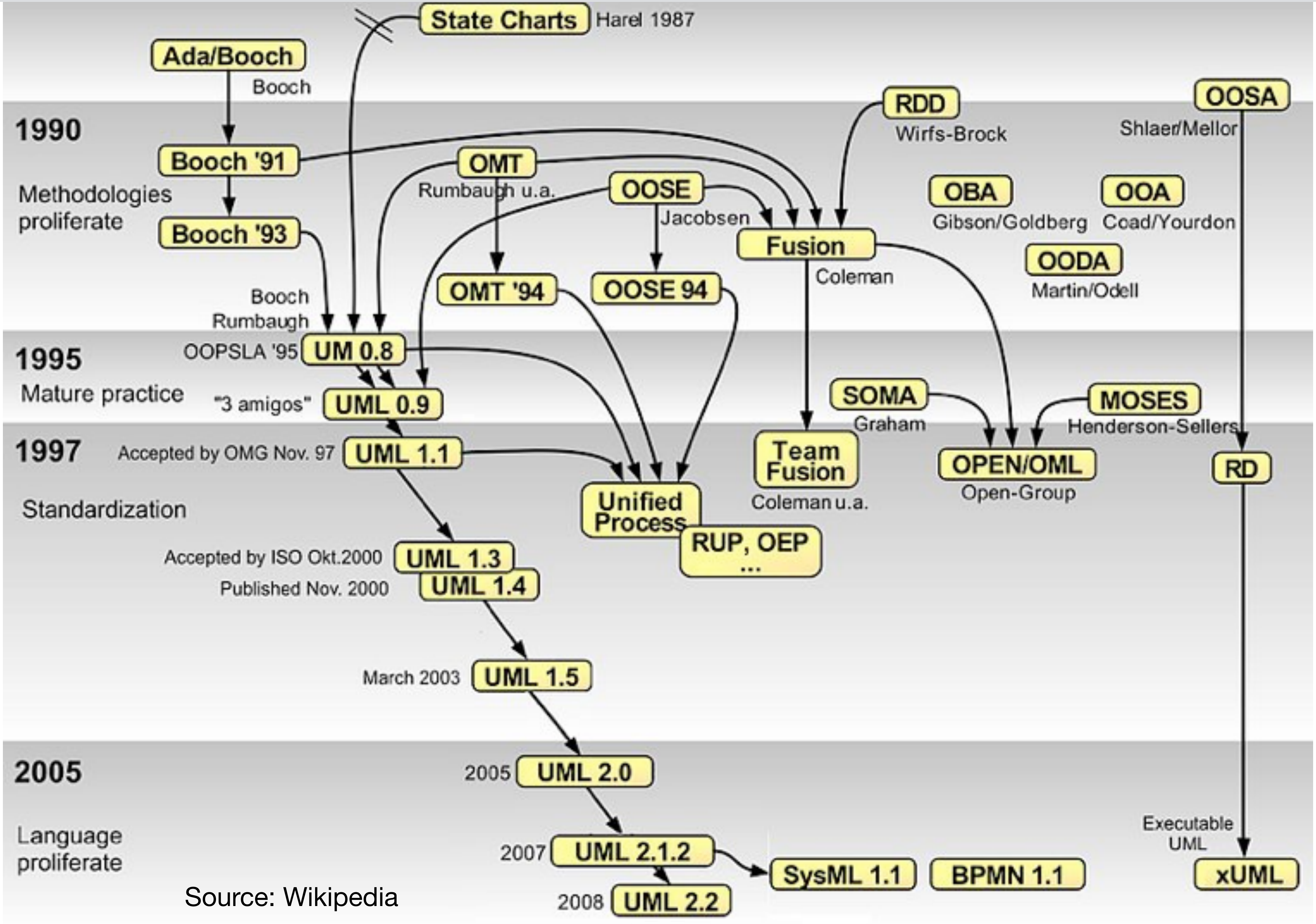
UML is a general-purpose modelling language used to create visual models of object-oriented software systems.

UML can be used to model:

- system structure
- system behaviour
- system architecture

and more.

