SOFTWARE ENGINEERING

Week 5

Software Project Management – 1

Prof. Dr. Muhittin GÖKMEN   Yard. Doç. Dr. A. Cüneyd TANTUĞ   Araş. Gör. Dr. Tolga OVATMAN
Istanbul Technical University
Computer Engineering Department

Agenda

1. Risk Management
2. Quality Management
3. Change Management

1. Risk Management
2. Quality Management
3. Change Management

Risk Management

5.1

Risk Management

- Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.
- A risk is a probability that some adverse circumstance will occur
  - Project risks affect schedule or resources;
  - Product risks affect the quality or performance of the software being developed;
  - Business risks affect the organisation developing or procuring the software.

Risk Management Styles

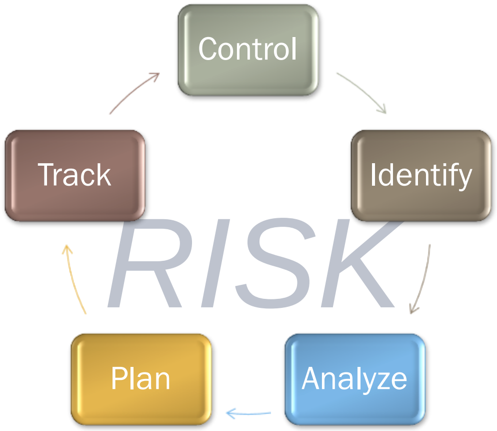
Reactive Risk Management

- project team reacts to risks when they occur
- mitigation—plan for additional resources in anticipation of fire fighting
- fix on failure—resource are found and applied when the risk strikes
- crisis management—failure does not respond to applied resources and project is in jeopardy

Proactive Risk Management

- formal risk analysis is performed
- organization corrects the root causes of risk
  - TQM concepts and statistical SQA
  - examining risk sources that lie beyond the bounds of the software
  - developing the skill to manage change

Risk Management Paradigm



Risk Estimation

- *Risk projection*, also called *risk estimation,* attempts to rate each risk in two ways
  - the likelihood or probability that the risk is real
  - the consequences of the problems associated with the risk, should it occur.
- The are four risk projection steps:
  - establish a scale that reflects the perceived likelihood of a risk
  - delineate the consequences of the risk
  - estimate the impact of the risk on the project and the product,
  - note the overall accuracy of the risk projection so that there will be no misunderstandings.

Risk Table

Risks	Category	Probability %	Impact	Remedy Plan
Size estimate may be significantly low	PS	60	2	
Larger number of users than planned	PS	30	3	
Less reuse than planned	PS	70	2	
End users resist system	BU	40	3	
Delivery deadline will be tightened	BU	50	2	
Funding will be lost	CU	40	1	
Customer will change requirements	PS	80	2	
Technology will not meet expectations	TE	30	1	
Lack of training on tools	TE	60	3	
Staff inexperienced	ST	30	2	
Staff turnover will be high	ST	60	2	
…….				

- Sort the table by probability and impact
- Cut-off low probability risks

Examples of Different Risk Types

Risk type	Possible risks
Technology	The database used in the system cannot process as many transactions per second as expected. (1) Reusable software components contain defects that mean they cannot be reused as planned. (2)
People	It is impossible to recruit staff with the skills required. (3) Key staff are ill and unavailable at critical times. (4) Required training for staff is not available. (5)
Organizational	The organization is restructured so that different management are responsible for the project. (6) Organizational financial problems force reductions in the project budget. (7)
Tools	The code generated by software code generation tools is inefficient. (8) Software tools cannot work together in an integrated way. (9)
Requirements	Changes to requirements that require major design rework are proposed. (10) Customers fail to understand the impact of requirements changes. (11)
Estimation	The time required to develop the software is underestimated. (12) The rate of defect repair is underestimated. (13) The size of the software is underestimated. (14)

