Creating UML Class models

Software Requirements and Design

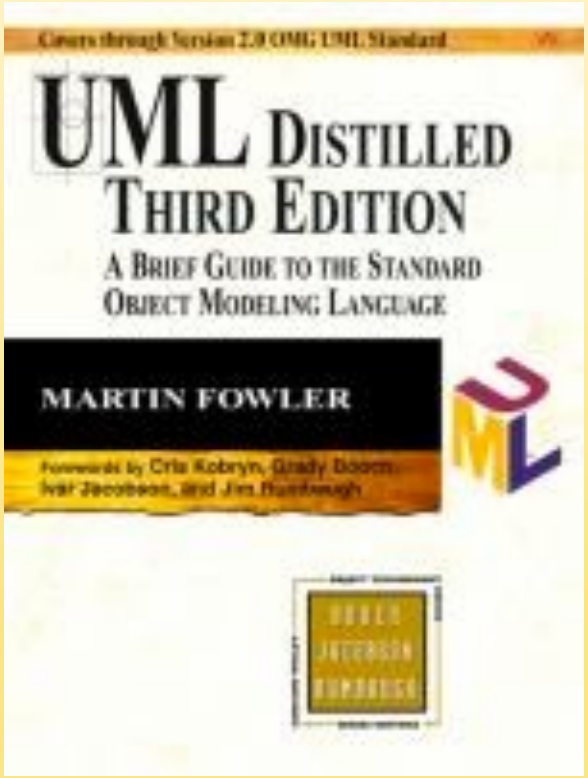
CITS4401

Lecture 10

Key Topics this week

1. UML Class diagrams (what they are for and how to read them)

2. Discovering **objects** (noun discovery method) and **associations** (Class, Responsibilities, Collaboration (CRC) method)



