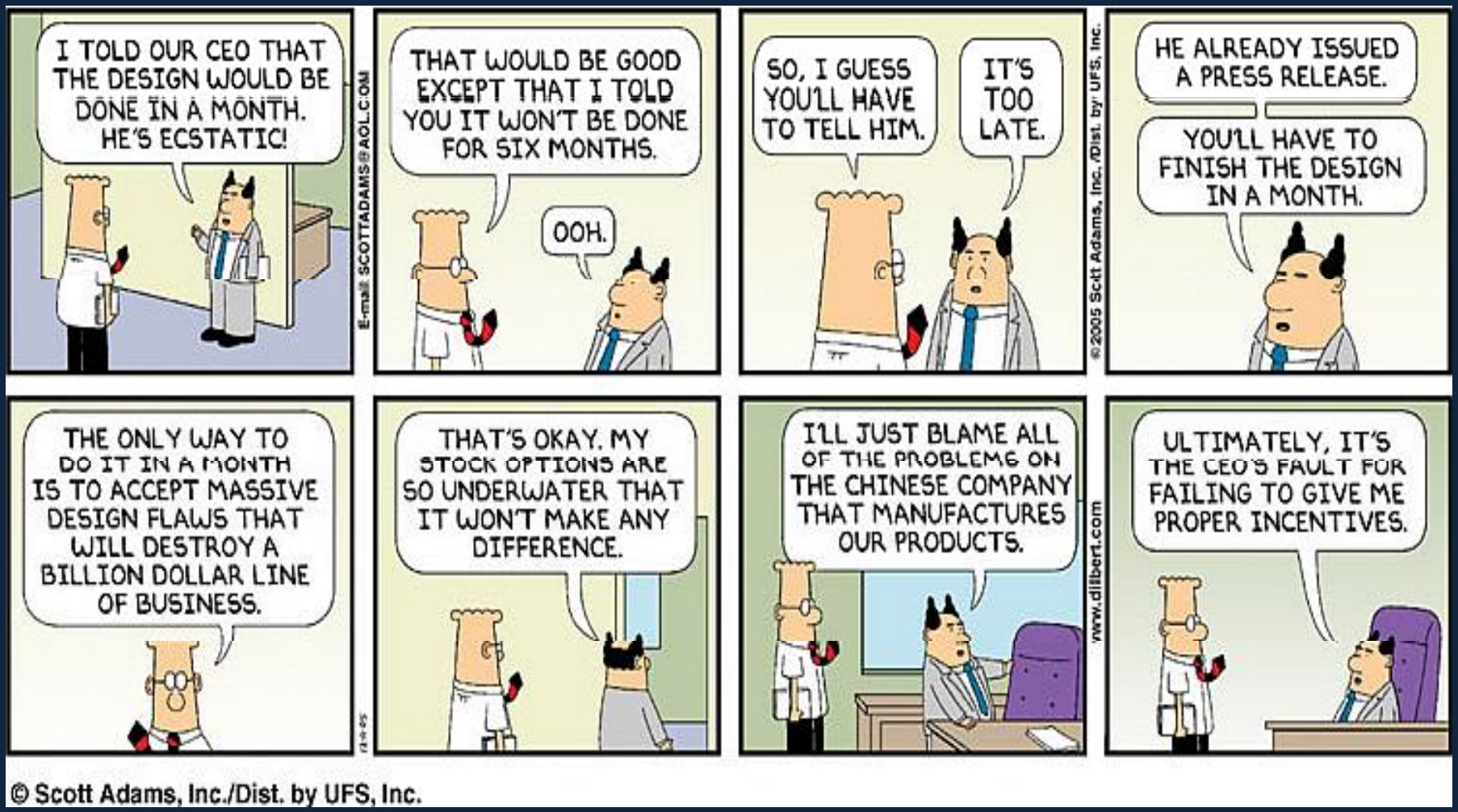
CS 123
Introduction to Software Engineering

04: Software Planning and Project Management

DISCS

SY 2013-2014

Planning


I TOLD OUR CEO THAT THE DESIGN WOULD BE DONE IN A MONTH. HE'S ECSTATIC!
E-mail: SCOTTADAMS@AOL.COM
THAT WOULD BE GOOD EXCEPT THAT I TOLD YOU IT WON'T BE DONE FOR SIX MONTHS.
OOH.
SO, I GUESS YOU'LL HAVE TO TELL HIM.
IT'S TOO LATE.
©2005 Scott Adams, Inc./Dist. by UFS, Inc.
HE ALREADY ISSUED A PRESS RELEASE.
YOU'LL HAVE TO FINISH THE DESIGN IN A MONTH.
THE ONLY WAY TO DO IT IN A MONTH IS TO ACCEPT MASSIVE DESIGN FLAWS THAT WILL DESTROY A BILLION DOLLAR LINE OF BUSINESS.
THAT'S OKAY. MY STOCK OPTIONS ARE SO UNDERWATER THAT IT WON'T MAKE ANY DIFFERENCE.
I'LL JUST BLAME ALL OF THE PROBLEMS ON THE CHINESE COMPANY THAT MANUFACTURES OUR PRODUCTS.
www.dilbert.com
ULTIMATELY, IT'S THE CEO'S FAULT FOR FAILING TO GIVE ME PROPER INCENTIVES.
© Scott Adams, Inc./Dist. by UFS, Inc.

Discussion

- Someone offers you a project to build their software. After explaining about the requirement, she asked:
  - How much does she need to pay you?
  - How long can you finish the project?
- How do you determine price and time duration for a project?
  - New project (no experience)
  - Similar to old project

Discussion

What are the consequences of wrong budget estimation?

- Over estimate budget
  - Cannot win the bid
  - Client feels software development is not worthy
- Under estimate budget
  - You lose money on project

Discussion

What are the consequences of wrong time estimation?

- Over estimate time
  - Clients will go elsewhere
- Under estimate time
  - Lose credibility
  - Get penalty (if clauses)
  - Manager need to explain a lot

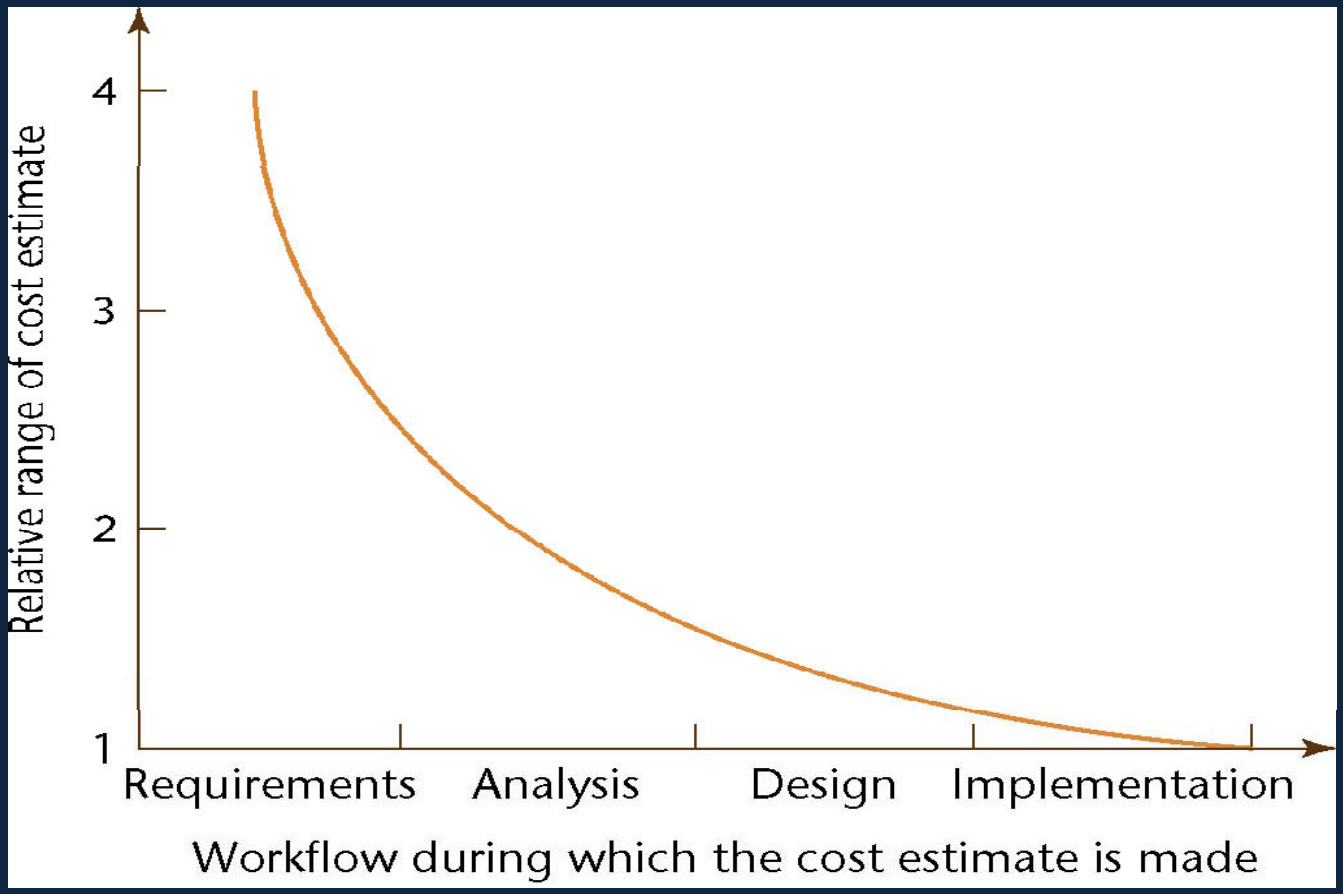
Overview

1. Planning and the software process
2. Break Even Point
3. Estimating duration and cost
4. Components of a software project management plan
5. Software project management plan framework
6. IEEE software project management plan
7. Project Scheduling
8. Planning testing
9. Planning object-oriented projects
10. Training requirements
11. Documentation standards
12. CASE tools for planning and estimating
13. Testing the software project management plan

