

INTRODUCTION TO SOFTWARE PROJECT MANAGEMENT

What is a project?

Some dictionary definitions:

“A *specific plan or design*”

“A *planned undertaking*”

“A *large undertaking e.g. a public works scheme*”

Longmans dictionary

Key points above are **planning** and **size** of task

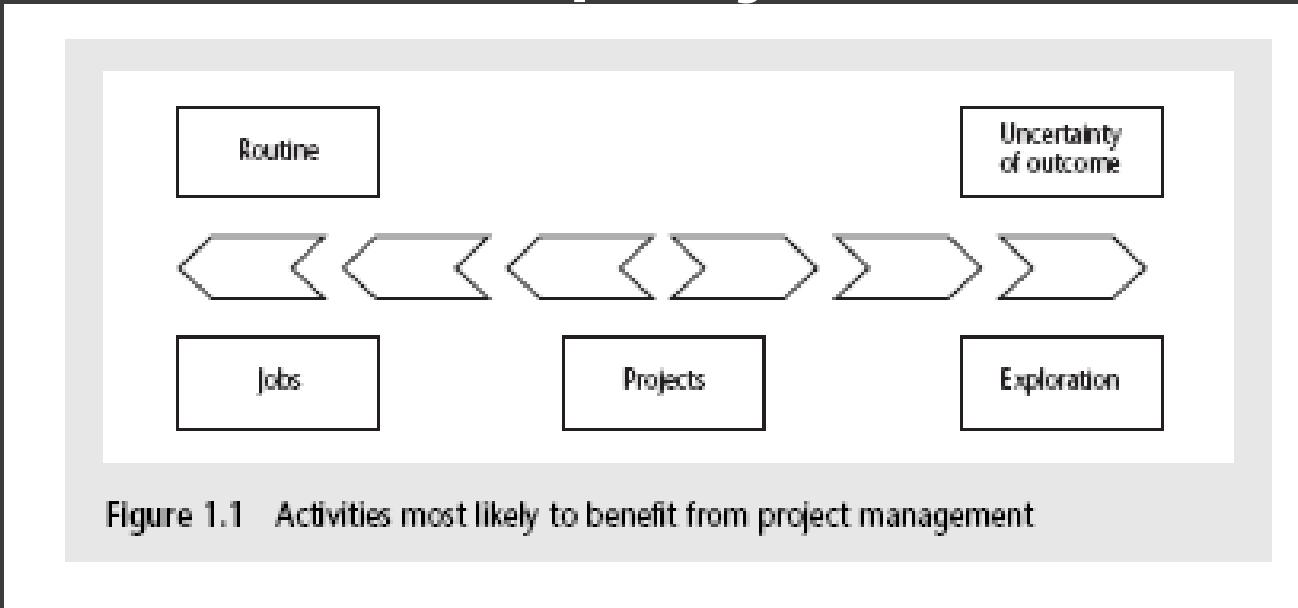
What's a project?

- A project being a planned activity or a specific plan or design.
- Characteristics which distinguish projects are:
 - Non-routine tasks are involved
 - Planning is required.
 - Specific objectives are to be met or specific products is to be created.
 - Project has a pre-determined time span
 - Work is carried out for someone other than yourself
 - Work is carried out in several phases
 - The resources available for use on the project are constrained
 - Project is large or complex.

Examples of Project

- Developing a new **Software**
- Implementing a new Decision Support System
- Developing a new office plan/layout
- Introducing a new product to the market
- Designing an airplane or a supercomputer
- Opening a new restaurant
- Constructing a bridge, dam, highway, or building
- Relocating an office or a factory
- Performing major maintenance or repair
- Producing or directing a movie

Jobs versus projects



'Jobs' – repetition of very well-defined and well understood tasks with very little uncertainty

'Exploration' – e.g. finding a cure for cancer: the outcome is very uncertain

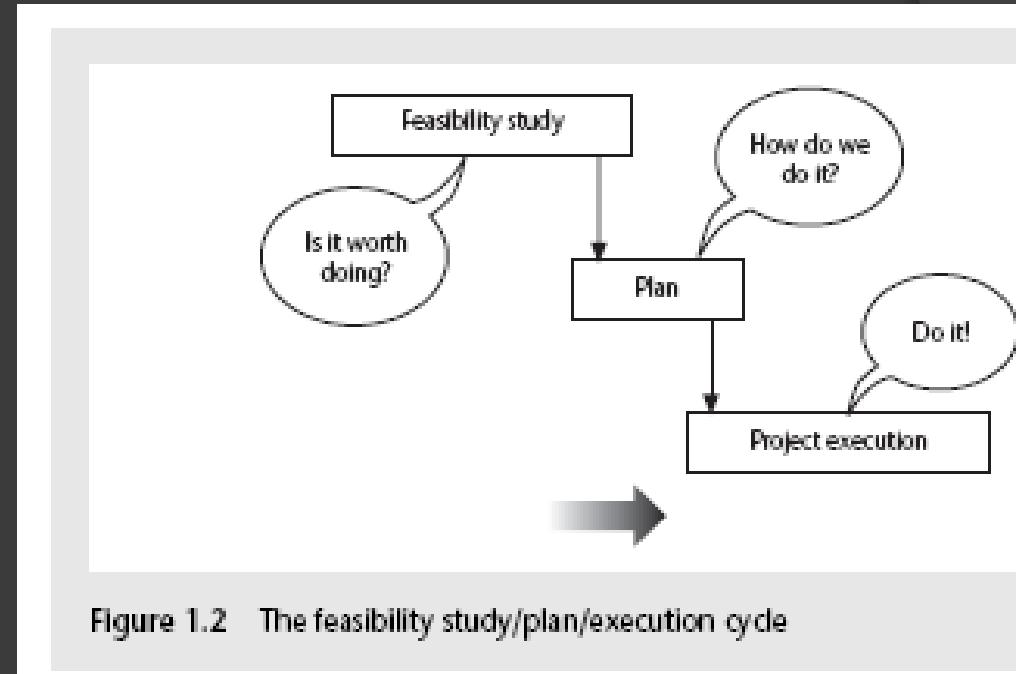
'Projects' – in the middle!

Software Project versus other types of Project

- **Invisibility**: When physical artifacts such as a bridge or road is being constructed the progress can be seen. With software, progress is not immediately visible.
- **Complexity**: Software products contain more complexity than other engineered artifacts.
- **Conformity**: other projects usually working with physical systems and materials like cement and steel. These are governed by physical laws that are consistent. Software developers have to conform to the requirements of human clients.
- **Flexibility**: The ease with which the software's can be changed

Activities covered by Software Project Management

- Feasibility Study
- Planning
- Project Execution



Activities covered by project management

Feasibility study

Is project technically feasible and worthwhile from a business point of view?

Planning

Only done if project is feasible

Execution

Implement plan, but plan may be changed as we go along

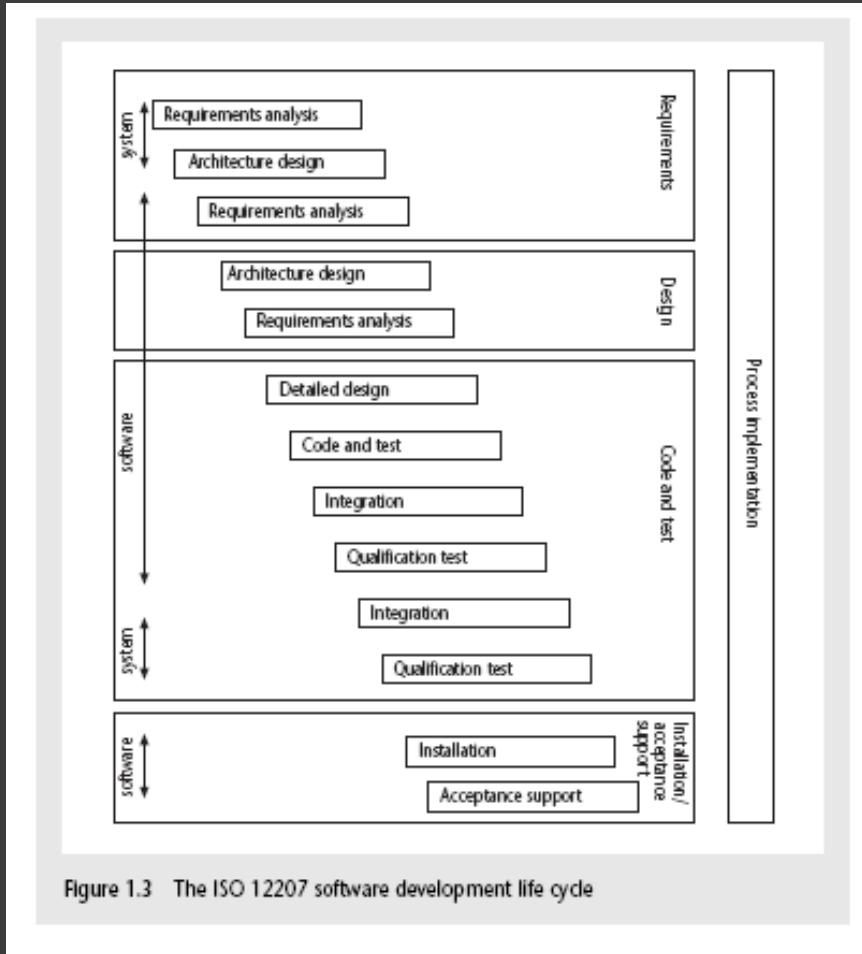
Plan Project

- A project plan answers:
 - What we are going to do?
 - Why are we going to do it?
 - How are we going to do it?
 - Who is going to be involved?
 - How long it will take?
 - How much it will cost?
 - What can go wrong and what can we do about it?
 - How did we estimate the schedule and budget?
- Initial plan is called – **baseline plan**

Execute project Plan

- q To put the plan into action.
- q As work progresses ensure that the project achieves its goal.
- q Project progress must be documented and compared with the baseline plan.

The software development life-cycle (ISO 12207)



Ways of categorizing projects

- **Information System Versus Embedded System**
 - **Information System**: system interfaces with organizations
 - **Embedded System**: Interfaces with a machine
- **Objectives versus products**
 - Projects may be distinguished by whether their aim is to produce a product or to meet certain objectives.
 - A project may be to create a product, the details of which have been specified by client.
 - Client has the responsibility for justifying the product.

