Unit 1 Learning Objectives

- In this unit,
  - Unit 1.1
    - Review software development processes & lifecycle
  - Unit 1.2
    - Discuss software engineering challenges
    - Discuss reuse-software development and landscape
    - Appraise the benefits & limitations of reuse
    - Case study for orientation: Failure of Ariane 5
    - Introduces the component-based software lifecycle and contrast it to generic lifecycles

Unit 2.1 Overview of Software Processes
Brainstorming Exercise

- What is your understanding of a “Software Process”?
- Have you used any “Software Process Model” in your practice?
  - Which models?
  - Examples?
  - Uses? Strengths/Weaknesses?
  - Observations?

Software Engineering - for Orientation

- Software Engineering is a branch of systems engineering concerned with the development of large and complex software intensive systems. It focuses on:
  - the real-world goals for, services provided by, and constraints on such systems;
  - the precise specification of systems structure and behaviour, and the implementations of these specifications;
  - the activities required in order to develop an assurance that the specifications and real world-world goals have been met,
  - the evolution of these systems over time, and across systems families,
  - It is also concerned with the processes, methods and tools for the development of software intensive systems in an economic and timely manner.

Reference: A. Finkelstein

Objectives

- Software processes models
- Waterfall, incremental, evolutionary, spiral
  - Advantages and disadvantages
- To describe the Rational Unified Process model
Software Process

- A structured set of activities required to develop a software system
  - Specification;
  - Design;
  - Validation;
  - Evolution.
- A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

Process Models: Examples

- The waterfall model
  - Separate and distinct phases of specification and development.
- Evolutionary development
  - Specification, development and validation are interleaved.
- Component-based software engineering
  - The system is assembled from existing components.

Waterfall Model
**Phase 1. Requirements analysis and definition**

- The process of establishing what services are required and the constraints on the system's operation and development.
  - **What is the system about?**
  - **Requirements engineering process**
    - Feasibility study;
    - Requirements elicitation and analysis;
    - Requirements specification;
    - Requirements validation.

**Phase 1 Requirements Engineering process**

Activities

- Feasibility study
- Requirements elicitation and analysis
- System models
- User and system requirements
- Requirements validation
- Requirements specification
- Requirements document

Output

---

**Phase 2. System and software design**

- i.e., How the requirements to be realised? Design a software structure that realises the specification:
  - Architectural design
  - Abstract specification
  - Interface design
  - Component design
  - Data structure design
  - Algorithm design...
The Software Design Process

Waterfall Model Phases

Phase 3. Implementation and unit testing
- Implementation: Executable code
- Unit testing (Component test)
  » Individual components (function/programs/classes) are tested independently;
  » Components may be functions or objects or coherent groupings of these entities.

Waterfall Model Phases

Phase 4. Integration and system testing
- System testing
  » Testing of the system as a whole. Testing of emergent properties is particularly important.
- Acceptance testing
  » Testing with customer data to check that the system meets the customer’s needs.
Waterfall Model Phases

Phase 5. Operation and maintenance

Evolutionary Development

- **Exploratory development**
  - Objective: to work with customers and to evolve a final system from an initial outline specification.
  - Start with well-understood requirements and add new features as proposed by the customer.

- **Throw-away prototyping**
  - Objective: to understand the system requirements. Should start with poorly understood requirements to clarify what is really needed.
Process Iteration

- System requirements **ALWAYS evolve in the course of a project** so process iteration where earlier stages are reworked is always part of the process for large systems
- Iteration can be applied to any of the generic process models (e.g., waterfall)
- Two (related) approaches
  - Incremental delivery;
  - Spiral development.

Incremental Delivery

- Rather than deliver the system as a single delivery,
  - the development and delivery is broken down into increments with each increment delivering part of the required functionality
- User requirements are prioritised
  - highest priority requirements are included in early increments
- Once the development of an increment is started, the requirements are frozen though requirements for later increments can continue to evolve
Incremental Development Advantages

- Early increments act as a prototype to help elicit requirements for later increments
- Lower risk of overall project failure
- The highest priority system services tend to receive the most testing
- Customer value can be delivered with each increment so system functionality is available earlier

Spiral Development

- Process is represented as a spiral rather than as a sequence of activities with backtracking
- Each loop in the spiral represents a phase in the process
- No fixed phases such as specification or design - loops in the spiral are chosen depending on what is required.
- Risks are explicitly assessed and resolved throughout the process
Spiral Model

Spiral Model Sectors

- Objective setting
  - Specific objectives for the phase are identified.
- Risk assessment and reduction
  - Risks are assessed and activities put in place to reduce the key risks.
- Development and validation
  - A development model for the system is chosen which can be any of the generic models.
- Planning
  - The project is reviewed and the next phase of the spiral is planned.

Exercise - The Rational Unified Process

- Use the Internet to understand RUP. Prepare a brief summary on RUP for class discussion.
RUP Model

**Disciplines**
- Business Modeling
- Analysis & Design
- Implementation
- Test
- Deployment
- Configuration & Change Mgmt
- Project Management
- Environment

**Phases**
- Inception
- Elaboration
- Construction
- Transition

**Iterations**

### RUP - Phases

- **Inception**
  - Establish the business case for the system

- **Elaboration**
  - Develop an understanding of the problem domain and the system architecture

- **Construction**
  - System design, programming and testing

- **Transition**
  - Deploy the system in its operating environment

### RUP - Class Discussion

It is claimed that RUP, if adopted, can:
- Develop software iteratively,
- Manage requirements,
- Support agile software development,
- Verify software quality,
- Control changes to software etc.

**What do you think?**
Do you agree/disagree & Why?
Summary of Unit 2.1

- Software processes are the activities involved in producing and evolving a software system.
- Software process models are abstract representations of these processes.
- General activities are specification, design and implementation, validation and evolution.
- Generic process models describe the organisation of software processes. Examples include the waterfall model, evolutionary development and component-based software engineering.
- Iterative process models describe the software process as a cycle of activities.
- The Rational Unified Process is a generic process model that separates activities from phases.