Risk Impact

- The overall *risk exposure,* RE, is determined using the following relationship

$$RE = P \times C$$

where

- P is the probability of occurrence for a risk
- C is the cost to the project should the risk occur

Risk Impact Example

- **Risk identification.** Only 70 percent of the software components scheduled for reuse will, in fact, be integrated into the application. The remaining functionality will have to be custom developed.
- **Risk probability.** 80% (likely).
- **Risk impact.** 60 reusable software components were planned. If only 70 percent can be used, 18 components would have to be developed from scratch (in addition to other custom software that has been scheduled for development). Since the average component is 100 LOC and local data indicate that the software engineering cost for each LOC is $14.00, the overall cost (impact) to develop the components would be 18 x 100 x 14 = $25,200.
- **Risk exposure.** RE = 0.80 x 25,200 ~ $20,200.

Risk Mitigation, Monitoring, Management

- Risk Mitigation,Monitoring,Management (RMMM)
- Mitigation : how can we avoid the risk?
- Monitoring : what factors can we track that will enable us to determine if the risk is becoming more or less likely?
- Management :what contingency plans do we have if the risk becomes a reality?

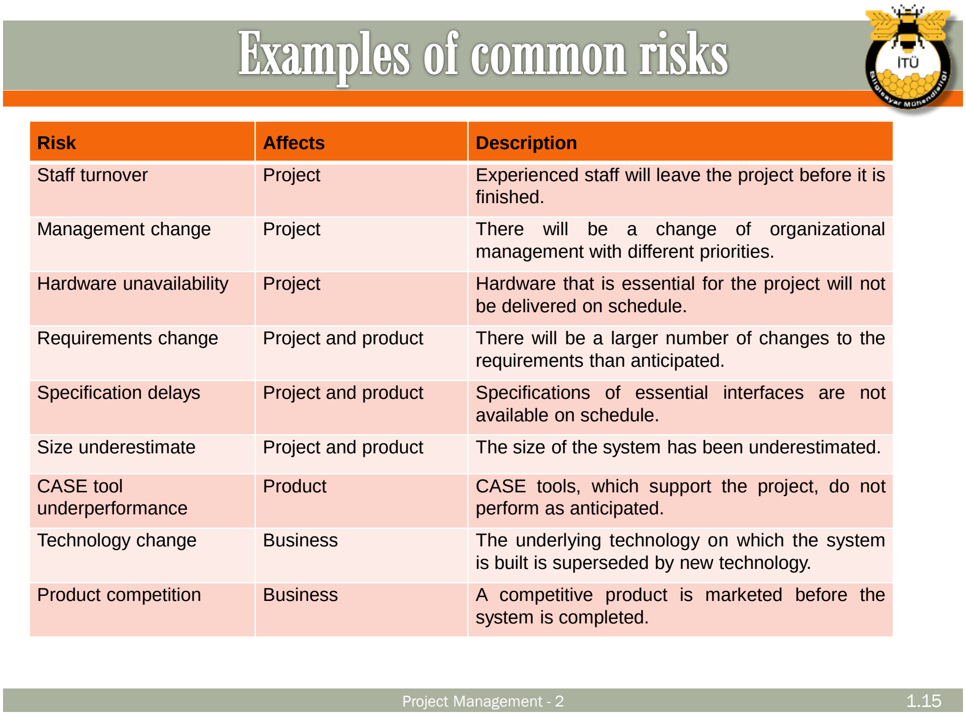
Risk Planning

- Consider each risk and develop a strategy to manage that risk.
- Avoidance strategies
  - The probability that the risk will arise is reduced;
- Minimisation strategies
  - The impact of the risk on the project or product will be reduced;
- Contingency plans
  - If the risk arises, contingency plans are plans to deal with that risk;