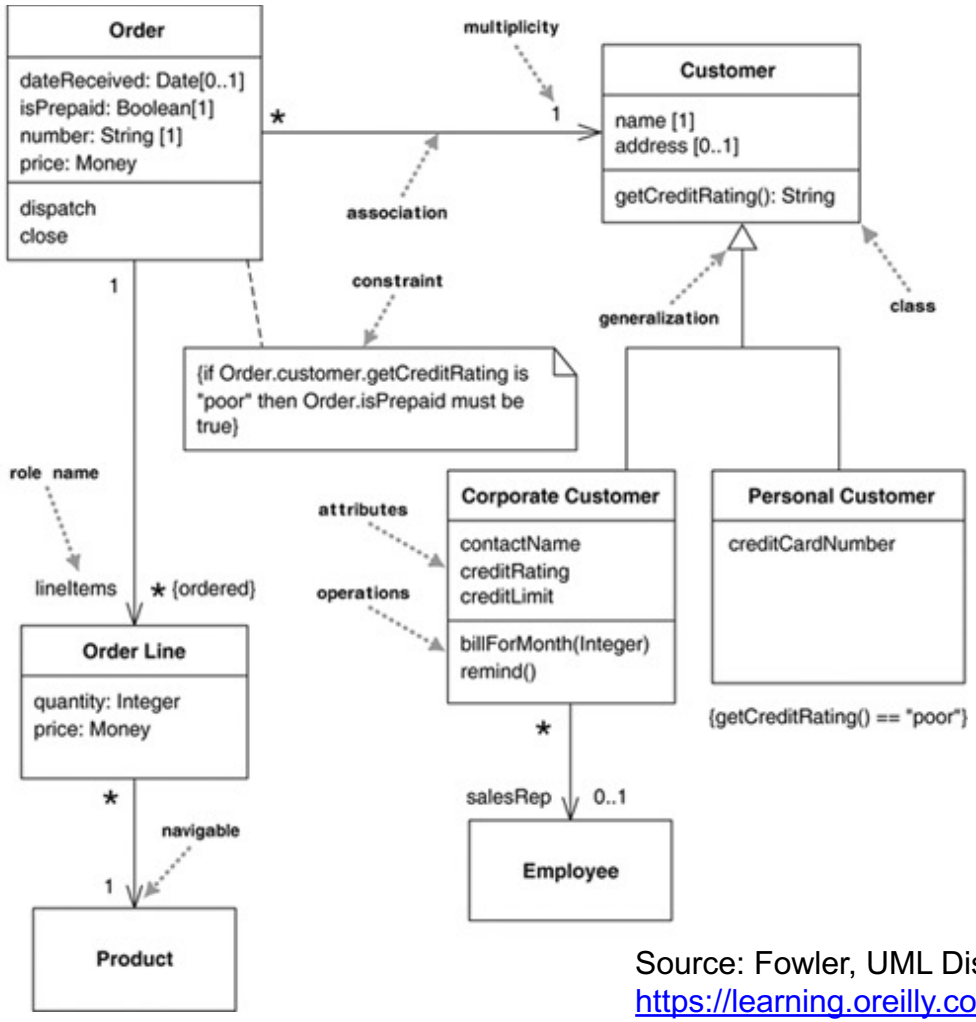
Requirements Analysis Framework

- Requirements analysis results in an analysis model of 3 parts
  - functional model: use cases & scenarios
  - static analysis object model: class & object diagrams
  - dynamic model: statechart & sequence diagrams (later in the unit)
- This week we will look at **static class models**
- Aim to investigate the problem domain as far as possible before moving to the solution domain (for design & implementation)

