Software Project Management

Fall, 2024

jehong@chungbuk.ac.kr
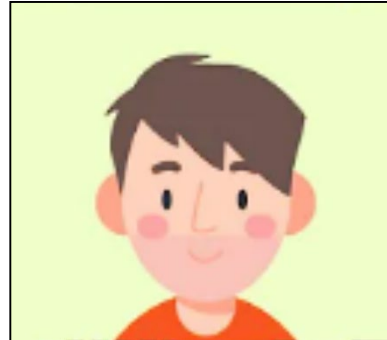
Prologue

What type of leader do you prefer?
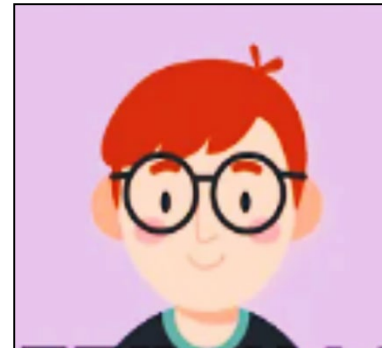
멍부	멍게	똑부	똑게
Stupid + Diligent	Stupid + Lazy	Smart + Diligent	Smart + Lazy

(?) 똑게 >> 멍게 >> 똑부 >> 멍부

Topics Covered

Software Project Management

- What is project management?
- Project planning
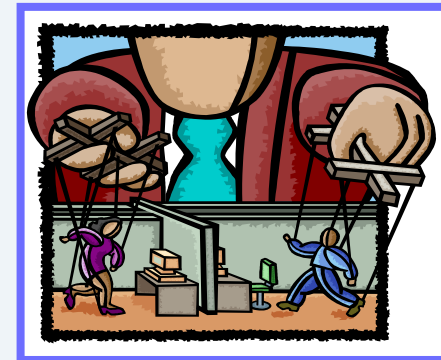- Project cost estimation
  - Function Point
  - COCOMO

Project Control Techniques

- Work breakdown structure
- Gantt chart
- PERT chart

Project Team Organization

Risk Management

Project Management Plan

Intro ..

Why software project fails ?

Lack of SW mind   Lack of appropriate SE skills

Insufficient software project management

Which issues to be managed in software project ?

Project Management

Needs

- Project failure due to ineffective management
- Project-late, unreliable, over budget, poor performance

Differences from other types of engineering

- Product is intangible
- Not have a clear understanding of the SW process
- Design intensive, as opposed to manufacturing intensive

Most Important Contributor to a Successful Software Project

" .. it's not the tools that we use, but it's the people"

"... having smart people .. very little else matters in my opinion."

" The only rule I have in management is to ensure I have good people"

Management Functions

Definition of Management

- **Creation and maintenance** of an internal environment in an enterprise where individuals, working together in groups, can perform efficiently and effectively toward the attainment of group goals

General functions of management

- Planning : objectives, resources, flow of information, people, artifacts
- Organizing : authority and responsibility for groups
- Staffing: hiring personnel
- Directing: leading subordinates
- Controlling: measuring and correcting activities

Management Steps

Step 1. Planning

- Understanding & documenting the goal.
- Developing a schedule, budget and other resource req.s.

Step 2. Acquisition of resources

- Space, computing resources, materials and human resources

Step 4. Monitoring

- Checking the progress of the project.
- Taking necessary actions to handle deviation from the plan.

Step 3. Execution

- Putting the plan into action.

Project Planning

Planning Issues

- Define and document the assumptions, goals, and constraints clearly
- Determine the required resources and budget
    - The number and skill level of the people
    - The amount of computer resources

Forecast in Planning

- How many engineer will be needed
- **Productivity** of software engineer

→ Software Cost Estimation

Software Productivity

For project planning, need to

- Estimate the difficulty of the task.

- Estimate how much of the task each engineer can solve.

Productivity metrics

- Amount of functionality

- LOC: not an ideal productivity metrics

How to quantify the concept of functionality?

- Function points.

Code Size (LOC)

The most commonly used metrics to measure productivity.

Two most common code size:

- DSI(delivered source instructions)
  - Only lines of code delivered to the customer.
- NCSS(non-commented source statements)
  - Comment lines are not counted.

Many problems to be solved, but easy to measure.

Currently not useful, but referenced.

Function Points

Attempt to quantify the functionality of a software system.

Characterize the complexity of the system.

Can be used to forecast

- how long it will take
- how many people will be needed to do it.

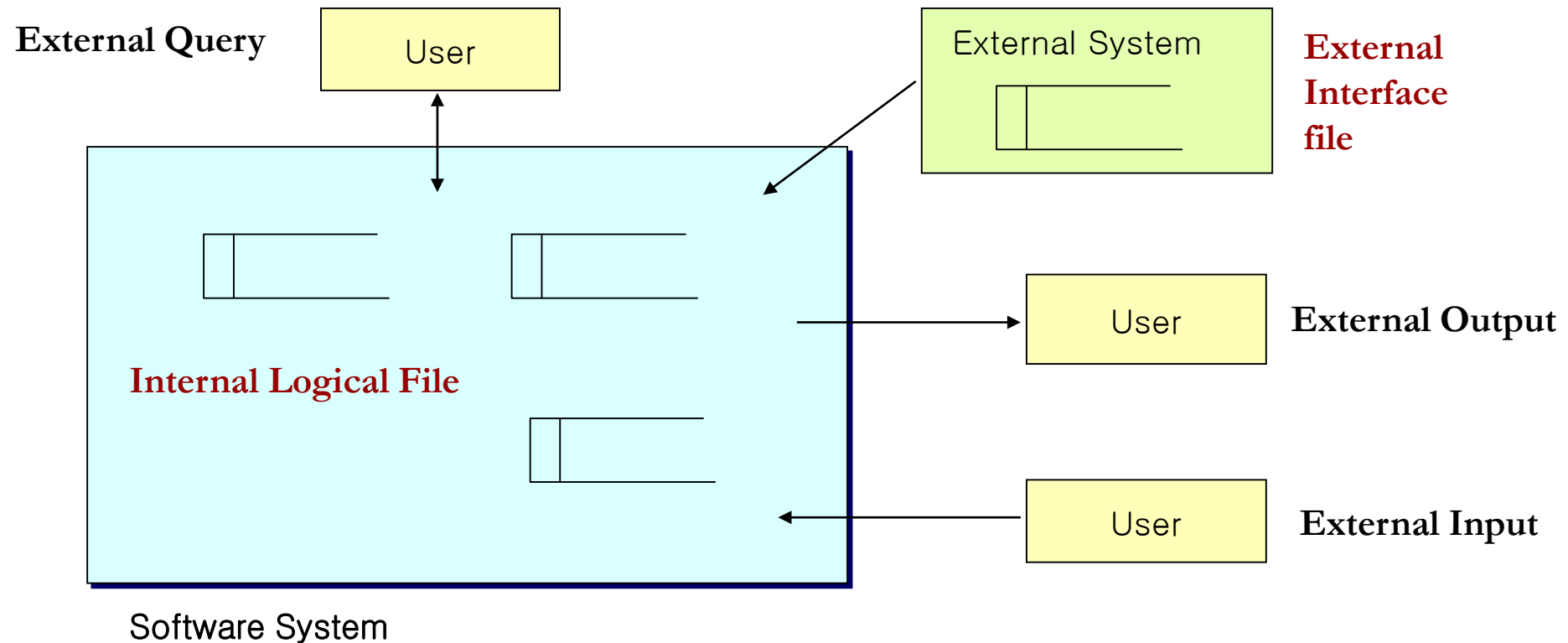Suitable for information processing applications.