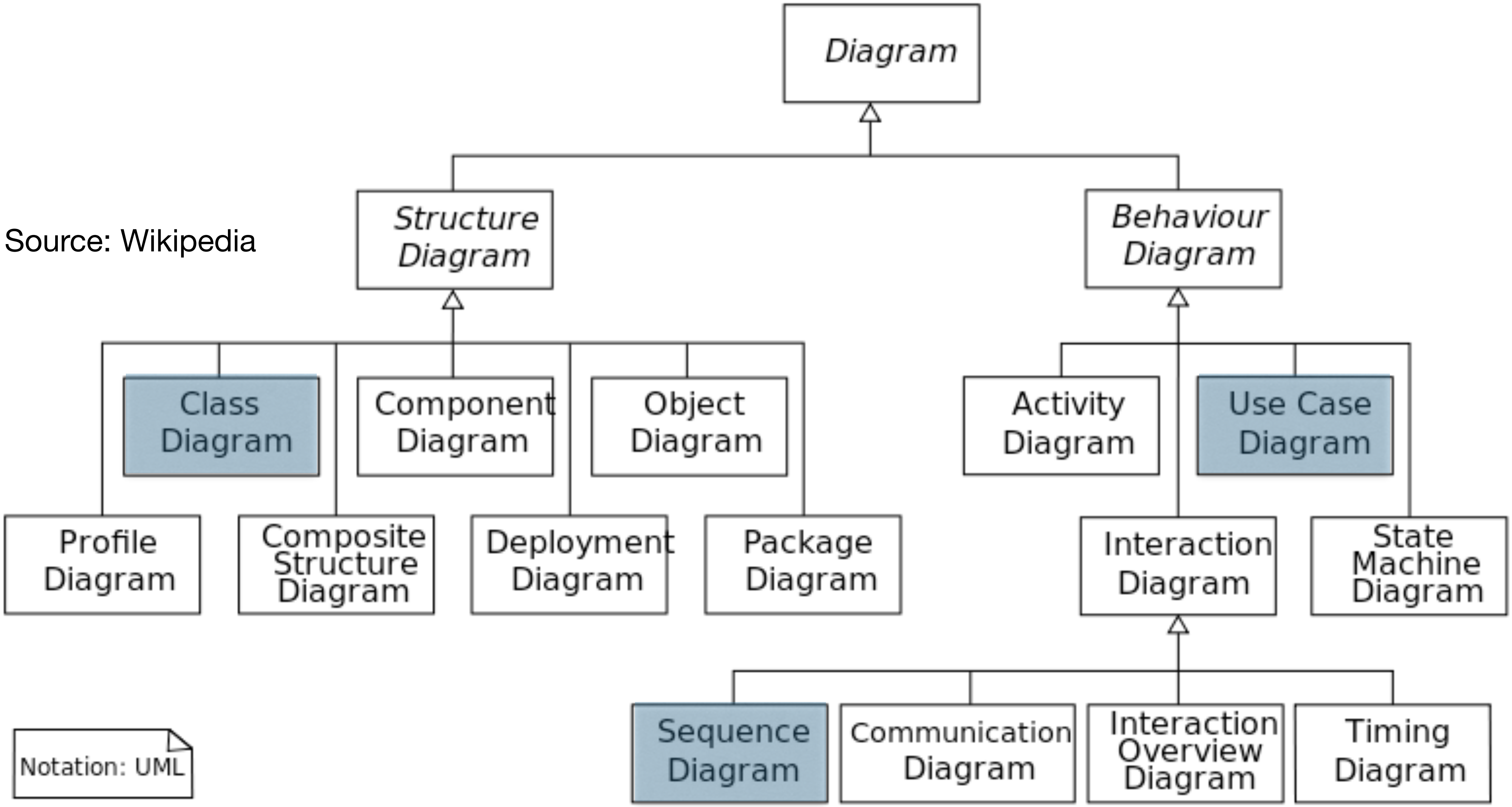
Where did UML come from?



