Project management

Project estimation: ✓
Estimate cost, duration & effort

Staff organization: Create a project team ✓

Project scheduling: Break down task —— assign task to team members. ✓

Monitoring & reporting progress.

Risk management identify, analysis ✓

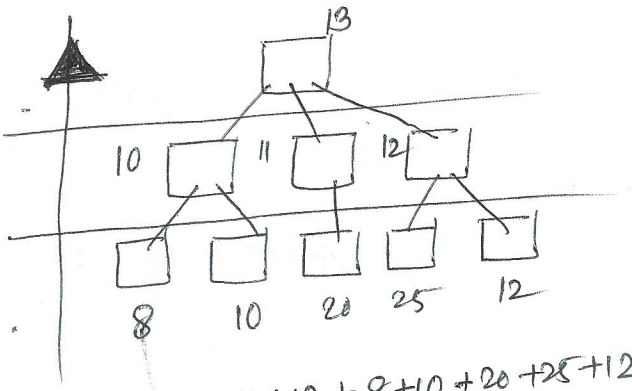
PROJECT ESTIMATION

Based on Project size / Problem size → [Effort, duration, cost] are estimated.

- Lines of code (LOC)
- Function point (FP) & Feature point

Lines of code (LOC)

Number of instructions in a source code
Ignore headers, comment etc.



LOC ← 18 + 10 + 11 + 12 + 8 + 10 + 20 + 25 + 12

Shortcomming of LOC

- Source code depends on programmer
- Depends on coding phase only + not considering challenges of other phase.
- Larger code size does not always imply better efficiency.
- LOC metric penalizes use of high level programming language, library inheritance etc.
- LOC Cannot identify complex logic, manual effort.
- Waits upto coding phase whereas development activity starts earlier.

Function point metric

Number of different functions/features supported by software

1. Compute unadjusted function point (UFP)
2. Compute technical complexity factor (TCF)
3. Compute total degree of influence (DI). [0-70]
   [0-6] transaction rate, [0-6] throughput, response time etc. [0-6]
   [0-6]
   $TCF \leftarrow 0.65 + 0.01 * DI$
4. $FP \leftarrow UFP * TCF$

DI

$UFP = 4 * \text{Number of inputs} + 5 * \text{Number of outputs}$
$+ 4 * \text{Number of inquiries (user's command)}$
$+ 10 * \text{Number of files (physical file or data structure)}$
$+ 10 * \text{Number of interfaces.}$

example.

	Factors	DI
1.		2
2.		3
3.		4
		0
14		1

DI → (40)

$0 \leq 40 \leq 70$ as $DI \in \{0, 1, \ldots 70\}$

$TCF \leftarrow 0.65 + 0.01 \times 0$ (DI ← 0)
$TCF \leftarrow 0.65$

$TCF \leftarrow 0.65 + 0.01 \times 70$
$TCF \leftarrow 1.35$ (DI ← 70)

Drawback

- Does not consider algorithmic complexity.
- For abstract data, ~~inputs~~ number of inputs & outputs can be subjective.

Feature point metric

Extension of function point metric
Takes algorithmic complexity into account.

Process based estimation

Function \ Activity	Customer communication	Risk analysis	Requirement analysis	Design	Coding	Testing	Total
							✓
f1							✓
f2							
f3							
							✓
fN							