Can measure productivity, amount of money, number of errors.

Can be used as bases for future planning.

Used to measure the relative power of different languages.

Many problems to be solved, not promising.

Functionality of General Software Systems

External Query

| User |

| External System |

External Interface file

Internal Logical File

| User | **External Output**

| User | **External Input**

Software System

- **Data Function : Internal Logical File + External Interface File**
- **Transaction Function : External Query + External Output + External Input**

FP Analysis Method

1

Determine the type of Project

2

Determine the scope of target system

3

Identify data function & determine the complexity

4

Identify transaction function & determine the complexity

5

Produce the unadjusted FP

6

determine the Value adjust. factors

7

Produce the adjusted FP
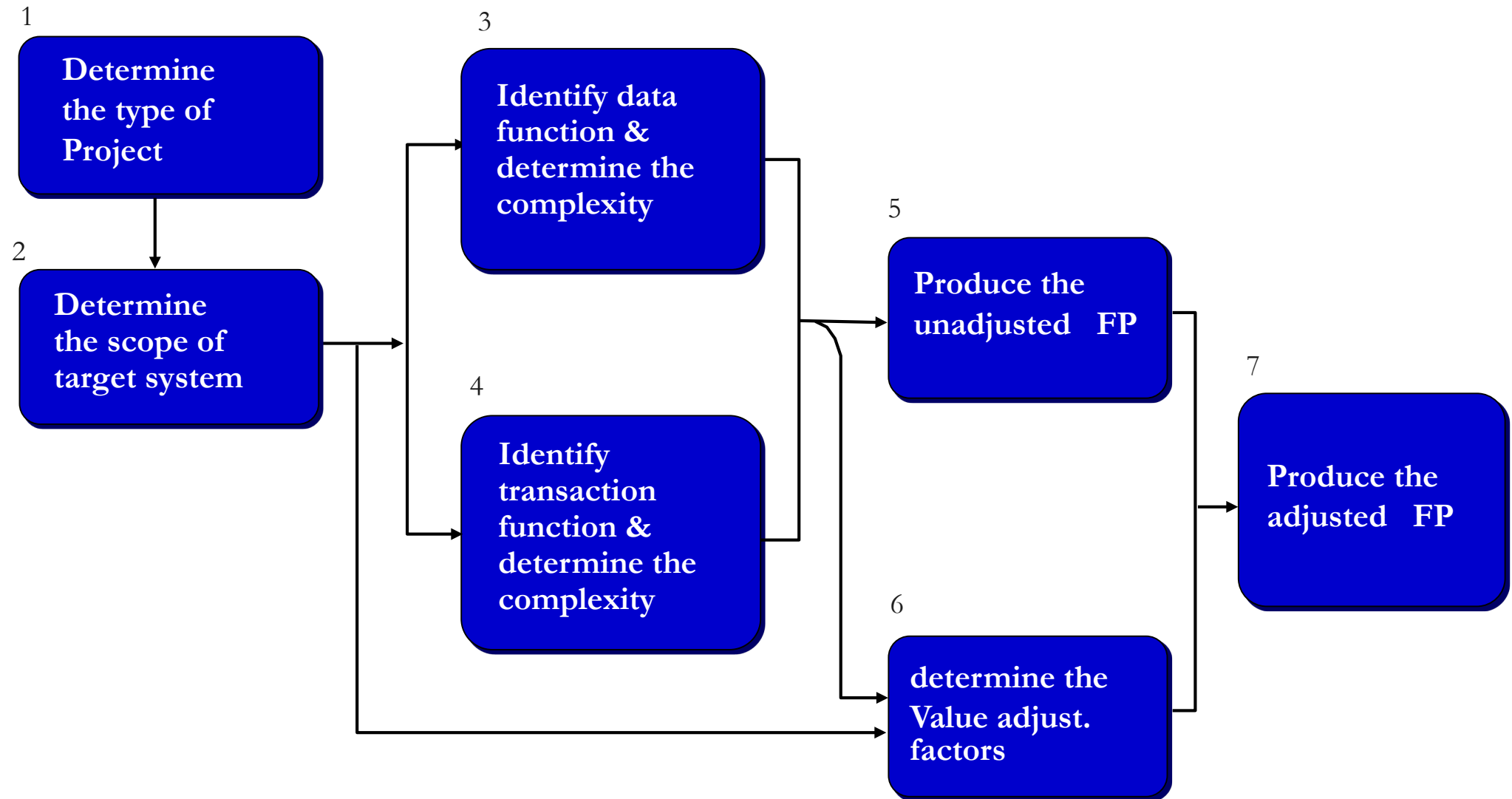
Activities in Each Step

1. Determine the type of project

- New project / Maintenance Project / Enhancement Project

2. Determine the scope of target system

- Whole or Part / In-house development, Outsourcing, Package acquisition, …

3. Identify data functions & their complexity

Internal Logical File		Data Element Type		
		1 - 19	20 - 50	>= 51
Record Element Type	1	Low	Low	Mid
	2 – 5	Low	Mid	High
	> 5	Mid	High	High

Activities in Each Step (Cont.)

