Project Management, Cost Estimation, and Team Organizations

- Software Project Management
- Cost Estimation
- Managing People

Project Management

- Poor managment is the downfall of many software projects
  - Delivered software was late, unreliable, cost several times the original estimates and often exhibited poor performance characteristics
- Software project management is different from other engineeering management
  - product is intangible
  - still no clear understanding of the software process or evaluation criteria
  - most software projects are new and technically innovative
- Good management cannot guarantee project success, but bad management usually results in project failure!

Management Activities

- Proposal writing
  - overview, estimates, justification
- Project costing
  - software cost estimation
- Project planning and scheduling
  - milestones, options to minimize risks
- Project monitoring and reviewing
  - progress, compare to schedule and planned costs, predict problems
- Personnel selection and evaluation
  - skill, experience, training, resources
- Report writing and presentation
  - primary summary documentation and progress reviews

Project Planning and Scheduling

- Project planning determines a project schedule based upon
  - project constraints (delivery, staff, budget)
  - project parameters (structure, size, functions)
  - project milestones and deliverables
- Planning and scheduling must estimate risk associated with each decision
- Project scheduling involves separating work into tasks and predicting task completion
  - coordinate parallel tasks to optimize work force
  - allow for problems
- Schedule must be periodically revised with progress

Types of Plans

- Quality plan
  - Describes the quality procedures and standards that will be used in the project
- Validation plan
  - Describes the approach, resources and schedule used for system validation
- Configuration management plan
  - Describes the configuration management procedures and strcutures to be used
- Maintenance plan
  - Predicts the maintenance requirements of the system, maintenance cost and effort required
- Staff development plan
  - Describes how the skills and experience of the project team members will be developed

The Project Plan

- Introduction
  - Briefly describes the objectives of the product and sets out the constraints (budget, time, etc.)
- Project Organization
  - Describes the way in which the development team is organized
- Risk analysis
  - Describes possible project risks and risk reduction strategies
- Hardware & Software resource requirements
  - Hardware/Software required to carry out the development
- Work breakdown
  - Breakdown of the project into activities, identification of milestones and deliverables
- Project schedule
  - Describes the dependencies between activities, the estimated time required to reach each milestone and the allocation of people to activites
- Monitoring and reporting techniques
  - Describes the management reports which should be produced

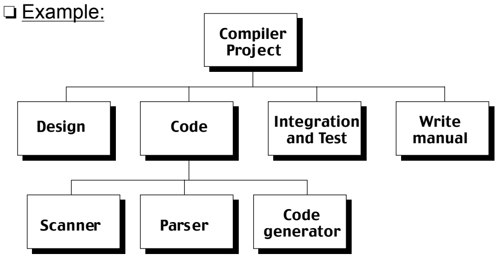
IEEE 1058.1,1987 SW Project Management Plan

1. Introduction
   1.1. Project Overview
   1.2. Project Deliverables
   1.3. Evoluation of the Software Project Management Plan
   1.4. Reference Materials
   1.5. Definitions and Acronyms
2. Project Organization
   2.1. Process Model
   2.2. Organizational Structure
   2.3. Organizational Boundaries and Interfaces
   2.4. Project Responsibilities
3. Managerial Process
   3.1. Management Objectives and Priorities
   3.2. Assumptions, Dependencies, and Constraints
   3.3. Risk Management
   3.4. Monitoring and Controlling Mechanisms
   3.5. Staffing Plan
4. Technical Process
   4.1. Methods, Tools, and Techniques
   4.2. Software Documentation
   4.3. Project Support Functions
5. Work Packages, Schedule, and Budget
   5.1. Work Packages
   5.2. Dependencies
   5.3. Resource Requirements
   5.4. Budget and Resource Allocation
   5.5. Schedule