Estimation with use case

	Use case	scenario	Pages	LOC
s1				
s2				
s3				

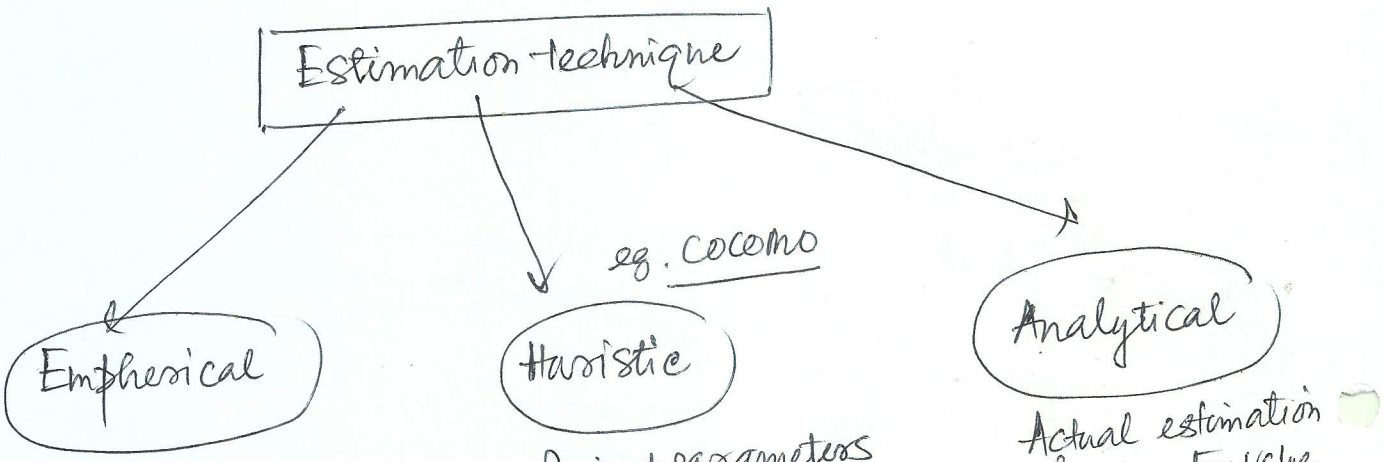
Software system ← {s1, s2, s3}

[Historical data]

	Scenario	Pages	LOC

Reconcilation estimate (Effort, Project duration, Cost)

Multiple estimates are combined to produce reconcilation estimate.



Educated guesswork
by expert
group of experts
a coordinator group of experts.

- Project parameters are modelled using mathematical expression.
- value of Constants are based on past experience

$$E = c_1 \times e^{d_1} + c_2 \times e^{d_2} + \cdots$$

Actual estimation of parameter value from source code.

COCOMO
(Constructive cost estimation model)

Type of project

1. Organic — Application (Program for data processing)
2. Semidetached — Utility (Compilers, linkers etc.)
3. Embeded — System program (operating system, realtime program etc.)

Development complexity = 1 : 3 : 9

Basic COCOMO

Based on product size

$$\text{Effort} = a_1 \times (\text{KLOC})^{a_2} \quad \text{PM (Person month)}$$

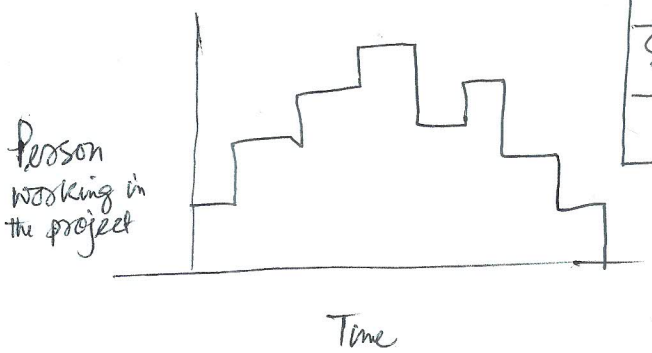
$$T_{dev} = b_1 \times (\text{Effort})^{b_2} \quad \text{month}$$

(Development time)

KLOC : Kilo line of Code

PM : Area under person month curve

	a_1	a_2	b_1	b_2
Organic	2.4	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32



Intermediate

Basic COCOMO is based on product size only.
But intermediate COCOMO is based on various cost drivers in addition to product size

These drivers are
- Complexity of the product
- Reliability requirement of the product
- Computer characteristics (execution speed, storage space etc.)
- Experience level of personnel
- Programming skill
- Development environment etc.

Complete COCOMO

A product may be made of application, utility & system program. Costs of these components are estimated separately and summed up to get overall costs of the system.

Staff organization

Organization structure

Big organization → Functional format

Team ← { FS, RE, D, C, M }

Small organization → Project format

Team ← { Project1, Project2 ..... , Project N }

Project-1	Project-2	...	Project-N
FS			
RE			
D			
C			
M			

Project format

Adv.

job rotation
+ handle skill shortage
+ domain ~~knowledge~~ efficiently export creation

Disadv.

- No job specialization

Functional format

Adv.

Ease of staffing + creating job specialization.

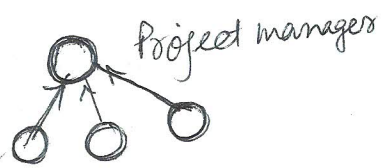
Disadv.