Planning and Estimating

- Today‘s class is addressed more for managers & owners, not engineers or programmers
- Before starting to build software, it is essential to plan the entire development effort in detail
- Planning continues during development and then post-delivery maintenance
  - Initial planning is not enough
  - Planning must proceed throughout the project
  - The earliest possible time that detailed planning can take place is after the **specifications** are complete

1. Planning and the Software Process



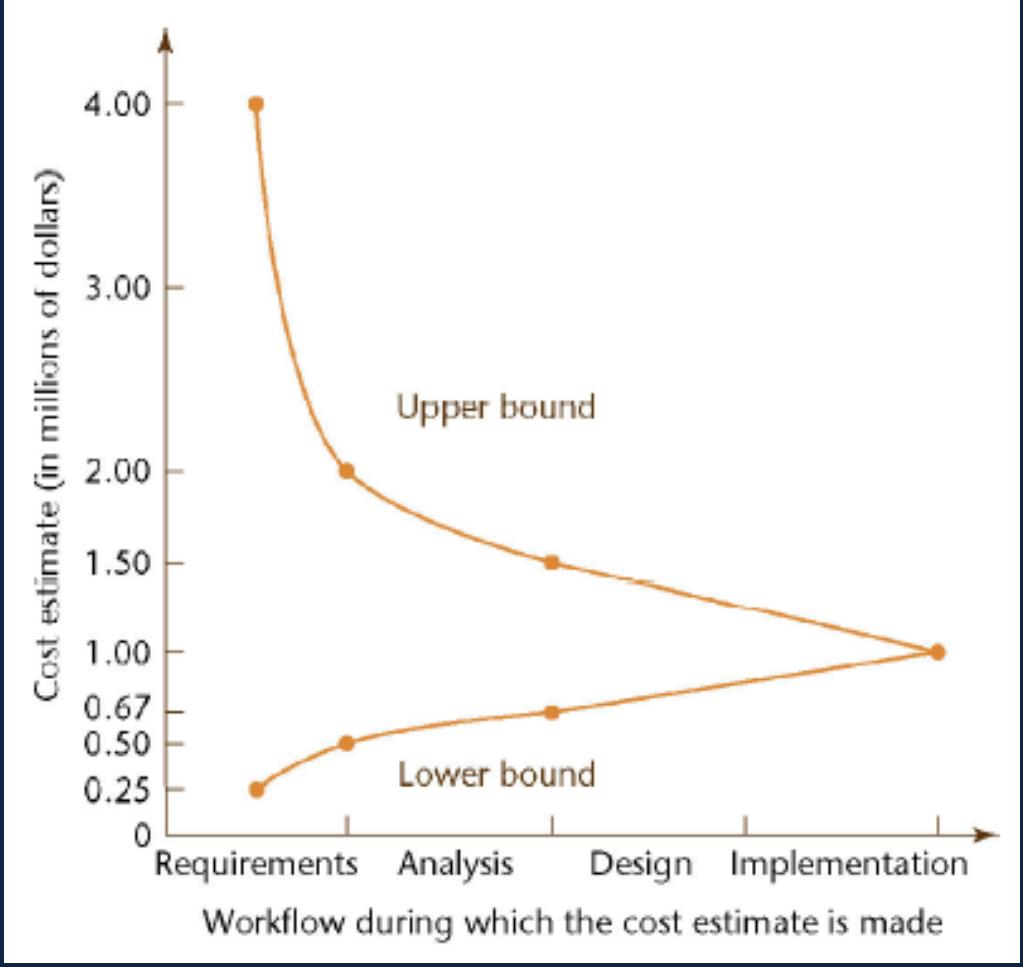
The accuracy of estimation increases as the process proceeds

1. Planning and the Software Process

- Example
  - Cost estimate of $1 million during the requirements workflow
    - Likely actual cost is in the range ($0.25M, $4M)
  - Cost estimate of $1 million at the end of the requirements workflow
    - Likely actual cost is in the range ($0.5M, $2M)
  - Cost estimate of $1 million at the end of the analysis workflow (earliest appropriate time)
    - Likely actual cost is in the range ($0.67M, $1.5M)

1. Planning and the Software Process

These four points are shown in the cone of uncertainty

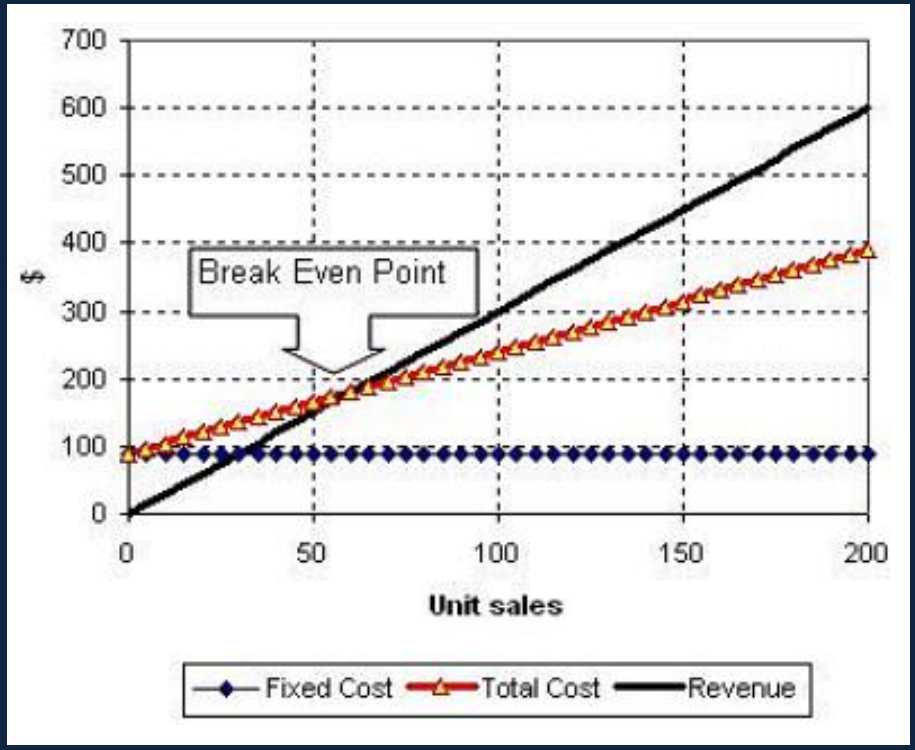


1. Planning and the Software Process

- This model is old (1976)
  - Estimating techniques have improved
  - But the shape of the curve is likely to be similar

2. BEP

- Fixed cost vs Variable cost
- Break even point (BEP) is the point where the revenue is equal to total cost



Fixed Cost

- What are the fixed cost items in a Software Engineering project?
  - Basic Salary of employees
  - Office rent
  - Computers & Software (e.g. IDE)
  - Utilities (electricity, water, AC)
  - Management cost (secretary, boss)
  - ....

Variable cost

- What are the variable cost items in a Software Engineering project?
  - Incentive per project
  - Progress based salary
    - Coding
    - Testing

BEP Formulas

- n = number of unit
- v = variable cost/unit
- V = variable cost = v . n
- F = fixed cost
- T = Total cost = V + F
- p = price/unit
- R = Revenue = n. p
- BEP if T = R, or v . n + F = n . p
- BEP unit n* = F / (p - v)
- Break even price BEP = (p. F) / (p - v)

HW: BEP

- You own a small software firm with 5 engineers who do all around jobs from requirements, analysis, design, coding, testing, and maintenance and 1 secretary. You do the management stuff & marketing.
- The basic monthly salary are P30,000 for each engineer, P60,000 for you and P20,000 for the secretary. Assume 13 months/year. You give incentive P20,000 for all engineers if a project is accomplished
- Suppose office rent, utilities & other fixed cost (aside from salaries) are P100,000 per month (12 months per year).
- If average project price is P1,000,000/Project, how many projects should you get per year to get at least Break-Even Point?
- If your team can only accomplish 1 project per year, how much is the minimum price of that project to be at least break-even?