Work Breakdown Structure

= one way of breaking down the major activity into smaller components

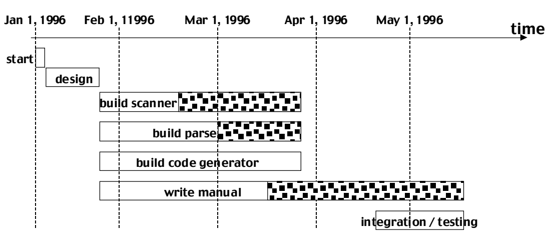
❏ Example:



GANTT Charts

❏ Project control technique for scheduling, budgeting, and resource planning

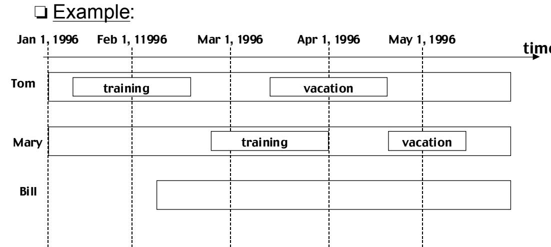
❏ Example: Gantt chart for simple compiler project



GANTT Charts - 2

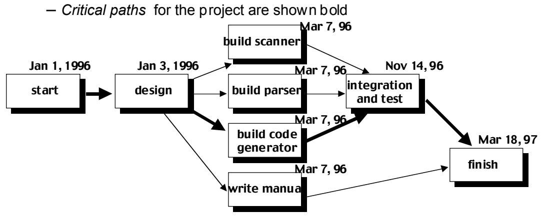
❏ Gantt charts can also be used for resource allocation and staff planning

❏ Example:



PERT Charts

❏ A PERT (**P**rogram **E**valuation and **R**eview **T**echnique) is a network of boxes (or circles) and arrows
  – Boxes (or circles) represent activities, arrows show the dependencies of activities on one another
  – Activity at the head of the arrow cannot start before activity at the tail of the arrow is finished
  – *Critical paths* for the project are shown bold



Software Cost Estimation

❏ Principal components of project costs derive from
  – hardware and software including maintenance
  – travel and training
  – effort (cost of paying software engineers)

❏ Initial cost estimation should be based on firm, complete requirements

❏ Continual cost estimation is required to ensure that spending is in line with budget

❏ Software Cost Estimation should use multiple techniques to predict costs:
  – historical cost information relating metrics and costs
  – analogies to similar systems
  – expert "guestimation"
  – hierarchical estimations

Factors affecting Software Pricing

Project Management, Cost Estimation, and Team Organizations

ICS 125

- Market opportunity
  - Moving into new markets --> low pricing
- Cost estimation uncertainty
  - If organization is unsure of its cost estimate, it may increase its price
- Contractual terms
  - Customer may be willing to allow the developer to retain ownership of the source code and reuse it in other projects
- Requirements volatility
  - If requirements are likely to change offer lower price to win the contract. After contract has been awarded, high prices may be charged for changes to the requirements
- Financial health
  - Sometimes it may be better to make a small profit or break even than to go out of business

13

Factors used in cost estimation models

Project Management, Cost Estimation, and Team Organizations

ICS 125

Group	Factor
Size attributes	Source instructions Number of routines Number of output formats Number of personnel
Program attributes	Type Complexity Language Required reliability
Personnel attributes	Personnel capability / continuity Hardware experience Application experience Language experience
Project attributes	Tools and techniques Customer interface Requirements definition

14

Algorithmic Cost Modeling

Project Management, Cost Estimation, and Team Organizations

ICS 125

- Model built by analyzing the attributes and costs of completed projects
  - metrics usually measure attributes of finished product (so predictions may be inaccurate)
  - metrics typically include size or function points (external interactions)
  - margin of error low if product is well-understood, model well calibrated to local organization, product is similar to previous projects, language and hardware choices are pre-defined
- Most algorithmic estimation models have an exponential component

$$Effort = C \times PM^{s} \times M$$

- C ...... complexity factor
- PM ... some product metric (size metric or functionality metric)
- M ...... multiplier, made up by combining different process, product and development attributes
- s .... exponent (usually close to 1) reflecting the increasing effort for large projects