Source: Wikipedia

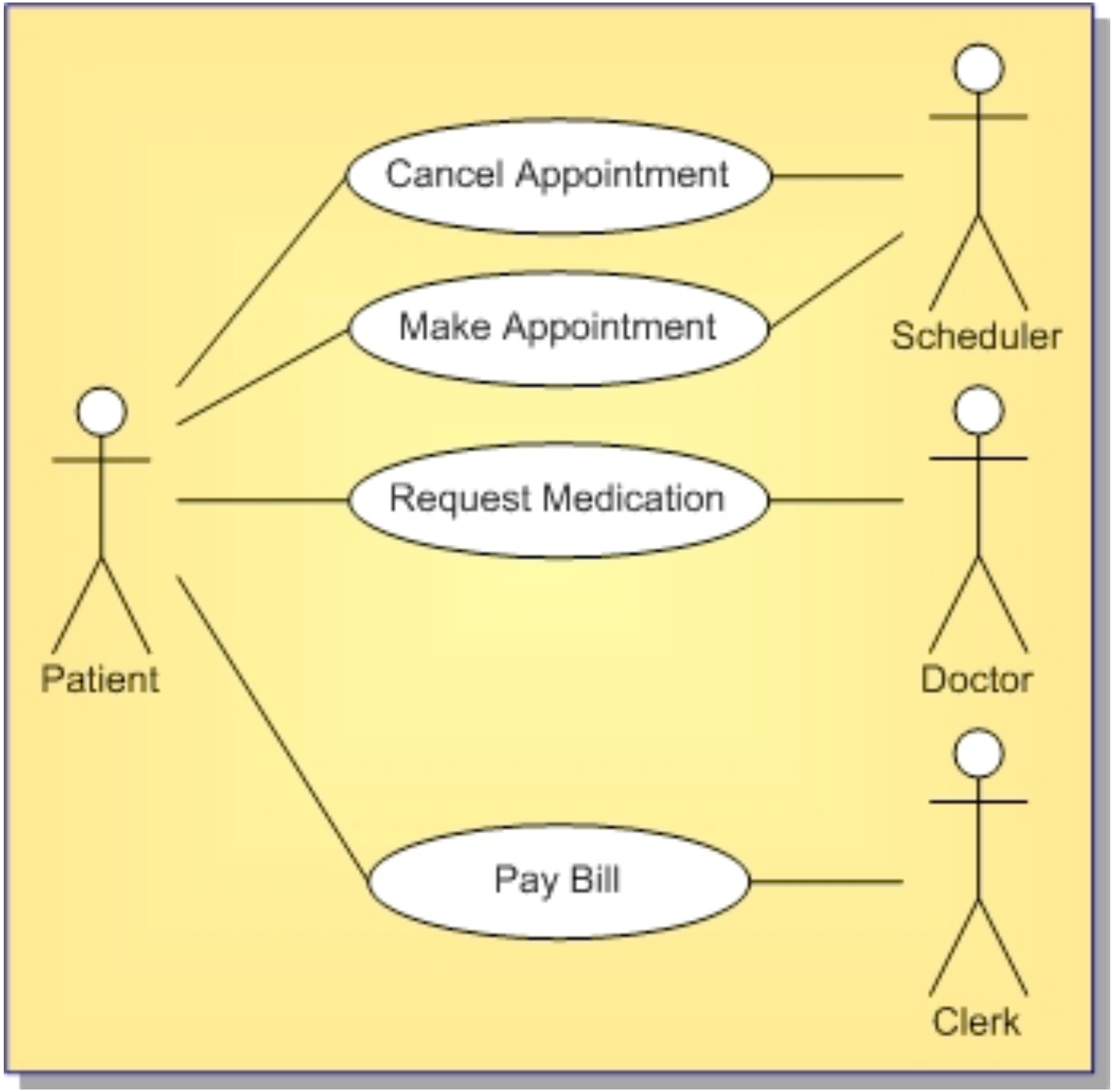
UML Diagram Types

Source: Wikipedia



14 diagram types in total! We'll look only at the three shaded diagram types.

Sample UML Use Case Diagram



Source: Embarcadero Technologies

Use Case or User Story?

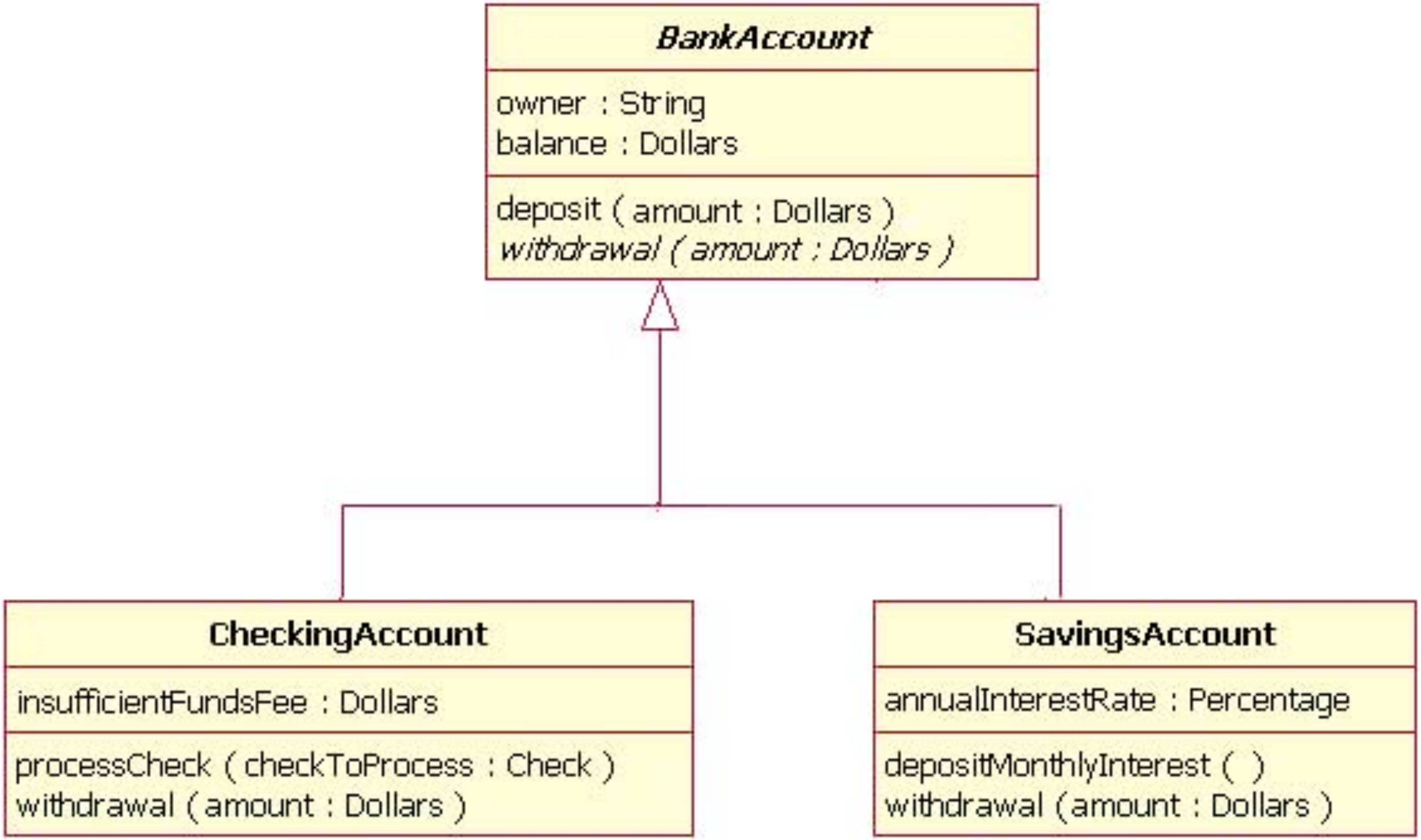
There is no exact definition that distinguishes between the two.