HW: Answer

	A	B	C
1	Break-Even Point		
2			
3	Number of Engineers	5	
4	Number of Management	1	
5	Number of Secretary	1	
6	Basic Salary of an Engineer	30,000.00	
7	Management Salary	60,000.00	
8	Salary of Secretary	20,000.00	
9	Rent & Utilities & Others	100,000.00	
10	Price/ Project	1,000,000.00	
11	Variable Cost/ Project	20,000.00	
12			
13	Total Yearly Salary	2,990,000.00	
14	Total Yearly Fixed Cost	4,190,000.00	
15			
16	BEP: Project Per Year	4.28	
17	Break-Even Point	4,275,510.20	
18			
19	Find the price such that BEP will be 1 project/ year		
20	Answer	4,209,732.01	
21			
22			

Number of Projects	Fixed Cost	Variable Cost	Total Cost	Revenue	Profit		
0	4,190,000.00	-	4,190,000.00	-	(4,190,000.00)	0	-4190000
1	4,190,000.00	20,000.00	4,210,000.00	1,000,000.00	(3,210,000.00)	1000000	-3210000
2	4,190,000.00	40,000.00	4,230,000.00	2,000,000.00	(2,230,000.00)	2000000	-2230000
3	4,190,000.00	60,000.00	4,250,000.00	3,000,000.00	(1,250,000.00)	3000000	-1250000
4	4,190,000.00	80,000.00	4,270,000.00	4,000,000.00	(270,000.00)	4000000	-270000
5	4,190,000.00	100,000.00	4,290,000.00	5,000,000.00	710,000.00	5000000	710000
6	4,190,000.00	120,000.00	4,310,000.00	6,000,000.00	1,690,000.00	6000000	1690000
7	4,190,000.00	140,000.00	4,330,000.00	7,000,000.00	2,670,000.00	7000000	2670000

Use goal-seek to find the answer

Break Even Price Analysis

- Reduce fixed cost as much as possible and exchange it into variable cost
  - Increase morale of employees
  - Increase overall profit
  - Productivity based wages

3. Estimating Duration and Cost

- Accurate duration estimation is critical
- Accurate cost estimation is critical
  - **Internal cost**: cost to the developer team, manager, personnel, hardware, software, utilities, rent
  - **External costs**: price that the client will pay
- There are too many variables for accurate estimate of cost or duration

How Not to Estimate Cost

- Work fill the time available:
  - You were given 12 months to do the job, so it will take 12 months
- Price to win:
  - We know our competitor put in bid of $1M, so we need to schedule a bid of $0.9M
- Budget method:
  - We need to show our product in trade show next year so the software need to be finished by 9 months though we realize that it is rather tight
- Settlement of bargain:
  - Actually the project need 1 year but my boss does not accept more than 10 months, so we settle for 10 months

Human Factors

- Sackman (1968) measured differences of up to 28 to 1 between pairs of programmers
  - He compared matched pairs of programmers (experience vs beginners) with respect to
    - Product size: 6:1
    - Product execution time: 8:1
    - Development time: 9:1
    - Coding time: 18:1
    - Debugging time: 28:1
- Presence of 1 or 2 very good (or very bad) team members can cause mark deviation from schedule and budget
- Conclusion: Human factor greatly affect accuracy of estimation (it‘s worth to pay much higher to experience programmer)

Discussion

- If you know the “size” of your project then you can use your previous cost and time to estimate the next project (i.e. cost/size, time/size etc). Do you have any idea on how to measure the “size” of your software?

3.1 Metrics for the Size of a Product

- Lines of code (LOC, KDSI, KLOC)
- FFP
- Function Points

3.1.1. Lines of Code (LOC)

- Simplest method but not reliable
- Metric
  - Thousand delivered source instructions (KDSI)
  - Thousand of Line of Code (KLOC)
- Example:
- Effort = 3.2 KLOC ^1.05 (in person month)

3.1.1. Lines of Code (LOC)

- Source code is only a small part of the total software effort
- Different languages lead to different lengths of code
- LOC is not defined for nonprocedural languages (like LISP)

3.1.1. Lines of Code (LOC)

- It is not clear how to count lines of code
  - Executable lines of code?
  - Data definitions?
  - Comments? If not, programmers might choose not put comments (bad for maintenance)
  - Changed/deleted lines? (refactor/enhancement)
- Not everything written is delivered to the client
- A report, screen, or GUI generator can generate thousands of lines of code in minutes

3.1.1. Lines of Code (LOC)

- LOC is accurately known only when the product finished
- Estimation based on LOC only is therefore doubly dangerous
  - To start the estimation process, LOC in the finished product must be estimated
  - The LOC estimate is then used to estimate the cost of the product — an uncertain input to an uncertain cost estimator

3.1 Metrics for the Size of a Product

- Metrics based on measurable quantities that can be determined early in the software life cycle
  - FFP (File-Flow-Process)
  - Function points

3.1.2. FFP Metric

- For cost estimation of medium-scale data processing products
- The three basic structural elements of data processing products
  - Files: logical & physical permanent transaction records
  - Flows: data interface between product & environment such as screen or report
  - Processes: logical or arithmetic manipulation of data (sort, validation, update etc.)

3.1.2. FFP Metric

- Given the number of files (Fi), flows (Fl), and processes (Pr)
  - The size (S), cost (C) are given by

    $S = Fi + Fl + Pr$

    $C = b \times S$
- The constant b (efficiency or productivity) varies from organization to organization (from previously developed similar projects by that organization)

3.1.2. FFP Metric

- The validity and reliability of the FFP metric were demonstrated using a purposive sample
  - However, the metric was never extended to include databases

3.1.3. Function Points

- Based on the number of
  - **Input types** (Inp): user input that results in change of data structure
  - **Output types** (Out)
  - **inquiries types** (Inq): inputs that control the execution of the program does not change internal data structure (e.g. menu, query criteria)
  - **master files** (Maf): internal data generated and used by the system (e.g. index file)
  - **interfaces** (Inf): output to another applications or shared with other applications

Over Simplified Function Point

- For any product, the size in “function points” is given by

$$\mathbf{FP = 4 \times Inp + 5 \times Out + 4 \times Inq + 10 \times Maf + 7 \times Inf}$$

- This is an oversimplification of a 3-step process

Analysis of Function Points

- Like FFP, maintenance can be inaccurately measured
- It is possible to make major changes without changing
  - The number of files, flows, and processes; or
  - The number of inputs, outputs, inquiries, master files, and interfaces
- In theory, it is possible to change every line of code with changing the number of lines of code

3.2 Techniques of Cost Estimation

- Expert judgment by analogy
- Bottom-up approach

3.2.1. Expert Judgment by Analogy

- Experts compare the target product to completed products
  - Guesses can lead to hopelessly incorrect cost estimates
  - Experts may recollect completed products inaccurately
  - Human experts have biases
  - However, the results of estimation by a broad group of experts may be accurate
- The Delphi technique is sometimes needed to achieve consensus

Delphi Method

- First round: all individuals are asked a series of questions and the results are gathered together.
- The aggregate results are fed back to each member of the panel
- Second round: each individual are asked if they wish to change their forecast
- Iterative up to several rounds until no individual change their forecast or level of general agreement exist

Delphi Practice

- Pretend that each of you is the expert
- Do several iteration of Delphi method to estimate a budget of software development of a web site for children educational games

3.2.2. Bottom-up Approach

- Break the product into smaller components
  - The smaller components may be no easier to estimate
  - However, there are process-level costs
- When using the object-oriented paradigm
  - The independence of the classes assists here
  - However, the interactions among the classes complicate the estimation process
- Weakness: a product is more than the sum of its components

