




# SOFTWARE ENGINEERING

## Week 4

### Software Project Management

Yard. Doç. Dr. A. Cüneyd TANTUĞ      Yard. Doç. Dr. Tolga OVATMAN  
Istanbul Technical University  
Computer Engineering Department

## Agenda



1. Project Management Concepts
2. Project Planning
3. Estimation
4. Process Metrics
5. Quality Management

Project Management - 1      2


1. Project Management Concepts
2. Estimation
3. Planning ←
4. Management


# Planning

4.3

Project Management - 1

## Project Planning



- Planning techniques, includes the following activities
  - Identify and organize resources (e.g. team, hardware) required for the project
  - Develop a work breakdown structure (WBS) and identify packages for the tasks in the work breakdown structure(WBS)
  - Define a schedule of objectively measurable milestones organized as a GANTT chart
  - Prepare a schedule network and identify the critical path(s)
- Planning activities should be continuously held during the project as a rolling wave
 

Project Management - 1      1.4


3. Planning
  - 3.1. Team Planning ←
  - 3.2. Task Planning
  - 3.3. Time Planning

# Team Planning

4.3.1

Project Management - 1

## Software Teams



- The following factors must be considered when selecting a software project team structure
  - the difficulty of the problem to be solved
  - the size of the resultant program(s) in lines of code or function points
  - the time that the team will stay together (team lifetime)
  - the degree to which the problem can be modularized
  - the required quality and reliability of the system to be built
  - the rigidity of the delivery date
  - the degree of sociability (communication) required for the project

Project Management - 1      1.6

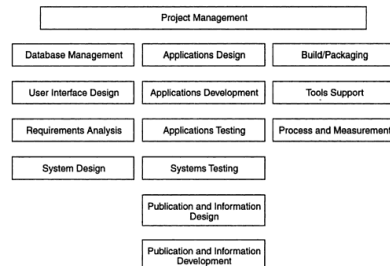
## Team Organizations

- ✎ There exist two main types of organizational structures:
  - Hierarchical organization: All the people associated with a project are grouped into functional departments that report directly within the vertical line of command.
  - Matrix organization: People are grouped based on the functions they perform.
- ✎ Also based on the location of the project personnel, teams may be:
  - Centralized
  - Distributed
    - Globally distributed
  - Mixed
- ✎ Small software engineering groups are usually organised informally without a rigid structure.
- ✎ For large projects, there may be a hierarchical structure where different groups are responsible for different sub-projects.

Project Management - 1

1.7

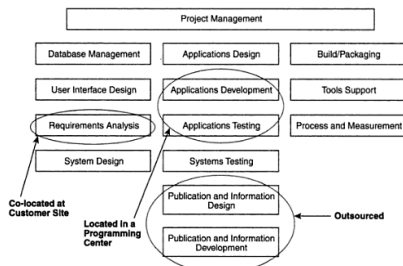
## Team Organizations



Project Management - 1

1.8

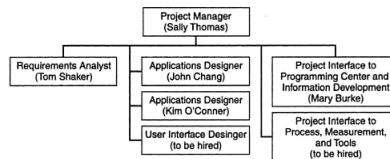
## An Example Team Organization



Project Management - 1

1.9

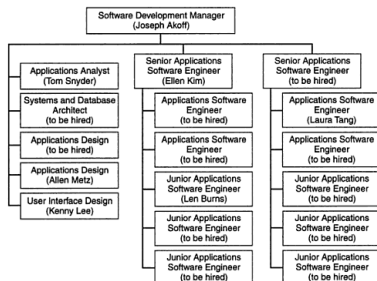
## An Example Team Organization



Project Management - 1

1.10

## Team Organizations



Project Management - 1

1.11

## Agile Teams

Agile teams have the following characteristics

- ✎ Co-location
- ✎ Engaged Customers
- ✎ Self-Organizing
- ✎ Accountable and Empowered
- ✎ Cross-Functional

Tips for Forming Your Agile Team

- ✎ Look for Generalists
- ✎ People Who Are Comfortable with Ambiguity
- ✎ Team Players Who Can Check Their Egos at the Door

Project Management - 1

1.12

- 3. Planning
- 3.1. Team Planning
- 3.2. Task Planning ←
- 3.3. Time Planning

## Task Planning

≈ 4.2 ≈

Project Management - 1

## Task Planning

- ⇒ In order to work your way down in decomposition towards atomic tasks, you need to come up with an architectural decomposition.
- ⇒ Architectural design of software is concerned with specifying the software modules, their interrelationships, and their connections to the environment of the software.
- ⇒ We will examine Software Architectures in detail later. But let's work on a trivial ATM software example to illustrate how we obtain WBS.