4. Identify transaction functions and their complexity

- Ex : Complexity matrix for "External Input"

External Input		Data Element Type		
		1 - 4	5 - 15	>= 16
Reference File Type	< 2	Low	Low	Mid
	2	Low	Mid	High
	> 2	Mid	High	High

5. Produce the unadjusted FP

- UFP = Sum of weights for all functions

| Functions	Complexity Level (Weight)		
	Low	Mid	High
ILF	X 7	X 10	X 15
EIF	X 5	X 7	X 10
EI	X 3	X 4	X 6
EO	X 4	X 5	X 7
EQ	X 3	X 4	X 6

Activities in Each Step (Cont.)

6. Determine the value adjustment factors (VAF)

- 14 System Characteristics

1. Data Communication	8. Online Update
2. Distributed Data Processing	9. Complex Processing
3. Performance	10. Reusability
4. Heavily Used Configuration	11. Installation Ease
5. Transaction Rate	12. Operational Ease
6. Online Data Entry	13. Multiple Sites
7. End User Efficiency	14. Facilitate Change

- Degree of Influence

0	No Required
1	Weak Influence
2	:
3	Strong Influence
4	
5	Very Strong …
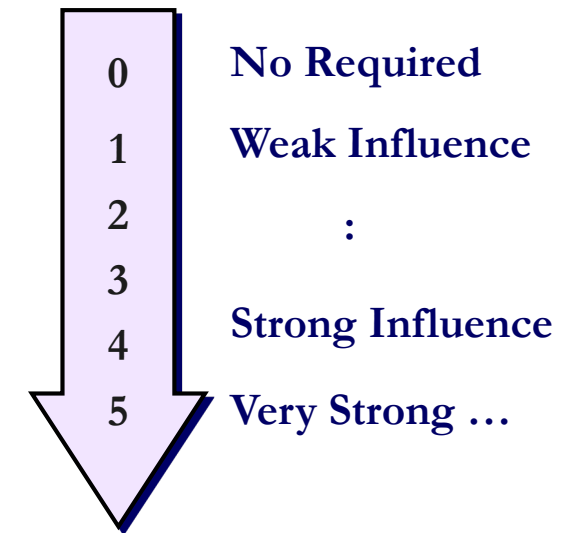
$$VAF = (TDI * 0.01) + 0.65$$

7. Produce the adjusted FP

- Adjusted Function Point,   AFP = UAF **X** VAF

LOC vs. FP by Programming Language

- **Implementation Lines of Code / 1 FP**

Imple. Languages		# Lines	Imple. Languages	# Lines
Assembly	Basic	320	FORTRAN	107
	Macro	213		
BASIC		107	HTML 3.0	15
Visual Basic		29	LISP	64
C		132	JAVA	53
C++		53	PL/I	80
COBOL		107	SQL	13
DELPHI		29	Power Builder	16

* Source : www.theadvisors.com/langcomparison.htm

Other Factors Affecting Productivity

The capability of the personnel

The complexity of the product

Required reliability

- Timing constraints (in real-time systems)

Schedule constraints

Language experience

Personnel turnover

Restructuring of the systems

...

Techniques of Software Cost Estimation

Algorithmic cost modeling

- COCOMO, COCOMO II

Expert judgment
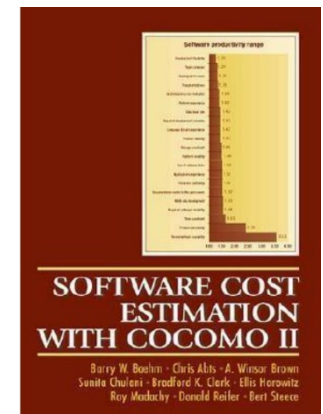
Estimation by analogy

- by similar project

Parkinson's Law

- determined by available resources

Top-down estimation

- overall functionality, first

Bottom-up estimation

SOFTWARE COST
ESTIMATION
WITH COCOMO II

Barry W. Boehm · Chris Abts · A. Winsor Brown
Sunita Chulani · Bradford K. Clark · Ellis Horowitz
Roy Madachy · Donald Reifer · Bert Steece

COCOMO

Constructive Cost Model, 1981 B. Boehm

Based on the delivered source instructions, KDSI

Based on a set of three different models of increasing complexity and level of detail

Nominal effort and schedule equations

Development Mode	Nominal effort	Schedule
Organic	$PM = 2.4(KDSI)^{1.05}$	$TDEV = 2.5(PM_{DEV})^{0.38}$
Semidetached	$PM = 3.0(KDSI)^{1.12}$	$TDEV = 2.5(PM_{DEV})^{0.35}$
Embedded	$PM = 3.6(KDSI)^{1.20}$	$TDEV = 2.5(PM_{DEV})^{0.32}$

PM = Person(programmer)-Month, PM_{DEV} = PM * Effort multiplier,

TDEV = Months required to complete the project

PM_{DEV} / TDEV = # of required persons

COCOMO

Effort Multiplier

$$= \prod_{i=1}^{15} D_i$$

Cost Driver (D_i)	Very low	Low	Nominal	High	Very High	Extra High
Product attributes						
Required software reliability	.75	.88	1.00	1.15	1.40	
Data base size		.94	1.00	1.08	1.16	
Product complexity	.70	.85	1.00	1.15	1.30	1.65
Computer attributes						
Execution time constraints			1.00	1.11	1.30	1.66
Main storage constraints						
Platform volatility		.87	1.00	1.15	1.30	
Computer turn around time		.87	1.00	1.07	1.15	
Personnel attributes						
Analyst capability	1.46	1.19	1.00	.86	.71	
Applications experience	1.29	1.13	1.00	.91	.82	
Programmer capability	1.42	1.17	1.00	.86	.70	
Virtual machine experience*	1.21	1.10	1.00	.90		
Programming language experience	1.14	1.07	1.00	.95		
Project attributes						
Use of modern programming practices	1.24	1.10	1.00	.91	.82	
Use of software tools	1.24	1.10	1.00	.91	.83	
Required development schedule	1.23	1.08	1.00	1.04	1.10	

Ratings

COCOMO (Cont.)

