

# Software Development

## Contents

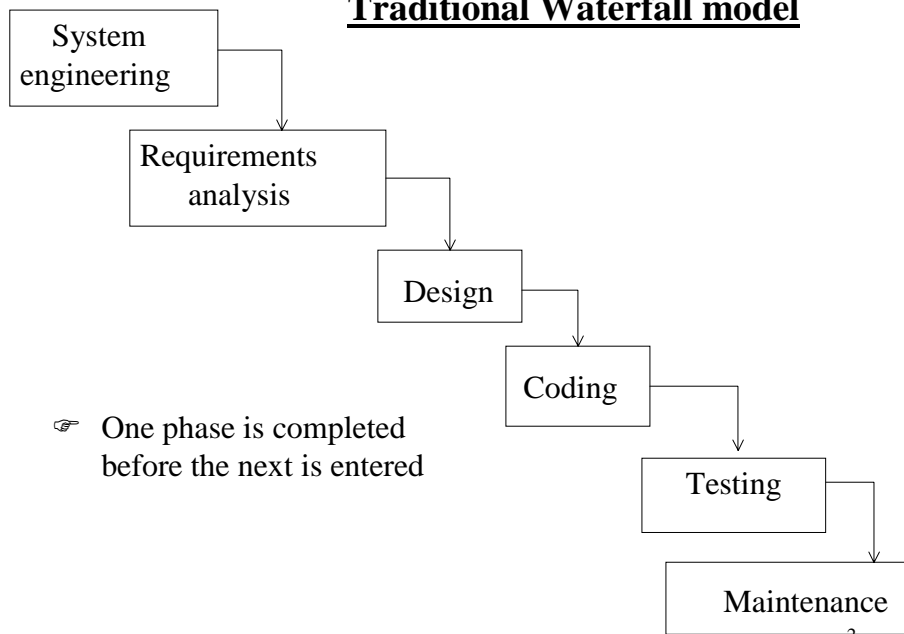
1. Traditional Waterfall model
2. Prototyping
3. Iterative Approach to Object-Oriented Development
4. Benefits of Object-Oriented Methodology

## Literature

- *Object-oriented systems analysis and design using UML*  
Simon Bennett, Steve McRobb and Ray Farmer.  
McGraw-Hill, 1999.

1

## Traditional Waterfall model



- **System engineering**

- High level specification
- Defines major elements: human, software, hardware and how they interact

3

- **Requirements analysis**

- Defines users requirements
- Use of fact finding techniques and techniques to document them (Interviewing, Data Flow Diagrams ...)
- Requirements specification contains:
  - detailed description of all the tasks
  - constraints (response time, memory utilisation)
  - design / implementation directives
  - +
  - maintenance support for the system
  - training of the customer' staff

4

- **Design**
  - Determines how to construct a system that delivers the requirements
  - Specification of a software architecture
  
- **Coding**
  - Translation of design into program code

5

- **Testing**
  - Ensures that the system meets requirements
  - Several levels of testing
  
- **Maintenance**
  - The system is subject to change
  - Corrective maintenance (corrections of the errors)
  - Perfective maintenance (implementation of certain aspects of system behaviour)
  - Adaptive maintenance (accomodation of changing requirements).