Project Management - 1

1.14

## Task Planning

Some prioritized requirements for ATM software

- ⇒ Essential requirements
  - E1 Financial transactions shall be authorized by an ATM card and a password
  - E2 Financial transactions shall be terminated if a customer fails to enter the correct password « settable » times
  - E3 Financial transaction shall allow quick cash withdrawals
  - E4 Financial transaction shall provide a printed receipt for each transaction
  - E5 The ATM shall retain the information listed in the requirements specification, for each customer transaction
  - E6 Financial transaction shall process Terminate requests from customers

Project Management - 1

1.15

## Task Planning

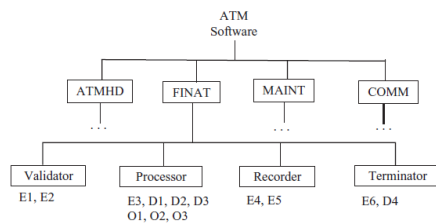
Some prioritized requirements for ATM software

- ⇒ Desirable requirements
  - D1 Financial transaction should accommodate balance inquiry transactions
  - D2 Financial transaction should accommodate standard withdrawal transactions
  - D3 Financial transaction should accommodate deposit transactions
  - D4 Customers should be allowed to conduct multiple transactions per session
- ⇒ Optional requirements
  - O1 Financial transaction could support debit card transactions
  - O2 Financial transaction could support payment of utility bills
  - O3 Financial transaction could allow customers to purchase postage stamps which will be disbursed by the ATM hardware

Project Management - 1

1.16

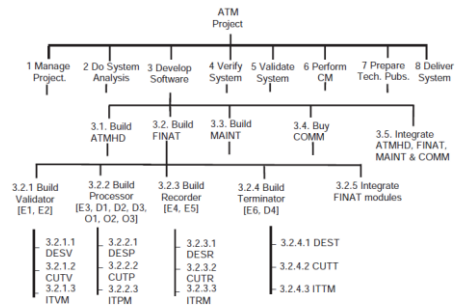
## Task Planning



Project Management - 1

1.17

## Task Planning - WBS



Project Management - 1

1.18

## Task Planning — Task Extraction

- Following WBS construction, tasks should be extracted to build a time plan and assign to resources(team). Tasks can also be assigned story points and can be used in estimation and tracking for agility.

TABLE 5.3B A work package example

Task identifier:	3.2.2.1 Design transaction processor
Task description:	Specify internal architecture of the transaction processor module
Estimated duration:	2 weeks
Resources needed:	2 senior telecom designers
Personnel:	Designers must know UML
Skills:	One workstation running Rapsody
Tools:	Three day design review in San Diego for 2 people
Travel:	3.2.1 Develop system architecture
Predecessor tasks:	3.3.2.2 Implement transaction processor
Successor tasks:	Architectural specification for transaction processor and test plan
Work products:	Architectural specification for transaction processor and test plan
Baselines created:	Designers not identified
Risk factors:	Successful design inspection by peers and approval of transaction processor design by the software architect
Acceptance criteria:	

Project Management - 1

1.19

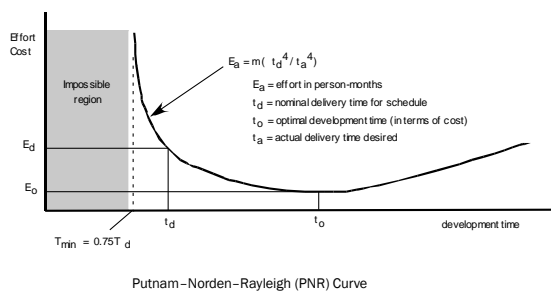
- 3. Planning
  - 3.1. Team Planning
  - 3.2. Task Planning
  - 3.3. Time Planning

## Time Planning

4.2

Project Management - 1

## Effort and Delivery Time



Project Management - 1

1.21

## For Scheduling

- **PERT Chart:**
  - Shows the relationships based on tasks or activities
  - Defines tasks that can be done concurrently or not and critical path
- **Gantt Chart:**
  - Shows schedule (timeline) information for each task as a bar chart

Project Management - 1

1.22

## PERT Chart

- Project Evaluation and Review Technique
- Also known as Task Network Diagram
- **Nodes:** activities / tasks and estimated duration
  - **Edges:** dependencies
- Critical path:** Longest path from start to finish. Any slippage on the critical path will cause project delay.