Both describe units of functionality from the end user perspective.

Use Cases tend to be higher-level blocks of functionality, and are usually documented more formally. Associated with a Waterfall process.

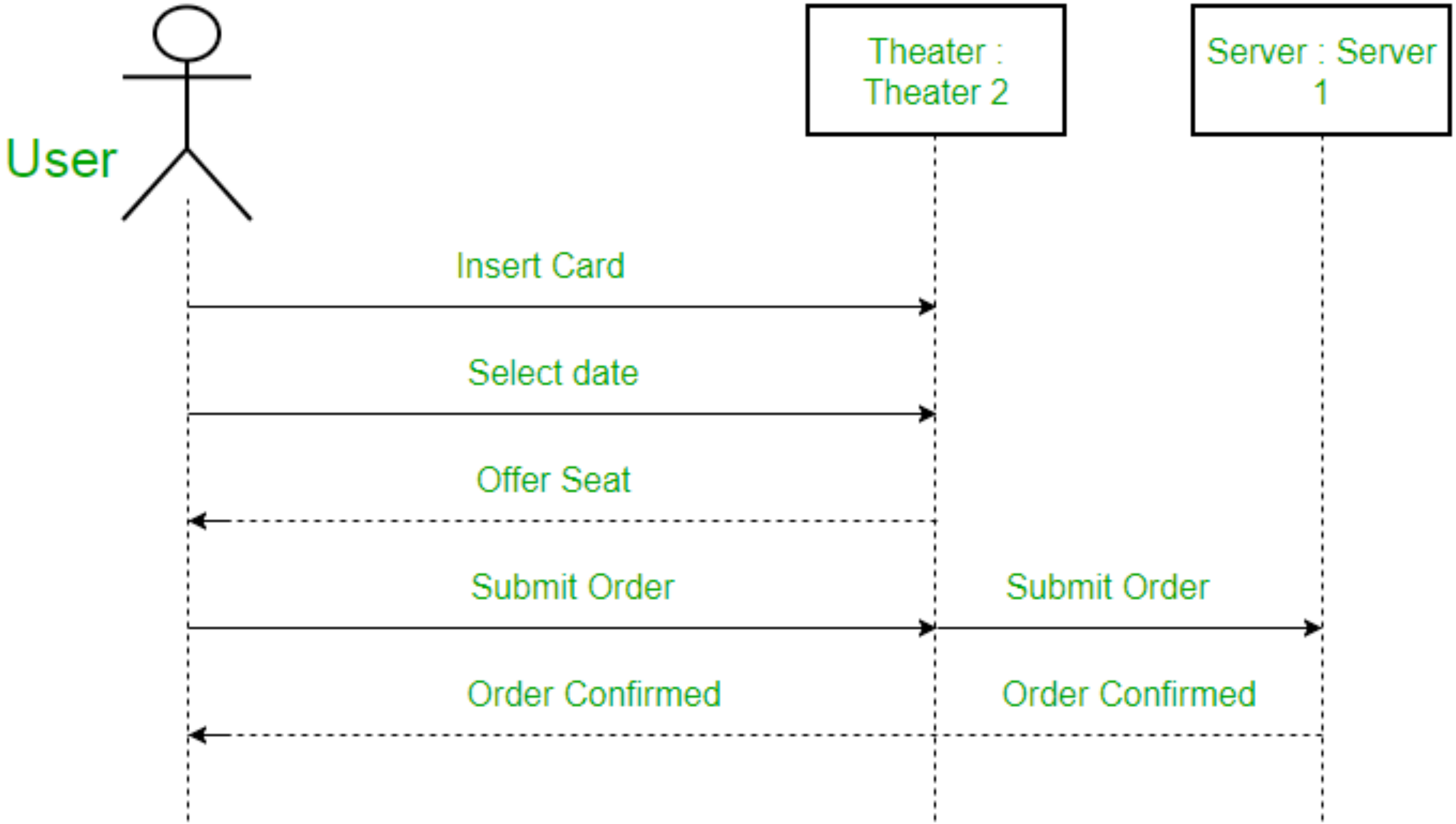
User Stories are minimal units of functionality. Associated with an Agile process.