Problem of skill shortage

Team structure

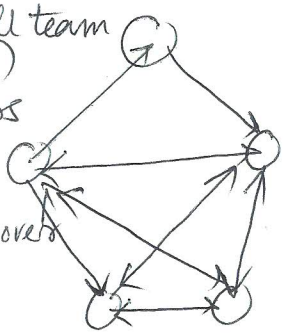
1. Chief programmers team / closed paradigm structure

Chief programmer partitions the task & assigns them to members. Then he verifies and integrates the task developed by members.



2. Democratic team

For small team (6-7)



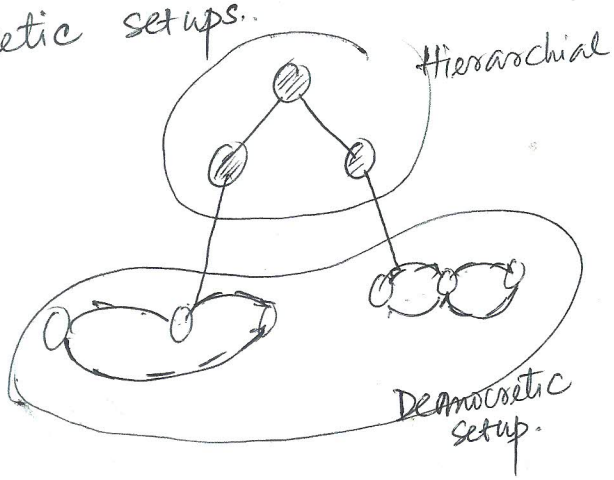
At different times different members provide leadership.

Adv. Suffers less from manpower turnover. Scope for better solution.

Disadv: Create chaos for large size project.

3. Mixed team

Hierarchial and democratic setups.



Factors behind team formation

1. Difficulty of the problem
2. Size of programs (in line of code / function point)
3. Team lifetime - time that team will stay together.
4. Degree to which problem can be modularized.
5. ~~Requirs~~ Quality requirement
6. Rigidity of delivery date
7. Degree of scalability
8. Environment

Skill of a Good software engineers

1. Exposure to software engineering principles
2. Domain knowledge (knowledge related to project work)
3. Programming ability
4. Oral, written & interpersonal skill
5. High motivation
6. Technical knowledge
7. Intelligence
8. Ability to work in a team
9. Discipline

Project scheduling

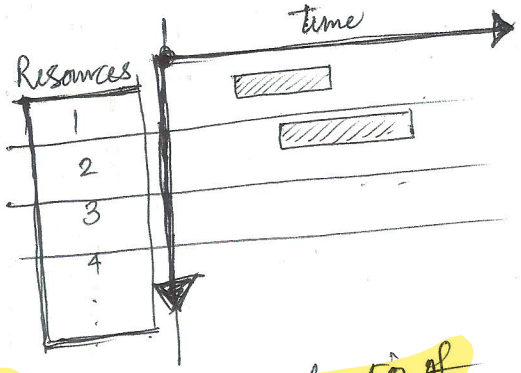
An activity in project planning
Schedule tasks in project

1. Identify tasks and break down large tasks into smaller activities. → Work breakdown structure

2. Determine dependancy among the tasks → Activity network / Task network

3. Estimate time duration needed to complete the activities.

4. Allocate resources to activities. → Grantt Chart
(Resource: staff, hardware, software)



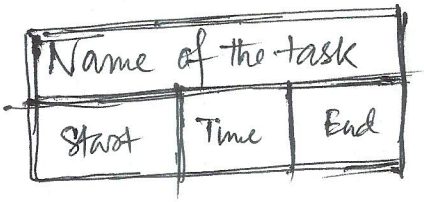
5. Plan starting and ending dates for various activities → PERT Chart