4. Components of a Software Project Management Plan

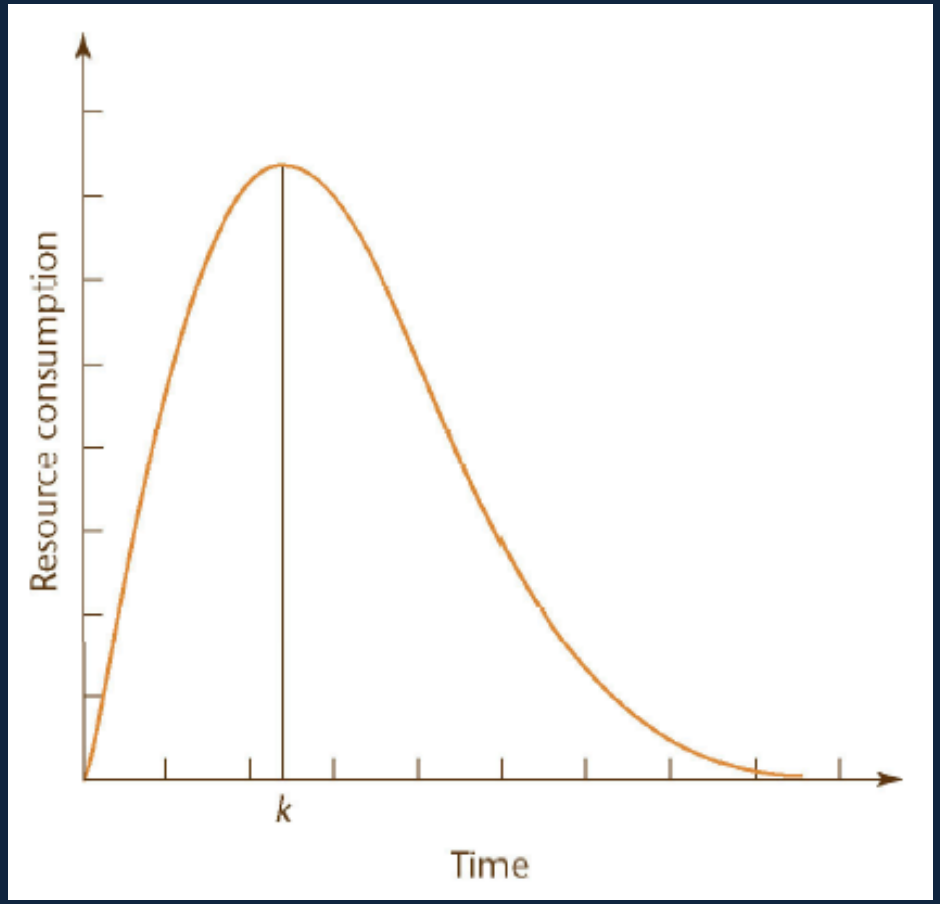
1. The work to be done
2. The resources with which to do it
3. The money to pay for it

Resources

- Resources needed for software development:
  - People
  - Hardware
  - Support software
  - Office facilities
  - Travel

Use of Resources Varies with Time

- Rayleigh curves accurately depict resource consumption
- The entire software development plan must be a function of time
- Effort = 0.3945 K
- K = total man power required = area under Rayleigh curve
- Close to rule of thumb: 40% for development & 60% for maintenance



Work Categories

- Three Work Categories

1. Project function (throughout project)
2. Activity (major units of work)
3. Task (minor units of work)

Work Categories

1. Project function
   - Work carried on throughout the project
   - Not related to specific workflow of software development
   - Examples:
     - Project management
     - Quality control

Work Categories

2. Activity
   - Work that relates to a specific phase
   - A major unit of work,
   - With precise beginning and ending dates,
   - That consumes resources, and
   - Results in work products like the budget, design, schedules, source code, or users' manual

Work Categories

3. Task
   - An activity comprises a set of tasks (the smallest unit of work subject to management accountability)

Completion of Work Products

- Milestone: The date on which the work product is to be completed
  - It must first pass reviews performed by
    - Fellow team members
    - Management
    - The client
- Once the work product has been reviewed and agreed upon, it becomes a baseline

Work Package

- Work product, plus
  - Staffing requirements
  - Duration
  - Resources
  - The name of the responsible individual
  - Acceptance criteria for the work product
  - The detailed budget as a function of time, allocated to
    - Project functions
    - Activities

5. Software Project Management Plan Framework

- There are many ways to construct an SPMP
- One of the best is IEEE Standard 1058.1
  - The standard is widely accepted
  - It is designed for use with all types of software products
  - It supports process improvement
  - Many sections reflect CMM key process areas
  - It is ideal for the Unified Process
    - There are sections for requirements control and risk management
  - Some of the sections are inapplicable to small-scale software
    - Example: Subcontractor management plan

IEEE Software Project Management Plan

1 Overview
- 1.1 Project summary
  - 1.1.1 Purpose, scope, and objectives
  - 1.1.2 Assumptions and constraints
  - 1.1.3 Project deliverables
  - 1.1.4 Schedule and budget summary
- 1.2 Evolution of the project management plan

2 Reference materials

3 Definitions and acronyms

4 Project organization
- 4.1 External interfaces
- 4.2 Internal structure
- 4.3 Roles and responsibilities

5 Managerial process plans
- 5.1 Start-up plan
  - 5.1.1 Estimation plan
  - 5.1.2 Staffing plan
  - 5.1.3 Resource acquisition plan
  - 5.1.4 Project staff training plan
- 5.2 Work plan
  - 5.2.1 Work activities
  - 5.2.2 Schedule allocation
  - 5.2.3 Resource allocation
  - 5.2.4 Budget allocation
- 5.3 Control plan
  - 5.3.1 Requirements control plan
  - 5.3.2 Schedule control plan
  - 5.3.3 Budget control plan
  - 5.3.4 Quality control plan
  - 5.3.5 Reporting plan
  - 5.3.6 Metrics collection plan
- 5.4 Risk management plan
- 5.5 Project close-out plan

6 Technical process plans
- 6.1 Process model
- 6.2 Methods, tools, and techniques
- 6.3 Infrastructure plan
- 6.4 Product acceptance plan

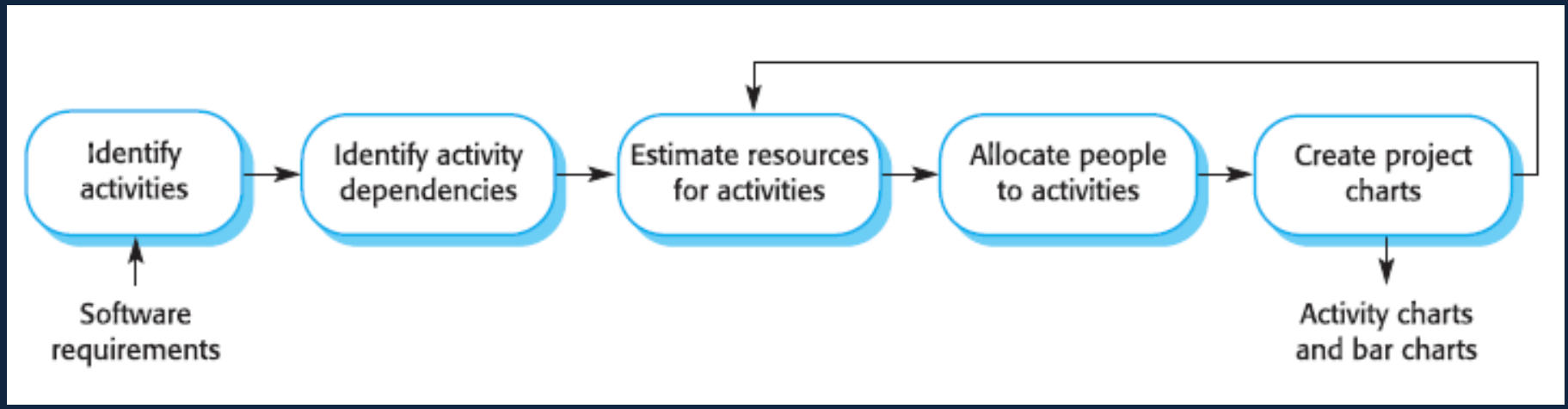
7 Supporting process plans
- 7.1 Configuration management plan
- 7.2 Testing plan
- 7.3 Documentation plan
- 7.4 Quality assurance plan
- 7.5 Reviews and audits plan
- 7.6 Problem resolution plan
- 7.7 Subcontractor management plan
- 7.8 Process improvement plan

8 Additional plans

Project scheduling

- A project usually consists of many activities or tasks.
- Each activity needs some time duration to accomplish.
- Some activity can only be performed after one or several other activities have been completed.
- This creates dependency between activities that can be modelled as a network

The project scheduling process



Project scheduling

- Split project into activities or tasks and estimate time and resources required to complete each activity.
- Organize activities concurrently to make optimal use of workforce.
- Minimize activity dependencies to avoid delays caused by one task waiting for another to complete.
- Dependent on project managers intuition and experience.

Scheduling problems

- Estimating the difficulty of problems and hence the cost of developing a solution is hard.
- Productivity is not proportional to the number of people working on a task.
- Adding people to a late project makes it later because of communication overheads.
- The unexpected always happens. Always allow contingency in planning.