Risk Monitoring / Risk Indicators

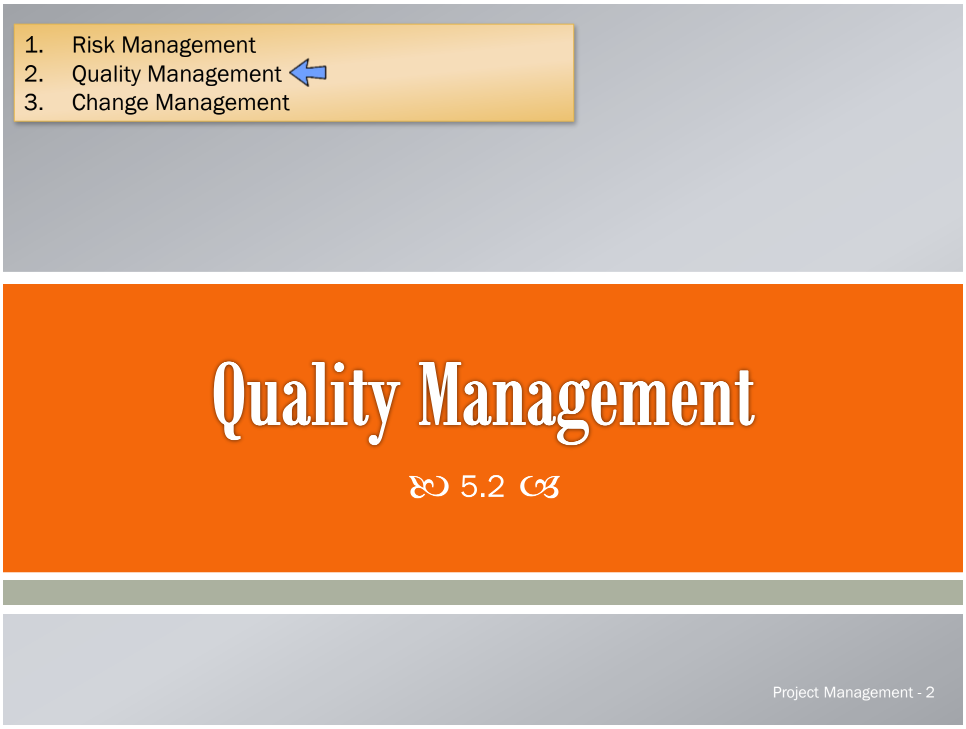
- Assess each identified risks regularly to decide whether or not it is becoming less or more probable.
- Also assess whether the effects of the risk have changed.
- Each key risk should be discussed at management progress meetings.

Risk type	Potential indicators
Technology	Late delivery of hardware or support software; many reported technology problems.
People	Poor staff morale; poor relationships amongst team members; high staff turnover.
Organizational	Organizational gossip; lack of action by senior management.
Tools	Reluctance by team members to use tools; complaints about CASE tools; demands for higher-powered workstations.
Requirements	Many requirements change requests; customer complaints.
Estimation	Failure to meet agreed schedule; failure to clear reported defects.



Examples of common risks

Risk	Affects	Description
Staff turnover	Project	Experienced staff will leave the project before it is finished.
Management change	Project	There will be a change of organizational management with different priorities.
Hardware unavailability	Project	Hardware that is essential for the project will not be delivered on schedule.
Requirements change	Project and product	There will be a larger number of changes to the requirements than anticipated.
Specification delays	Project and product	Specifications of essential interfaces are not available on schedule.
Size underestimate	Project and product	The size of the system has been underestimated.
CASE tool underperformance	Product	CASE tools, which support the project, do not perform as anticipated.
Technology change	Business	The underlying technology on which the system is built is superseded by new technology.
Product competition	Business	A competitive product is marketed before the system is completed.