Allows to analyze sensitivity

- by changing parameters from effort multipliers

Difference with COCOMO II

- Assumption about the process model
  - COCOMO : Sequential development process (waterfall model)
  - COCOMO II : iterative approach, RAD, Reuse-driven approach, etc
- Estimator
  - COCOMO : Source Instructions (KDSI)
  - COCOMO II : Source Instructions and Function points

COCOMO II

B. Boehm, 1995

Use different model by project progress

- by changing parameters from effort multipliers

3 Models of COCOMO II

	STEP 1	**STEP 2**	**STEP 3**
When	Prototyping phase	Preliminary Design	Post architecture
Metrics	App. point	FP	FP & LOC
Reuse	Implicit	Explicit	Explicit
Req. Changes	Implicit	One of Cost Drivers	One of Cost Drivers

* Application point : # of components, # of screens in input/output interface

Project Control

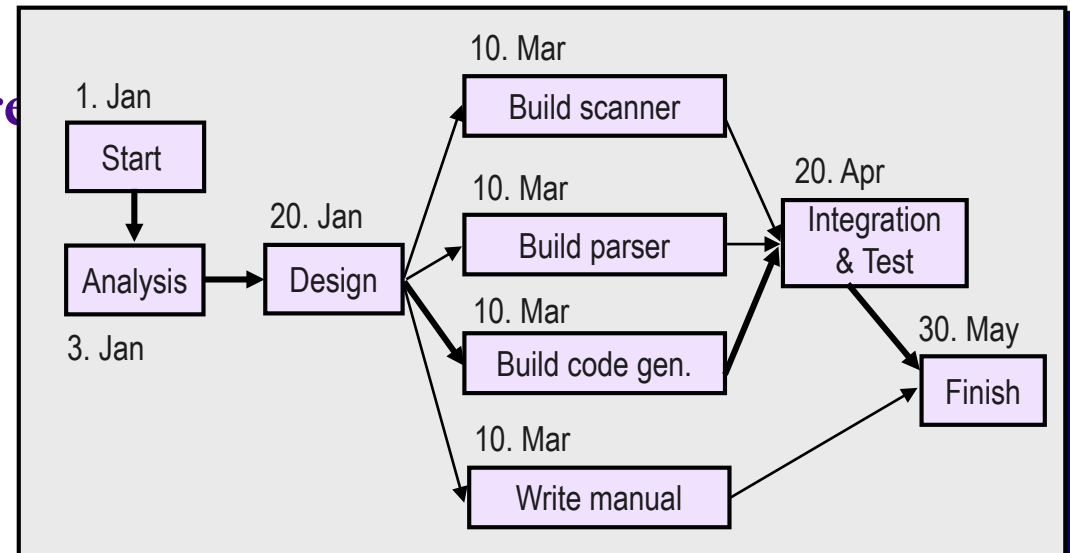**Monitor the progress of the activities.**

Detect when the deviations from the plan are occurring.
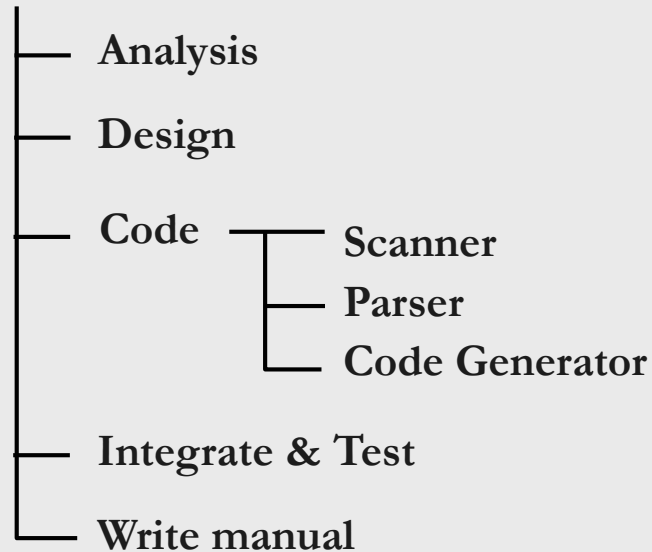
Use project control techniques:

- Work breakdown structure
- Gantt chart
- PERT chart(program evaluation & review technique)
- Kanban board

Project Control Techniques

WBS (Work Breakdown Structure
Gantt chart
PERT chart

Complier Project
- **Analysis**
- **Design**
- **Code**
  - **Scanner**
  - **Parser**
  - **Code Generator**
- **Integrate & Test**
- **Write manual**

PERT chart:
- 1. Jan — Start
- 3. Jan — Analysis
- 20. Jan — Design
- 10. Mar — Build scanner
- 10. Mar — Build parser
- 10. Mar — Build code gen.
- 10. Mar — Write manual
- 20. Apr — Integration & Test
- 30. May — Finish

Gantt chart:
- 1 Jan. | 1 Feb. | 1 Mar. | 1. Apr. | 1 May. | 1.Jun
- Start
- Analysis
- Design
- Build scanner
- Build parser
- Build code gen.
- Write manual
- Int. & test
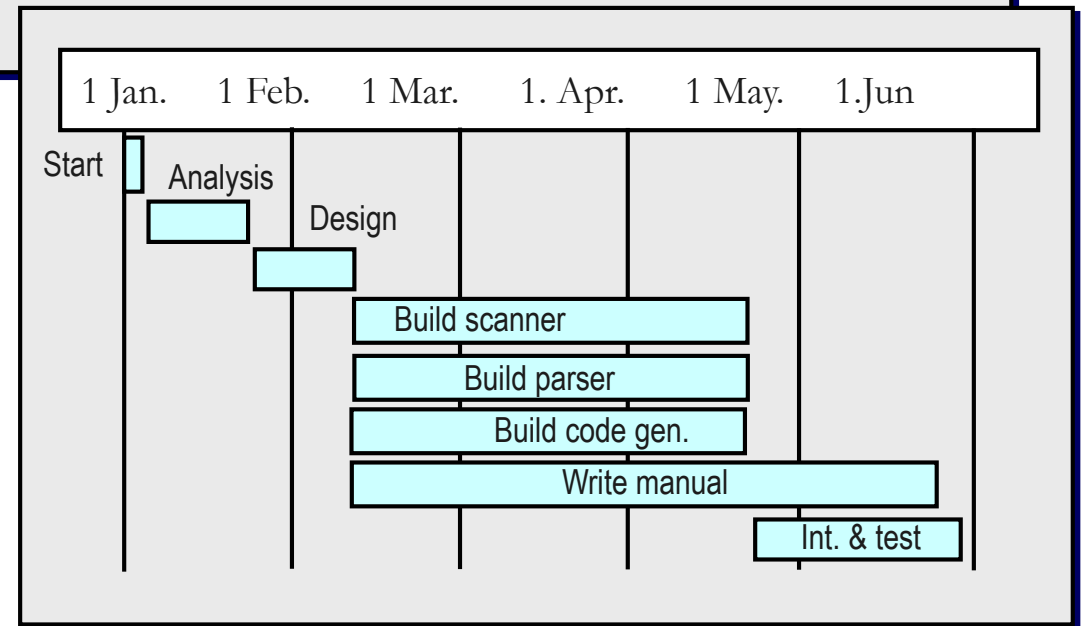
Project Control Techniques – WBS (1)