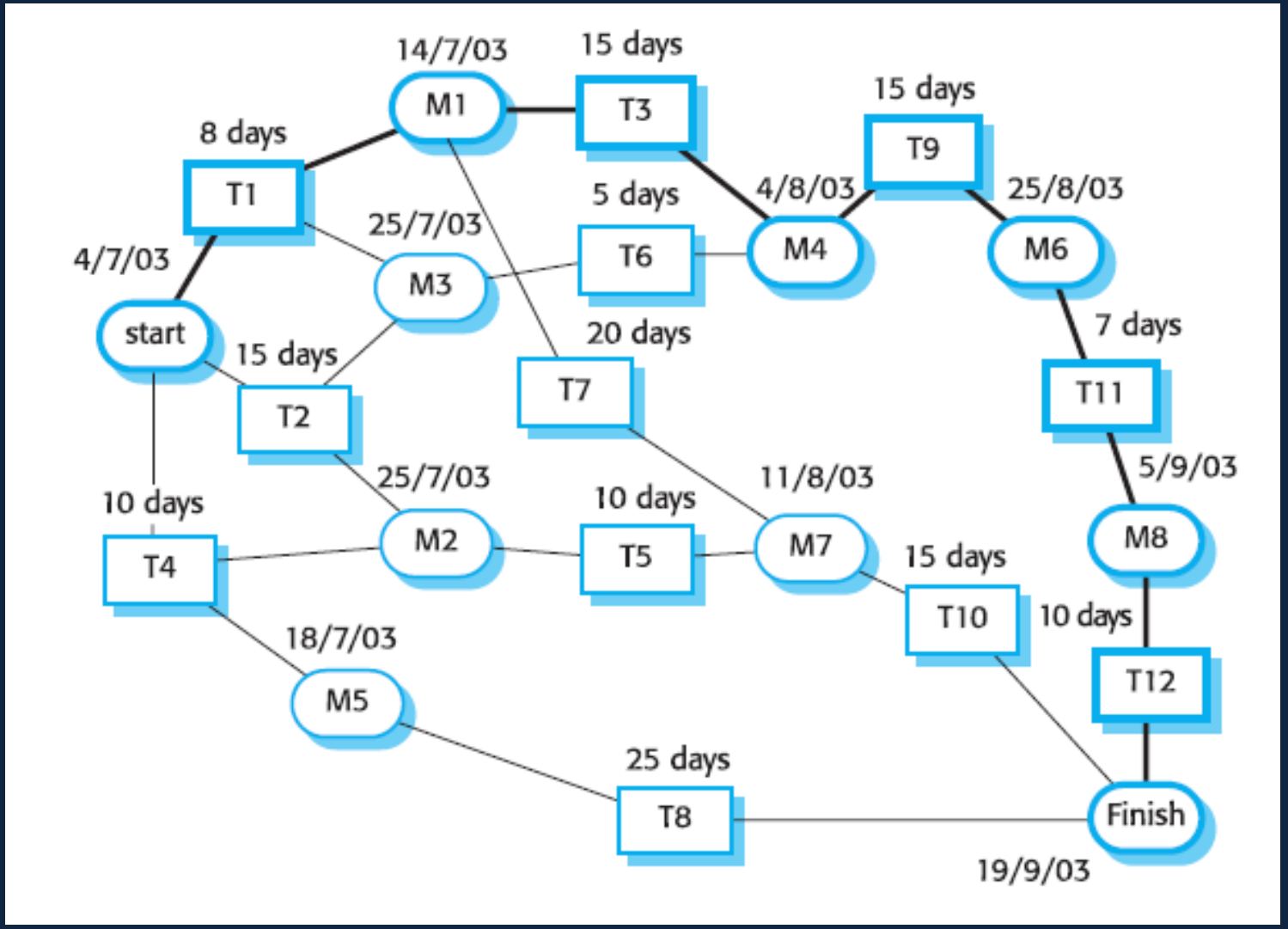
Bar charts and activity networks

- Graphical notations used to illustrate the project schedule.
- Show project breakdown into tasks. Tasks should not be too small. They should take about a week or two.
- Activity charts show task dependencies and the critical path.
- Bar charts show schedule against calendar time.

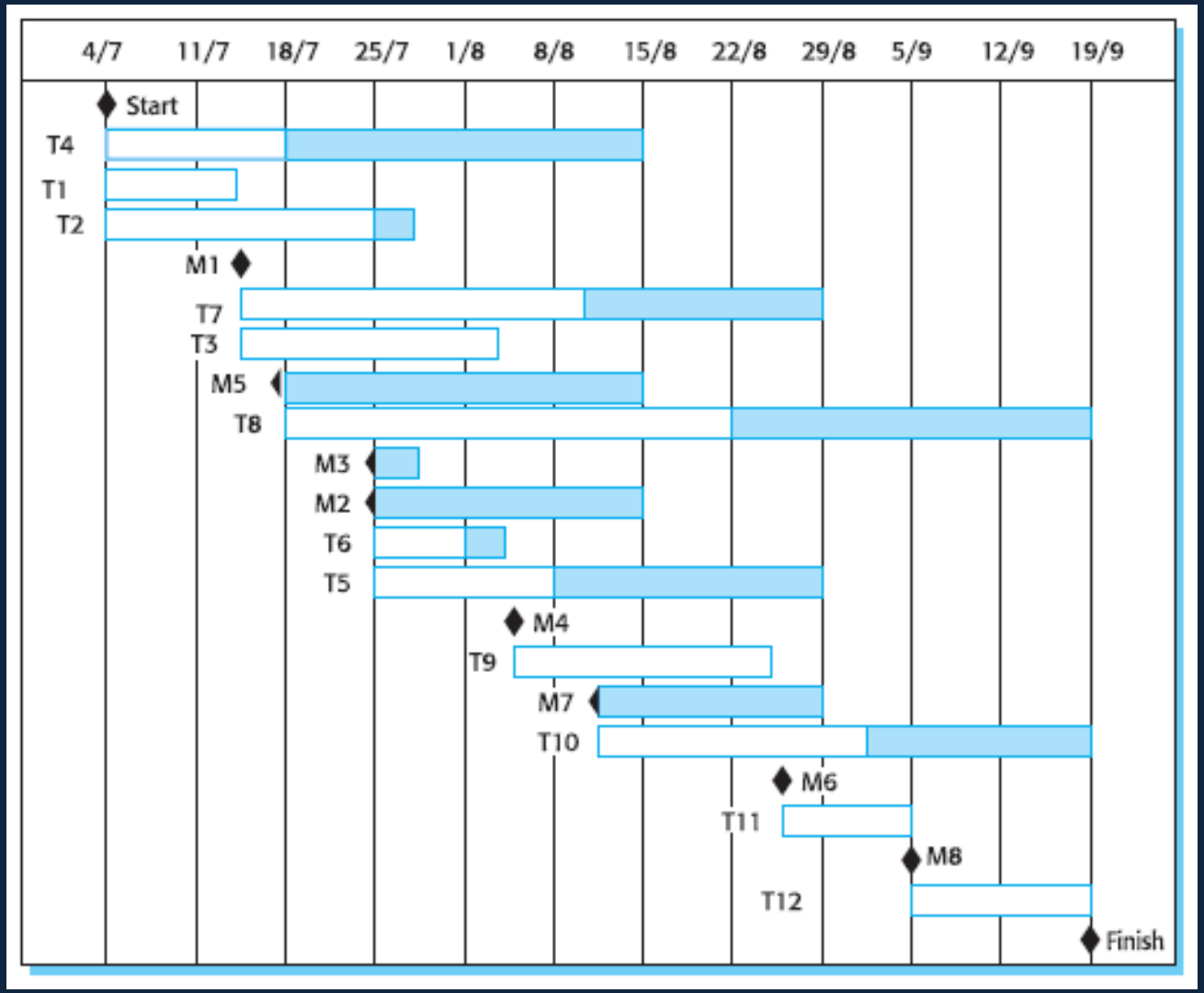
Task durations and dependencies

Activity	Duration (days)	Dependencies
T1	8	
T2	15	
T3	15	T1 (M1)
T4	10	
T5	10	T2, T4 (M2)
T6	5	T1, T2 (M3)
T7	20	T1 (M1)
T8	25	T4 (M5)
T9	15	T3, T6 (M4)
T10	15	T5, T7 (M7)
T11	7	T9 (M6)
T12	10	T11 (M8)