Project Management - 1

1.23

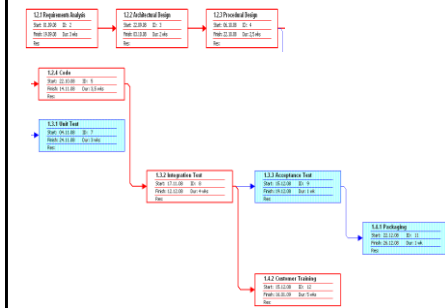
## PERT Example : Tasks and Durations

TASKS	WEEKS
<b>1.2 Software Development</b>	<b>11</b>
1.2.1 Requirements Analysis	3
1.2.2 Architectural Design	2
1.2.3 Procedural Design	2.5
1.2.4 Code	3.5
<b>1.3 Testing</b>	<b>7</b>
1.3.1 Unit Test	3
1.3.2 Integration Test	4
1.3.3 Acceptance Test	1
<b>1.4 Operations</b>	<b>5</b>
1.4.1 Packaging	1
1.4.2 Customer Training	5

Project Management - 1

1.24

## PERT Chart

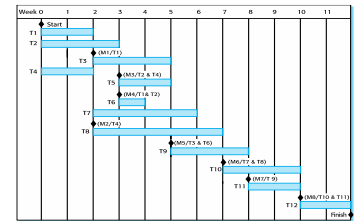


Project Management - 1

1.25

## GANTT Chart

- Also known as timeline chart
- Shows high level view of whole project
  - Horizontal bars show duration of tasks
  - Triangles show milestones
  - Vertical lines show dependencies



Project Management - 1

1.26

## An example

- Let's assume we identify the following tasks

TABLE 5.4 A task list

Activity number	Description	Predecessors	Duration	Staff number
2.1	Receive approval to proceed	—	—	—
3.1	Analyze requirements	2.1	1	2
3.2	Design			
3.2.1	Redesign existing components	3.1	6	4
3.2.2	Design new components	3.1	3	1
3.2.3	Design interfaces	3.2.2	1	2
3.3	Implement code			
3.3.1	Implement new code	3.2.2	6	2
3.3.2	Modify existing code	3.2.1, 3.2.3	5	1
3.4	Finish implementation			
3.4.1	Develop integration plan	3.2.2	2	2
3.4.2	Finish unit testing	3.3.1, 3.3.2	2	2
3.4.3	Update documentation	3.3.1, 3.3.2	2	3
3.5	Integrate and test			
3.5.1	Develop integration tests	3.4.1	1	3
3.5.2	Perform integration tests	3.4.2, 3.4.3, 3.5.1	1	2
3.6	Perform acceptance tests	3.5.2	1	1

Project Management - 1

1.27

## An example

- Following milestones can be extracted

TABLE 5.5 Milestones for the schedule network in Figure 5.6

Event	Description
1	Project initiation
2	Requirements analysis completed
3	Design of new components completed
4	Existing components redesigned; interfaces to new components designed
5	Integration plan completed
6	New code implemented; existing code modified
7	Documentation updated
8	Unit testing completed; documentation updated; integration tests ready
9	Integration tests completed
10	Acceptance tests completed

Project Management - 1

1.28

## An example

- Sometimes a task network is constructed to identify **critical paths**.

