Object Oriented Design

Software Development Processes

Dr. Seán Russell

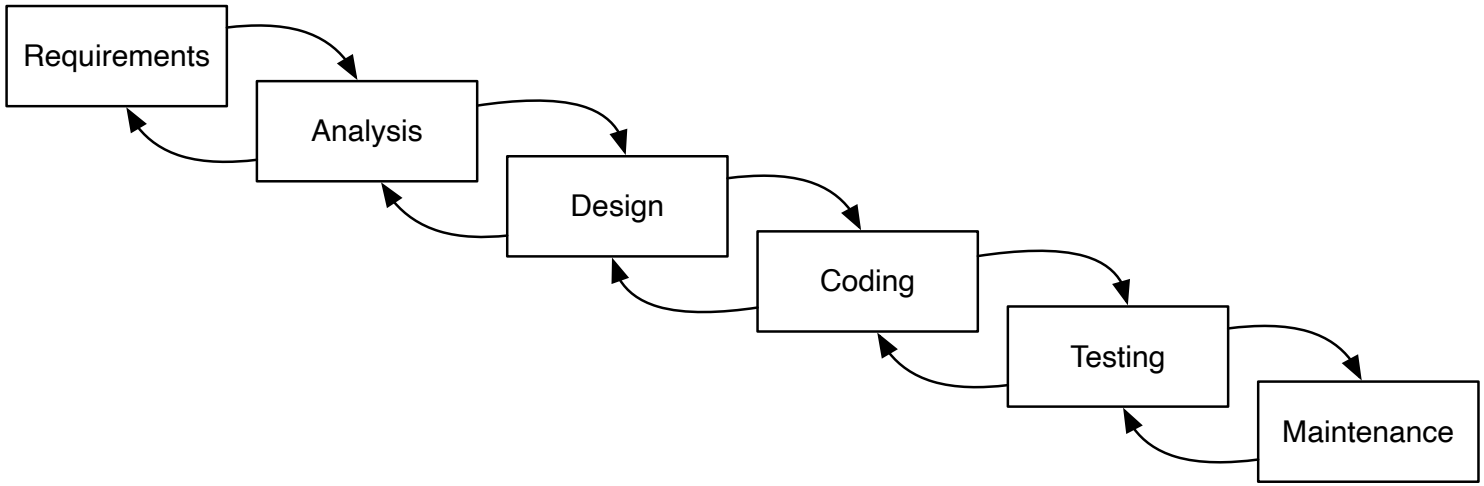
School of Computer Science,
University College Dublin

Lecture 02

Table of Contents

Section Contents

- Early development processes based on how engineering disciplines work
- Distinct activities were identified
  - Requirements, analysis, design, coding, etc.
- The order they should be complete was defined
- This led to the development of the classic **waterfall** model



- Different versions have different stages
- All move systematically from high-level to low-level
- Documentation produced for next stage

- The waterfall model is a good management tool
  - Clear structure with milestones
  - Documentation read for the next stage
- Not always good, two major reasons for failure
  1. Management of risk in projects
  2. Treatment of requirements

- Testing is the best way of finding problems in software
- Planned testing is very late in the waterfall model
- If a large problem is discovered, we may need to return to analysis, design or coding and repeat a lot of work
- There is no limit to the cost of solving the problem

- Requirements gathering is a single step at the beginning of the waterfall model
- But, requirements gathering is very hard
  1. The requirements of many systems are complex
  2. Environment of the software may change during development
  3. Involvement in the process can change the requirements