Sample UML Class Diagram



Source: Rational

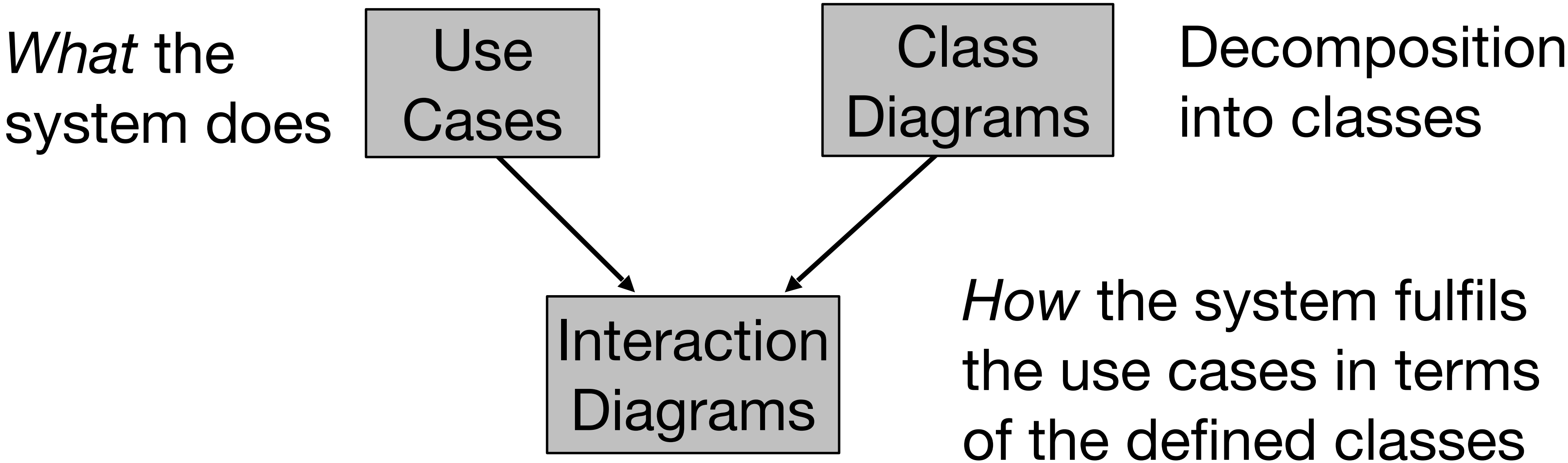
Sample UML Sequence Diagram



Source: geeksforgeeks

How are the diagrams related?

The relationship between the key diagrams we have seen can be visualised as follows:



Note: You would only ever build these models fully as part of a BDUF/Waterfall process.

UML and Agile

UML fits well with the Waterfall/BDUF approach -- it's the modelling language of choice for analysis & design.

Few teams use Waterfall anymore, so what is the role of UML in an Agile context?

(Recall the Agile manifesto: "**working software over comprehensive documentation**").

UML *is* used when discussing design options. So a basic familiarity with it is essential.

Roadmap

In the coming slides, we'll briefly look the three most commonly-used UML diagram types:

1. Use Case Model
2. Class Model
3. Interaction Model (Sequence Diagram)

Note for 2025: Will cover only the **Class Model** as it's the most relevant.

