Meeting the Challenges

Based on Chapter 03
Bennett, McRobb and Farmer

Object Oriented Systems Analysis and Design Using UML

4th Edition, McGraw Hill, 2010

In This Lecture You Will Learn:

- about indicative responses to the challenges discussed in Chapter 2
- about prototyping and incremental life cycles
- the importance of project management
- how users may be involved in a project
- the role of software development tools in systems development

3.2 Responding to the problems

- **Quality Problems**
  - mostly be managed by adopting a systematic approach to systems development.
- **Installation and Operation Problems**
  - may be a consequence of quality problems during the development process.
- **Productivity Problems**
  - typically addressed by using a systematic approach to systems development.

P. 66

Quality Problems

Figure 3.1 Causes of IS project failure and indicative solutions .

Problem	How to minimize risk
The wrong problem is addressed	Strategic Information Systems Planning, Business Modelling, Systematic Approach
Project undertaken for wrong reason	Strategic Information Systems Planning, Business Modelling, Systematic Approach, Prototyping
Wider influences are neglected	Systematic Approach, Requirements, Prototyping
Missing or inappropriate functionality	
Incorrect requirements analysis	Systematic Approach, RUP, AUP, EUP
Users change their minds	Prototyping, User involvement, RUP, AUP
External events change the environment	
Poor interface design	Prototyping, User involvement
Inappropriate data entry	
Software causes inappropriate ways of working	Systematic Approach, User Involvement
Incomprehensible error messages	User involvement
Unhelpful 'help'	
Requirements changed before project delivery	Systematic Approach, RUP, Agile, AUP

續下頁

Installation and Operation Problems

Problem	How to minimize risk
Poor installation	Systematic Approach, Test, Deployment
Unreliability in operation	
Operational Issues	Systematic Approach, Test
Poor response times	
Poor documentation inhibits maintenance	Systematic Approach, Software Development Tools
Local resistance to new information system	User involvement

續下頁

Productivity

Problem	How to minimize risk
Implementation is not feasible	Prototyping
Impossible targets	Systematic Approach, Agile, AUP, DSDM
Time constraints	
Requirements Drift	
Poor project control	Manage IS development, Reuse
Late delivery	
Failure to deliver any system	
Cost overrun	
Developers not familiar with OO	Manage IS development, Training

Problem Solving Model

- Main phases are
  - Data gathering
  - Problem redefinition
    - These focus on understanding what the problem is about
  - Finding ideas
    - Concerned with understanding more about the nature of the problem and possible solutions
  - Finding solutions
  - Implementation

Problem Solving Model

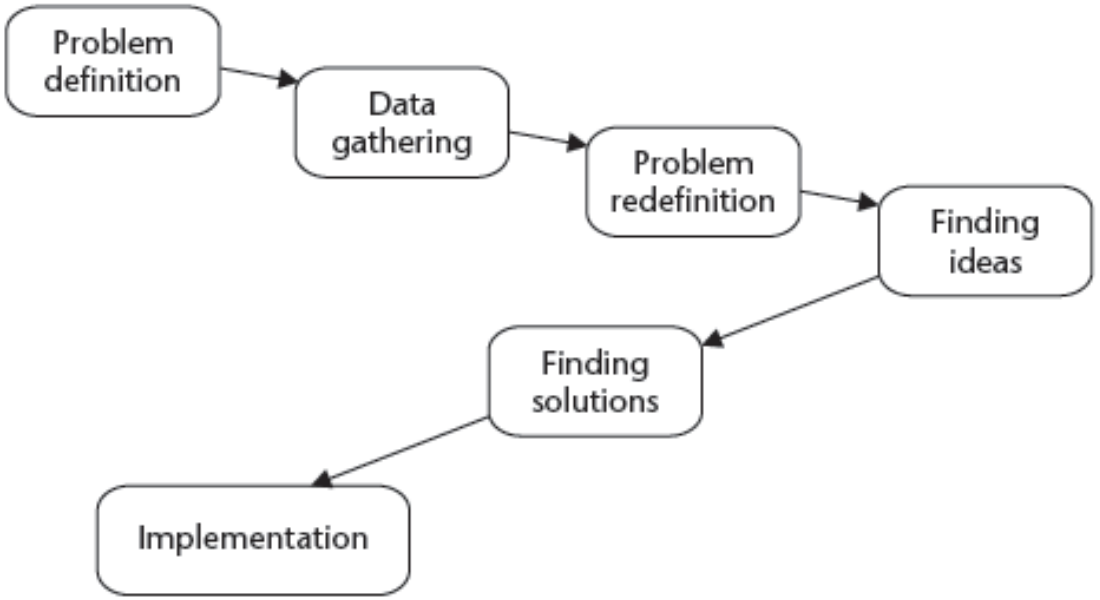

Figure 3.2 General problem-solving model (adapted from Hicks, 1991)

General problem solving model (adapted from Hicks, 1991).