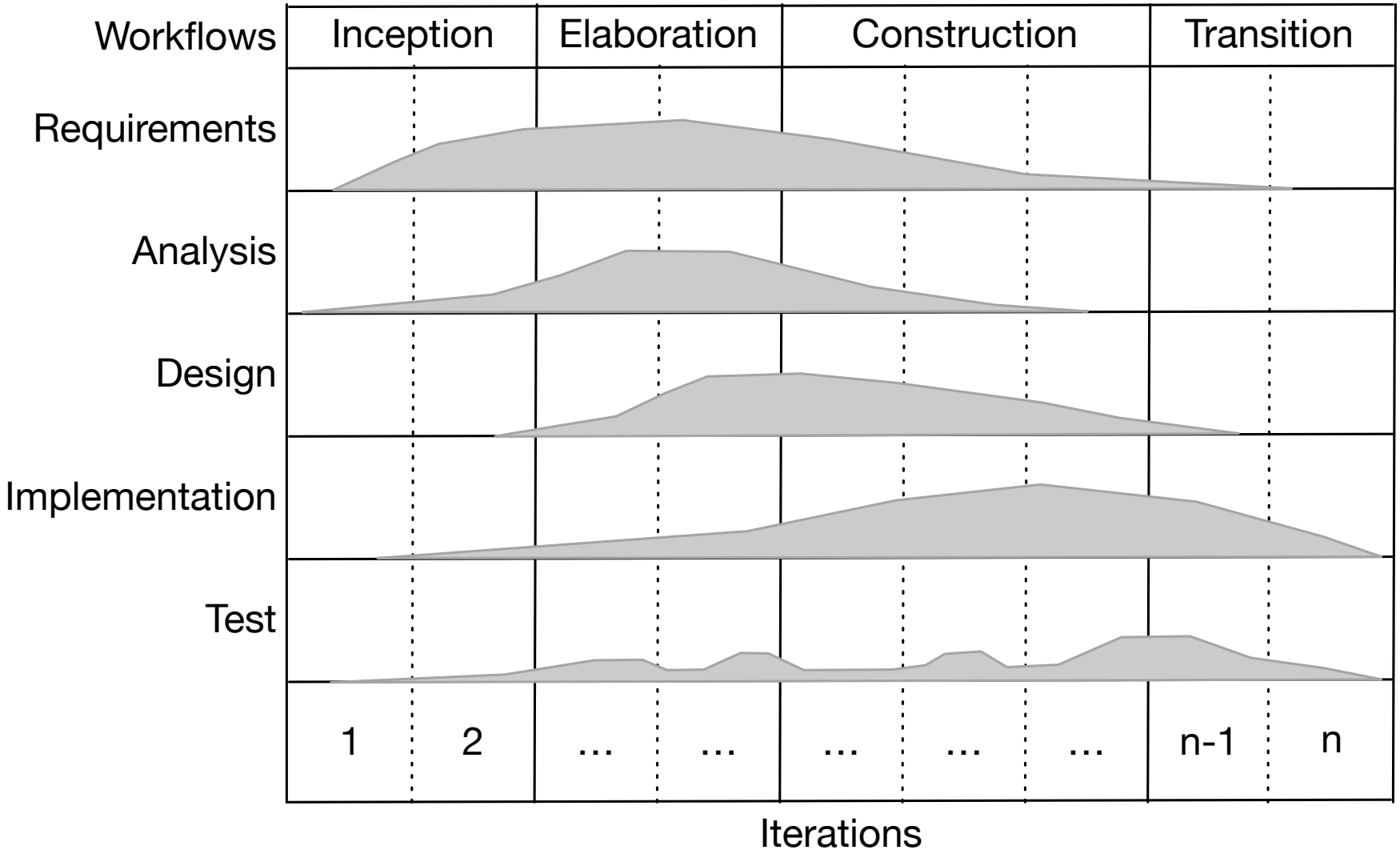
Section Contents

- Most modern processes incorporate an element of **iterative** development
- This is where development is organised as a repeated completion of the development activities
- Each iteration of the development process will achieve a part of the overall development or potentially refactor a part that has already been completed

- Modern process will also usually incorporate an element of **incremental** development
- This similarly organises the development into phases
- However, here the focus of the development is initially developing a prototype
- Then subsequent phases will add new functionality to the prototype or improve the existing functionality

- These two ideas are very compatible, both break development into iterations but have a different focus
  - Iterative development is focused on the organisation of different activities
  - Incremental development is focused on what functionality is developed and when

- We will be loosely following the Unified Process (UP) as an example of a design methodology
- UP includes activities such as requirements analysis, analysis, design, implementation, and testing
- It is not clear how these activities should fit into an iterative and incremental process
- So UP allows for a variable amount of work to be completed on each activity in an iteration


Workflows
Inception
Elaboration
Construction
Transition
Requirements
Analysis
Design
Implementation
Test
1
2
...
...
...
...
...
n-1
n
Iterations