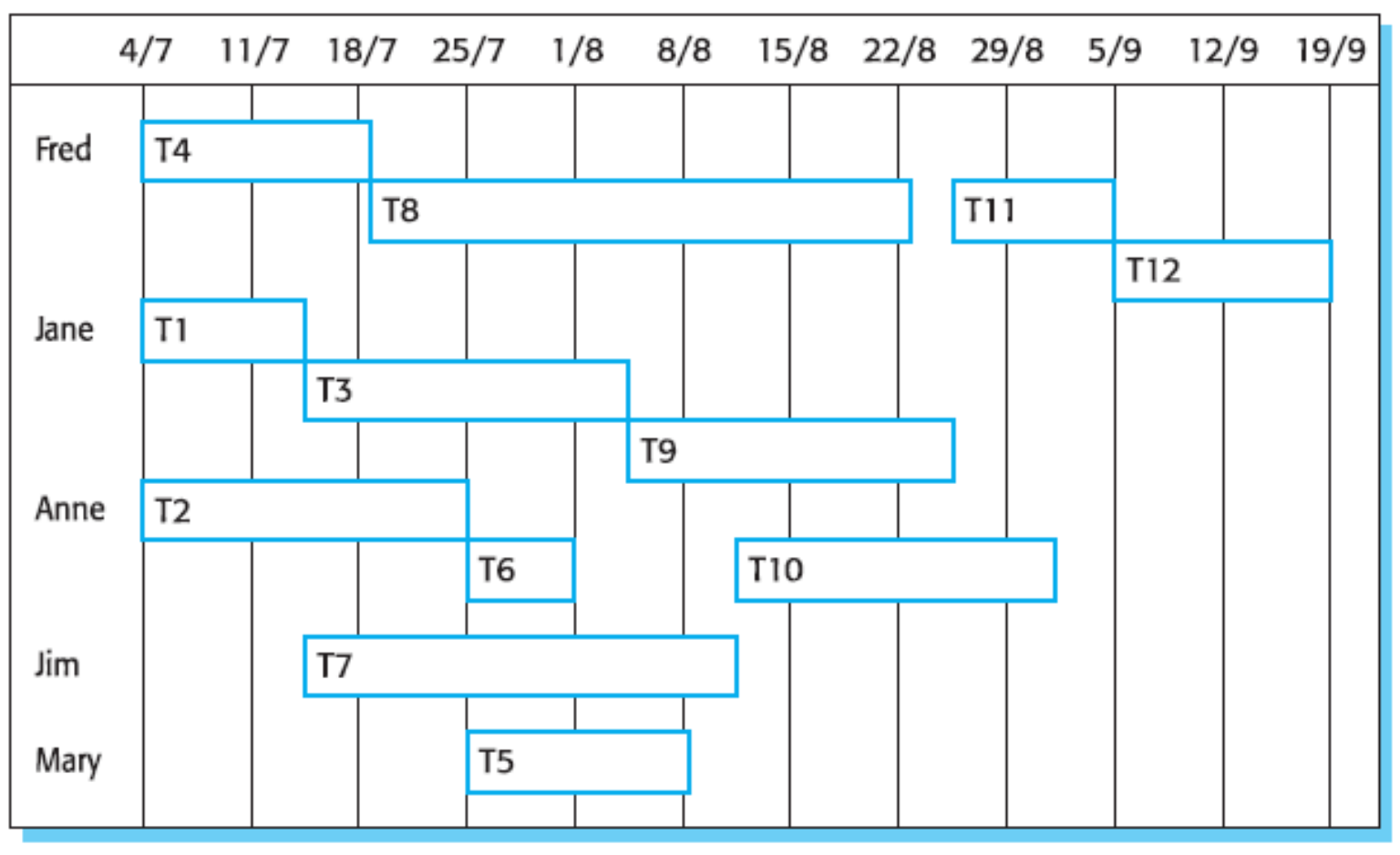
Activity Network



Activity Timeline



Staff Allocation



CPM

- Critical Path Method or CPM is a project management technique to schedule activities within a project.
- A project can be broken down into activities
- Each activity requires some amount of time to be accomplished, Tk
- Each activity is represented by a node in a project network

Activities Dependency

- Dependency between activities is modelled through precedence relationship.
- Precedence relationship between activities is represented by a directed link between nodes.
- When activity A is the immediate predecessor of activity B, we draw a link from node A to node B. In this case, node B is called immediate successor of activity A.

Goal of CPM

- The goal of CPM is to determine
  - How much time is needed for the project to be accomplished?
  - When each activity must be scheduled to begin?
  - Which activities are critical that must be performed exactly as schedule to keep the project on schedule?
  - How long the non critical activities can be delayed without affecting project completion time?

Assumptions

- A project network must have only a single start activity and a single final activity.
- When the project starts or ends at several activities, we must create dummy start node or dummy final node to compute the CPM

4 Variables of CPM

- Given the time duration for each activity k has to finish, T_k, to compute CPM, we need to define 4 variables:
  - Earliest possible start time for activity k is denoted by ES_k
  - Earliest possible finish time for activity k is denoted by EF_k
  - Latest possible start time for activity k is denoted by LS_k
  - Latest possible finish time for activity k is denoted by LF_k

CPM computation

- Once we develop our network, the manual computation of CPM has three steps
  - Forward pass: to determine earliest start and finish time of each activity
  - Backward pass: to determine latest start and finish time of each activity
  - Critical path is determined from the set of nodessequence that has zero slack

Forward Pass

- The start activity node has time equal to zero that is the time where the project begins. Then we compute forward from the start activity node until the final activity node.
- The earliest time to finish an activity is equal to the earliest start time plus the time duration of the activity.

$$EF_k = ES_k + T_k$$

- The earliest time to start an activity is equal to the maximum earliest finish time of the activities that directly preceding it

$$ES_k = \max \{Ef_{j \rightarrow k}\}$$

- Time needed for the project can be accomplished is equal to the earliest finish time of the final activity node in the project.

Backwad Pass

- To perform backward pass, first we set the latest finish time of the final activity node equal to the earliest finish time of the same node. Then we compute backward from the final activity node until the start activity node.
- The latest start time of an activity is equal to the latest finish time minus the time duration of the activity.

$$LS_k = LF_k - T_k$$

- The latest finish time of an activity is equal to the min latest start time of the activities that directly following or succeeding it

$$LF_k = \min \{LS_{k \rightarrow j}\}$$

- To check the correctness of the computation, the latest start time of the start node must equal to zero.

Critical Path & Slack

- The key computation of CPM is to determine the critical path. Critical path is set of nodes sequence that has zero slack. Slack is the amount of time of that the starting time of an activity can be delayed without affecting the overall project schedule. Slack of an activity is equal to

$$Slack_k = LS_k - ES_k$$

- Or equivalently,

$$Slack_k = LF_k - EF_k$$

- Zero slack activity means the activity must be started immediately without delay. Thus, critical path is the longest path through a project network.

Critical Path

- The nodes on the critical path must be identified as critical activities that must be put into attention because any delay on these activities will delay the project schedule.
- Nodes that are not on the critical path are called non-critical activities, they have positive slack.
- A project network has at least one critical path and it can have more than one critical paths

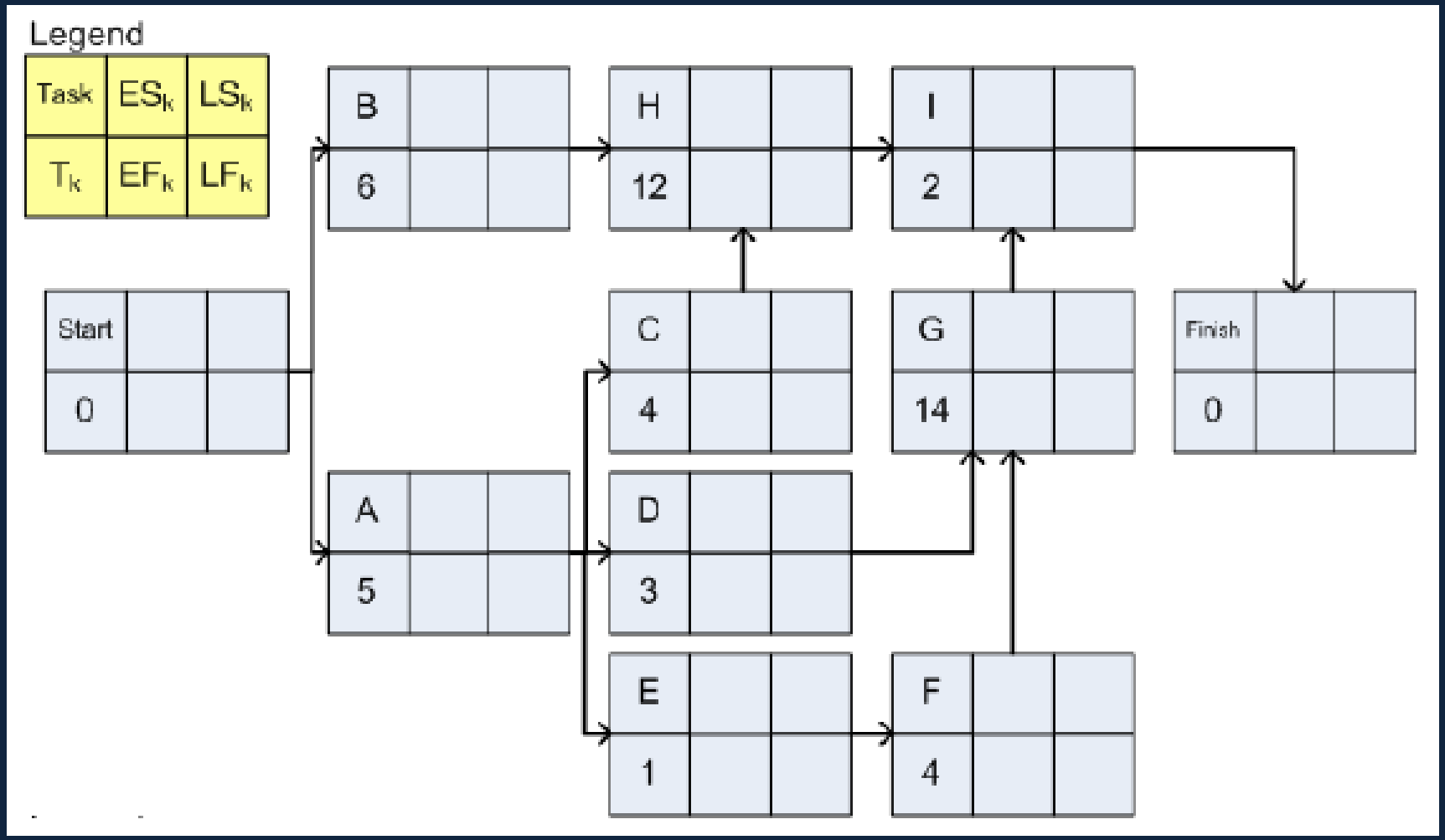
Practice

Compute CPM for the following activities

Activity	Immediate predecessor	Activity Time
A	-	5
B	-	6
C	A	4
D	A	3
E	A	1
F	E	4
G	D, F	14
H	B, C	12
I	G, H	2