WBS (Work Breakdown Structure)

- Based on breaking down the goal of the project into several intermediate goals
- WBS Goal
  - Identify all activities that a project must undertake
- A tree whose root is labeled by the major activity of the project
  - Broken down into smaller components
  - Until each leaf in the tree represents a piece of work that the manager feels confident to estimate in term of size, difficulty, and resource requirements
- Used for
  - Summary of the project plans
  - Input into the scheduling process

Project Control Techniques – WBS (2)

An Example of WBS

ID	ℹ	작업 완	작업 이름	기간	시작 날짜	완료 날짜	선행 작업	자원 이름
1		0%	OO은행 SPI Project	424 일	04-01-05 (월)	05-08-29 (월)		
2		0%	현황진단단계	17 일	04-01-05 (월)	04-01-30 (금)		
3	▦	0%	SPI TFT 구성	1 일	04-01-05 (월)	04-01-05 (월)		
4	▦	0%	SPI 운영위원회 구성	3 일	04-01-06 (화)	04-01-08 (목)	3	
5	▦	0%	SPI 세미나	2 일	04-01-06 (화)	04-01-07 (수)	3	
6	▦	0%	현황진단	9 일	04-01-08 (목)	04-01-20 (화)	3,7	
7		0%	프로세스 교육	2 일	04-01-06 (화)	04-01-07 (수)	3	
8		0%	PIT 구성	4 일	04-01-09 (금)	04-01-14 (수)	4	
9	▦	0%	우선순위 결정	1 일	04-01-26 (월)	04-01-26 (월)	6	
10		0%	SPI 계획 수립 및 승인	5 일	04-01-26 (월)	04-01-30 (금)	6	
11		0%	SPI TFT 멘터링	15 일	04-01-06 (화)	04-01-29 (목)	3	
12	▦	0%	단계 종료 보고	1 일	04-01-30 (금)	04-01-30 (금)	11	
13		0%	구축 및 초기적용 단계	126 일	04-02-02 (월)	04-07-30 (금)	2	
14		0%	표준 프로세스 개발	70 일	04-02-02 (월)	04-05-12 (수)		
15		0%	프로세스 개발	40 일	04-02-02 (월)	04-03-29 (월)		
16		0%	템플릿 개발	12 일	04-03-30 (화)	04-04-15 (목)	15	
17		0%	가이드라인 개발	12 일	04-04-16 (금)	04-05-03 (월)	16	
18		0%	체크리스트 개발	6 일	04-05-04 (화)	04-05-12 (수)	17	
19		0%	방법론 개발	50 일	04-02-02 (월)	04-04-13 (화)	2	
20	▦	0%	객체지향 방법론	25 일	04-02-02 (월)	04-03-08 (월)		
21	▦	0%	정보공학 방법론	25 일	04-03-09 (화)	04-04-13 (화)	20	
22		0%	SPI 교육	12 일	04-05-13 (목)	04-05-31 (월)	14,19	
23		0%	방법론 교육	4 일	04-05-13 (목)	04-05-18 (화)	14,19	
24		0%	프로세스 파일럿 적용	43 일	04-06-01 (화)	04-07-29 (목)	22,23	
25		0%	프로젝트 멘터링 및 코칭	43 일	04-06-01 (화)	04-07-29 (목)	22,23	
26		0%	단계 종료 보고	1 일	04-07-30 (금)	04-07-30 (금)	24,25	

Project Control Techniques – Gantt chart (1)