1. UML Class diagrams

Last lecture we learned how to READ and understand a UML class diagram

UML Class Diagram

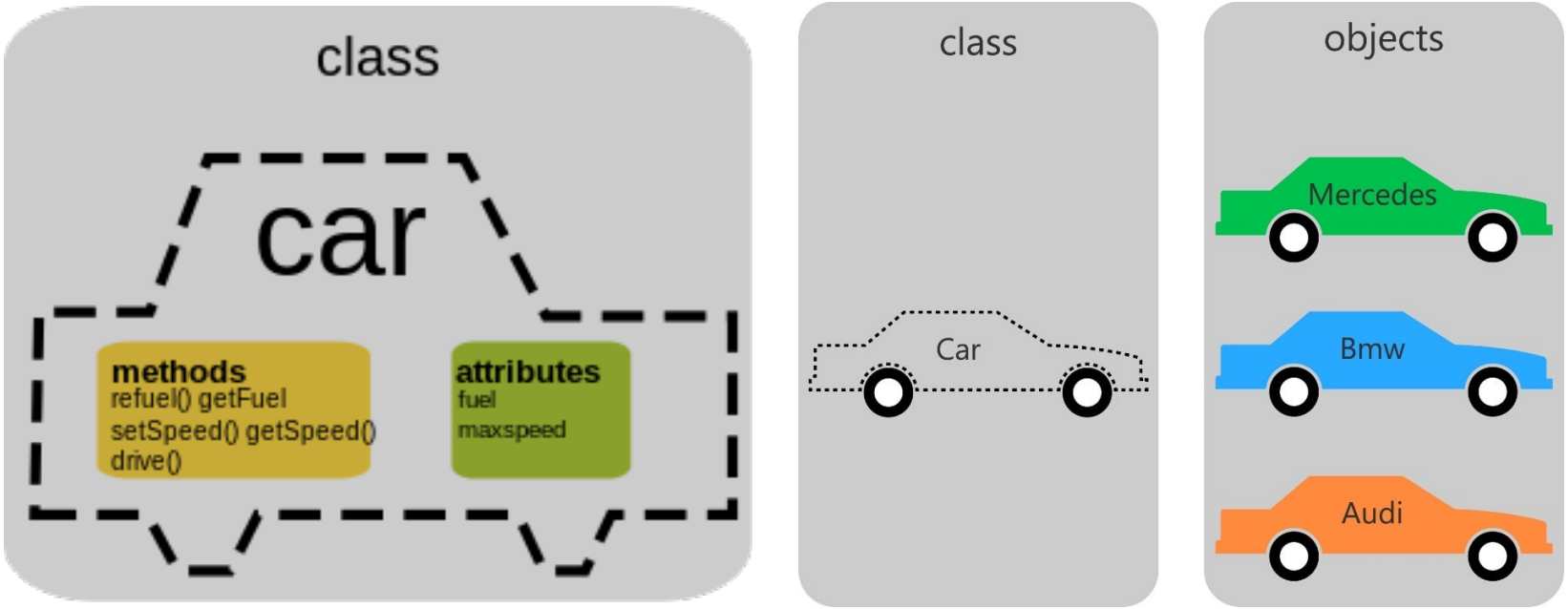


Source: Fowler, UML Distilled, Figure 3.1
https://learning.oreilly.com/library/view/uml-distilled-a/0321193687

2. Discovering objects and associations

Now we know how to READ UML class models, we'll look at how to discover classes and associations from problem descriptions. Then you'll be able to WRITE UML class models.

OOP concept


class
car
methods
refuel() getFuel
setSpeed() getSpeed()
drive()
attributes
fuel
maxspeed
class
Car
objects
Mercedes
Bmw
Audi

Actors, Objects and Classes

- What is the difference between an actor and a class and an object?
- Actor:
  - An entity outside the system to be modeled, interacting with the system (“Driver”)
- Class:
  - An abstraction modeling of an entity in the problem domain, inside the system to be modeled (“Car”)
- Object:
  - A specific instance of a class (“my white Toyota Corolla hybrid”).

Examples of Class Components

- **Name:** employee
- **Attributes:** name, address, dateofbirth, employeenum, department, manager, salary, status, taxcode
- **Operations:** join, leave, retire, changedetails

Video References

Barnes and Kolling, Objects First with Java

Chapter 1: VN 1.1 Introduction to key concepts (in object oriented design)
https://www.youtube.com/watch?v=CPUaTT0Xoo4&t=7s

Chapter 13: VN 13.1 Using the noun-verb method for application design
https://www.youtube.com/watch?v=sbQqb_XXOK8

Noun-Verb Method

- In a problem description …

- NOUNS may represent
  - Objects and Attributes
- VERBS may represent
  - operations or services

- The noun-verb method throws up noise
- So it needs to be used alongside other methods

Heuristics

Heuristics for Mapping parts of speech to model components
Source Abbot 1983 quoted in Bruegge and Dutoit Object-Oriented SW Eng

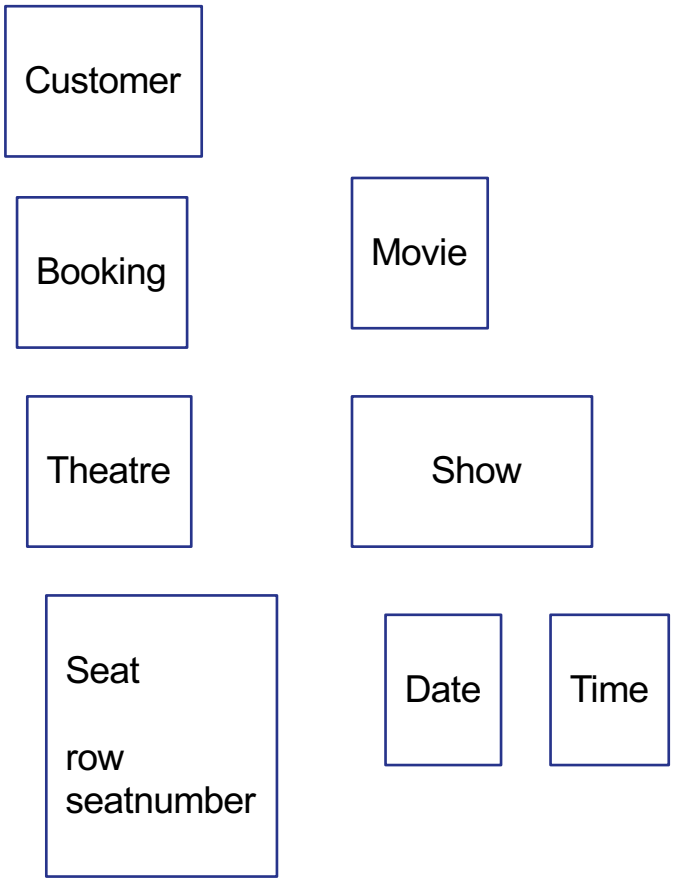
Part of Speech	Model component	Examples
Proper noun	Instance	Alice
Common noun	Class	Field officer
Doing verb	Operation	Creates, submits, selects
Being verb	Inheritance	Is a kind of
Having verb	Aggregation	Has, includes
Modal verb	Constraints	Must be
Adjective	Attribute	Incident description