What is management?

- Management involves the following activities:
 - **Planning:** deciding what is to be done
 - **Organizing:** Making arrangements
 - **Staffing:** Selecting the right people for the job
 - **Directing:** giving instructions
 - **Monitoring:** Checking on progress
 - **Controlling:** taking actions to remedy hold-ups
 - **Innovating:** coming up with solutions
 - **Representing:** liaising with users.

Problems with Software

- Poor Estimates and plans
- Lack of quality standards and measures
- Lack of guidance about making organizational decisions
- Lack of techniques to make progress visible
- Poor role definition-who does what?
- Incorrect success criteria

Problems identified by students

- Inadequate specification of work
- Lack of knowledge of specification area
- Management ignorance of IT
- Lack of standards
- Lack of communication between users and technicians
- Lack of communication leading to duplication of work
- Lack of commitment
- Changing software environment
- Deadline pressure
- Lack of quality control
- Lack of training.

Setting Objectives

- Projects objectives should be clearly defined.
- Setting objectives can be used to guide and motivate individuals and group of staff.
- In order to achieve the objective we must achieve certain goals first.

Goals/sub-objectives

These are steps along the way to achieving the objective. Informally, these can be defined by completing the sentence...

Objective X will be achieved

IF the following goals are all achieved

A.....

B.....

C..... etc

Goals/sub-objectives continued

Often a goal can be allocated to an individual.

Individual may have the capability of achieving goal, but not the objective on their own e.g.

Objective – user satisfaction with software product

Analyst goal – accurate requirements

Developer goal – software that is reliable

Objectives should be SMART

- S** – specific, that is, concrete and well-defined
- M** – measurable, that is, satisfaction of the objective can be objectively judged
- A** – achievable, that is, it is within the power of the individual or group concerned to meet the target
- R** – relevant, the objective must relevant to the true purpose of the project
- T** – time constrained: there is defined point in time by which the objective should be achieved

Measures of effectiveness

How do we know that the goal or objective has been achieved?

Measure of effectiveness tells us how successful the project has been.

By a practical test, that can be objectively assessed.

e.g. for user satisfaction with software product:

- Repeat business – they buy further products from us
- Number of complaints – if low etc

Stakeholders

- These are people who have interest in the project.
- Stakeholders are to be identified as early as possible as we have to set up adequate communication channels with them right from the start.
- Stakeholders might be **internal to the project team, external to the project team but in the same organization, or totally external to the organization.**

Stakeholders (Cont..)

- Internal to the project team: This means they will be under the direct managerial control of the project leader. *Include:*
 - ❑ Project Sponsor
 - ❑ Project staff
 - ❑ Support staff
 - ❑ Top management
- External to the project team but within the same organization:
 - ❑ Project leader need assistance of information management group in order to add to database.
- External to both the project team and the organization:
Include
 - ❑ Customer\client
 - ❑ Suppliers
 - ❑ Competitor
 - ❑ Regulatory agencies

The business case

Benefits



Costs

£

Benefits of delivered
project must outweigh
costs

Costs include:

- Development
- Operation

Benefits

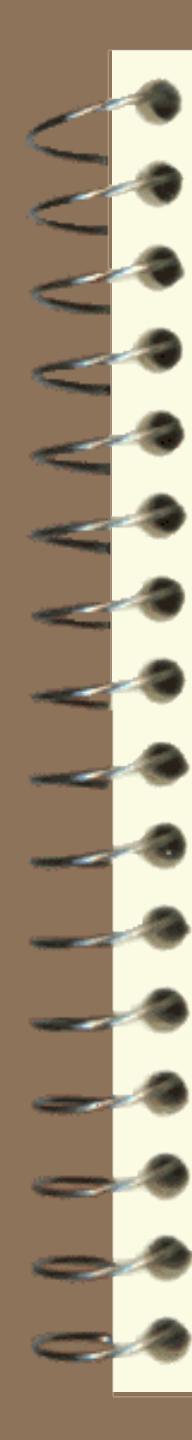
- Quantifiable
- Non-quantifiable

Business Case

- Purpose is to provide the top management with all the information needed to decide, whether the project should be funded.
- E.g. A new web-based application might allow customers from all over the world to order a firm's products via the internet, increasing sales, revenue and profits.
- Project plan must ensure that the business case is intact. For e.g.
 - Development cost should not exceed.
 - The features of the system should not be reduced
 - Delivery of the project is not delayed.



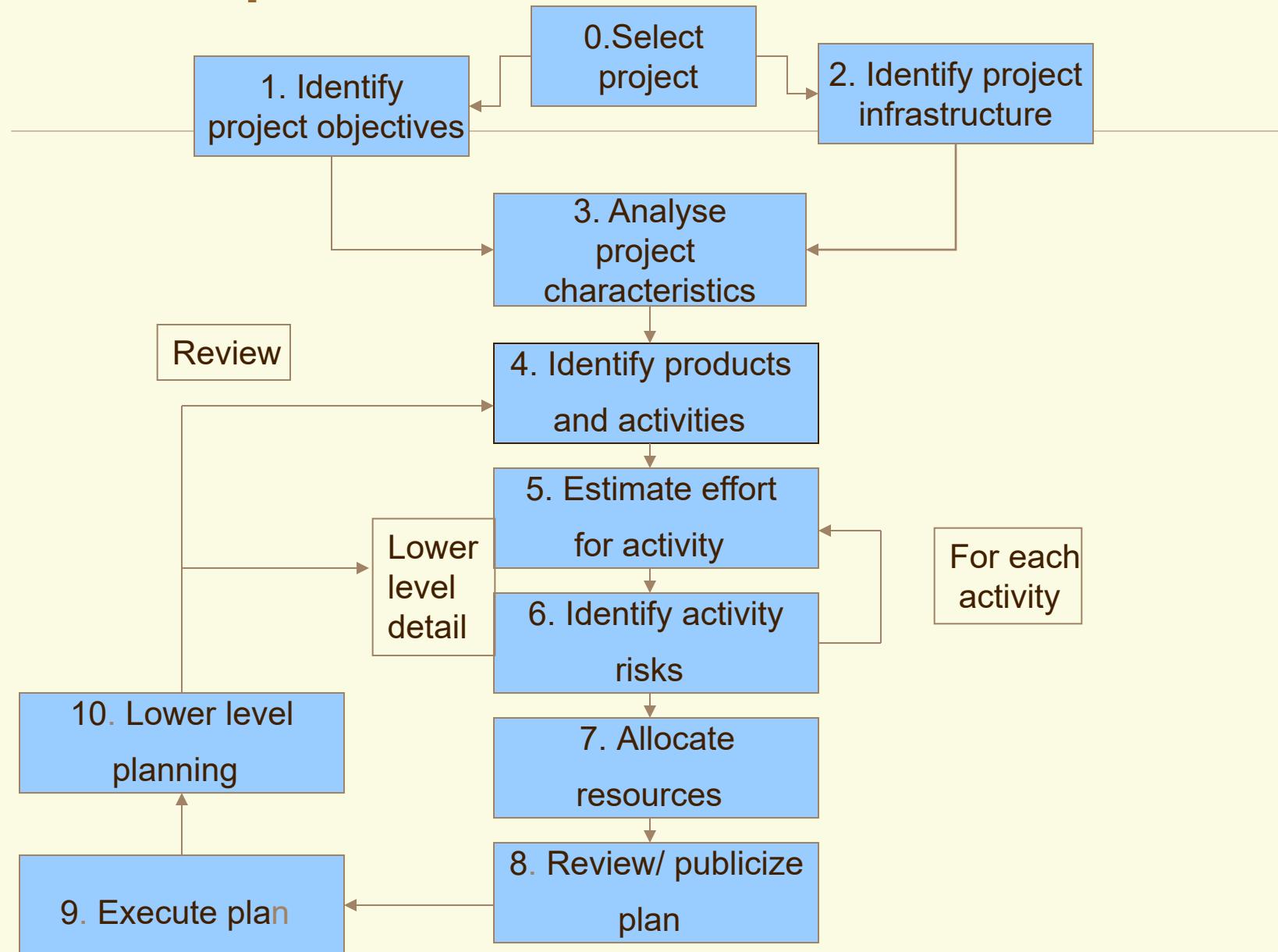
Step Wise Overview of Project Planning



Questions from customer

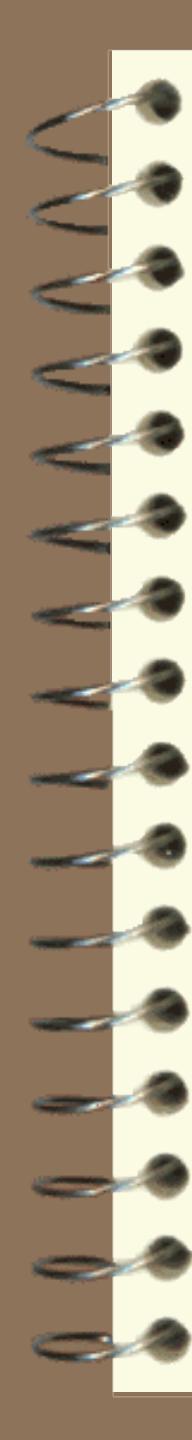
- do you understand my problem and my needs?
- can you design a system that will solve my problem and satisfy my needs?
- how long will it take to develop such a system?
- how much will it cost to have you develop such a system?

'Step Wise' - an overview



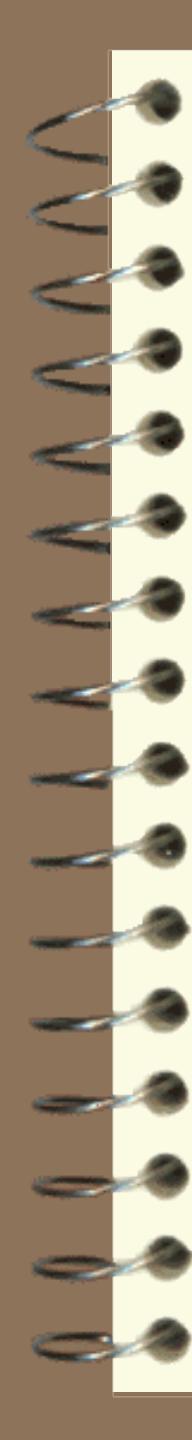
Project Planning

- 4 Project Manager reviews contractual commitment.
- 4 Creates plan.
- 4 Project plan involves:
 - Life-cycle process to be followed.
 - Estimating the effort.
 - scheduling



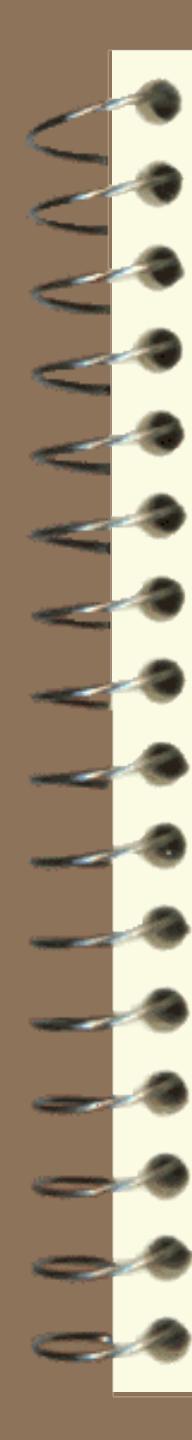
Planning (cont...)