1. Risk Management
2. Quality Management ⇦
3. Change Management

Quality Management

℘ 5.2 ℘

Software Quality

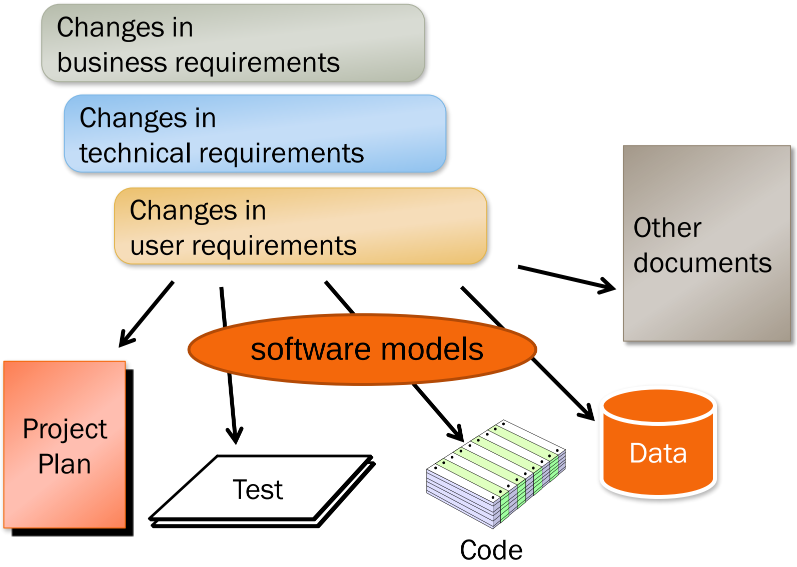
1. Risk Management
2. Quality Management
3. Change Management ⇦

Change Management

5.3

Change Management

What are changes?



Changes in business requirements

Changes in technical requirements

Changes in user requirements

software models

Project Plan

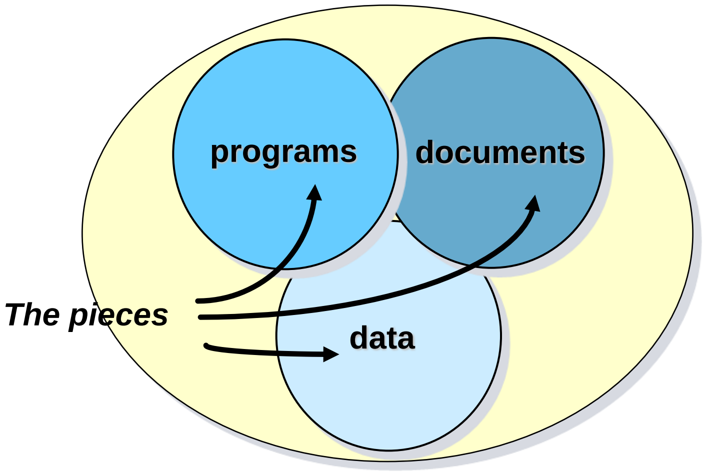
Test

Code

Data

Other documents

The Software Configuration



SCM Questions

- How do we manage requests for changes in software?
- What and where are the software components?
- What is the status of each software component?
- How does a change to one component affect others?
- How do we resolve conflicting to changes?
- How do we maintain multiple versions?
- How do we keep the system up to date?