Section Contents

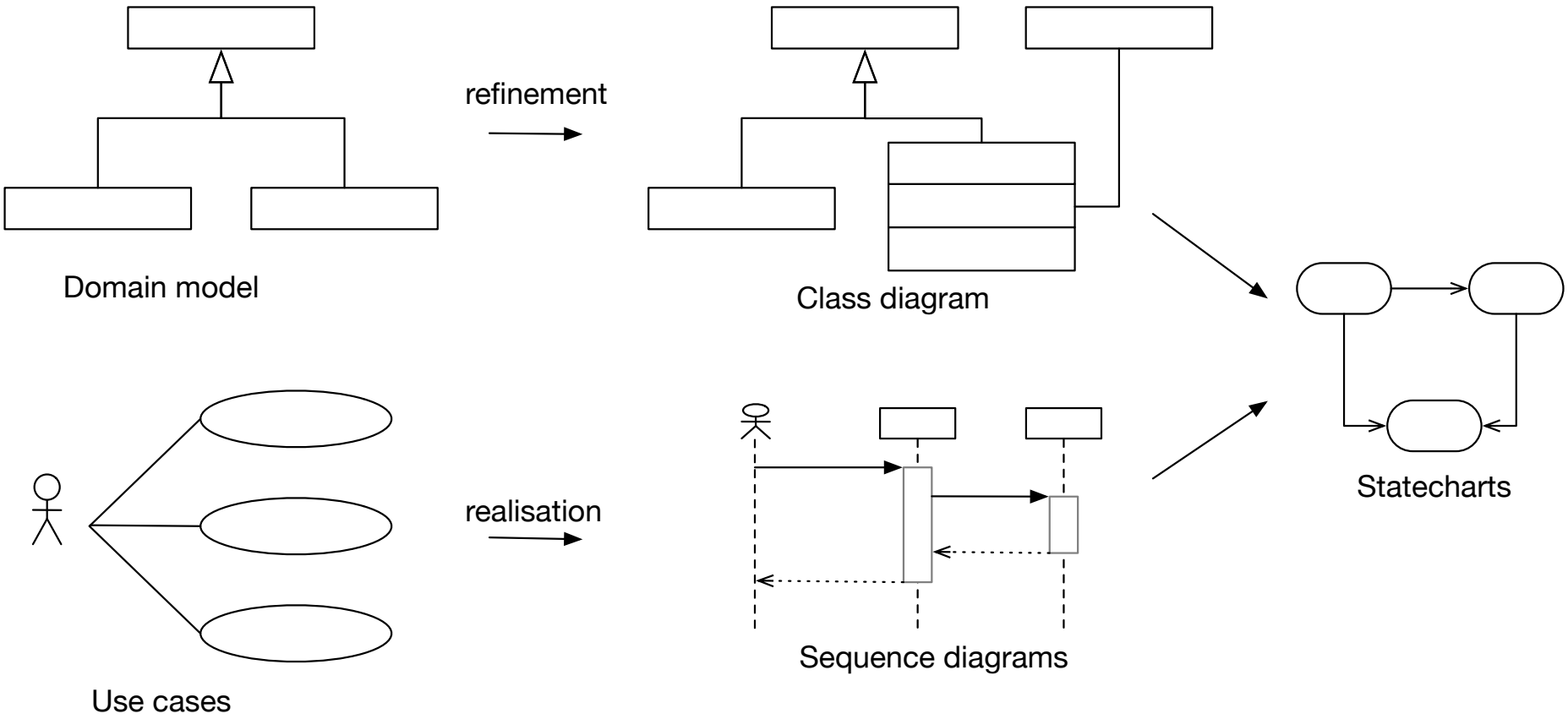
- UP assumes that models are important in software development
- It was developed with UML, so specific diagram types are expected to be used
- The diagrams used will vary depending on the workflow

- Some processes like Extreme Programming (XP) make little use of models
  - XP does not recommend the use of models in the same way as UP
- XP is often thought of as being anti-model

- Keeping code and models consistent can be difficult in iterative processes
  - Designs are often adapted/changed when being implemented
  - If the model is not also updated, it will not match the code in the next iteration
- Two possible solutions recommended
  1. Round-trip engineering
  2. Only one documentation

Section Contents

- UP emphasises the capture of system requirements
- Functionality is recorded in use case descriptions
- Actors and use cases are represented in a use case diagram
- Usually this is also supported with a **domain model**
  - This is a very basic class diagram


refinement
Domain model
Class diagram
Statecharts
realisation
Use cases
Sequence diagrams

- During the Analysis and Design phases the information in the use case documents is used to refine the design
- Sequence diagrams are produced to show how the components of the system will interact
- The details of these interactions are added to the classes in the class diagram