Case Study: Cinema booking system

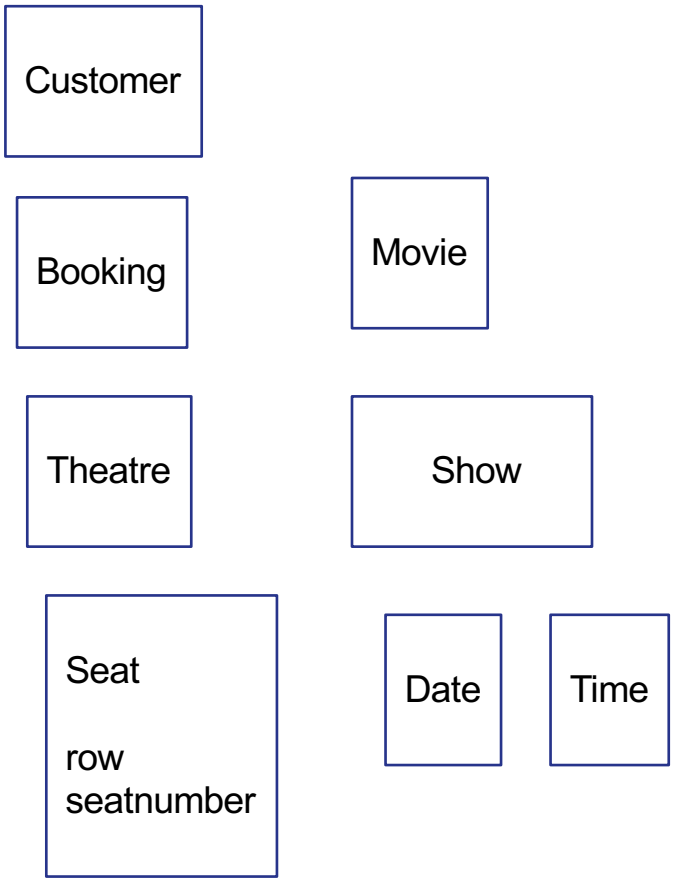
- The cinema booking system should store seat bookings for multiple theatres. Each theatre has seats arranged in rows.
- Customers can reserve seats and are given a row number and a seat number. They may request bookings of several adjoining seats.
- Each booking is for a particular show (that is a screening of a given movie at a certain time).
- Shows are at an assigned date and time and are scheduled for a theatre where they are screened.
- The system stores the customer's telephone number.

- Reference Barnes and Kolling Objects First Chapter 15
- Video Notes: VN 13.1 Using the noun-verb method for application design