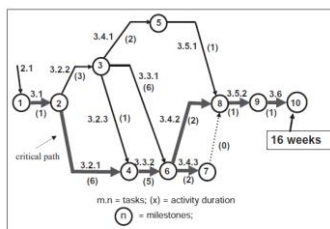


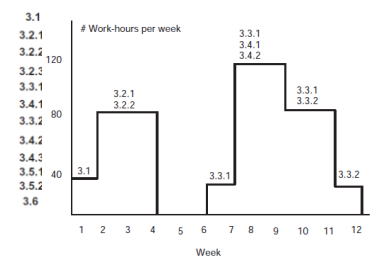
FIGURE 5.6 A critical-path activity network

Project Management - 1

1.29

## An example

- A final remark is to identify dependencies and pay attention to workloads



Project Management - 1

1.30

## Project Plans

- ⇒ In a plan-driven development project, a project plan sets out the resources available to the project, the work breakdown and a schedule for carrying out the work.
- ⇒ Plan sections
  - Introduction
  - Project organization
  - Risk analysis
  - Hardware and software resource requirements
  - Work breakdown
  - Project schedule
  - Monitoring and reporting mechanisms

Project Management - 1

1.31

## A Typical Software Project Plan Document

1. Introduction
  - A. Purpose of Plan
  - B. Project Scope and Objectives
2. Project Estimates
  - A. Historical Data Used for Estimates
  - B. Estimation Techniques
  - C. Estimates of Effort, Cost, Duration
3. Risks Management Strategy
  - A. Risk Table
  - B. Discussion of Risks to be Managed
  - C. RMMM Plan
4. Schedule
  - A. Project Work Breakdown Structure
  - B. Task Network
  - C. Timeline Chart
  - D. Resource Table
5. Project Resources
  - A. People
  - B. Hardware and Software
  - C. Special Resources
6. Staff Organization
  - A. Team Structure
  - B. Management Reporting
7. Tracking and Control Mechanisms
  - A. Quality Assurance and Control
  - B. Change Management and Control
8. Appendices

Project Management - 1

1.32

- 4. Management
  - 4.1. Risk Management
  - 4.2. Quality Management
  - 4.3. Change Management

## Risk Management

⇒ 4.4 ⇄

Project Management - 2

## 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.
- ⇒ There 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.

Project Management - 2

1.34

## 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

Project Management - 2

1.35

## 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)

Project Management - 2

1.36

## 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

Project Management - 2

1.37

## 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  $18 \times 100 \times 14 = \$25,200$ .
- **Risk exposure.**  $RE = 0.80 \times 25,200 \sim \$20,200$ .

Project Management - 2

1.38

## 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?

Project Management - 2

1.39

## 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.

Project Management - 2

1.40

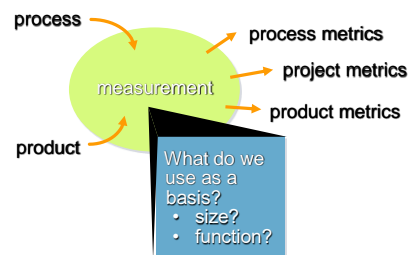
- 4. Management
  - 4.1. Risk Management
  - 4.2. Quality Management ←
  - 4.3. Change Management

## Quality Management and Metrics

4.4.203

Project Management - 2

## Why Do We Measure?



Project Management - 1

42

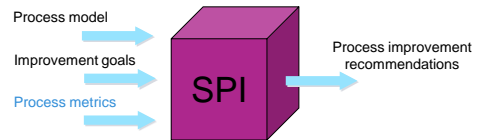
## Why Do We Measure?

- ⇒ assess the status of an ongoing project
- ⇒ track potential risks
- ⇒ uncover problem areas before they go "critical,"
- ⇒ adjust work flow or tasks,
- ⇒ evaluate the project team's ability to control quality of software work products.

Project Management - 1

43

## Software Process Improvement



Project Management - 1

44

## Typical Project Metrics

- ⇒ Effort/time per software engineering task
- ⇒ Errors uncovered per review hour
- ⇒ Scheduled vs. actual milestone dates
- ⇒ Changes (number) and their characteristics
- ⇒ Distribution of effort on software engineering tasks

Examples:

- ⇒ total FP estimation (Function Points)
- ⇒ total LOC estimation (Lines of Code)
- ⇒ FP per person-month
- ⇒ LOC per person-month
- ⇒ errors or defects per KLOC (thousand LOC)
- ⇒ pages of documentation per KLOC

Project Management - 1

45

## Object-Oriented Metrics

- ⇒ Number of **scenario scripts** (use-cases)
- ⇒ Number of **support classes** (required to implement the system but are not immediately related to the problem domain)
- ⇒ Average number of **support classes per key class** (analysis class)
- ⇒ Number of **subsystems** (an aggregation of classes that support a function that is visible to the end-user of a system)