- Quality and configuration management
 - Risk management
- 4 Principle: plan outline



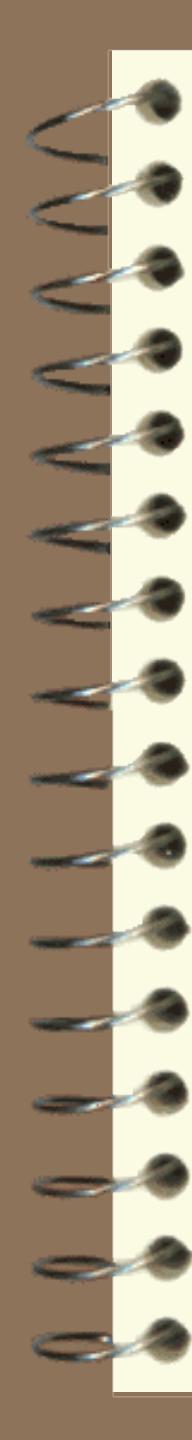
Step 0: Select Project

- 4 Outside the main project planning process.
- 4 Initiation is required.
- 4 Feasibility study.



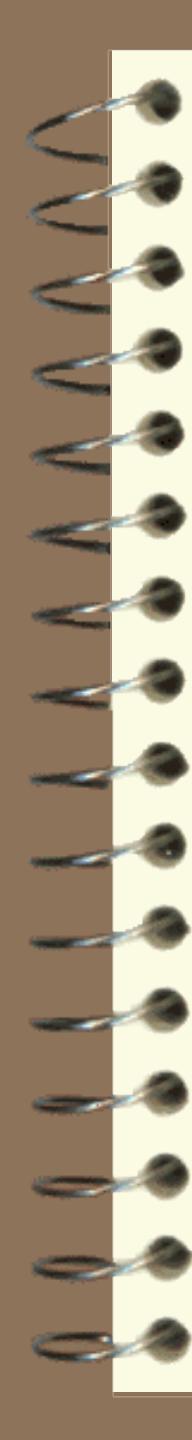
Step 1: Identify project scope and objectives

- 4 Scope of s/w project
- 4 Project manager defines the scope.
- 4 Agreement of all the parties.
- 4 Ensure commitment.



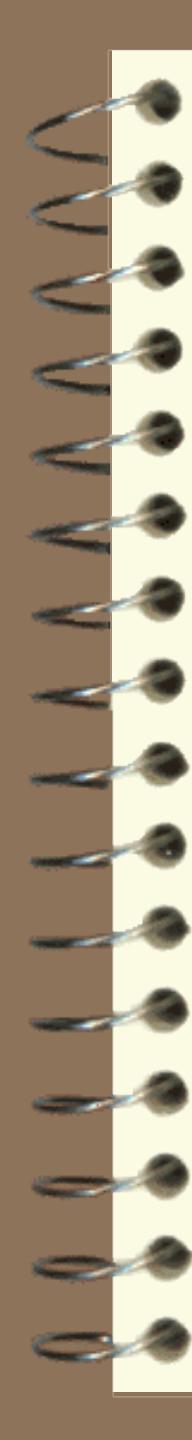
Step 1.1 Identify objectives and practical measure of effectiveness in meeting those objectives.

- 4 Measure of effectiveness: tells how successful the project has been.
- 4 Success in achieving those objectives.



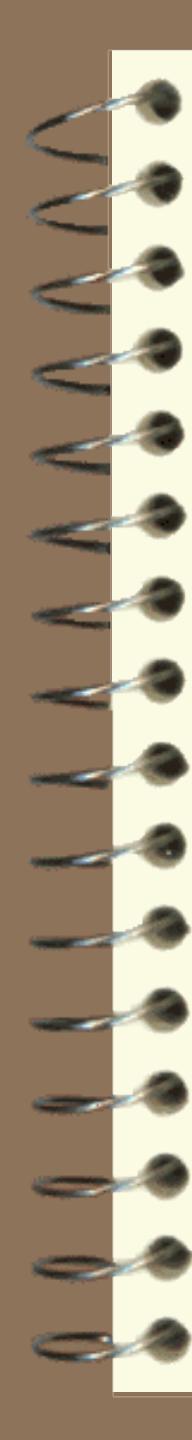
Step 1.2 Establish a project authority.

- 4 Single overall authority is needed for the unity of work done by all the people.
(project steering committee or project board or project management board)



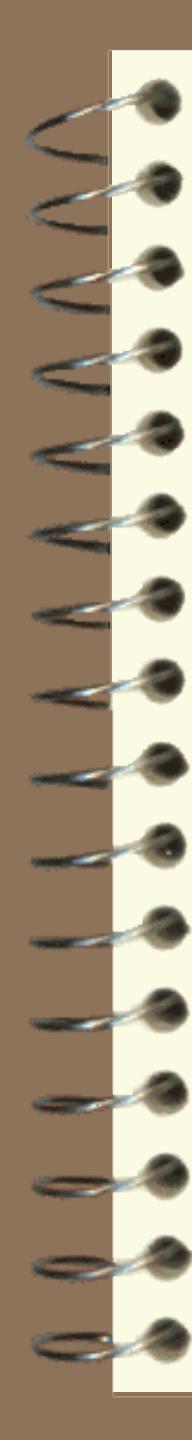
Step 1.3 Stakeholder analysis

- 4 Identify all the stakeholders.
- 4 Their interest in project.



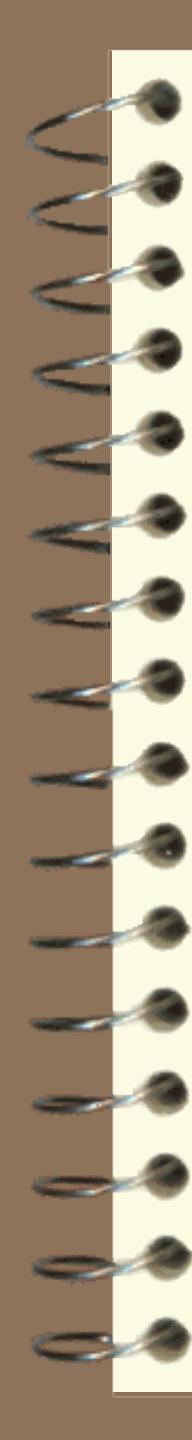
Step 1.4 Modify objectives in the light of stakeholder analysis

- 4** Adding new features suggested by the stakeholders.
- 4** Full cooperation and commitment could be assured.



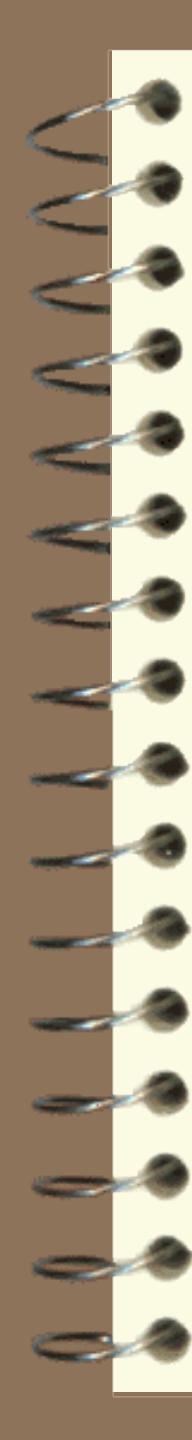
Step 1.5 Establish methods of communication with all parties.

4 Communication with all the people involved in the project.



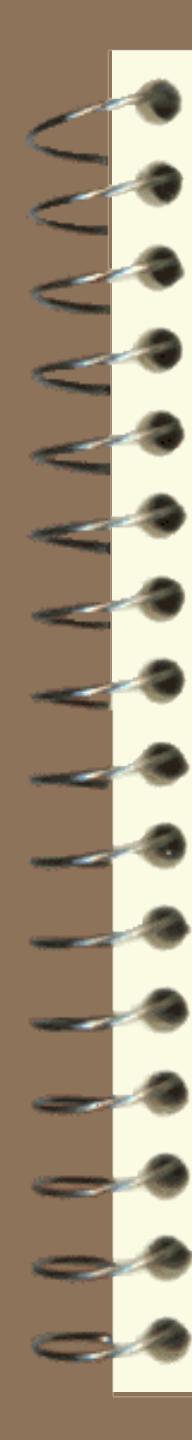
Step 2: Identify project infrastructure

- 4 Infrastructure needed for a particular project.
- 4 The project leader responsible for finding out the infrastructure.



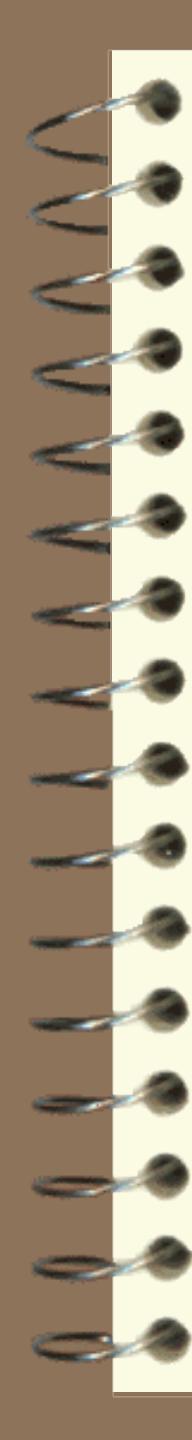
Step 2.1 Identify relationship between the project and strategic planning

- 4** What order the project should be carried out.
- 4** Establish a framework.