Project Life Cycles

- A distinction should be made between
  - Systems development which incorporates human, software and hardware elements
  - Software development which is primarily concerned with software systems
- Two important phases are
  - Strategic Information Systems Planning
  - Business Modelling

3.3.1 Waterfall Lifecycle Model

- So called because of the difficulty of returning to an earlier phase.
- The model shown here is one of several more or less equivalent alternatives.
  - Typical deliverables are shown for each phase.

P. 71

Waterfall Lifecycle Model

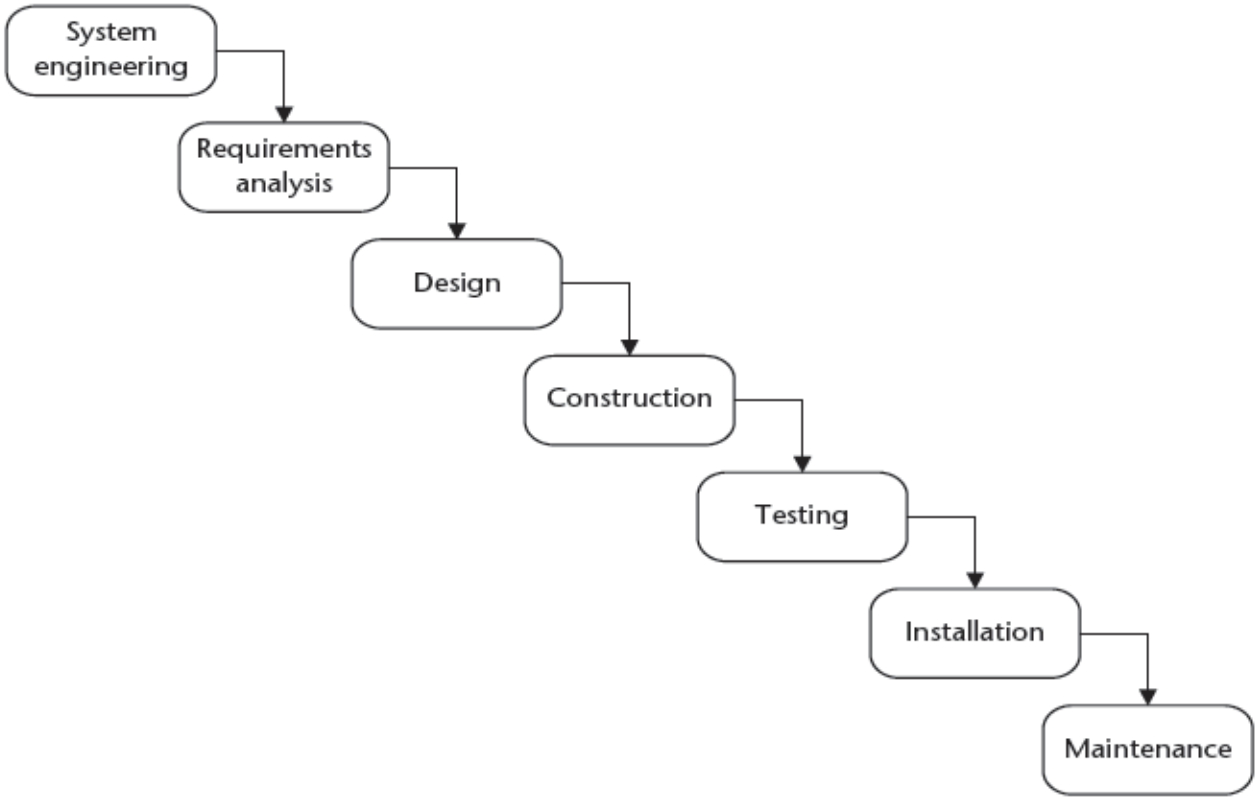

Figure 3.3 Waterfall lifecycle model.