Project Management - 1

1.46

## Web Project Metrics

- ⇒ Number of **static Web pages** (the end-user has no control over the content displayed on the page)
- ⇒ Number of **dynamic Web pages** (end-user actions result in customized content displayed on the page)
- ⇒ Number of **internal page links** (internal page links are pointers that provide a hyperlink to some other Web page within the WebApp)
- ⇒ Number of **persistent data objects**
- ⇒ Number of **external systems interfaced**
- ⇒ Number of **static content objects**
- ⇒ Number of **dynamic content objects**
- ⇒ Number of **executable functions**

Project Management - 1

1.47

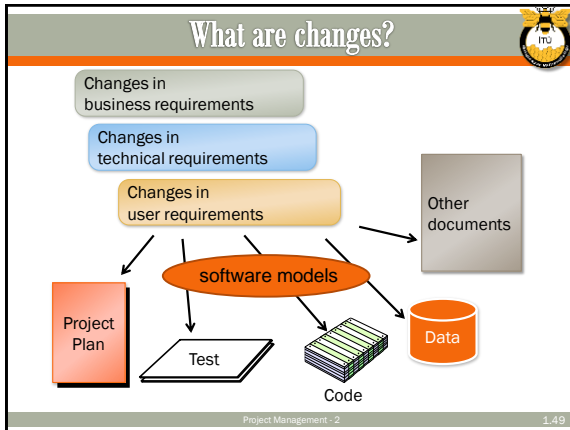
- 4. Management
  - 4.1. Risk Management
  - 4.2. Quality Management
  - 4.3. Change Management ←

## Change Management

⇒ 4.4.3.3

Project Management - 2





## 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 1.50

## Terminology - II

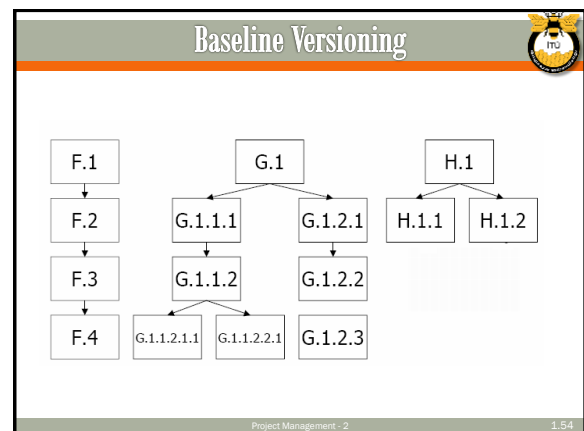
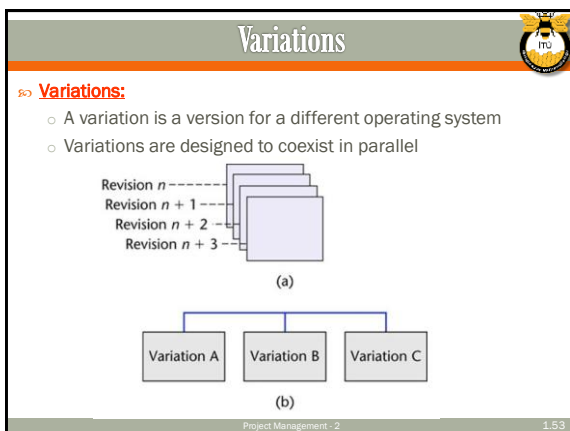
Term	Explanation
Mainline	A sequence of baselines representing different versions of a system.
Release	A version of a system that has been released to customers (or other users in an organization) for use.
Workspace	A private work area where software can be modified without affecting other developers 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.

Project Management - 2 1.51

## 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

Project Management - 2 1.52



## Configuration-Control Tools



### Configuration-Control Tools

- Gradle
- Maven
- Build.NET

### Versioning Tools

- CVS
- SVN
- GIT

### Other commercial tools

- TFS (Microsoft)
- ClearCase (IBM Rational Software)

Project Management - 2

55

## Wrap-up



This week we present

- Project Estimation: How to forecast about the project before the project goes underway
- Project Planning: What kind of activities should be perform to organize the project resources-activities in a plan-driven way
- Project Management: How can we measure-monitor-assess various indicators of project progress

Introduction &amp; UML

1.56

## Next Week



- We will introduce various techniques and contemporary issues in *Requirements Engineering!!!*

Introduction &amp; UML

1.57