Cinema Booking System
Nouns = potential classes



Cinema Booking System



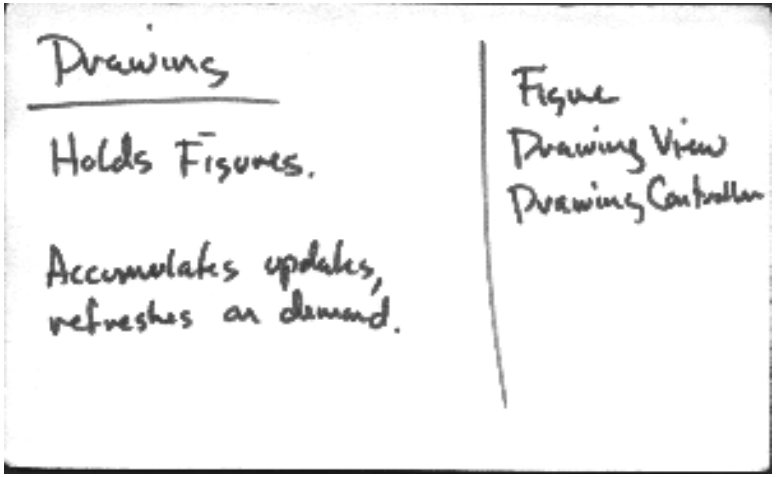
Verbs = potential associations

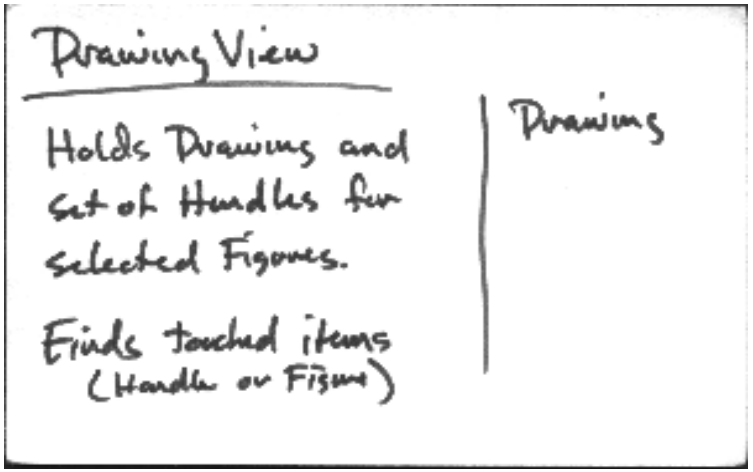
- stores (seat booking)
- has (seats)
- reserves (seats)
- Is given (row and seat number)
- Requests (seat booking)
- is scheduled (in theatre)

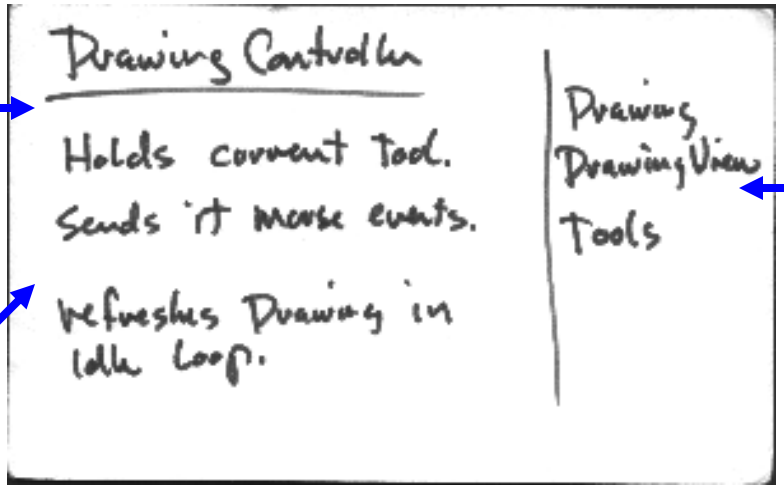
Class-Responsibility-Collaborator Model

- the class name of an object
  - creates a vocabulary for discussing a design
- responsibilities of an object
  - identify problems to be solved
  - a handful of short verb phrases, each containing an active verb
- collaborators of an object are
  - other objects which will send or be sent messages in the context of satisfying responsibilities
- CRC cards are a brainstorming tool for OO design
- Recommended for Extreme Programming XP methodology