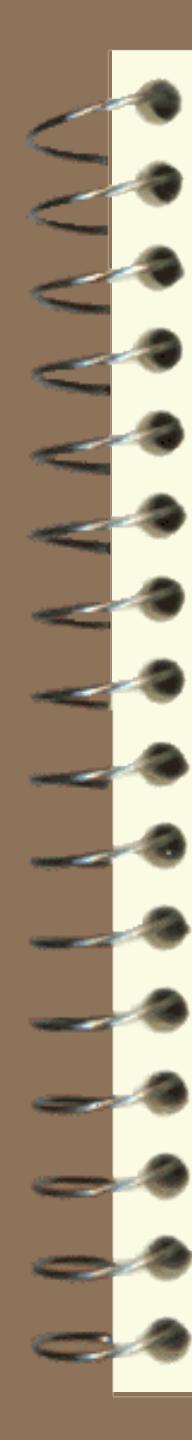
Step 2.2 Identify installation standards and procedures.

- 4 Should define their development procedures.
- 4 Should be documented as the product is created at each stage.



Step 2.3 Identify project team organization

- 4 Project is divided among different teams.
- 4 Should identify the project team.
- 4 programmers and system analysts put into different teams.
- 4 Eg:Development of PC application and mainframe application in different groups.



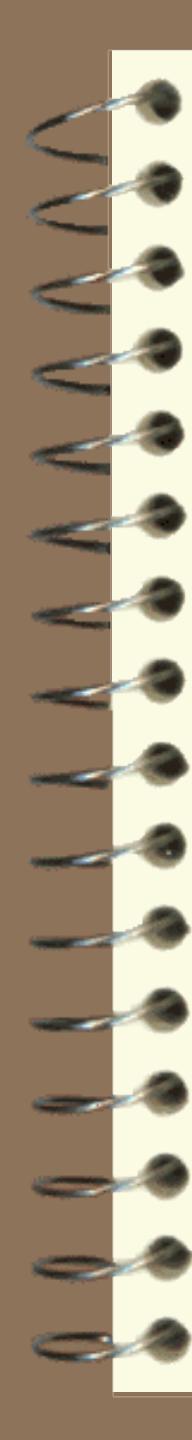
Step 3: Analyze Project characteristics.

- 4 Ensure that appropriate methods are used for the project.



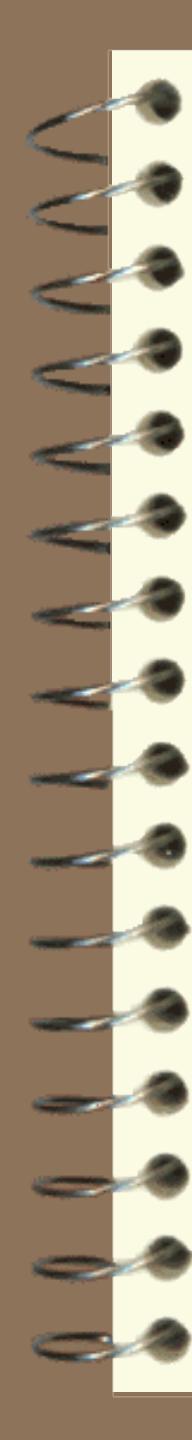
Step 3.1 Distinguish the project as either objective or product driven

- 4** Identify the project by their aim whether the project is required for meeting some objectives or to produce a product.



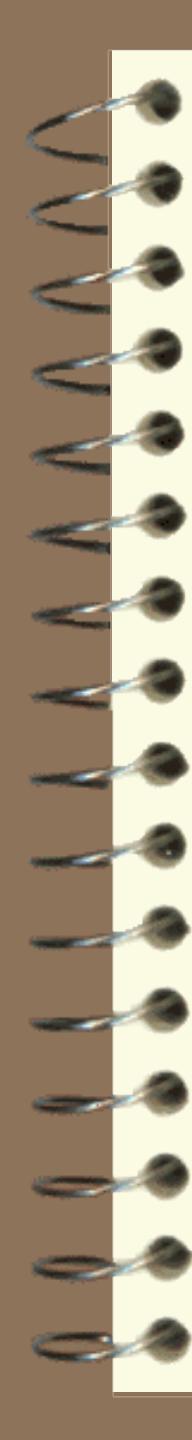
Step 3.2 Analyze other project characteristics.

- 4 Including quality based-ones.
- 4 Will the system be safe critical.
4 i.e. where human life could be threatened by a malfunction.



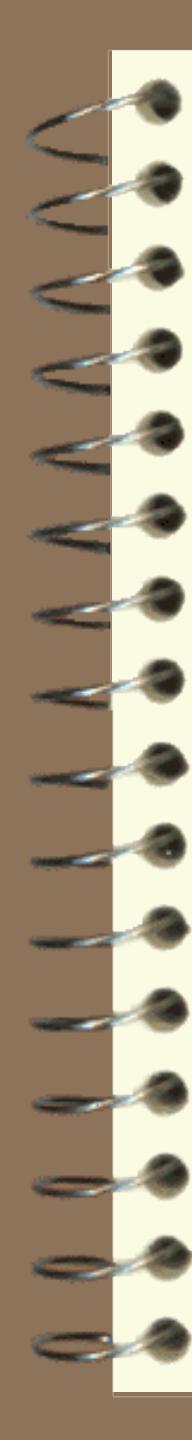
Step 3.3 Identify high level project risks

- 4 Risk that threaten the successful outcome of the project.**



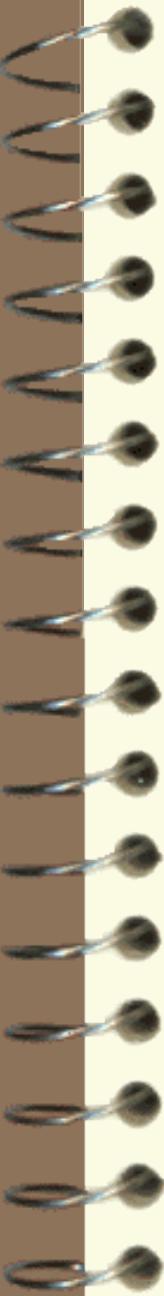
Step 3.4 Take into account user requirements concerning implementation

4 User requirement in order to fulfill their needs.



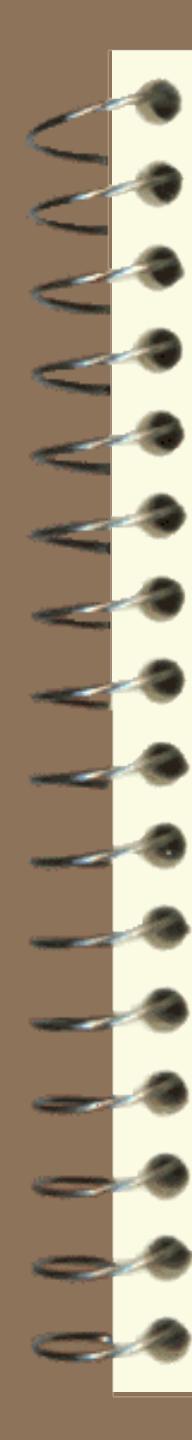
Step 3.5 Select development methodology and life cycle approach

- 4 How the development methods is carried out
- 4 **Methodology** :group of methods to be used in a project.
- 4 and life cycle approach to be used.



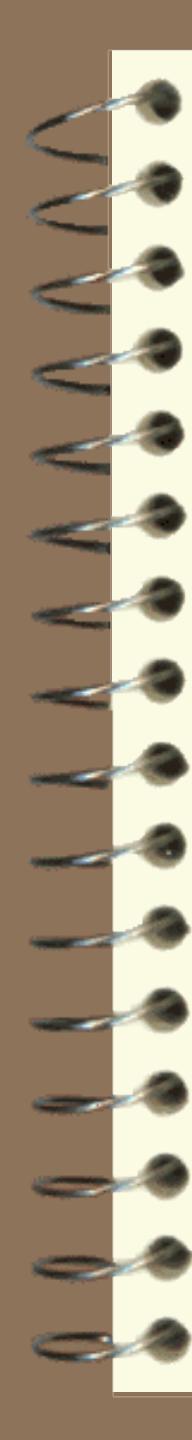
Step 3.6 Review overall resource estimates

- 4 After estimating the risks and approach good point is to re-estimate the effort and other resources required for the project.



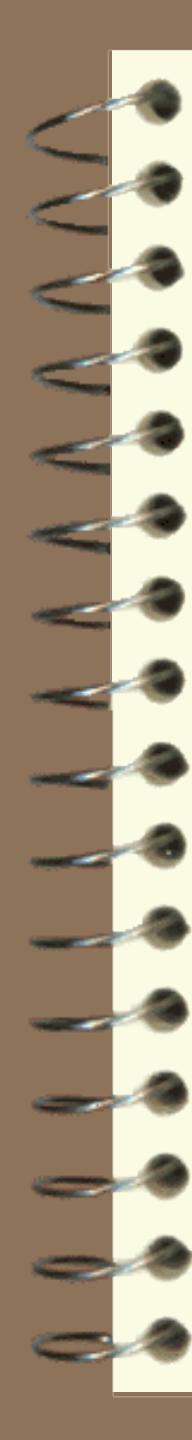
Step 4: Identify the project products and activities

- 4 More detailed planning of the process takes place.
- 4 Longer term planning is broad and outline
- 4 Immediate tasks are planned in detail.



Step 4.1 Identify and describe project products(or deliverables)

- 4 Products form an hierarchy. (Product is result of an activity)
- 4 Main products-set of component products-sub component products and so on...
- 4 The relationships are documented in PBS (Product Breakdown Structure)



Product Breakdown Structure

- 4 It's an exhaustive ,hierarchical tree structure of components that make up an item arranged in a whole-part relationship.
- 4 Help clarifying what is to be delivered by the project.