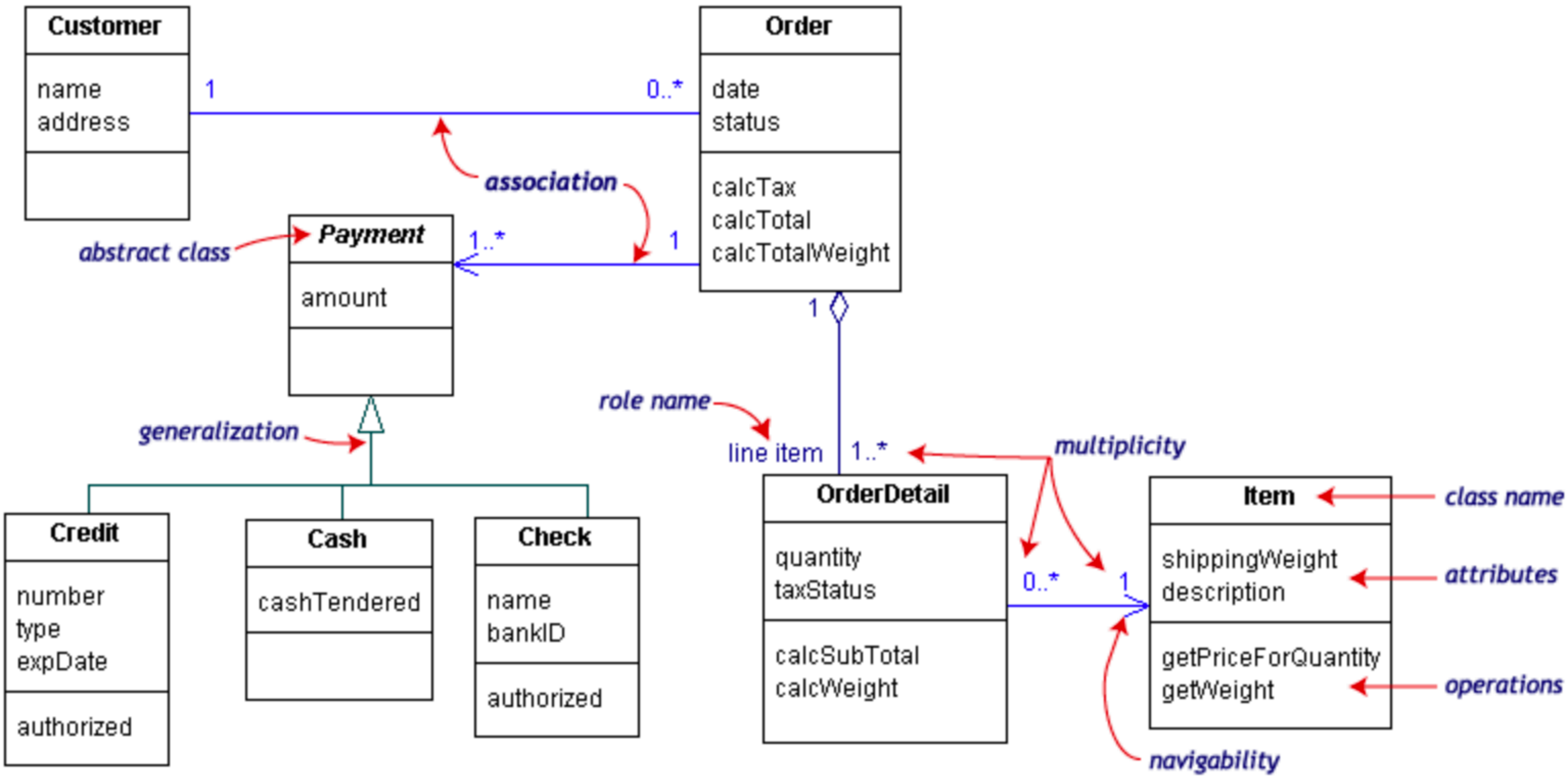
2. Class Diagrams

Class Diagrams depict the main classes and their associations.

High-level modelling is the goal.

Leave implementation details to the code.

Sample Class Diagram



Possible Cardinalities

Symbol	Meaning
0	None
1	One
m	An integer value
0..1	Zero or one
m, *n*	*m* or *n*
m..*n*	At least *m*, but not more than *n*
*	Any nonnegative integer (zero or more)
0..*	Zero or more (identical to *)
1..*	One or more

Domain Models

A **domain model** is a class model that shows only the key abstractions of the domain.

In low-level design, new implementation-related classes may be added to the class model.