CRC Example







class name

responsibilities

collaborators

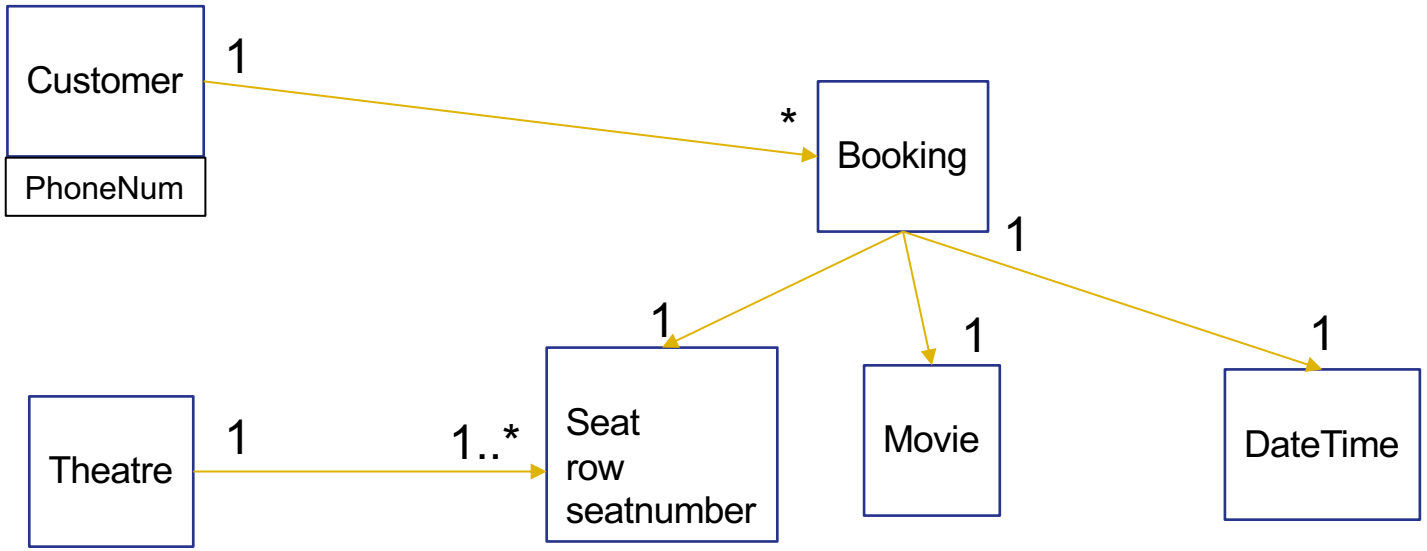
Cinema Booking Scenario CRC

- Alice selects the movie Star Wars at the Apollo Theatre
- She requests the 8pm showing on 24 March 2020
- The booking system sends the request to Apollo
- Alice is allocated seat R18

Customer	
Selects theatre, movie and show date and time	Show
Receives tickets	Theatre

Theatre	
Receives booking request	Customer, Show
Allocates tickets	Customer, Show

Cinema Booking System
UML class diagram (version 0)



BookingSystem
(could store a collection of shows to book)

This is a model in progress
Ideas and changes welcome

Drawing Options

There are many free and paid software tools for drawing UML diagrams

See https://www.guru99.com/best-uml-tools.html for a list

I tried out StarUML, UML designer and Lucidchart.
But I found all of them were hard to use and had too much detail for our purposes.
ArgoUML was popular in the past but is not supported in modern Java