Phase	Output deliverables
System engineering	High-level architectural specification
Requirements analysis	Requirements specification Functional specification Acceptance test specification
Design	Software architecture specification System test specification Design specification Subsystem test specification Unit test specification
Construction	Program code
Testing	Unit test report Subsystem test report System test report Acceptance test report Completed system
Installation	Installed system
Maintenance	Change requests Change request report

Figure 3.4 Lifecyc deliverables(abapted from Sommervulle, 1992).

Waterfall Lifecycle Deliverables

- Systems Engineering
  - High Level Architectural Specification
- Requirements Analysis
  - Requirements Specification
  - Functional Specification
  - Acceptance Test Specifications

Life cycle deliverables (adapted from Sommerville, 1992).

Waterfall Lifecycle Deliverables

- Design
  - Software architecture specification
  - System test specification
  - Design specification
  - Sub-system test specification
  - Unit test specification

Life cycle deliverables (adapted from Sommerville, 1992).

Waterfall Lifecycle Deliverables

- Construction
  - Program code
- Testing
  - Unit test report
  - Sub-system test report
  - System test report
  - Acceptance test report
  - Completed system

Life cycle deliverables (adapted from Sommerville, 1992).

Waterfall Lifecycle Deliverables

- Installation
  - Installed System
- Maintenance
  - Change requests
  - Change request report

Lifecycle deliverables (adapted from Sommerville, 1992).

Problems with Waterfall Lifecycle

- Real projects rarely follow such a simple sequential life cycle.
- Iterations are almost inevitable.
- Lapsed time between systems engineering and the final installation
- Unresponsive to changes during project.

Strengths of Waterfall Lifecycle

- Tasks in phases may be assigned to specialiszed teams.
- Project progress evaluated at the end of each phase.
- Manage projects with high levels of risks.

Prototyping Life Cycle

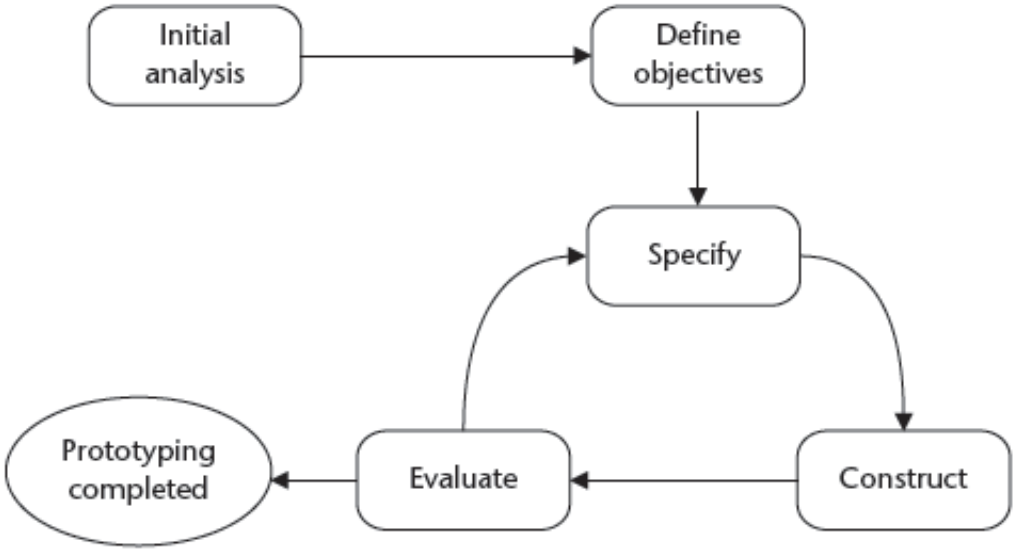

Figure 3.5 A prototyping lifecycle.

Prototyping Advantages:

- early demonstrations of system functionality help identify any misunderstandings between developer and client;
- client requirements that have been missed are identified;
- difficulties in the interface can be identified;
- the feasibility and usefulness of the system can be tested, even though, by its very nature, the prototype is incomplete.

Prototyping – Problems:

- the client may perceive the prototype as part of the final system;
- the prototype may divert attention from functional to solely interface issues;
- prototyping requires significant user involvement;
- managing the prototyping life cycle requires careful decision making.