A techniques that can be used for scheduling, budgeting, and resource planning.
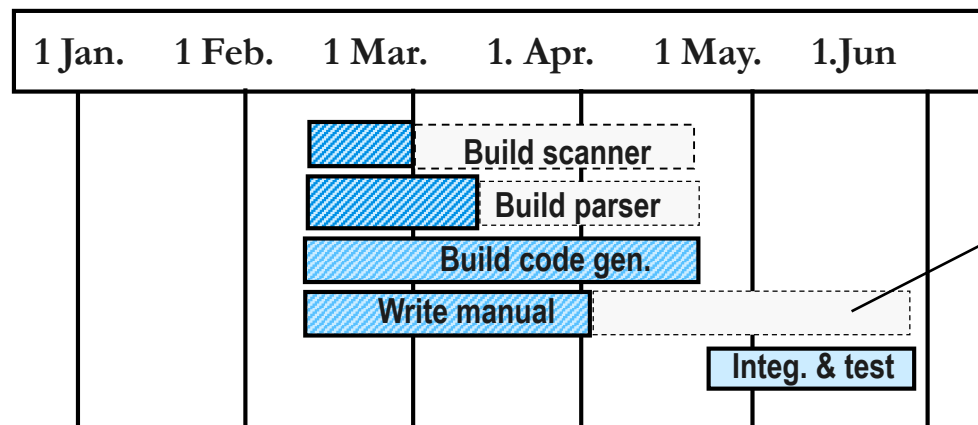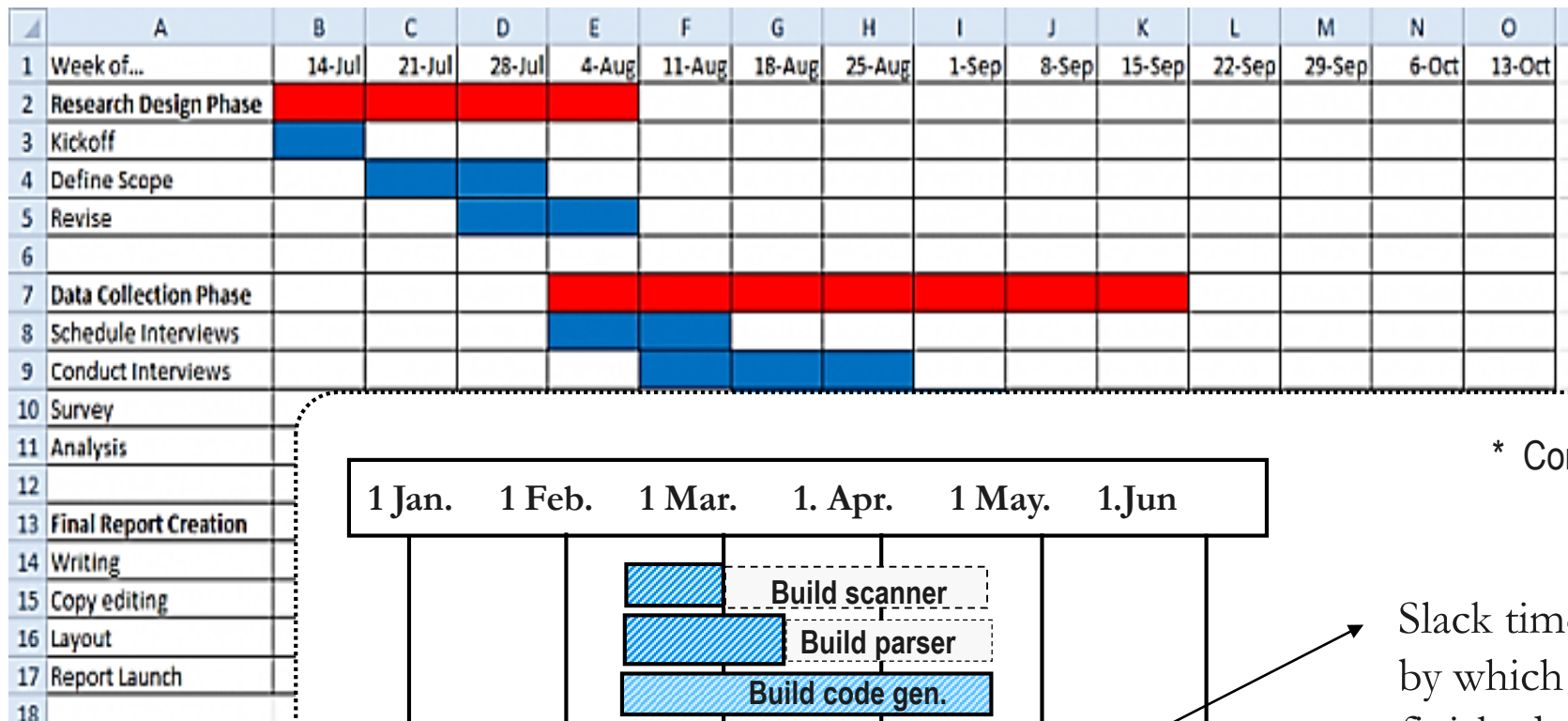
A kind of bar chart

- Each bar represents an activity
- Drawn against a timeline (proportional to the length of time planned for the activity)

Used to allocate resources and plan staffing

Do not highlight inter-task dependencies

Project Control Techniques – Gantt chart (2)

Example of Gantt chart

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Week of...	14-Jul	21-Jul	28-Jul	4-Aug	11-Aug	18-Aug	25-Aug	1-Sep	8-Sep	15-Sep	22-Sep	29-Sep	6-Oct	13-Oct
2	Research Design Phase														
3	Kickoff														
4	Define Scope														
5	Revise														
6															
7	Data Collection Phase														
8	Schedule Interviews														
9	Conduct Interviews														
10	Survey														
11	Analysis														
12															
13	Final Report Creation														
14	Writing														
15	Copy editing														
16	Layout														
17	Report Launch														
18															

* Compiler Project *

1 Jan.	1 Feb.	1 Mar.	1. Apr.	1 May.	1.Jun

Build scanner

Build parser

Build code gen.

Write manual

Integ. & test

Slack time : the latest time by which a task must be finished

Project Control Techniques – PERT chart (1)

Program Evaluation and Review Techniques, PERT
   Critical Path Method, CPM

Represents the dependencies of activities which should performed in a project

- A network of boxes (activities) and arrows (dependencies of activities)

Salient features

- Expose all possible parallelism in the activities
- Allows scheduling and simulation of alternative schedules
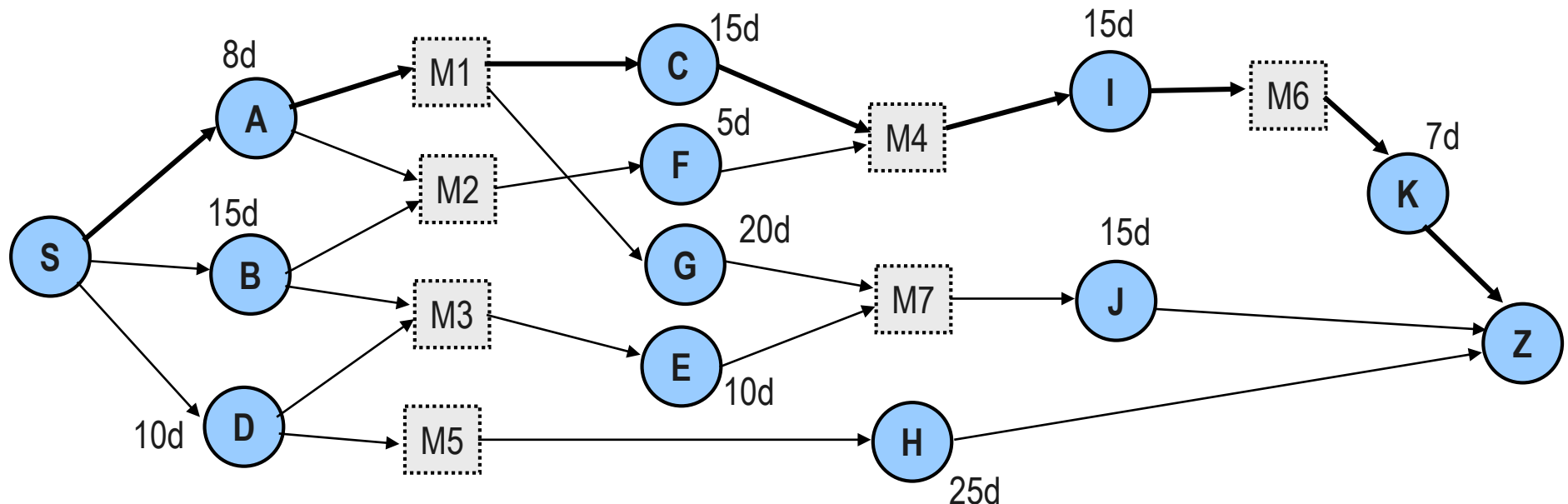- Enables the manager to monitor and control the project

