Software engineering - systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software – cost-effective, timely, predictable, high-quality

Software process - structured set of activities required to develop a software system.

Following process activities

- Specification – defining what the system should do;
- Design and implementation – defining the organization of the system and implementing the system;
- Validation – checking that it does what the customer wants;
- Evolution – changing the system in response to changing customer needs.

Types of processes

- Plan-driven: planned in advance and progress is measured against this plan
- Agile: planning is incremental
- Most approaches nowadays combine both methods

Why define processes?

- Efficiency – helps to keep focus and structure
- Consistency - results likely to be similar
- Basis for Improvement - gathering data on your work -> room for improvement

0.1 Process Activities

Software specification (or requirements engineering)

- Requirements elicitation and analysis - What do the system stakeholders require or expect from the system?
- Requirements specification - Defining the requirements in detail
- Requirements validation - Checking the validity of the requirements

Software design and implementation Design - design a software structure that realises the specification

- Architectural design – overall structure
- Database design – system data structure
- Interface design – between system components
- Component (or detailed) design – design of each component individually

Implementation - translate the design into an exec. prog.

Software verification and validation (testing) the system conforms to its specification (verification) meets the requirements and customer needs (validation).

Testing

- Unit (or component) testing – individual component testing
- Integration testing – testing of interaction between components
- System testing – general testing (performance, usability, etc. . .)
- Acceptance testing – live testing with data

Software evolution (or maintenance) – after development

- Corrective – bug fixing
- Adaptive – adapt to new platforms, technologies
- Perfective – new functionalities

0.2 Software process models

Waterfall model (plan-driven) – Separate specification and development

- Inflexible partitioning of the project – hard to respond to changing requirements
- Used for large systems engineering projects where a system is developed at several sites – plan-driven aspect helps with coordination

Incremental development (& delivery) (agile or plan-driven) - Specification, development and validation are interleaved.

- Easier to adapt to changing requirements
- More feedback – reduced risk of failure

- Can be delivered staggered
- Needs constant refactoring (due to the multiple increments)
- Suboptimal reusability

Integration and configuration (agile or plan-driven) - The system is assembled from existing configurable components.

- Reduced costs and risks – less software developed from scratch
- Faster system delivery
- Needs requirement compromises to fit existing components
- Loss of control over the evolution of the used components

Software prototyping - Not actually a model but an approach to cope with uncertainty

- A prototype is an initial version of a system used to demonstrate concepts and try out design options – reduced uncertainty

0.3 RUP - Rational Unified Process

0.3.1 Best Practices

- Develop iteratevely
- Manage requirements
- Use component architectures
- Model visually (UML)
- Continuously verify quality
- Manage change

0.3.2 Phases

Each phase has several iterations, that walk through all disciplines.

- Inception
  Define the project scope (understand the problem)
- Elaboration
  Define the solution architecture (understand the solution)
- Construction
  Build the product
- Transition
  Transition the product into the end-users

0.3.3 Disciplines

- Business modeling
- Requirements
- Analysis & design
- Implementation
- Test
- Deployment
- Configuration & change management
- Project management
- Environment

0.3.4 Basic elements

- Role
- Activity
- Artifact
- Workflow & Workflow Detail