SCM Activities

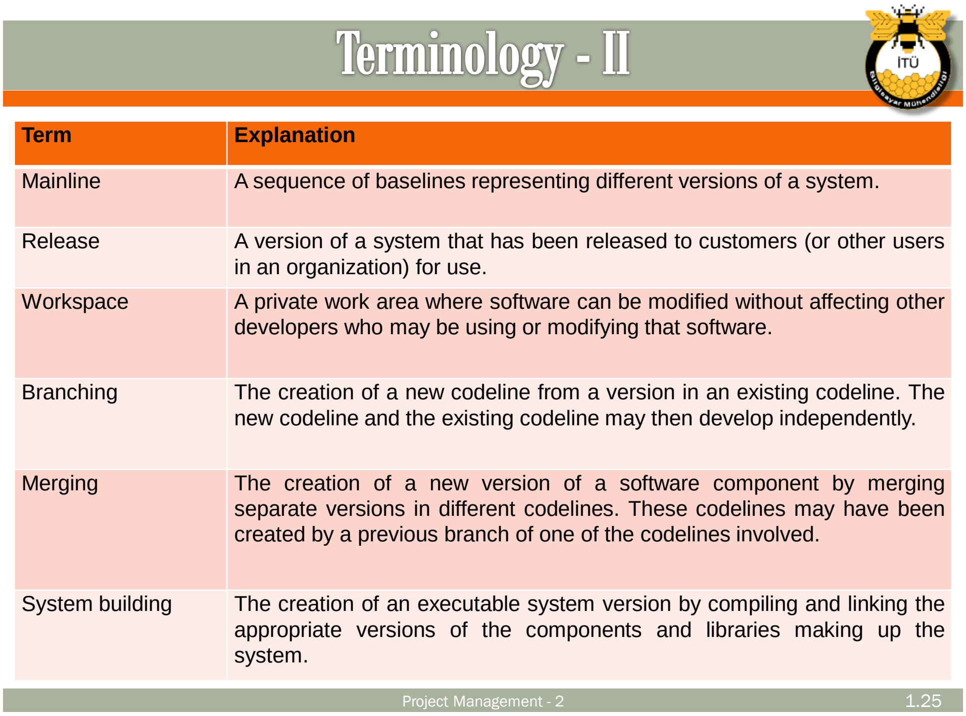
- Configuration item identification
  - is the modeling of the system as a set of evolving components
- Promotion management
  - is the creation of versions for other developers
- Release management
  - is the creation of versions for the clients and users
- Branch management
  - is the management of concurrent development
- Variant management
  - is the management of versions intended to coexist
- Change management
  - is the handling, approval and tracking of change requests

Project Management - 2 1.23

Terminology

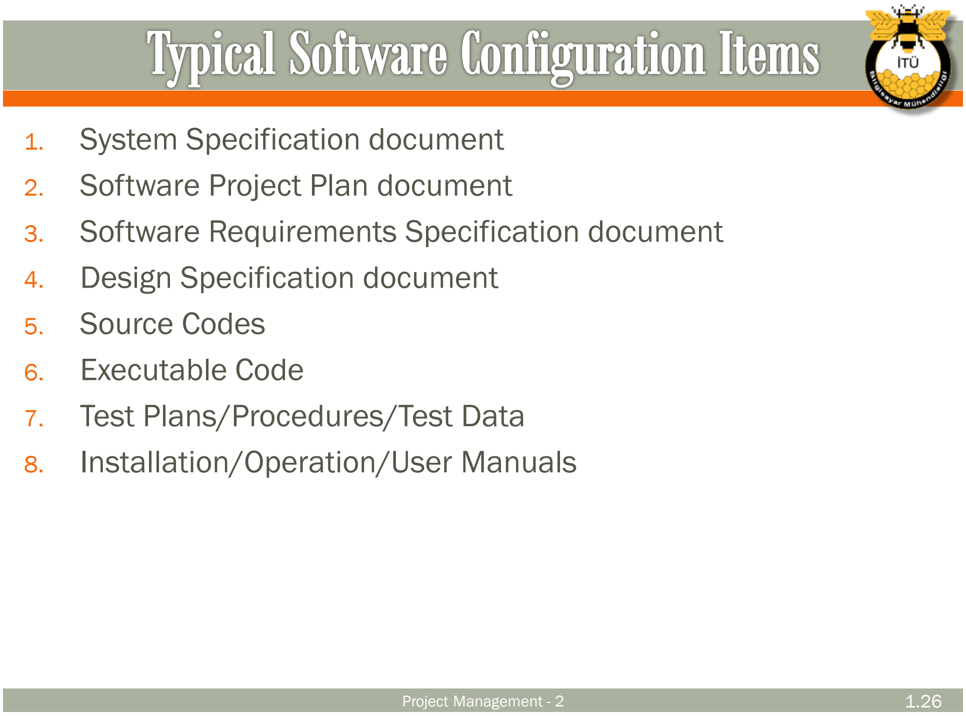
Term	Explanation
Configuration item or software configuration item (SCI)	Anything associated with a software project (design, code, test data, document, etc.) that has been placed under configuration control. There are often different versions of a configuration item. Configuration items have a unique name.
Configuration control	The process of ensuring that versions of systems and components are recorded and maintained so that changes are managed and all versions of components are identified and stored for the lifetime of the system.
Version	An instance of a configuration item that differs, in some way, from other instances of that item. Versions always have a unique identifier, which is often composed of the configuration item name plus a version number.
Baseline	A baseline is a collection of component versions that make up a system. Baselines are controlled, which means that the versions of the components making up the system cannot be changed. This means that it should always be possible to recreate a baseline from its constituent components.
Codeline	A codeline is a set of versions of a software component and other configuration items on which that component depends.