Example of PBS

4 E.g. PBS of a computer.

4 Main unit

- Housing
- Motherboard
 - CPU
 - RAM chips
 - ...
- FDD
- HDD
- Video card
- Sound card
- Network card
- LPT port card

4 Monitor

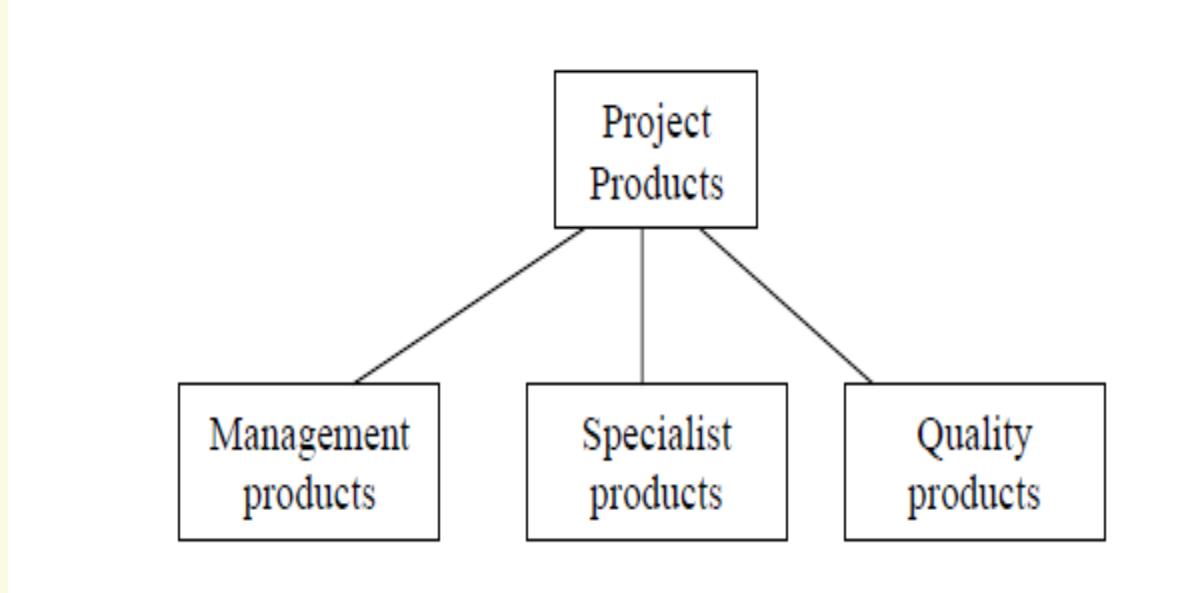
- CRT
- Housing
- Electronic components

4 Mouse

- Body
- Marble
- Cable

4 Keyboard

- ...



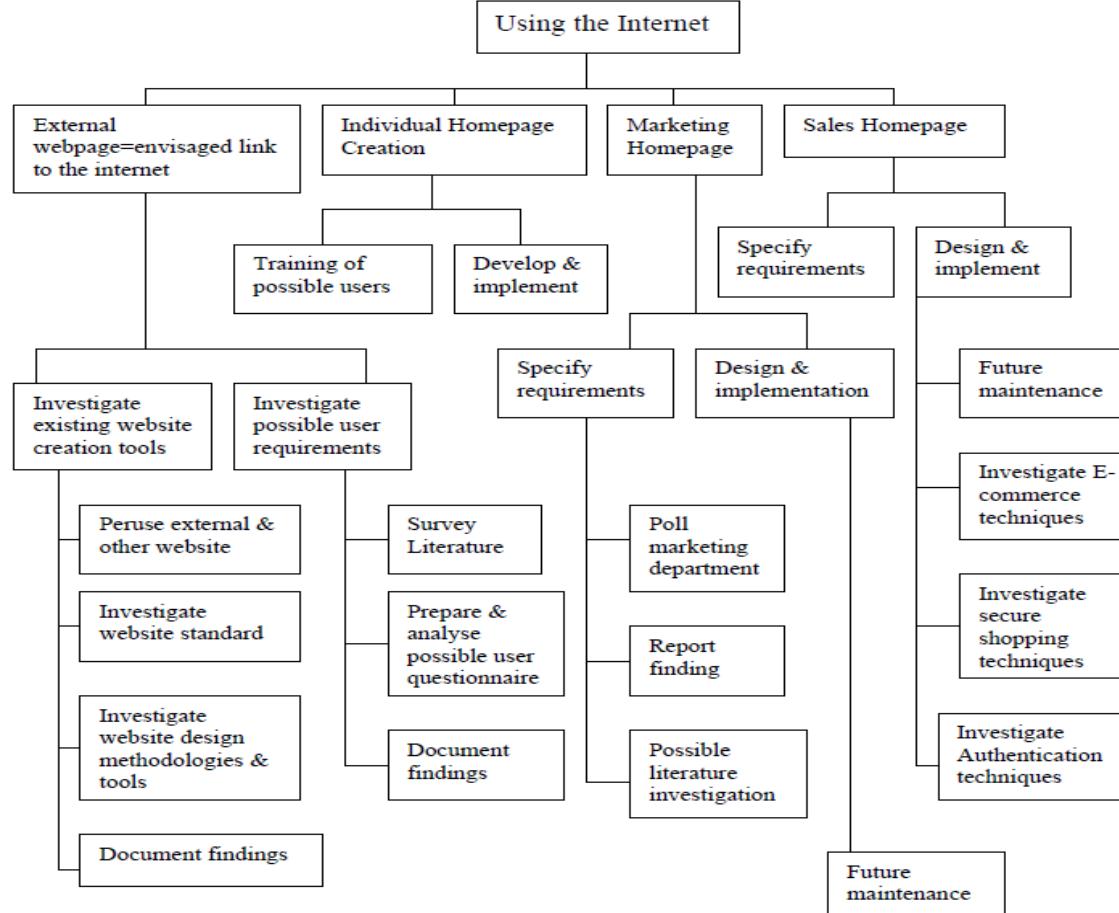
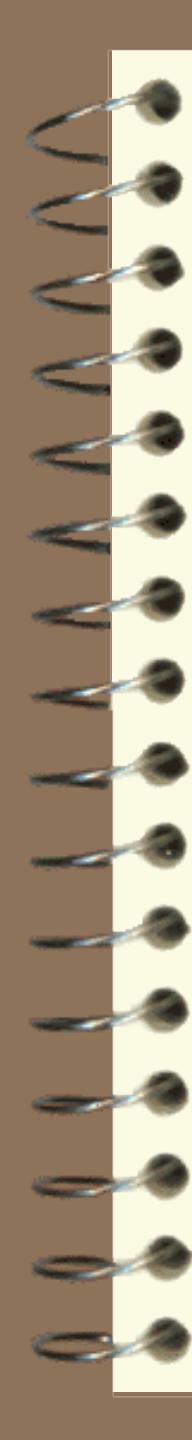


Figure 1: PBS for "Using the Internet"

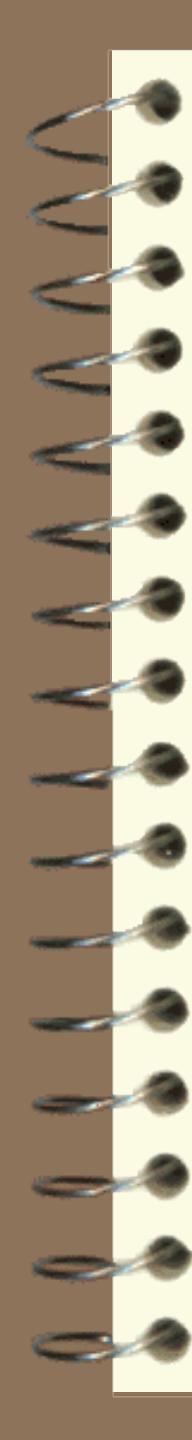
Step 4.2: Document generic product flows

- 4 Existence of one or more product.
- 4 E.g. program design should be there before coding, documentation should be there before design.
- 4 These relationships can be shown using PFD (Product Flow Diagram)