6

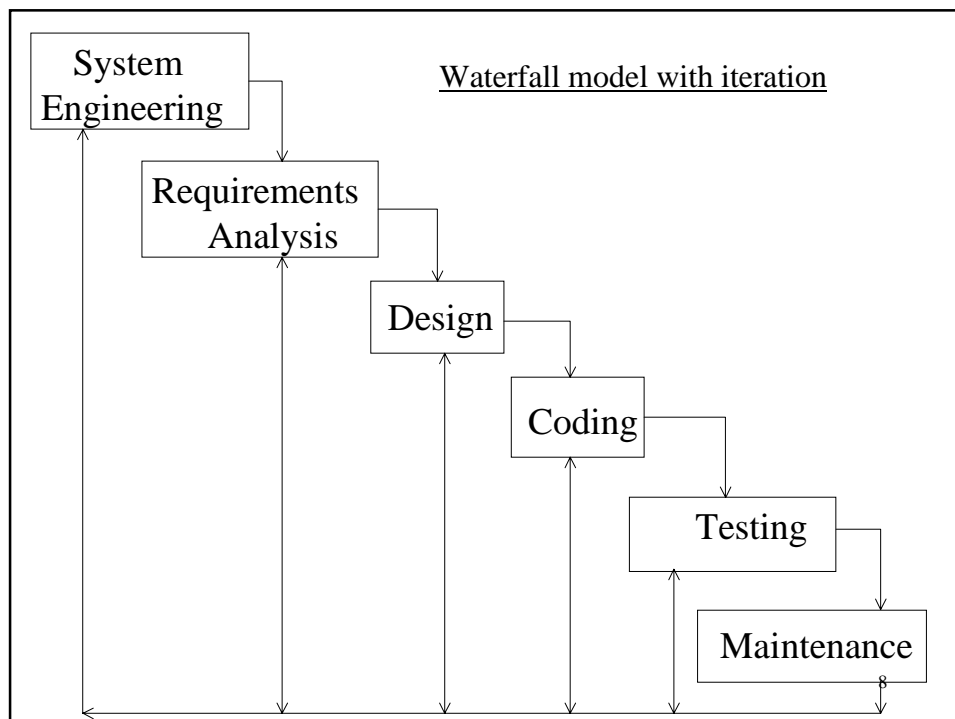
### Advantages

- Enables allocation of tasks within a phase.
- The progress can be evaluated at the end of each phase.

### Disadvantages

- Projects rarely flow in a sequential process.
- Difficult to define all requirements at the beginning of a project.
- Unresponsive to changes.
- A working version of the system is not seen until late in the project's life.
- Repairing problems further along the lifecycle becomes progressively more expensive.
- Maintenance costs can be as much as 70% of systems costs

7



## **Prototyping**

- A **prototype** is a system or partially complete system that is built quickly to explore some aspects of the system requirements .
- Constructed with various objectives

9

## **Advantages**

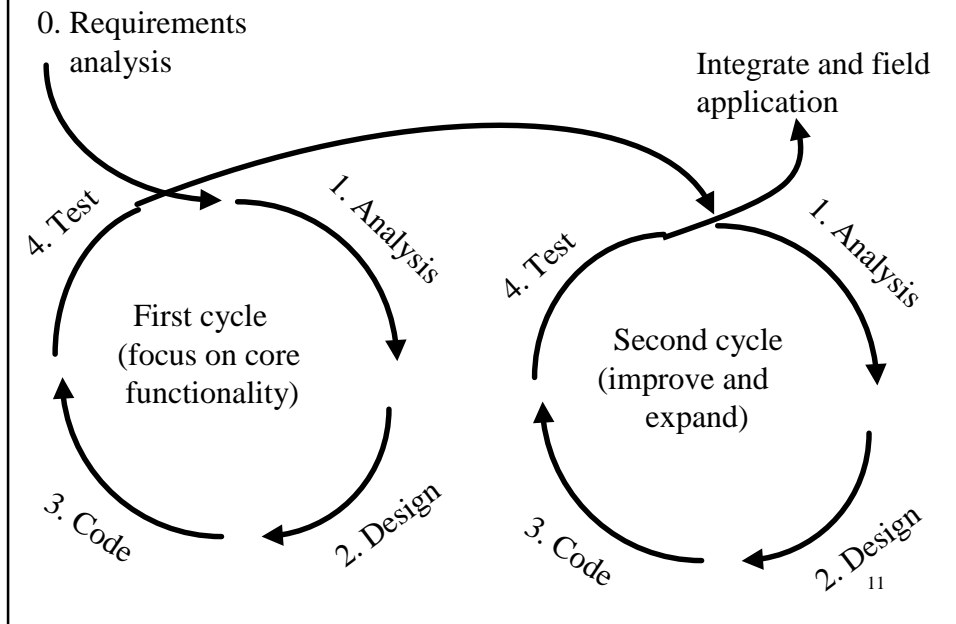
- Early identification of misunderstandings between developers and users.
- Identification of difficulties in the interface.
- Feasibility and usefulness of the system can be tested.

## **Disadvantages**

- The user may perceive the prototype as part of the final system.
- May divert attention from functional to interface issues .
- Requires significant user involvement.
- Difficult management of the system life cycle.

