Overview

- Prescriptive process models prescribe a distinct set of activities, actions, tasks, milestones, and work products required to engineer high quality software.
- Prescriptive software process models are adapted to meet the needs of software engineers and managers for a specific project.
- Prescriptive software models provide stability, control, and organization to a process that if not managed can easily get out of control.
- Framework activities for a particular process model may be organized into a process flow that may be linear, incremental, or evolutionary.
- The software engineer's work products (programs, documentation, data) are produced as a consequence of the activities defined by the software process.
- The best indicators of how well a software process has worked are the quality, timeliness, and long-term viability of the resulting software product.

Prescriptive Process Models

- Originally proposed to bring order to the chaos of software development
- They brought order to software engineering work and provide reasonable guidance to software teams
- Yet, they have not provided a definitive answer to the problems of software development in an ever changing computing environment

Software Processes

- Every software engineering organization needs to describe a set of framework activities for the processes it adopts
- Each framework activity needs to be populated with software engineering actions
- Each engineering action needs to be defined in terms of a task set that defines the work and work products needed to meet the development goals
- The resulting process model should be adapted to accommodate the nature of the specific project, people doing the work, and the work environment
- Regardless of the process model selected, software engineers will chose a generic process framework that includes these activities: communication, planning, modeling, construction, and deployment
- Each process model will prescribe a set of process elements (framework activities, software engineering actions, tasks, work products, quality assurance, and change control mechanisms) and a workflow (the manner in which the process elements are interrelated)
- All software process models discussed in this chapter can accommodate the generic framework activities described previously

Software Process Models

- Waterfall Model (old fashioned but reasonable approach when requirements are well understood)

Incremental Models

- Incremental Model (delivers software in small but usable pieces, each piece builds on pieces already delivered)
- Rapid Application and Development (RAD) Model (makes heavy use of reusable software components with an extremely short development cycle)
Evolutionary Process Models

- Prototyping Model (good first step when customer has a legitimate need, but is clueless about the details, developer needs to resist pressure to extend a rough prototype into a production product)
- Spiral Model (couples iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model)
- Concurrent Development Model (similar to spiral model often used in development of client/server applications)

Specialized Process Models

- Component-Based Development (spiral model variation in which applications are built from prepackaged software components called classes)
- Formal Methods Model (rigorous mathematical notation used to specify, design, and verify computer-based systems)
- Aspect-Oriented Programming (provides a process for defining, specifying, designing, and constructing software aspects like user interfaces, security, and memory management that impact many parts of the system being developed)

Unified Process

- Use-case driven, architecture centric, iterative, and incremental software process
- Attempts to draw on best features of traditional software process models and implements many features of agile software development
- Phases
  - Inception phase (customer communication and planning)
  - Elaboration phase (communication and modeling)
  - Construction phase
  - Transition phase (customer delivery and feedback)
  - Production phase (software monitoring and support)

Unified Process Work Products

- Inception phase
  - Vision document
  - Initial use-case model
  - Initial project glossary
  - Initial business case
  - Initial risk assessment
  - Project plan (phases and iterations)
  - Business model
  - Prototypes
- Elaboration phase
  - Use-case model
  - Functional and non-functional requirements
  - Analysis model
  - Software architecture description
  - Executable architectural prototype
  - Preliminary design model
  - Revise risk list
  - Project plan (iteration plan, workflow, milestones)
  - Preliminary user manual
- Construction phase
  - Design model
  - Software components
  - Integrated software increment
  - Test plan
  - Test cases
  - Support documentation (user, installation, increment)
• Transition phase
  o Delivered software increment
  o Beta test reports
  o User feedback

Unified Process Workflows (not covered in SEPA, 6/e but provided for information purposes)

• Requirements workflow
  o Goal is to answer 4 questions
    1. What are the relevant characteristics of customer domain?
    1. What will the software product accomplish?
    1. What features and functions will be used to accomplish it?
    1. What constraints will be placed on the system?
  o Requirements are determined iteratively during inception and elaboration phases
  o Software team must learn enough to establish project scope, set an iterative and incremental project plan, and develop use-cases recognizable to end-users

• Analysis workflow
  o Begins during inception phase and culminates in the construction phase
  o Goal is to perform architectural analysis and produce the work products required by the analysis model
  o The analysis model uses abstractions that are recognizable to the customer or end-user

• Design workflow
  o Begins in last part of elaboration phase and continues to first stages of construction phase
  o Objective of design is to transform analysis model (the what) into a design model (the how - an implementation model of the software)
  o Modeling moves from the problem (customer) domain into solution (engineer) domain

• Implementation workflow
  o Work begins in elaboration phase and dominates construction phase
  o Incorporates the software engineering tasks required to translate design classes into executable software components, testing and integrating these components, and deliver those required for the planned increment
  o Build/reuse decisions made during this phase

• Testing workflow
  o During elaboration the intent of testing work flow is to exercise the executable architecture and demonstrate its viability
  o During construction testing workflow focuses on testing software components as they are integrated in the build and testing integrated software before releasing to customer
  o During transition testing the customers or end-users assume the responsibility for acceptance tests

• Project management workflow
  o Spans all software increments
  o Focus is on planning, risk management, project tracking and control
  o Phase plan provides a rough estimate of the effort required to accomplish workflow across each UP phase, the major milestones for each phase and increment, and the number of increments required
  o Iteration plan defines the tasks associated with analysis, design, implementation, testing, and deployment for one software increment