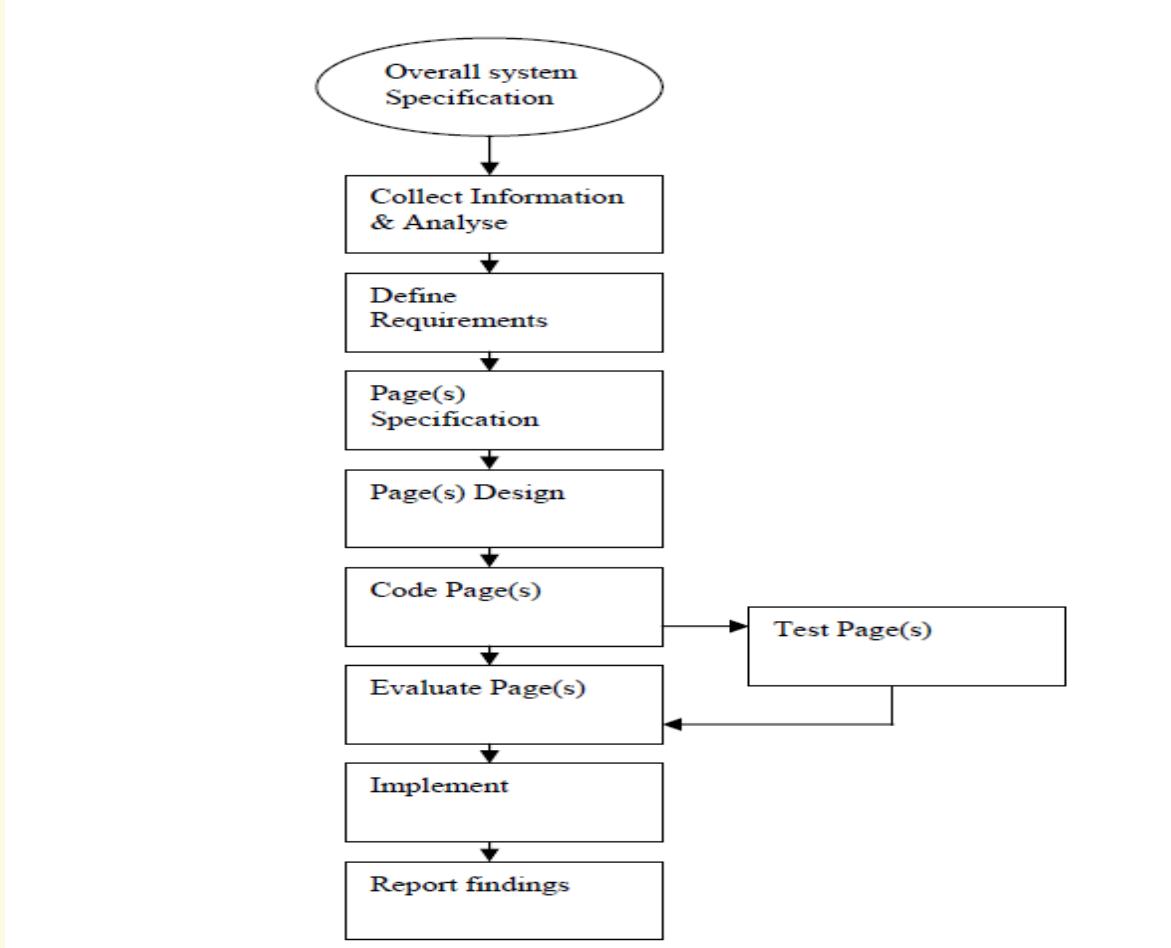
Product Flow Diagram

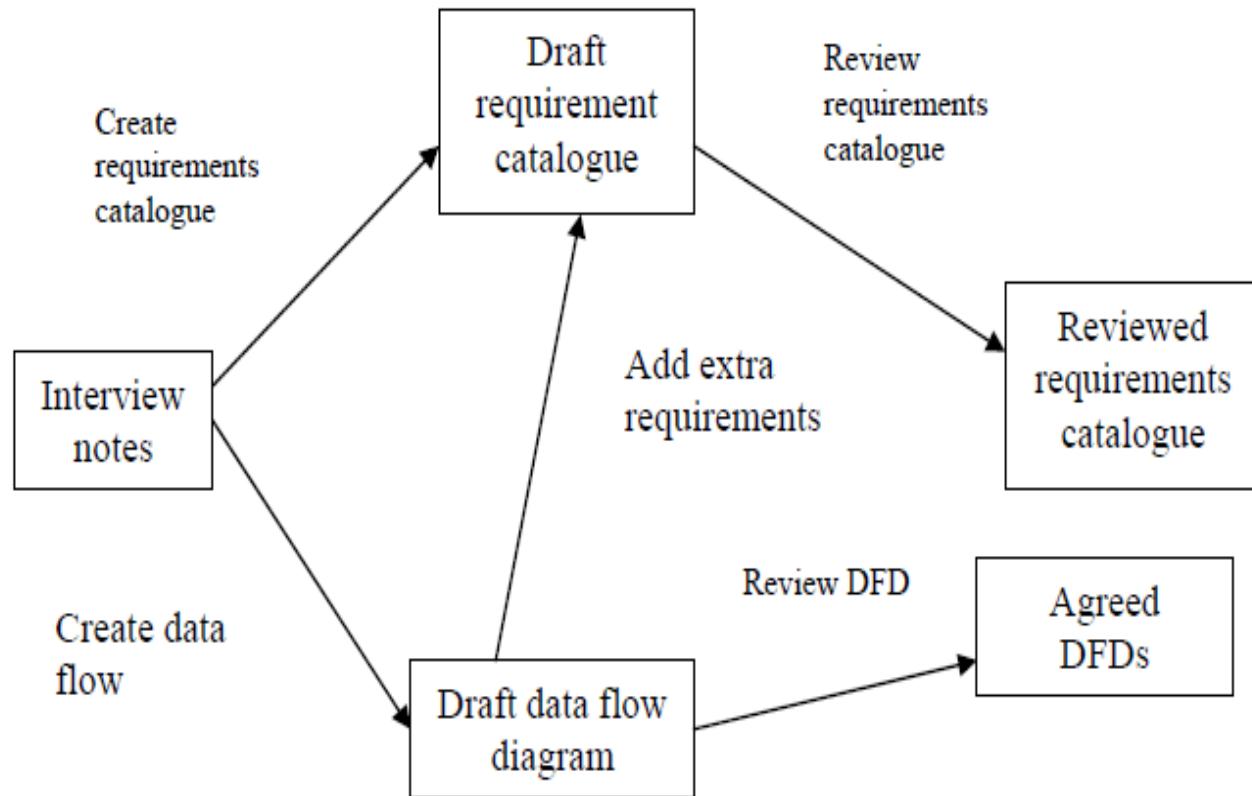
- 4 PFD specifies which product must be completed before next can be produced.
- 4 Flow in the diagram is from top to bottom and left to right.

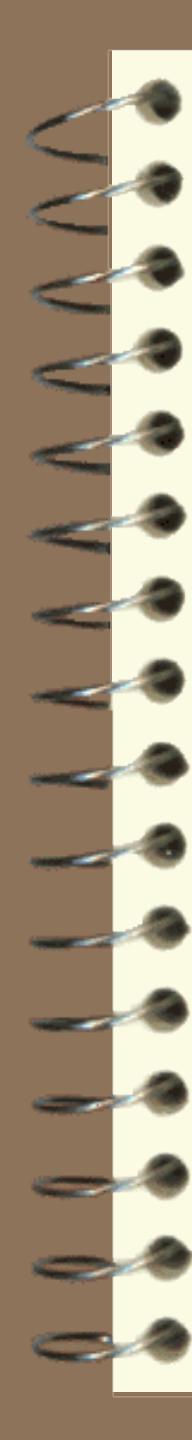


PFD (continues..)

- 4 Some items are intermediate products, needed only to help produce other products.
 - Indicated by boxes.
- 4 Some items will exist already
- 4 Used by project but is not created by it
 - Feasibility study
 - Indicated by ovals (ellipse)

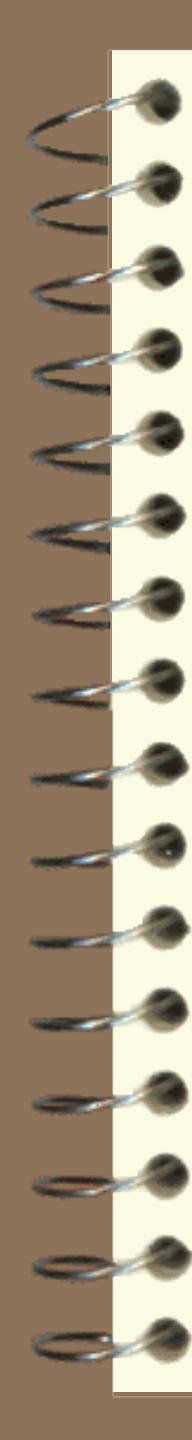






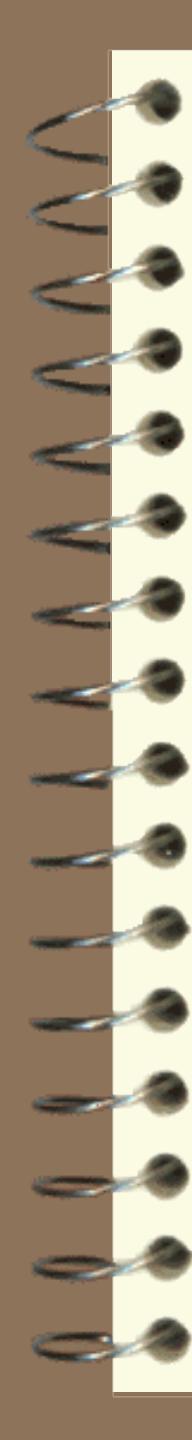
Step 4.3 Recognize product instances.

- 4 When same PFD fragment relates to more than one instances of a product.
- 4 Attempt to identify those instances.



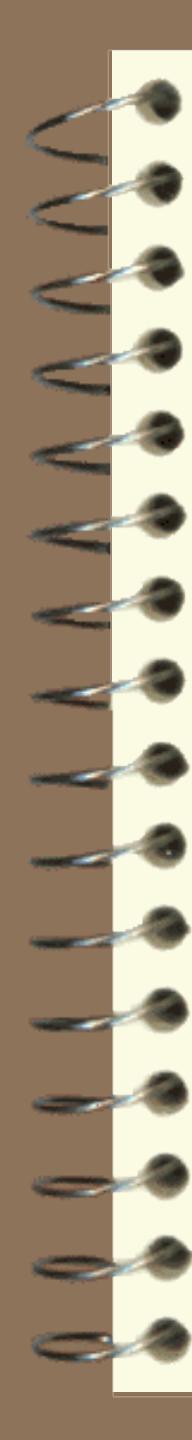
Step 4.4: Produce ideal activity network.

- 4 Identifying all activities that produce a product can create an activity network.
- 4 Activity network: shows the tasks carried out and the order in which they are executed.



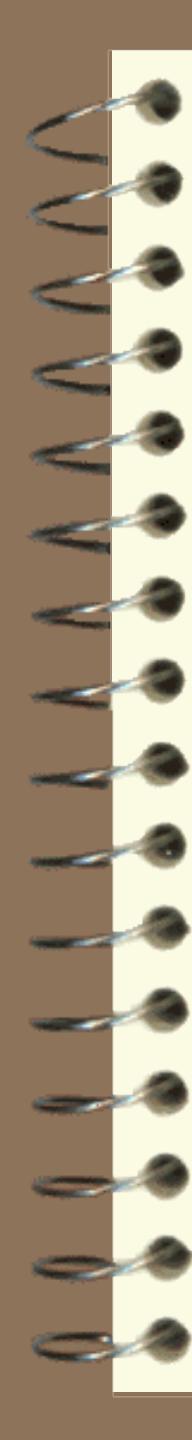
Step 4.5: Modify the ideal to take into account need for stages and checkpoints

- 4** Ideal activity network should be modified by dividing into stages and inserting checkpoints.
- 4** Checkpoints draws together the products of proceeding activities to check that they are compatible.
- 4** Delay some work.



Activities and Milestones

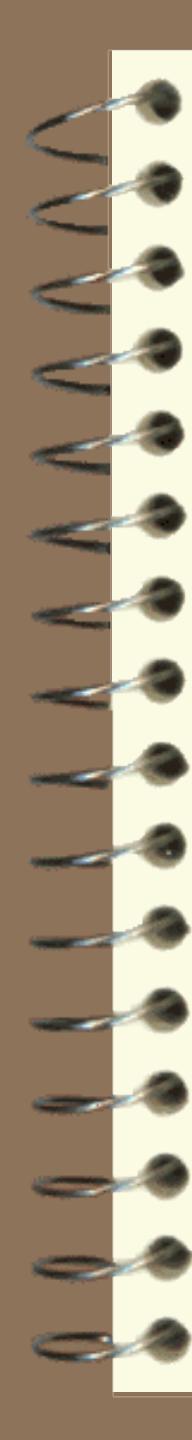
- 4 Milestones indicate measurable level of progress
- 4 Each milestone completed can be reported or demonstrated to the customer.
- 4 An **activity** is part of a project that takes place. A milestone is a completed activity



Step 5: Estimate the effort for each activity.