15

Algorithmic Cost Modeling: The COCOMO Model [Boehm,1981]

Project Management, Cost Estimation, and Team Organizations

ICS 125

- Uses 3 different models depending on the complexity of the project:
  - *Simple project:* Well-understood applications developed by small teams.
    $PM = 2.4\ (KDSI)^{1.05} \times M$
  - *Moderate project:* More complex projects where team members may have limited experience of related systems.
    $PM = 3.0\ (KDSI)^{1.12} \times M$
  - *Embedded project:* Complex projects where the software is part of a strongly coupled complex of hardware, software, regulations, and operational procedures
    $PM = 3.6\ (KDSI)^{1.20} \times M$

KDSI ..number of thousands of delivered source instructions

16

The COCOMO Model – 2

Project Management, Cost Estimation, and Team Organizations

ICS 125

- In the basic COCOMO model the multiplier M is 1 (starting point for cost estimation)
- Intermediate model adds other product attributes as factors (multipliers):
  - product attributes (reliability, database size, etc.)
  - computer attributes (storage constraints, stability of hard/software)
  - personnel attributes (experience of personnel)
  - project attributes (use of software tools, development methods)
- Complete model decomposes total system in estimating costs (system is made up of sub-systems)

17

Project Duration and Staffing

Project Management, Cost Estimation, and Team Organizations

ICS 125

- Project managers have also to estimate
  - how long a software product will take to develop
  - when how many people will be needed to work on the project
- More people working on a project also requires more communication overhead
- COCOMO estimation:

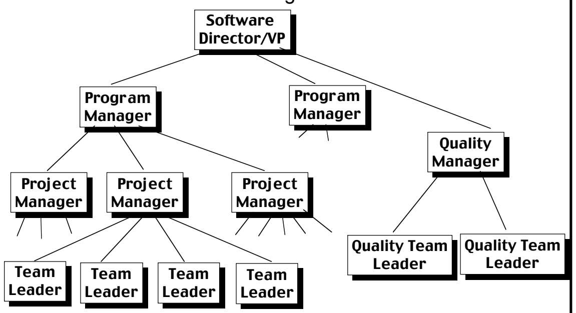
Simple projects $TDEV = 2.5\ (PM)^{0.38}$
Intermediate projects $TDEV = 2.5\ (PM)^{0.35}$
Embedded projects $TDEV = 2.5\ (PM)^{0.32}$

  - NOTE: COCOMO model does not include number of project engineers working on the project !

18

Management Structure

❑ Traditional hierarchical managment structure



Team Organization

❑ Large software systems require a coordinated team of software engineers for effective development

❑ Team organization involves devising roles for individuals and assigning responsibilities

❑ Organizational structure attempts to facilitate cooperation

❑ For long-term projects, job satisfaction is extremely important for reduced turnover

❑ Need mix of senior and junior engineers to facilitate both accomplishing the task and training

❑ Adding people to a project introduces further delays

Team Organizations

❑ Hierarchical organizations minimize and discourage communication, while democratic organizations encourage it

❑ Appropriate organization depends on project length and complexity

- small teams lead to cohesive design, less overhead, more unity, higher morale
- but some tasks too complex
- optimal size between 3 and 8

❑ Appropriate design leads to appropriate assignment of tasks and appropriate team organization

Group Cohesiveness

❑ In a cohesive group, members think of the group as more important than the individuals in it

❑ Advantages of cohesive groups:

- + A group quality standard can be developed
- + Team members work closely together. Members can learn from each other
- + Team members can get to know each other´s work (continuity can be maintained should a team member leave)
- + Egoless programming can be practised (programs are regarded as team property rather than personal property)

❑ Problems:

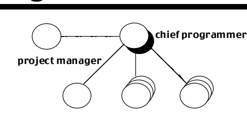
- Irrational resistance to a leadership change
- Groupthink (consideration of alternatives is replaced by loyalty to group norms and decisions)