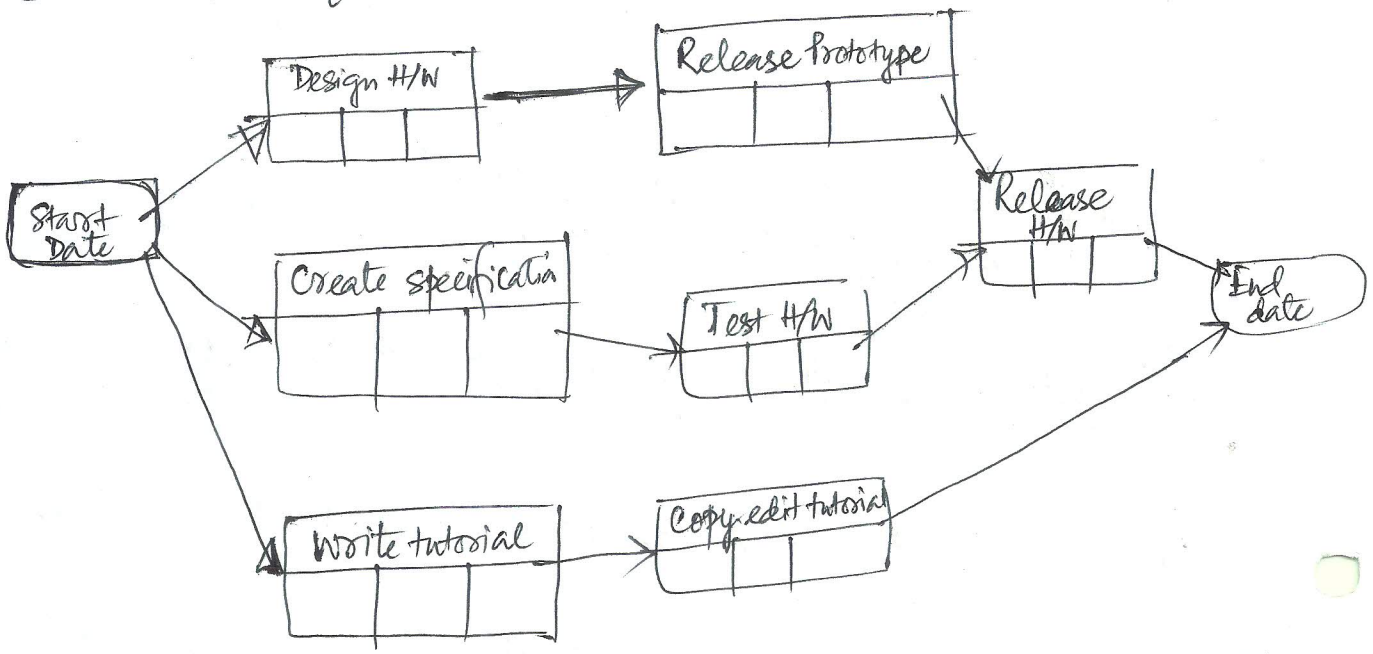
6. Determine critical path
(a chain of activities that determines duration of the project)
→ CPM (Critical path method)

Milestone
End of activity

PERT chart

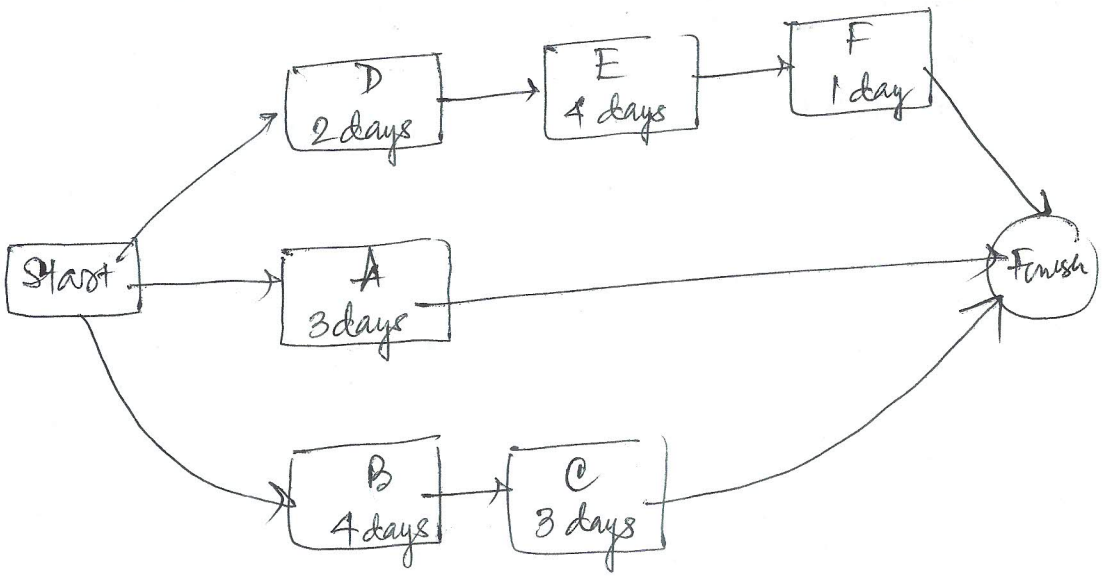
Project evaluation & Review technique





Example

Task ID	Task	Duration (day)
A	Create outline	1
B	Write draft	2
C	Edit & create final draft	4
D	Design positive visual	2
E	Add animation to visual	1
F	Upload post	