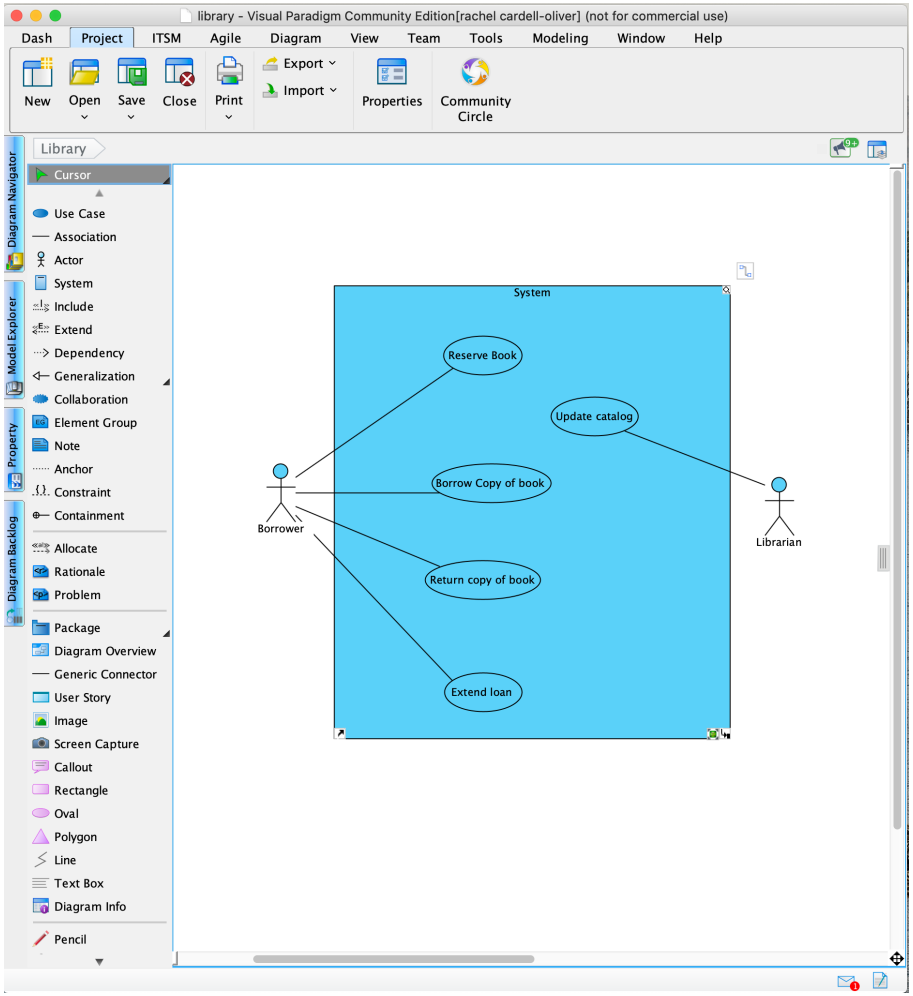
Option: you can use any editor with text boxes and arrows – as I have done with powerpoint in the notes

Option: you can draw class diagrams by hand

If you find an easy to use and free tool, or good ppt template do let me know!