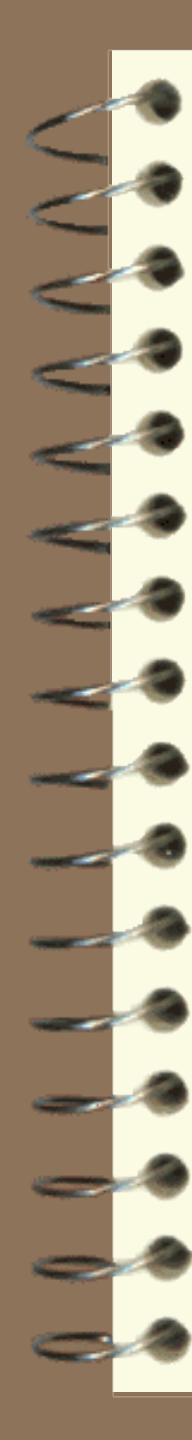
Step 5.1: Carry out bottom-up estimates.

- 4 Estimation of the staff effort required.
- 4 The probable elapsed time.
- 4 Elapsed time-it's time between start and end of a task.
- 4 Non-staff resource needed.
- 4 Estimation may vary according to the activity.



Step 5.2:Revise plan to create controllable activities.

- 4 some activity takes long time.
- 4 Long activity make project difficult to control.



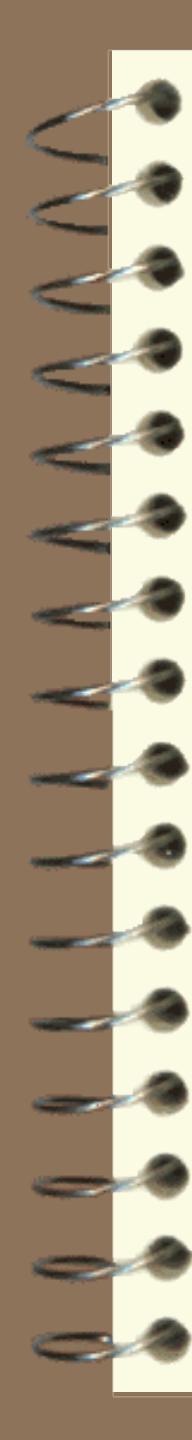
Step 6: Identify activity risk

Step 6.1: identify and quantify activity-based risks

- 4 Project plan is based on huge assumptions.
- 4 To identify risk is more important.
- 4 Damage caused by each risk should be identified.
- 4 If risk occurs, it make the task longer or more costly.

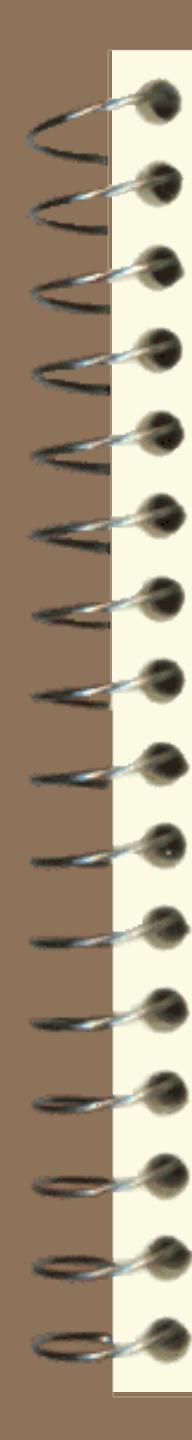
Step 6.2: Plan risk reduction and contingency measures where appropriate

- 4 Avoid or at least reduce some of the identified risks.
- 4 Contingency measures: action that is to be taken if risk materializes.
 - 4 E.g. contract staff.



Step 6.3: Adjust overall plans and estimates to take account of risks.

- 4 Change our plan by adding new activities that reduce risks.
- 4 E.g. new programming language requires training of developers.



Step 7 :Allocate resources

Step 7.1: Identify and allocate Resources

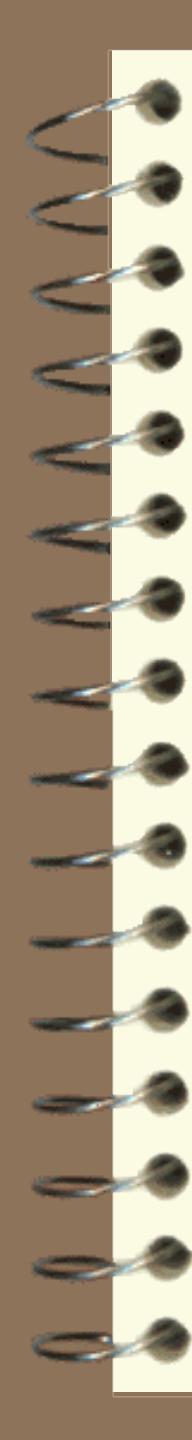
- 4 Type of staff needed for each activity is recorded.
- 4 Staff available for the project is identified.
- 4 Allocated to tasks.

Step7.2:Revise plan and estimates to take into account resource constraints.

- 4 Ensuring that staffs are available as soon as the proceeding work is completed.
- 4 Gantt chart.

Gantt Charts

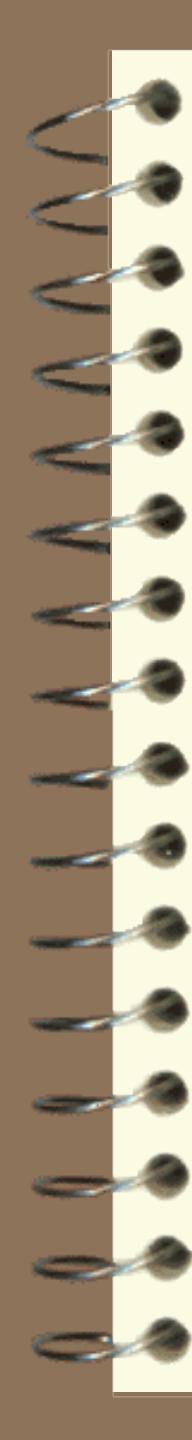
- 4 Shows all project activities and their
 - start, finish and slippage time
 - total duration
 - slack time
 - critical periods
 - dates associated within these times
- 4 in Bar Chart form



Step 8: Review/publicize plan

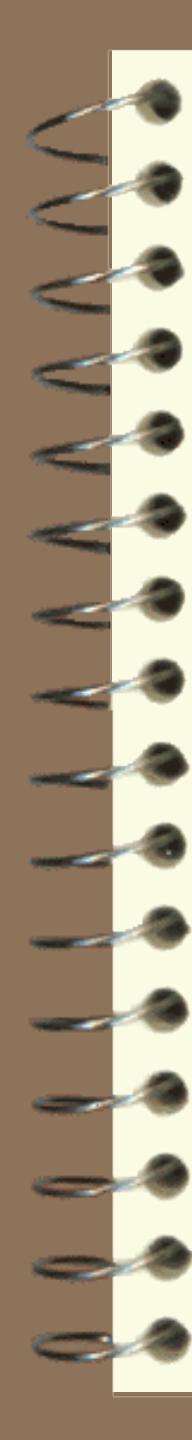
Step 8.1: Review quality aspects of the project plan

- 4 Ensure that activity is properly completed not giving a chance to re-work.
- 4 Each task should have ‘exit requirement’.
- 4 These are quality checks that have to be done before the activity can be signed off as completed.



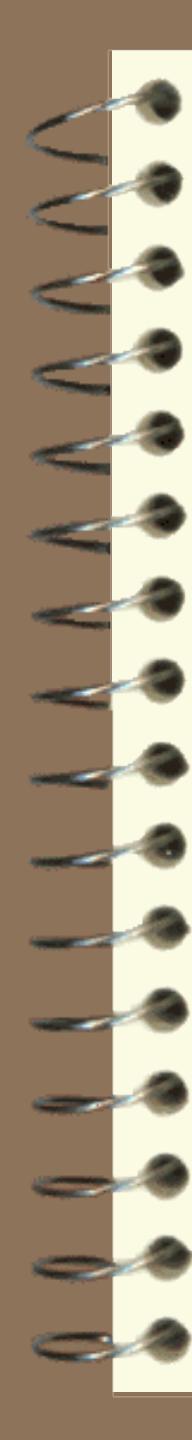
Step 8.2: Document plan and obtain agreement

- 4** Careful documentation of the plan.
- 4** All the parties to project understand and agree to the commitment in the plan.



Step 9 and 10: Execute plan and lower level planning

- 4 Plan drawn up in detail for each stage.
- 4 Detailed planning of the later stages is delayed as more information is available as we reach nearer the start of the stage.