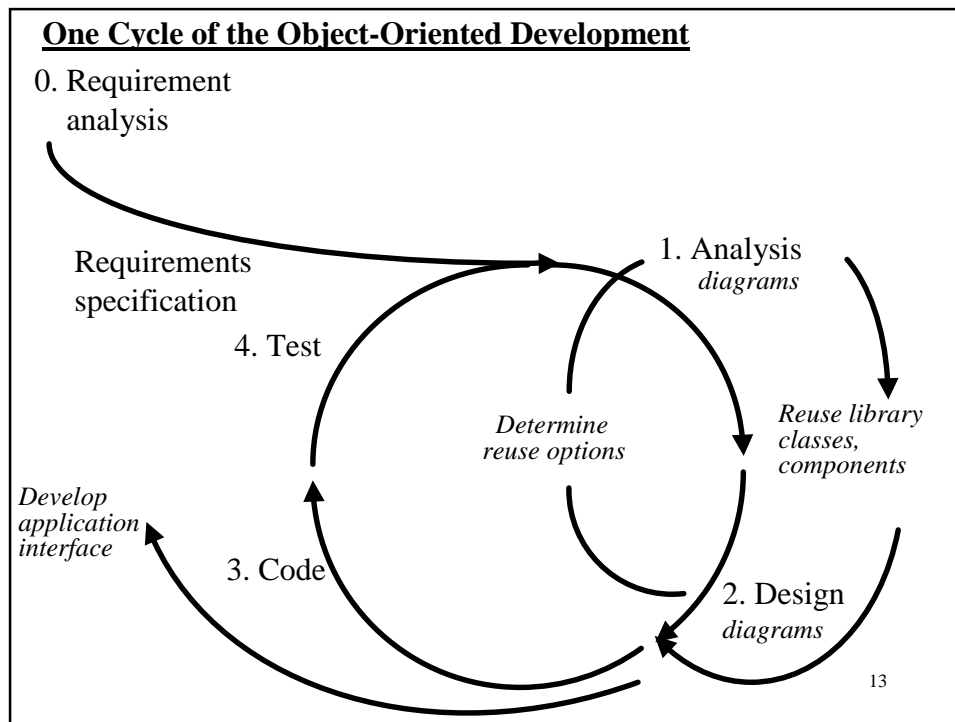
10

## An Iterative Approach to Object-Oriented Development



1. Define the scope and requirements for the entire project and create a high-level description of the entire application.
2. Select a portion of the entire application to develop first
  - the hardest part of the application or
  - a portion of the system that supports some of the functionality that the end users are most interested in.
3. Plan what part of application will be developed during the first prototyping cycle, the second cycle, and so on.

12



- Phases of a development cycle**
- ◆ **Requirements analysis**
    - specification of the scope and requirements
    - accompanied by scenarios of the system's behaviours
    - a good requirements statement enables easier modularization of the application.
  - ◆ **Analysis Phase**
    - diagrams
    - no rigid distinction between analysis (what the application should do) and design (how to do it)
    - classes from the domain of the application refer to real things in the application
    - infrastructure classes handle the details of programming
- 14

◆ **Design Phase**

- extended diagrams + infrastructure objects
- shift from logical relationship between objects to the physical layout
- reuse
  - class libraries
  - development of classes to be reused

◆ **Coding Phase**

- no sharp break between design and coding
- object-oriented modelling tool
- development of the code for the application interface

◆ **Testing Phase**

- testing the code to see if it functions properly.
- testing leads back to analysis and design

15

**Benefits of object-oriented methodology:**

**1. Productivity**

- object-oriented methodology can increase the productivity of the project team and can reduce the project schedule as much as an order of magnitude
- intrinsic power of the object-oriented programming languages
- reuse of classes and objects

**2. Rapid development**

- reusability
- prototyping

**3. Quality**

- absence of defects
- better in object-oriented methodology (reuse, encapsulation)

**4. Maintainability**

- greater in object-oriented methodology (encapsulation) than in conventional architectures based on hierarchy of functions

## Summary

- Different approaches to software development are discussed: traditional Waterfall model, prototyping, and object-oriented.
- The arguments in favour of object-oriented methodology are: productivity, rapid development, quality and maintainability.

17