Project Management - 2 1.24



Terminology - II

Term	Explanation
Mainline	A sequence of baselines representing different versions of a system.
Release	A version of a system that has been released to customers (or other users in an organization) for use.
Workspace	A private work area where software can be modified without affecting other developers who may be using or modifying that software.
Branching	The creation of a new codeline from a version in an existing codeline. The new codeline and the existing codeline may then develop independently.
Merging	The creation of a new version of a software component by merging separate versions in different codelines. These codelines may have been created by a previous branch of one of the codelines involved.
System building	The creation of an executable system version by compiling and linking the appropriate versions of the components and libraries making up the system.



Typical Software Configuration Items

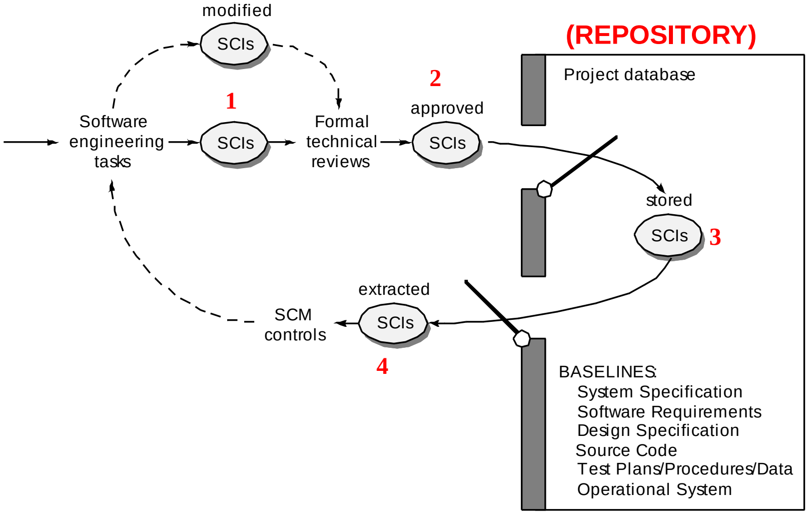
1. System Specification document
2. Software Project Plan document
3. Software Requirements Specification document
4. Design Specification document
5. Source Codes
6. Executable Code
7. Test Plans/Procedures/Test Data
8. Installation/Operation/User Manuals

Baselines

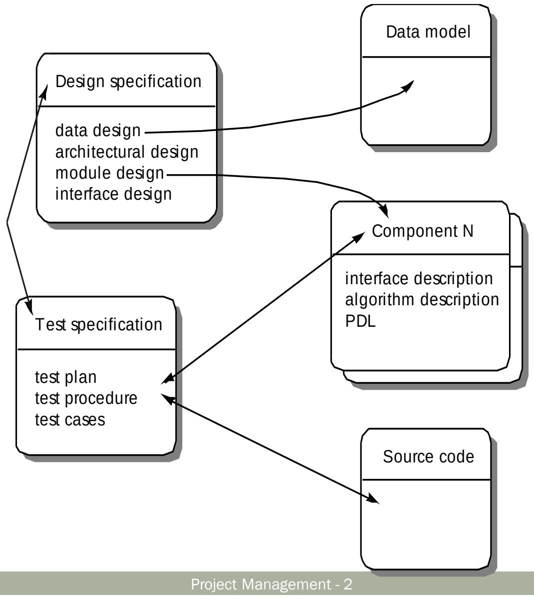
- The IEEE defines a baseline as:
  - *A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures.*