Project Control Techniques – PERT chart (2)

Dates from PERT chart
- Earliest start date and latest start date
- Earliest finish date and latest finish date

Critical path
- A path that cause a delay in the entire project because of any delay in any activity

Kanban Board

Kanban board

- Management tool designed to help visualize work, limit work-in-progress, and maximize efficiency
- Visualization tool that shows work in progress to help identify bottlenecks and overcommitments

Kanban Origin

- Toyota Kanban System for JIT production (1963)
  - Used to organize its manufacturing operations, such as logistics, supplier management and customer delivery.

Application Areas

- Agile and DevOps teams to establish order in daily work
- Marketing teams, HR teams, Personal Kanban, ..

Kanban Board

Digital Kanban board by JIRA

Dealing with Deviations from the Plan

The manager must decide how to handle the deviation from the schedule

- Not adding engineer, But right engineer
- Temporarily reassigning senior engineer or Hiring expert troubleshooting consultants
- Requirement scrubbing
- Admit the incorrectness of the original plans and schedules
- Accept a delay as the right course of action

Team Organization

Aim : to facilitate cooperation towards a common goal

Types of organizational structure

- Centralized-control team
- Decentralized-control team
- and More ..

cf) function-oriented, Project-oriented and Matrix structure

Considerations that affects the choice of an org. structure

- The length of the project
- The nature of task and how much communications
- The appropriate size for the team
  - A team should be large enough, but not too large, and small enough, but not too small

Organization Structure

Centralized-control team

- Chief programmer team
- Chief programmer is responsible for the design and all the technical details of the project
- Works well when the task is well understood.
- The chief programmer may be overloaded.

Communication pattern:

Project Manager or
Chief programmer

Librarian   Programmers   Specialist
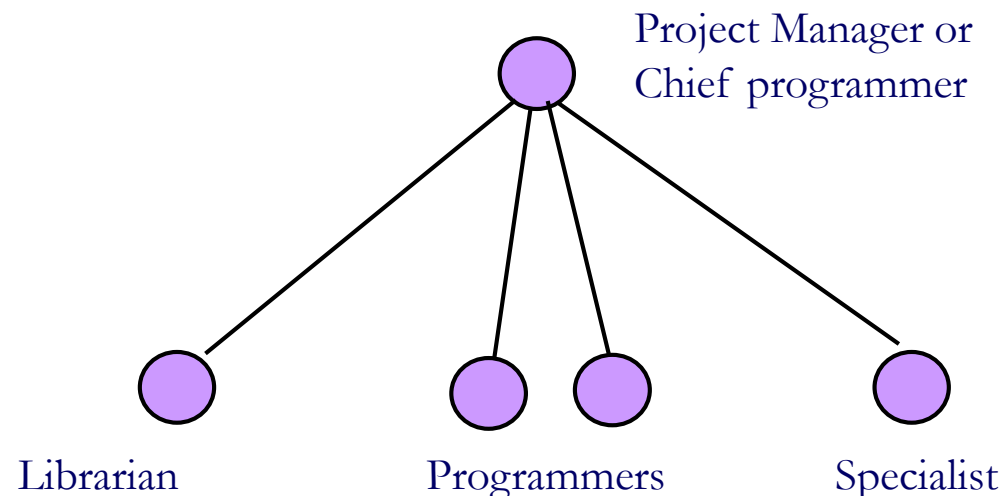
Organization Structure (cont.)

Decentralized-control team

- Democratic team
- Decisions are made thru consensus.
- Review each other's work.
- Leads to higher morale and less turnover.
- Suitable for long-term projects.
- Appropriate for less understood and more complicated problems.
- Not suitable for large teams.

Communication pattern:
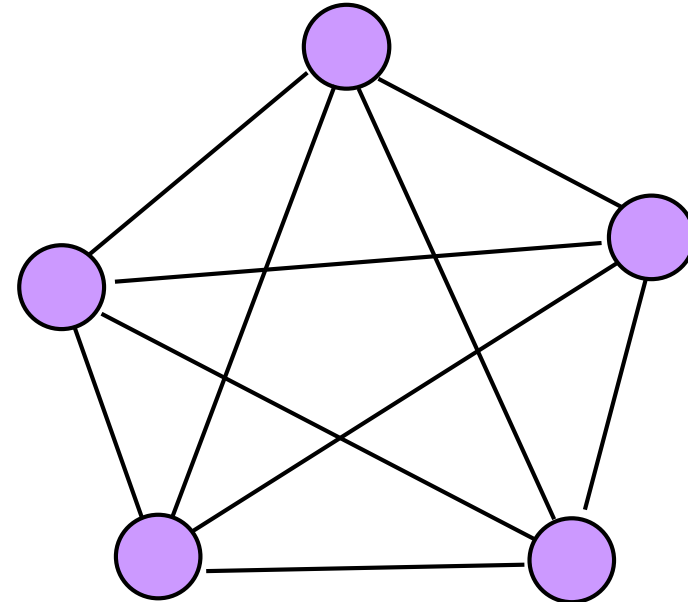
Organization Structure (cont.)

Mixed-control team

- Hierarchical team

- Combination of centralized and decentralized teams.

- Attempts to have advantages of both.

Communication pattern:

Project manager

Senior managers

Junior Engineers

Organization Structure (cont.)