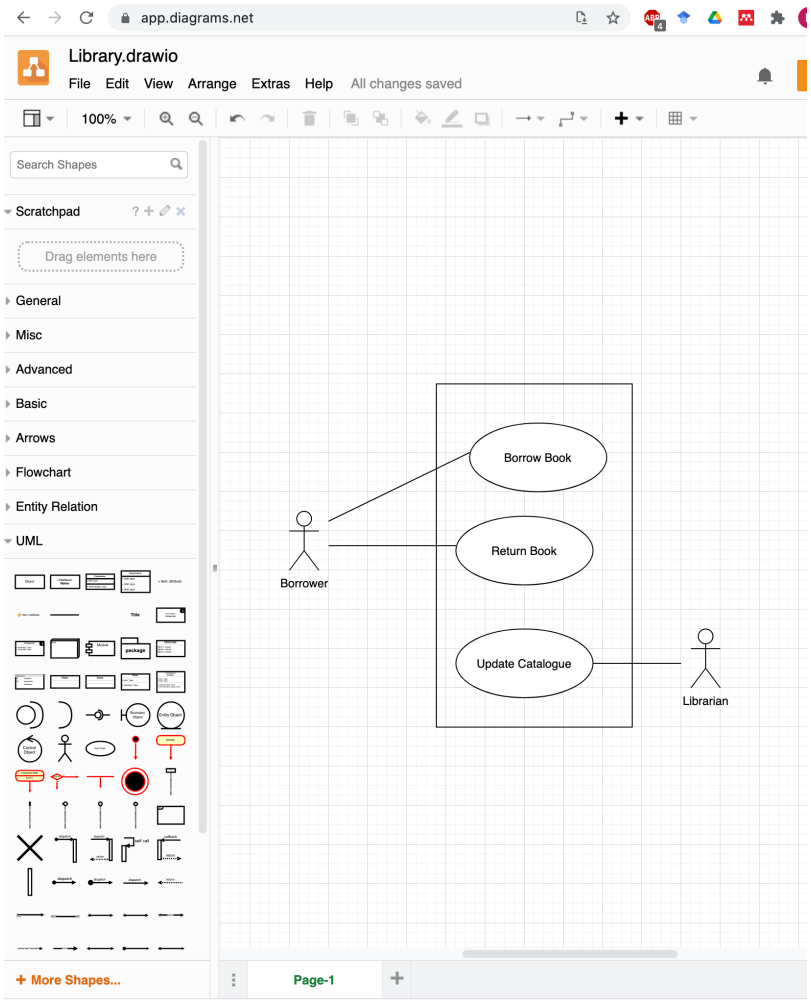
UML drawing tools

- https://www.visual-paradigm.com/download/community.jsp
- Make sure you download the FREE community edition (free for non-commercial use)



UML drawing tools (cont)

- https://app.diagrams.net/
- draw.io
- Online portal



Summary

1. UML Class diagrams (what they are for and how to read them)
2. Discovering objects (noun discovery method)
3. Discovering associations (Class, Responsibilities, Collaboration (CRC) method)

UML class diagrams

UML Class Diagrams describe the **static** structure of the system

- classes,
- class attributes,
- associations between classes,
- association roles and multiplicity

classes: *features and facts about the problem domain which matter in the system we are building to support it*

Reference: UML Distilled by Martin Fowler, Chapter 3

Our focus

- Class diagrams are the backbone of the UML, so you will find yourself using them all the time.
- But the trouble with class diagrams is that they are so rich, they can be overwhelming to use.
- In CITS4401 we will study these elements:

  classes, associations, attributes, generalization, and constraints

- For more detail: see UML Distilled by Martin Fowler, Chapter 3 & 5
- And some info on the hidden slides – not examinable but for interest

When to Use Class Diagrams

[Martin Fowler]

- Class diagrams are the backbone of the UML, so you will find yourself using them all the time.
- The trouble with class diagrams is that they are so rich, they can be overwhelming to use. Here are a few tips.
- **Don't** try to use all the notations available to you.

  Start with the **simple stuff**: classes, associations, attributes, generalization, and constraints. Introduce other notations only when you need them.
- I've found conceptual class diagrams very useful in exploring the language of a business. For this to work, you have to work hard on **keeping software out of the discussion** and keeping the notation very simple.

When to use (2)

- **Don't** draw models for everything; instead, concentrate on the key areas. It is better to have a few diagrams that you use and keep up to date than to have many forgotten, obsolete models.

- The biggest danger with class diagrams is that you can focus exclusively on **structure** and ignore **behaviour.**

  Therefore, when drawing class diagrams to understand software, always do them in conjunction with some form of behavioural technique. If you're going well, you'll find yourself swapping between the techniques frequently.

- Next week we will study 2 behavioural UML models: sequence diagrams and state charts.

Recommended reading

UML Distilled by Martin Fowler, Chapter 3

UWA Unit Readings or
https://my.safaribooksonline.com/book/software-engineering-and-development/uml/0321193687

Barnes and Kolling, Objects First with Java, Chapter 15

B. Bruegge and A. H. Dutoit, *Object-Oriented Software Engineering – Using UML, Patterns, and Java,* 3rd ed., Prentice Hall, 2010

- Section 2.2
- Section 2.4
- Section 5.3