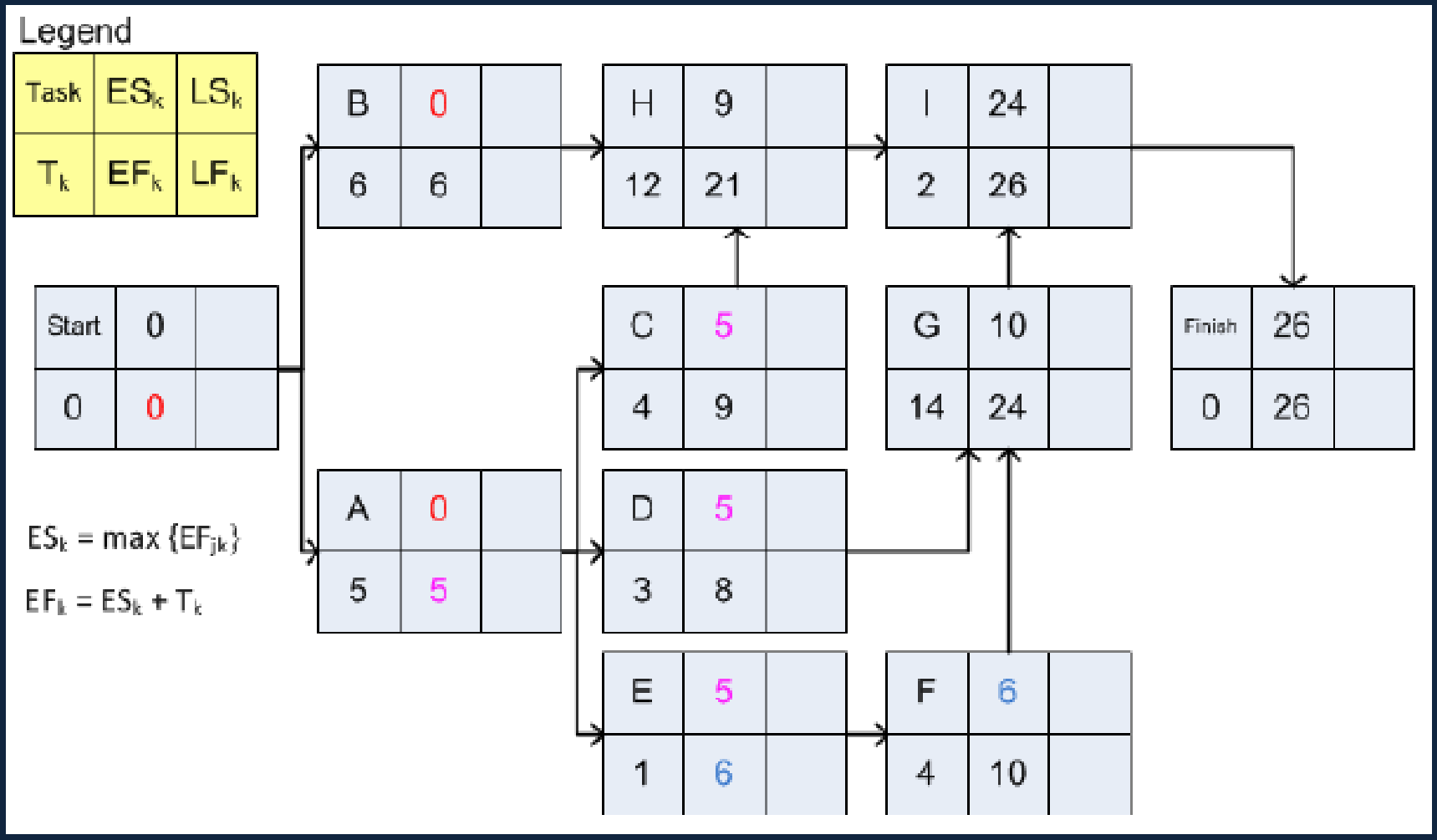
Answer

1. Prepare your network and form



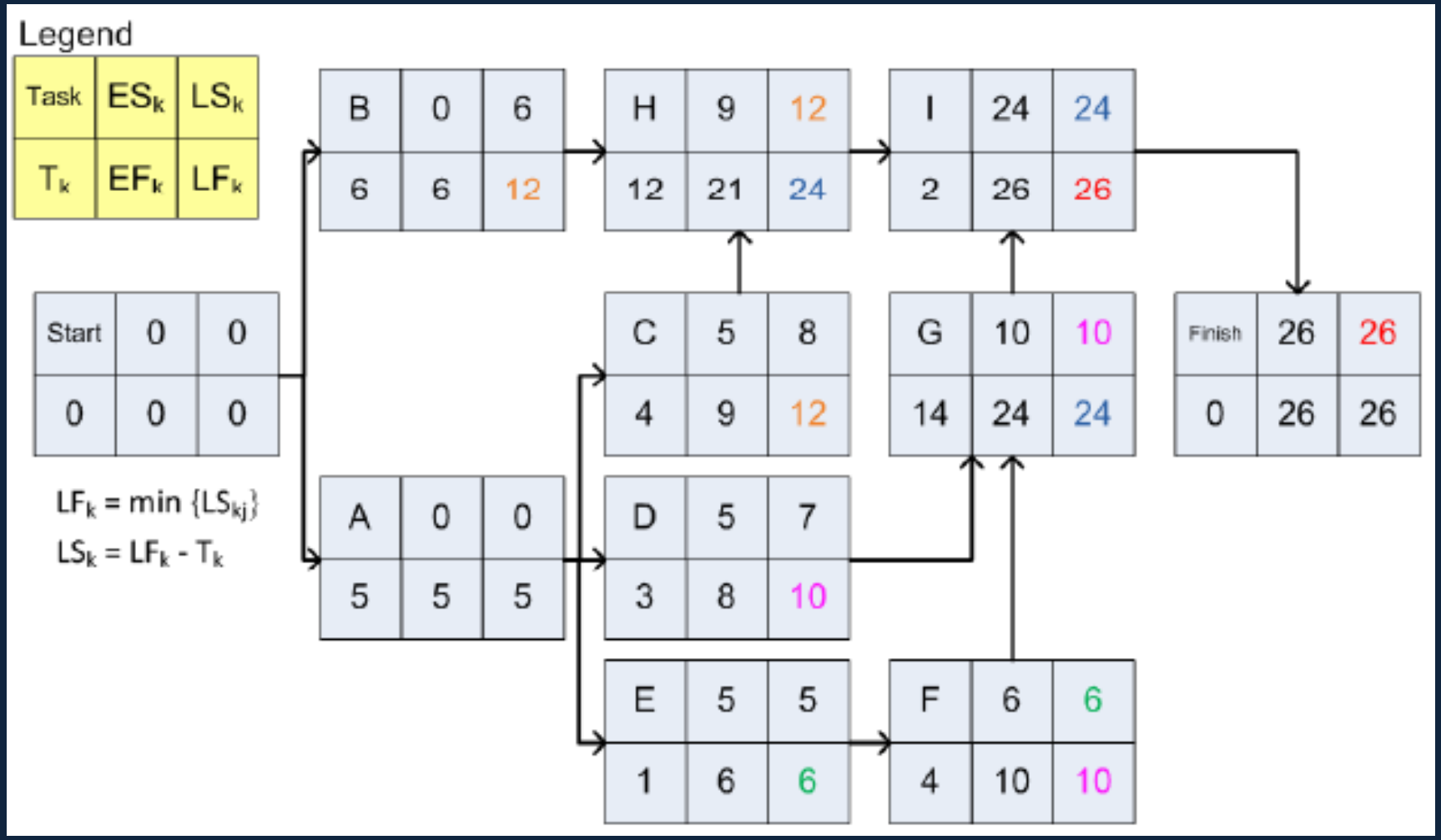
Answer

2. Do forward pass



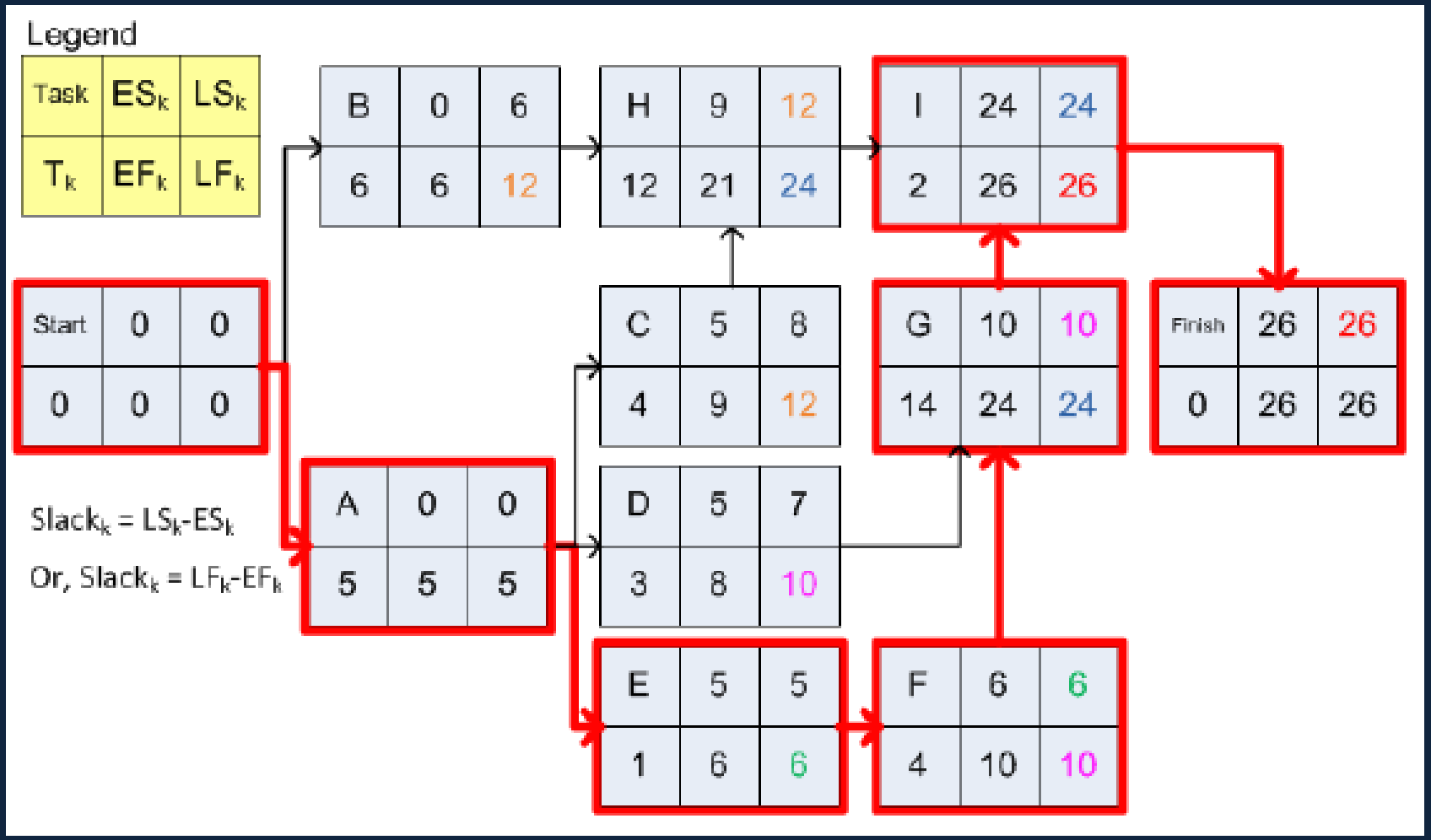
Answer

3. Do backward pass



Answer

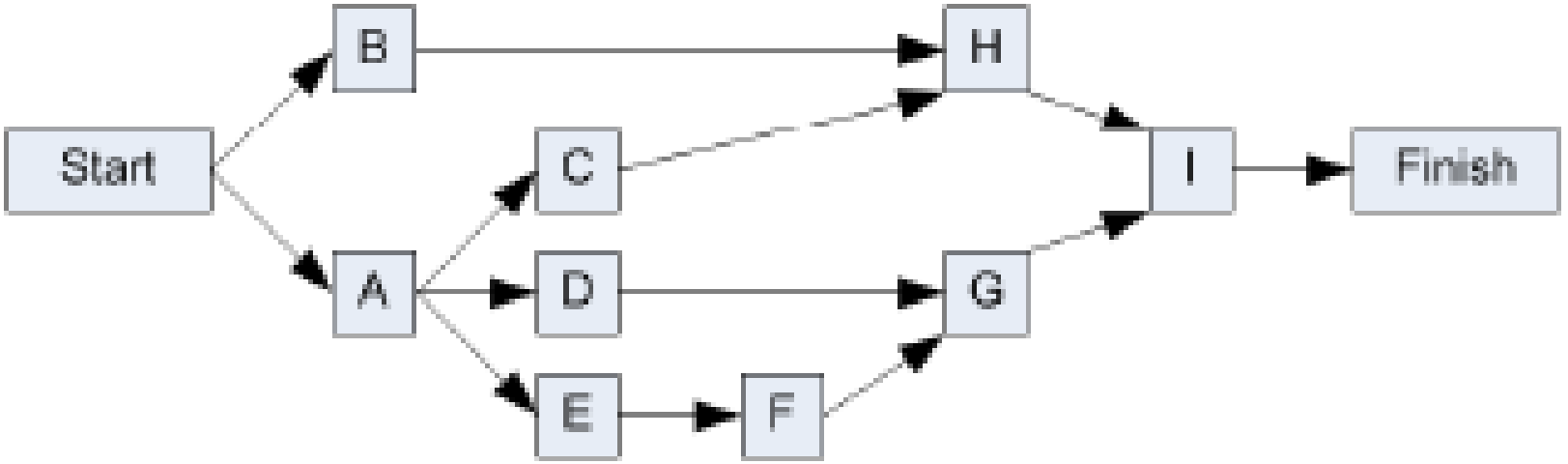
4. Find the Slack and Critical Path



Answer

Critical path: A-E-F-G-I

Project completion time: 26



6. Planning Testing

- The Software Project Management Plan (SPMP) must explicitly state what testing is to be done
- Traceability is essential
- All black box test cases must be drawn up as soon as possible after the specifications are complete

Planning Object-Oriented Projects

- An object-oriented product consists of largely independent pieces
- Consequently, planning is somewhat easier
- The whole is more than the sum of its parts

Planning of Object-Oriented Projects

- However, reuse induces errors in cost and duration estimates
  - Reuse of existing components during development (-)
  - Production of components for future reuse (+)
- These work in opposite directions
- In the long run, the savings (-) due to reuse outweigh the costs (+)

7. Training Requirements

- “We don’t need to worry about training until the product is finished, and then we can train the user”
- Not only users will need training
- Training is generally needed by the members of the development group, starting with training in software planning
- A new software development method requires training for every member of the group

Training Requirements

- Introduction of hardware or software tools of any sort necessitates training