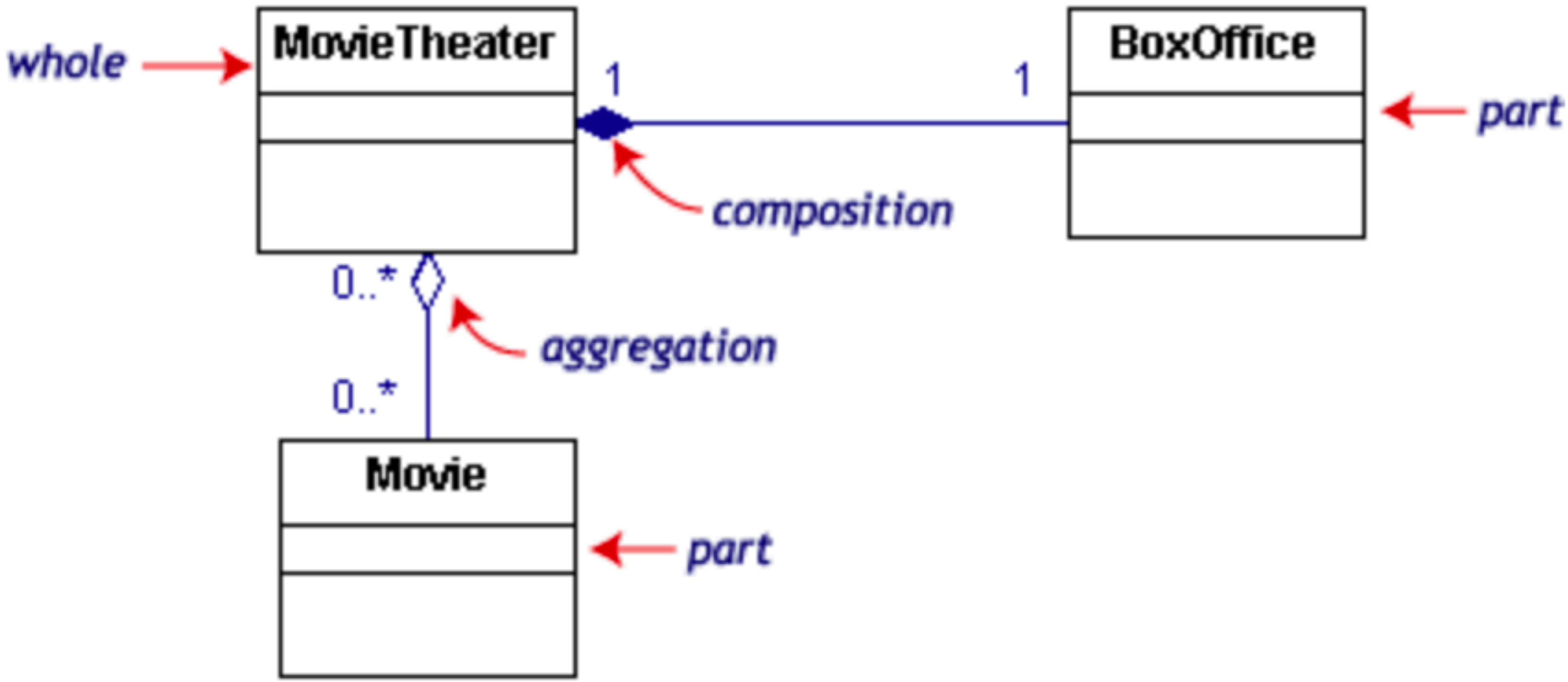
E.g. the earlier Customer/Order example is a domain model.

The implementation will require e.g. a List of Orders, but that is not part of the domain model.

Aggregation and Composition

Aggregation means a whole-part association.

Composition means full ownership. Deletion of the composite => deletion of components



Some Advice

“All models are wrong, some are useful.” — George Box

Modelling with UML can help you understand the problem better, or help you in developing a solution.

If you feel you could solve the problem more easily in code, just do it!

Modelling is pervasive in engineering (blueprints, drawings etc.), why don’t we do it more in software?

Empirical data on UML Usage (1)

Summary of a 2013 study of UML usage:

TABLE I. DECLARED CURRENT UML USE AMONG 50 PROFESSIONAL SOFTWARE DEVELOPERS FROM 50 COMPANIES.

Category of UML Use	Instances of Declared Current Use
no UML	35
retrofit	1
automated code generation	3
selective	11
wholehearted	0

It seems UML is not heavily used in practice.

Source: Petre, Marian, UML in practice, In: 35th International Conference on Software Engineering (ICSE 2013), 18-26 May 2013, San Francisco, pp. 722–731.

Empirical data on UML Usage (2)

From another 2014 study of ~400 software professionals:

"Informal" UML *is* used.

Sketches and diagrams mainly used for designing, explaining, and understanding.

Usually done on paper or whiteboard. Often archived.

Most sketches relate to methods, classes, or packages, but not to source code artefacts.

Source: Baltes and Diehl, Sketches and diagrams in practice, Proceedings of the International Symposium on Foundations of Software Engineering, 2014.

"Diagramming"

While the industry has turned away from UML, *diagramming* or *sketching* is gaining some interest.

The idea is to develop diagrams in specific situations where code is less effective, e.g.

- to visualise the system architecture
- to describe complex object interactions
- to visualise the effect of a multi-step refactoring

The Baltes and Diehl paper from the previous slide provides evidence to support this.