Group communication

❑ Several factors affect communications in a group

- Status of the group members
  - Higher-status members tend to dominate communications with lower-status members
- Personalities in the group
  - If there are too many people in the group who are task-oriented, this may inhibit effective communications (all are concentrated on technical issues and nobody discusses problems)
- Sexual composition of the group
  - Studies have shown that men and women prefer to work in mixed-sex groups. Within these groups, communications are better than in single-sex groups
- Communication channels
  - Communications are more effective if anyone in a group can easily contact anyone else

Centralized-Control: Chief Programmer Team Approach

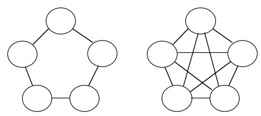


❑ Hierarchical organizational structure and matching pattern of communication

- chief programmer reports to peer project manager
- programmers report to chief programmer
- librarian responsible for central repository
- specialists added as needed

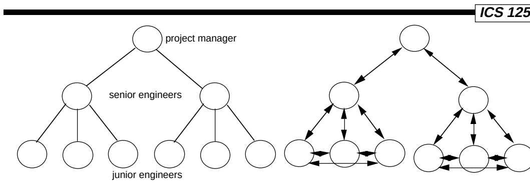
❑ Works well with simple tasks that can be grasped by one *good* engineer, but "single point of failure"

Decentralized Control: Democratic Team Approach



- ❑ Ring organization and connected communication
  - decisions made by consesus
  - all work is group work: "egoless programming"
    - leads to higher morale and job satisficaction
    - not appropriate for large teams
- ❑ More appropriate for less understood and more complex programs with longer term project

Mixed Control



- ❑ Hierarchy with extra communication
  - senior engineers report to project manager
  - junior engineers report to senior engineers
  - control is vested in project manager and senior engineers
  - communication is decentralized among each set of peers and their supervisor
- ❑ Limits communication to a small group and realizes benefits of group decisions by vesting authority

Assessment of Team Organizations

- ❑ No team organization is appropriate for all tasks
- ❑ Decentralized control is best when communication among engineers is necessary for achieving a good solution
- ❑ Centralized control is best when speed of development is the most important goal and the problem is well understood
- ❑ An appropriate approach tries to limit amount of communication overhead
- ❑ An appropriate approach also has to take into account other goals, such as personnel turnover, development of junior engineers, dissemination of specialized knowledge among personnel

Training and Motivation

- ❑ Training is not only a way of ensuring that engineers have necessary skills, but also demonstrates an organization´s interest in its staff
- ❑ Training may also cause problems:
  - Learning new languages may also be very difficult, especially for older programmers (switch of the paradigm, e.g. from FORTRAN to C++/LISP/Prolog, etc.)
  - Programmer training requires consideration of different educational background and experience of programmers
- ❑ Motivation of people requires
  - satisfaction of their social needs (time for meeting co-workers, providing places to meet)
  - recognition of achievements
  - giving people responsibility for their own work

Risk Management

- ❑ An engineering project is
  - expected to produce a reliable product
  - within a limited time
  - using limited resources
- ❑ Risk management
  - identifying project risks
  - assessing their impact
  - monitoring and controlling the risks
- ❑ Approaches to reduce risks in Software Engineering
  - Prototyping
  - Incremental delivery
  - Modular design (i.e. handling risks of late changes)

Common Risk Items in Software Engineering [Boehm,1989]

Risk items	Risk management techniques
Personnel shortfalls	Staffing with top talent; job matching; team-building;
Unrealistic schedules & budgets	Detailed multisource cost & schedule estimation; incremental development; software reuse
Developing the wrong software functions	Organization analysis; mission analysis; user surveys; prototyping; early users manuals
Developing the wrong user interface	Prototyping; scenarios; task analysis; user characterization
Continuing stream of requirements changes	Information hiding; incremental development (defer changes to later increments)
Real-time performance shortfalls	Simulation; benchmarking; modeling; prototyping