- Programmers may need training in the operating system and/or implementation language
- Documentation preparation training may be needed
- Computer operators require training

8. Documentation Standards

- How much documentation is generated by a product?
  - IBM internal commercial product (50 KDSI)
    - 28 pages of documentation per KDSI
  - Commercial software product of the same size
    - 66 pages per KDSI
  - IMS/360 Version 2.3 (about 166 KDSI)
    - 157 pages of documentation per KDSI
  - [TRW] For every 100 hours spent on coding activities, 150–200 hours were spent on documentation-related activities

Types of Documentation

- Planning
- Control
- Financial
- Technical
- Source code
- Comments within source code

Documentation Standards

- Reduce misunderstandings between team members
- Aid SQA
- Only new employees have to learn the standards
- Standards assist maintenance programmers
- Standardization is important for user manuals

Documentation Standards

- As part of the planning process
  - Standards must be set up for all documentation
- In a very real sense, the product is the documentation

9. CASE Tools for Planning and Estimating

- It is essential to have
  - A word processor; and
  - A spreadsheet
- Management tools assist with planning and monitoring
  - MacProject
  - Microsoft Project

10. Testing the Software Project Management Plan

- We must check the SPMP as a whole
- Paying particular attention to the duration and cost estimates

References

- CS 123 Lectures: Kardi Teknomo, PhD
- Schach. Object-Oriented and Classical Software Engineering, 8th Ed.