- A baseline is a milestone in the development of software that is marked by the delivery of one or more software configuration items and the approval of these SCIs that is obtained through a formal technical review

Mechanism of SCM

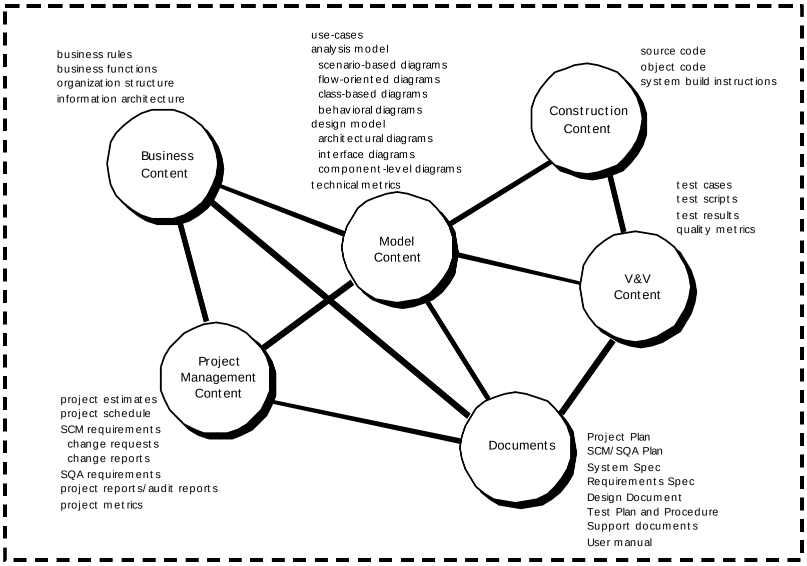


Software Configuration Objects



- The SCM repository is the set of mechanisms and data structures that allow a software team to manage change in an effective manner
- The repository performs or precipitates the following functions
  - Data integrity
  - Information sharing
  - Tool integration
  - Data integration
  - Methodology enforcement
  - Document standardization

Repository Content



Business Content

business rules
business functions
organization structure
information architecture

Model Content

use-cases
analysis model
scenario-based diagrams
flow-oriented diagrams
class-based diagrams
behavioral diagrams
design model
architectural diagrams
interface diagrams
component-level diagrams
technical metrics

Construction Content

source code
object code
system build instructions

V&V Content

test cases
test scripts
test results
quality metrics

Project Management Content

project estimates
project schedule
SCM requirements
change requests
change reports
SQA requirements
project reports/audit reports
project metrics

Documents

Project Plan
SCM/SQA Plan
System Spec
Requirements Spec
Design Document
Test Plan and Procedure
Support documents
User manual

Repository Features

- Versioning
  - Saves all of these versions to enable effective management of product releases and to permit developers to go back to previous versions
- Dependency tracking and change management
  - The repository manages a wide variety of relationships among the data elements stored in it.
- Requirements tracing
  - Provides the ability to track all the design and construction components and deliverables that result from a specific requirement specification
- Configuration management
  - Keeps track of a series of configurations representing specific project milestones or production releases. Version management provides the needed versions, and link management keeps track of interdependencies.
- Audit trails
  - Establishes additional information about when, why, and by whom changes are made.