Spiral Model & Incremental Development

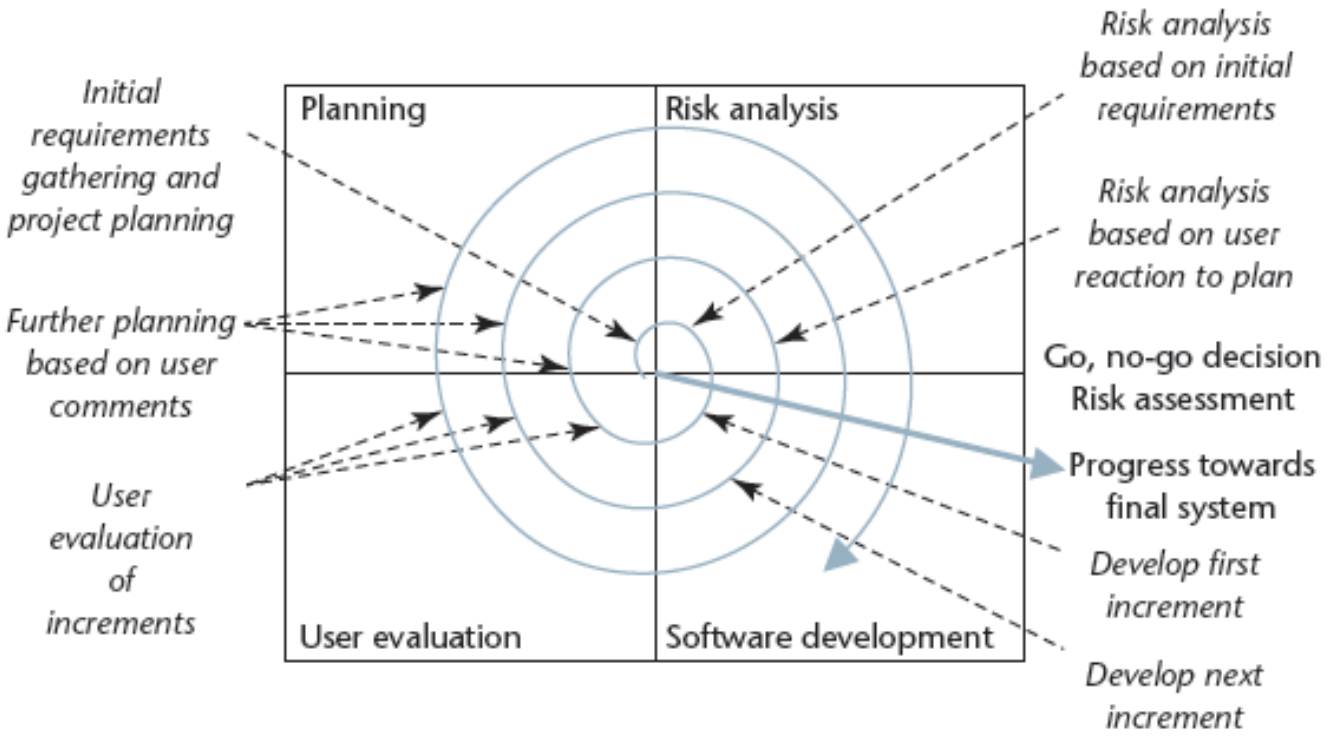

Figure 3.6 Spiral model for incremental delivery (adapted from Boehm, 1988)

Unified Software Development Process

- Captures many elements of best practice
- Main phases are:
  - *Inception* is concerned with determining the scope and purpose of the project;
  - *Elaboration* focuses requirements capture and determining the structure of the system;
  - *Construction's* main aim is to build the software system;
  - *Transition* deals with product installation and rollout.