Assessment of Team Organizations
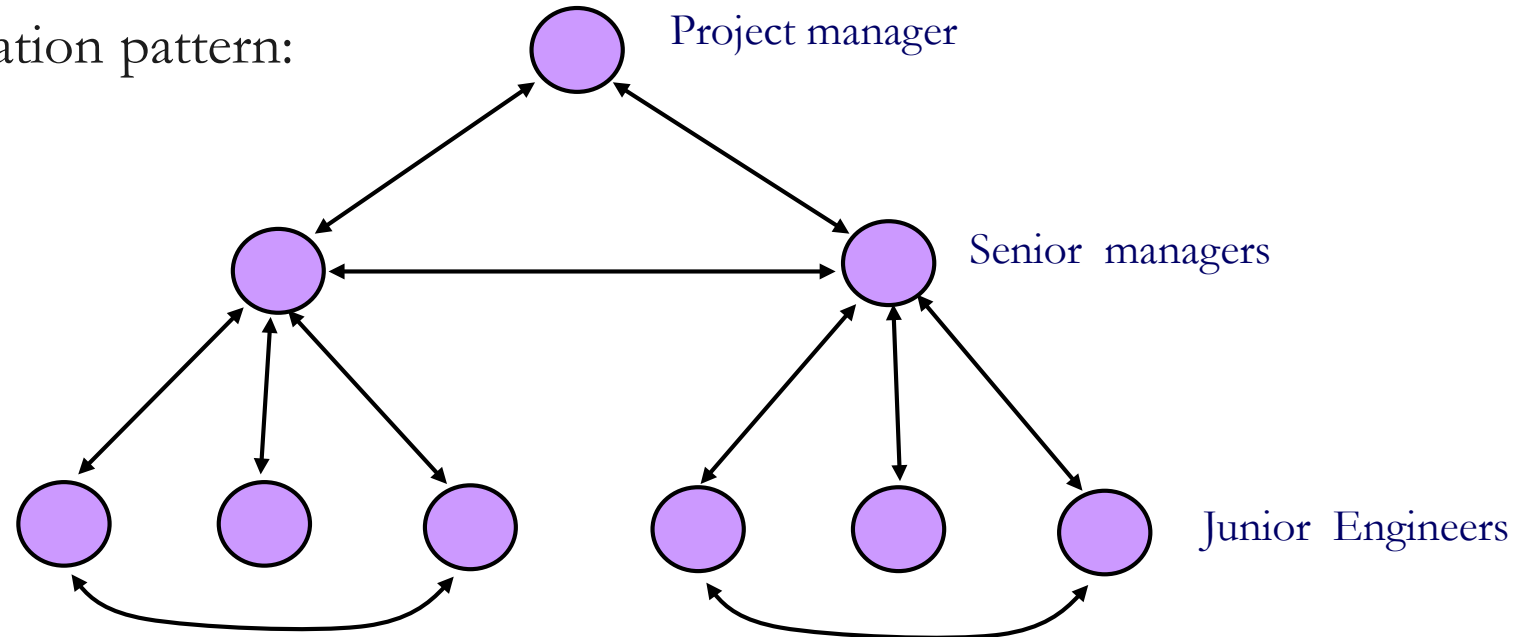
- No team organization is appropriate for all tasks.
- Decentralized control is best when communication among engineers is necessary.
- Centralized control is best when the speed of development is the most important goal and the problem is well understood.
- Limit the amount of communication to what is necessary.
- Consider goals other than speed of development
  - lower life cycle costs
  - reduced personnel turnover
  - development of junior engineers to senior
  - widespread dissemination of specialized knowledge and expertise.

f: Governance Model

Chief Officer
- Project Prioritization
- Program Review
- Program Control

IT Control Group

Business/Cust.
- BC Develop
- BC Request

Biz. Liaison Group
- BC Review
- IT Approval on BC
- Non-BC Control

PMO (Project Management Office)
- Project Tracking
- Data Gathering
- Status Reporting

SQA
- Quality Assurance

IT Develop / Maintenance Group

Development Team
- Analysis, Design, Implement,
- Unit Testing

Sub-Project
- Component Development

Su... •Co... •Co... •Co... •C...

XXX Project

Project Team X   **Project Team Y**

Other Projects

IT TWG

Architect
- Architecture Mentoring
- Architecture Verification

Testing Group
- System Testing
- Integration Testing Guide

Corp. CM Group
- Organization-level SCM

Risk Management

Typical management risks in software engineering:
- Changes in requirements.
- Not having the right people working in the project.

How to reduce risks:
- Prototyping
- Incremental delivery
- Modular design (to accommodate changes easily)

Risk Handling Form (example)

No	Risk	Critical.	Occur.	Mitigation Plan	Alternative	Respon.
1	Staff turnover	High	Mid	Reduce workload	Multi-Role Assign	J.E.Hong
2	Requirements change					
	Schedule delay					

Common Risks in Software Engineering Areas

Risk Items	Risk Management Techniques
Personnel shortfalls	Staffing with top talent; Job matching; Teambuilding; Key-personnel agreements; Cross training; Pre-Scheduling key person
Unrealistic schedule and budgets	Detailed multi-source cost & schedule estimation; Incremental development; Software reuse; Requirement scrubbing
Developing the wrong software functions	Organization analysis; Mission analysis; Operational concept formulation; User survey; Prototyping; Early user manuals
Gold plating	Requirements scrubbing; Prototyping; Cost benefit analysis; Design to cost
Continuing stream of requirements	High change threshold; Information hiding; Incremental development (defer changes to later incremental)

And any more ??

(Source, Boehm 1989)

Contents of SPMP

*The contents of document depends on the organization's standards

1. Introduction
- Purpose of this document
- Project overview
- Related documents, terms, abbreviations

2. Development Plan
- Resource : Staffing, Cost
- WBS
- Schedule (Gantte chart)

3. Organization
- Team structure
- Role and Responsibility

4. Technical Management
- Change management
- Configuration management
- Technology management

5. Quality control
- Review method
- Review periodic
- Other quality control techniques

6. Development Environment
- Required software and spec.
- Hardware spec.
- Space and security
- so on

7. Deliverables
- define the documents
- date and destination

8. Others
- Considerable issues

9. References and Appendix

Summary and Discussion

Project Management Steps

- Planning $\rightarrow$ Acquisition of resources $\rightarrow$ Execution $\rightarrow$ Monitoring

Software Productivity (or Size) Estimation

- LOC, FP

Project Control Techniques

- WBS, Gantt chart, PERT chart

Team Organization

Project Risk Management

Can effective project management improves the software quality ?