Notes: SE-Radio Episode 566 (May 2023) covers the use of diagramming in software development. Based on the book *Creating Software with Modern Diagramming Techniques*, Ashley Peacock, Pragmatic Bookshelf, 2023.

Diagramming Tooling

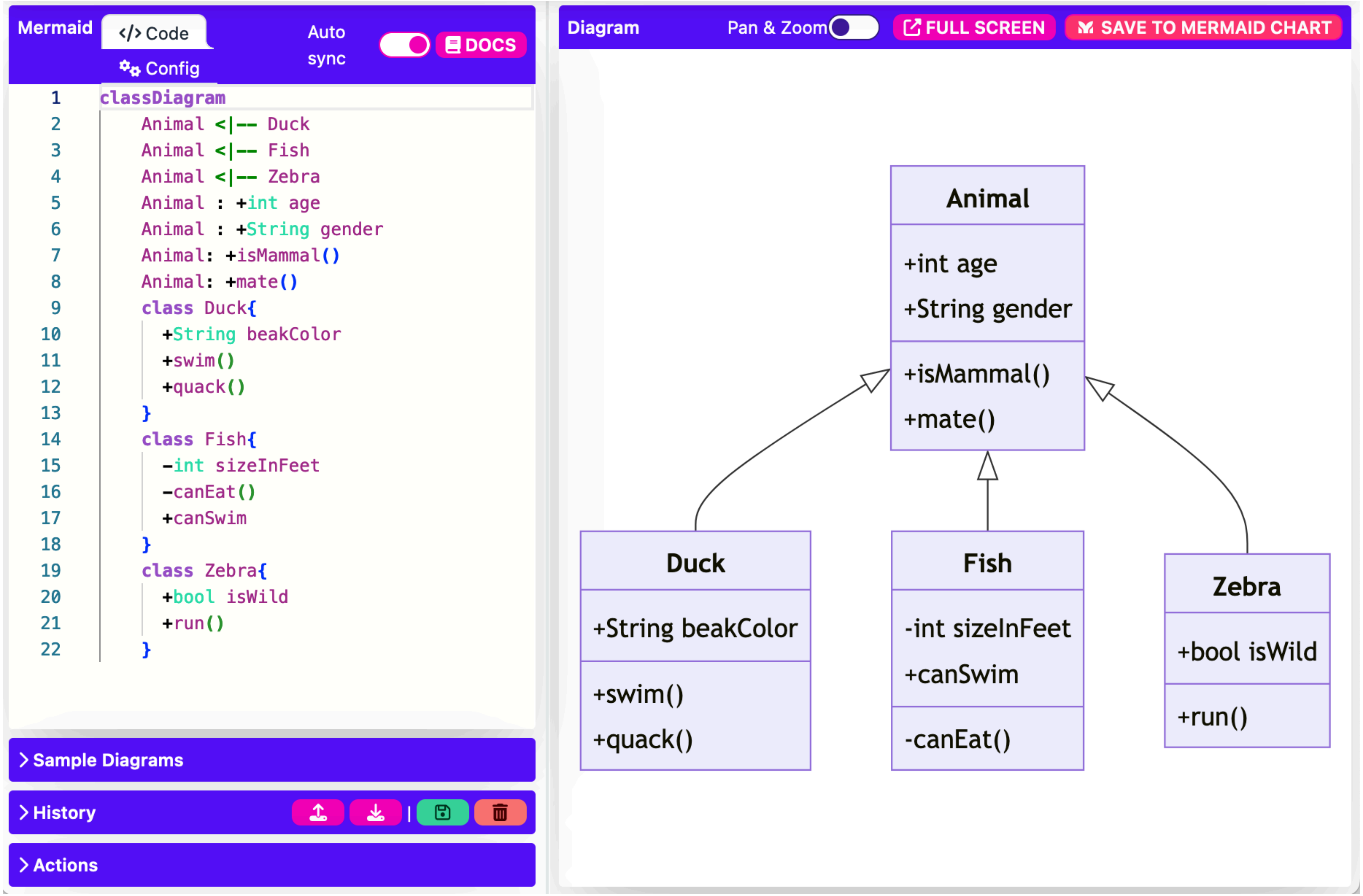
While a UML-like language is used, notation details are not important.

The main diagram types used are **class diagrams** and **sequence diagrams.**

Tools like PlantUML or Mermaid enable diagrams to be drawn using a markdown-like language.

- Easy to store alongside code
- Can be versioned easily in a Git repo
- Example on next slide

Mermaid Example



Summary

UML is the standard object-oriented modelling language.

Although it is not used to drive development, developers are usually acquainted with it.

We looked at three key UML diagram types:
Use Cases: what the system does
Class Diagrams: key classes of the system
Sequence Diagrams: how objects interact

You should be able to do simple modelling using these diagram types.