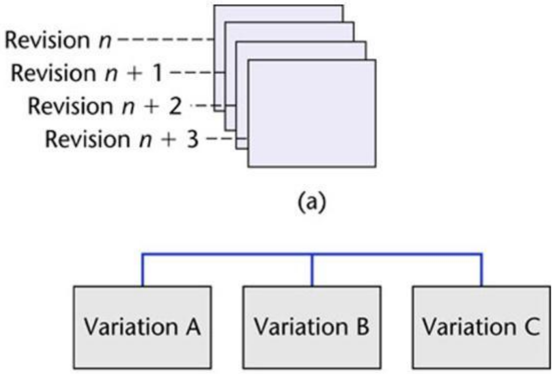
Versioning

- During maintenance, at all times there are at least two versions of the product:the old version, and the new version

- There are two types of versions: *revisions* and *variations*
- Revisions:
  - A version to fix a fault in the artifact
  - We cannot throw away an incorrect version
    - The new version may be no better
    - Some sites may not install the new version
  - Perfective and adaptive maintenance also result in revisions

Variations

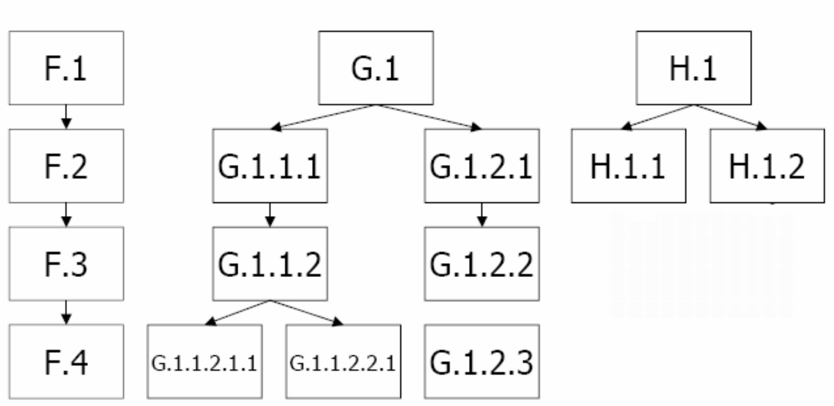
- Variations:
  - A variation is a version for a different operating system
  - Variations are designed to coexist in parallel



(a)

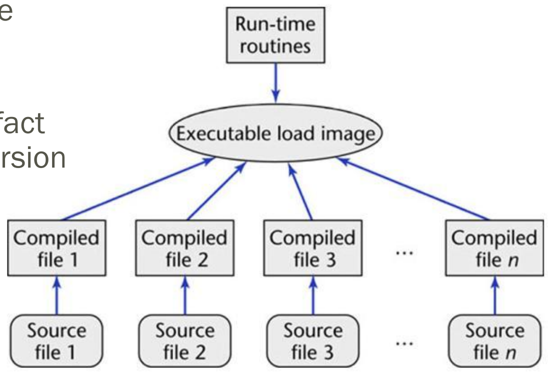
(b)

Baseline Versioning



Configuration Control

- Every code artifact exists in three forms
  - Source code
  - Object code
  - Executable load image
- Configuration
  - A version of each artifact from which a given version of a product is built



Configuration-Control Tools

- UNIX tools
  - SCCS (Source Code Control System)
  - RCS (Revision Control System)
  - CVS (Concurrent Versions System)

- Other commercial tools
  - SourceSafe (Microsoft)
  - ClearCase (IBM Rational Software)

- Reduces the effort required to manage a change: (improved productivity)

- Leads to better software integrity and security: (increased quality)

- Generates information about the process: (enhanced control)

- Maintains a software development database (repository): (better record keeping)

CASE Tools

- Computer-Aided Software Engineering (CASE) tools can assist software engineers with every activity associated with the software development process.
  - Automating management activities.
  - Assisting engineers with analysis, design, coding, and testing work.
  - Complementing solid software engineering practices and leading to improved software quality.

CASE Tools - II

Components of CASE Tools

- Diagram tools
- Model analysis tools
- Repository
- Data dictionary
- Automatic code generation tools
- Form / Report definition and generation tools
- Import/export utilities

Example Tools

- IBM Rational Rose (ROSE = Rational Object Oriented Software Engineering)
- ArgoUML (Free - Open Source)
- Visual Paradigm for UML
- Oracle Designer
- PowerBuilder
- Microsoft Project
- Microsoft Visio
- SmartDraw