Calculate critical path

Step 1: Write down start time & end time next to each activity

First activity	Start time – 0	End time – Duration of first activity
Next activity	Start time – End of previous activity	End time – Start time + Duration

Do this for all activities.

Step 2: Find end time of last activity in each sequence

Step 3: Sequence of activities with longest duration is the critical path.

Max (3+2+4+1, 3 , 4+3) = 10 days.

Float or slack calculation

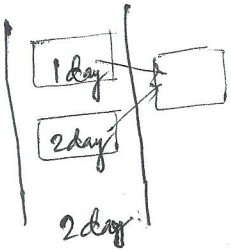
Amount of flexibility ~~given to~~ of a given task.

ES: Early start
EF: Early finish
LS: Late start
LF: Late finish

Free float

Two or more activities share common successor activity.
Next activity cannot start ~~in~~ until ~~activities~~ previous activities are completed.

ES (next task) – EF (current task)



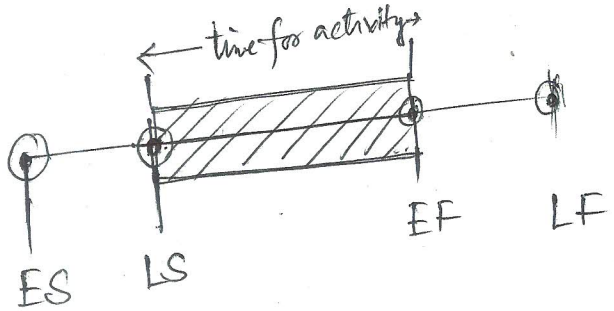
Free float = 2 – 1 day.

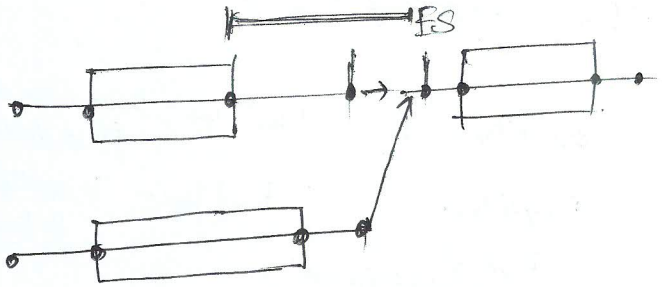
Total float

LS – ES

LF – EF

ES: Earliest date activity ~~can~~ can begin.




FS

Risk management

Risk ← Unfavourable event that hampers successful & timely completion of a project.

- Risk identification
- Risk projection
- Risk assessment
- Risk containment

Listing down risks.

Rank the risks in terms of causing damage.

Plan against the risk.

Types of risks

- **Project risk**
  Budgetary, schedule, personnel, customer-related problem
- **Technical risk**
  Design, implementation, interfacing testing, maintainace problem.
- **Business risk**
  Building excellent product which no one wants.

$p = r \times s$

p: priority of the risk
r: probability of the risk.
s: consequence associated with the risk.

- Avoid risk
  eg. discussion with customer
- Transfer the risk
  eg. outsourcing
- Risk reduction
  eg. new recruitement may be planned considering some employee may leave the job.

Risk leverages

$$\text{Risk leverage} = \frac{(\text{risk exposure before reduction} - \text{risk exposure after reduction})}{\text{cost of reduction}}$$

Risk information sheet (RIS) Maintained using database

Risk ID:	Date:	Probability:	Impact: eg. High

Description:

Context/Condition

Avoiding/Mitigation/Monitoring PROACTIVE PLAN

Management/Contingency plan REACTIVE PLAN

Current status

Risk table

Risk Mitigation, Monitoring & Management

Risks	Category	Probability	Impact	RMMM

[0-1] [1-4]

Impact assessment

Category / Components	Performance	Support	Cost	Schedule	Average Impact value
Catastrophic (R&D)					
Critical					
Marginal					
Negligible					