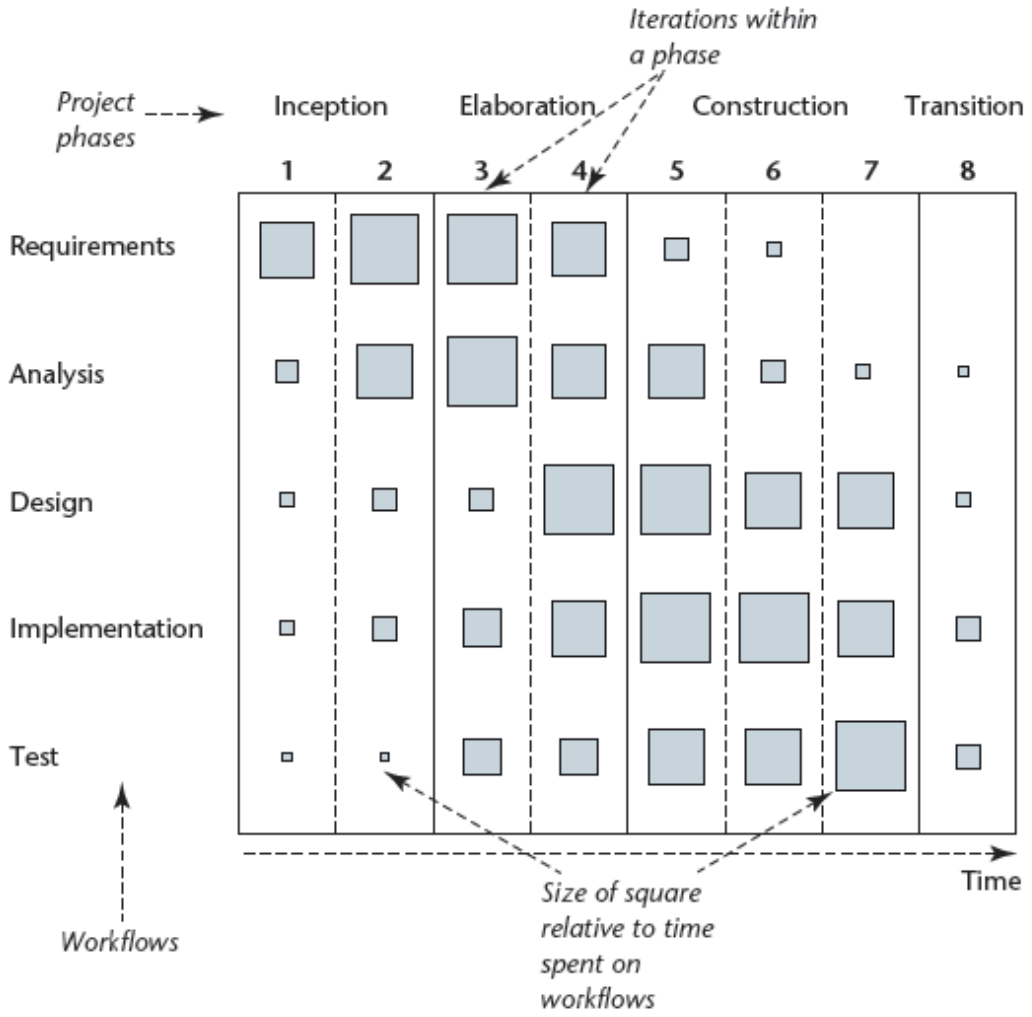

Figure 3.7 The Unified Software Development process (adapted from jacobson et al.,1999).

Figure 3.8 The Manifesto for Agile Software Development .

3.6 User Involvement

- User's can be involved at various levels
  - As part of the development team (DSDM)
  - Via a consultative approach
  - In fact gathering

P. 82

Deliverable/Role	Project Sponsor	Project Manager	Applications Development	Analyst	User Representative	Development Manager
Project Initiation Document	A	R	I	C	C	C
Project Plan	C	A	C	C	C	C
Use Case Model	C	A	I	R	C	C
Priority Requirement List	A	I	I	R	C	C
Software Increments	I	I	R	C	I	A

Key: R – Responsible
A – Accountable
C – Consulted
I – Informed

Figure 3.9 RACI matrix .

Computer Aided Software Engineering

- CASE tools typically provide a range of features including:
  - checks for syntactic correctness;
  - repository support;
  - checks for consistency and completeness;
  - navigation to linked diagrams;
  - layering;
  - traceability;
  - report generation;
  - system simulation;
  - performance analysis;
  - code generation.

References

- Hicks (1991)
- Sommerville (1992, 2007) and Pressman (2009)
- Jacobson, Booch and Rumbaugh (1999)
- Chapters 5 and 22 of Bennett, McRobb and Farmer includes more detail about the Unified Process

(For full bibliographic details, see Bennett, McRobb and Farmer)