Key Points

- 4 Effective management depends on planning
- 4 Planning and estimating are iterative
- 4 Project milestones should be dispersed throughout the project
- 4 Managers must analyse options thoroughly
- 4 Project scheduling must account for interrelationships

Software Project Management

4th Edition

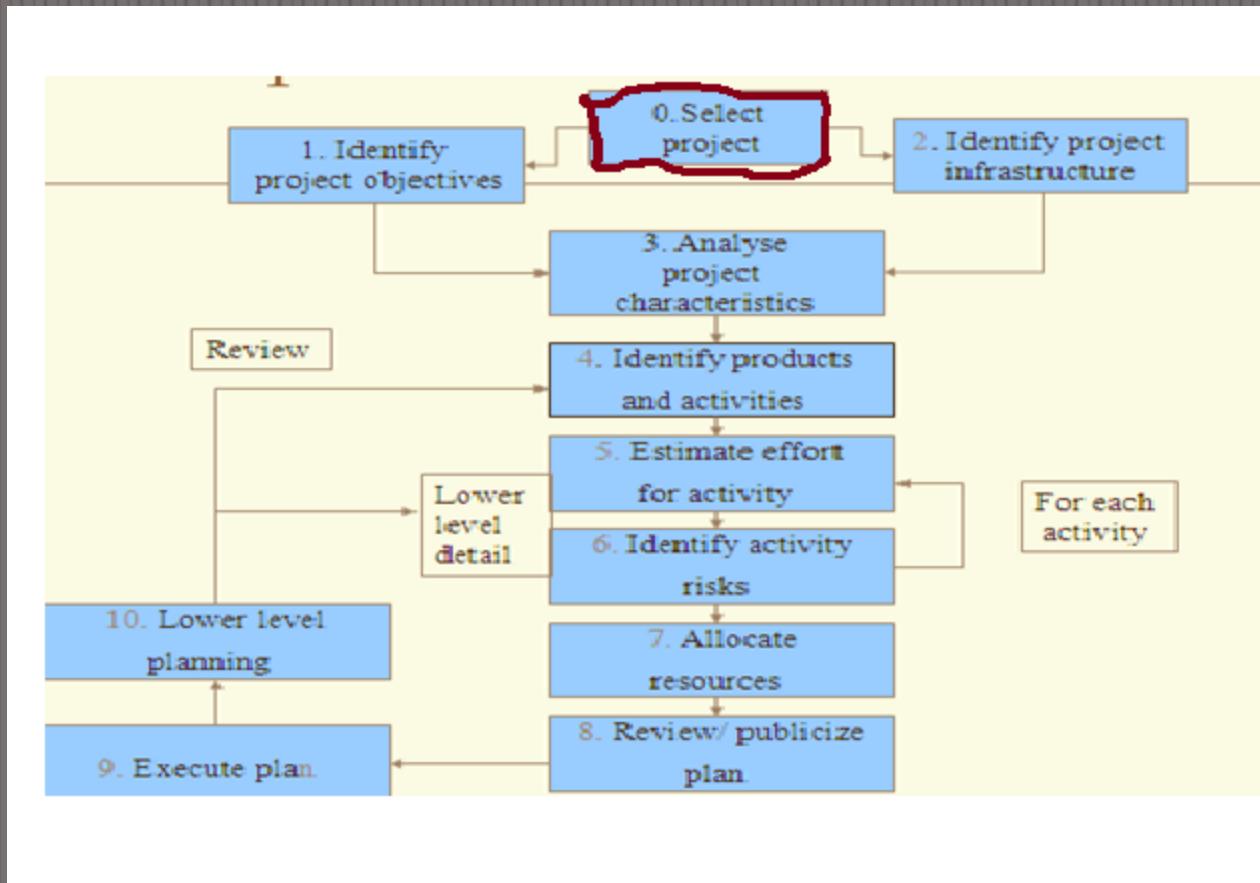


Chapter 3

Programme management and project evaluation

Programme management

- One definition:
'a group of projects that are managed in a co-ordinated way to gain benefits that would not be possible were the projects to be managed independently' Ferns



Evaluation

- Deciding to go ahead with project based on evaluation.
- Based on:
 - Strategic planning or a feasibility study
 - Technical assessment
 - Economic criteria
- So, project evaluation is carried out in Step 0 of Planning process.

Feasibility Study

- A **feasibility study** is a preliminary study undertaken to determine and document a project's viability.
- Results make a decision whether to proceed with the project.
- It is an analysis of possible alternative solutions to a problem and a recommendation on the best *alternative*.

Strategic Assessment

- Strategic plan-clearly defining the organizations objectives.
- Defining programme and programme goal.
- Project manager or programme director or programme executive be responsible for strategic assessment of a proposed project.

Technical Assessment

- Evaluating the required functionality against the hardware and software .
- This involves questions such as
 - Whether the technology needed for the system exists, how difficult it will be to build
 - Whether the firm has enough experience using that technology.
 - Based on an outline design of system requirements in terms of Input, Output, Fields, Programs.

Economic Criteria

- This involves questions such as
 - Whether the firm can afford to build the system, whether its benefits should substantially exceed its costs.
 - Whether the project has higher priority and profits than other projects that might use the same resources.
 - Also includes whether the project is in the condition to fulfill all the eligibility criteria and the responsibility of both sides in case there are two parties involved in performing any project.

Cost-benefit analysis

- Common way of carrying out an economic assessment.
- Comparing the expected costs of development and operation of the system with benefits.
- Assessment based on:
 - Whether the estimated costs are exceeded by estimated income and other benefits.
 - Whether or not the project under consideration is the best of a number of

Cost benefit analysis (CBA)

You need to:

- Identify all the costs which could be:
 - Development costs
 - Set-up
 - Operational costs
- Identify the value of benefits
- Check benefits are greater than costs

(cont..)

- Expressing these costs and benefits in common units:
 - evaluate the net profit
 - Net profit: difference between total benefit accruing from the system and the total cost creating and operating it.
- Common unit of measurement is “Money”.

Costs

- Helpful to categorize costs according to where they originate in the project.
- **Development costs:** Include the salaries and other employment costs of the staff.
- **Setup costs:** cost of putting the system into place.
 - Hardware , recruitment and staff training.
- **Operational costs:** cost of operating

Cash Flow Forecasting

- Indicate when expenditure and income will take place.
- Need to spend money, such as Staff wages during the development stages of project.
- Expenditure from company's resources or from bank.

(Cont..)

- Assumed that cash flows take place at the end of each year.
- Short term projects or candidate projects will be seasonal cash flow.
- Quarterly, or even monthly.

Cost-benefit evaluation techniques

- Proceed when benefits outweigh the cost.
- Methods for comparing projects on the basis of their cash flow forecasts:
 - Net profit
 - Payback period
 - Return on investment (ROI)
 - Net present value
 - Internal rate of return (IRR)

Net Profit

- Difference between the total costs and the total income over the life of the project.
- In simplistic terms, net profit is the money left over after paying all the expenses of an endeavor.

Net profit

Year	Cash-flow
0	-100,000
1	10,000
2	10,000
3	10,000
4	20,000
5	100,000
Net profit	50,000

‘Year 0’ represents all the costs before system is operation

‘Cash-flow’ is value of income less outgoing

Net profit value of all the cash-flows for the lifetime of the application

Table 3.2 *Four project cash flow projections – figures are end of year totals (£)*

<i>Year</i>	<i>Project 1</i>	<i>Project 2</i>	<i>Project 3</i>	<i>Project 4</i>
0	-100,000	-1,000,000	-100,000	-120,000
1	10,000	200,000	30,000	30,000
2	10,000	200,000	30,000	30,000
3	10,000	200,000	30,000	30,000
4	20,000	200,000	30,000	30,000
5	100,000	300,000	30,000	75,000

Net Profit

Table 3.2 *Four project cash flow projections – figures are end of year totals (£)*

<i>Year</i>	<i>Project 1</i>	<i>Project 2</i>	<i>Project 3</i>	<i>Project 4</i>
0	-100,000	-1,000,000	-100,000	-120,000
1	10,000	200,000	30,000	30,000
2	10,000	200,000	30,000	30,000
3	10,000	200,000	30,000	30,000
4	20,000	200,000	30,000	30,000
5	100,000	300,000	30,000	75,000
Net profit	50,000	100,000	50,000	75,000

Pay back period

This is the time taken to break even or pay back the initial investment.

Year	Cash-flow	Accumulated
0	-100,000	-100,000
1	10,000	-90,000
2	10,000	-80,000
3	10,000	-70,000
4	20,000	-50,000
5	100,000	50,000

Payback Period

- Period of time required for the return on an investment to "repay" the sum of the original investment .
- For example, a \$1000 investment which returned \$500 per year would have a two year payback period.
- Measure that describes how long something takes to "pay for itself"
- Shorter payback periods are obviously preferable than longer payback periods

Payback Period

Table 3.2 *Four project cash flow projections – figures are end of year totals (£)*

<i>Year</i>	<i>Project 1</i>	<i>Project 2</i>	<i>Project 3</i>	<i>Project 4</i>
0	-100,000	-1,000,000	-100,000	-120,000
1	10,000	200,000	30,000	30,000
2	10,000	200,000	30,000	30,000
3	10,000	200,000	30,000	30,000
4	20,000	200,000	30,000	30,000
5	100,000	300,000	30,000	75,000
Net profit	50,000	100,000	50,000	75,000
Payback Period	4.5 yrs	4.7 yrs	3.3 yrs	4
yrs				

Return on Investment (ROI)

- Also known as Accounting rate of return (ARR)
- Ratio of money gained or lost on an investment relative to the amount of money invested.
- The amount of money gained or lost may be referred to as **profit/loss**.
- The money invested may be referred to as the **asset, capital, principal**.

(Cont..)

$$\text{ROI} = \frac{\text{average annual profit}}{\text{Total investment}} * 100$$

ROI

Table 3.2 *Four project cash flow projections – figures are end of year totals (£)*

<i>Year</i>	<i>Project 1</i>	<i>Project 2</i>	<i>Project 3</i>	<i>Project 4</i>
0	-100,000	-1,000,000	-100,000	-120,000
1	10,000	200,000	30,000	30,000
2	10,000	200,000	30,000	30,000
3	10,000	200,000	30,000	30,000
4	20,000	200,000	30,000	30,000
5	100,000	300,000	30,000	75,000
Net profit	50,000	100,000	50,000	75,000
Payback Period yrs	4.5 yrs	4.7 yrs	3.3 yrs	4
ROI	10%	2 %	10 %	12.5%

Net present value

Would you rather I gave you £100 today or in 12 months time?

If I gave you £100 now you *could* put it in savings account and get interest on it.

If the interest rate was 10% how much would I have to invest now to get £100 in a year's time?

This figure is the *net present value* of £100 in one year's time

Net present value (NPV)

- Take into account the net profitability of a project and the timing of cash flows.
- It does so by discounting future cash flows with a percentage known as discount rate.
- Present value of any cash flow = value in year t

$$(1+r)^t$$

r-discount rate, t- no. of years into the future when the cash flow occurs.

(Cont..)

□ Present value of cash flow is calculated by multiplying the cash flow with discount factor.

□ Discount factor =
$$\frac{1}{(1+r)^t}$$

q The annual rate by which we discount future earnings is known as *discount rate*.

q Disadvantage: Difficulty for deciding appropriate discount rate.

Applying discount factors

Year	Cash-flow	Discount factor @ 10%	Discounted cash flow
0	-100,000	1.0000	-100,000
1	10,000	0.9091	9,091
2	10,000	0.8264	8,264
3	10,000	0.7513	7,513
4	20,000	0.6830	13,660
5	100,000	0.6209	62,090
		NPV	618

NPV

Table *Calculating the net present value of projects 1, 2, 3 and 4*

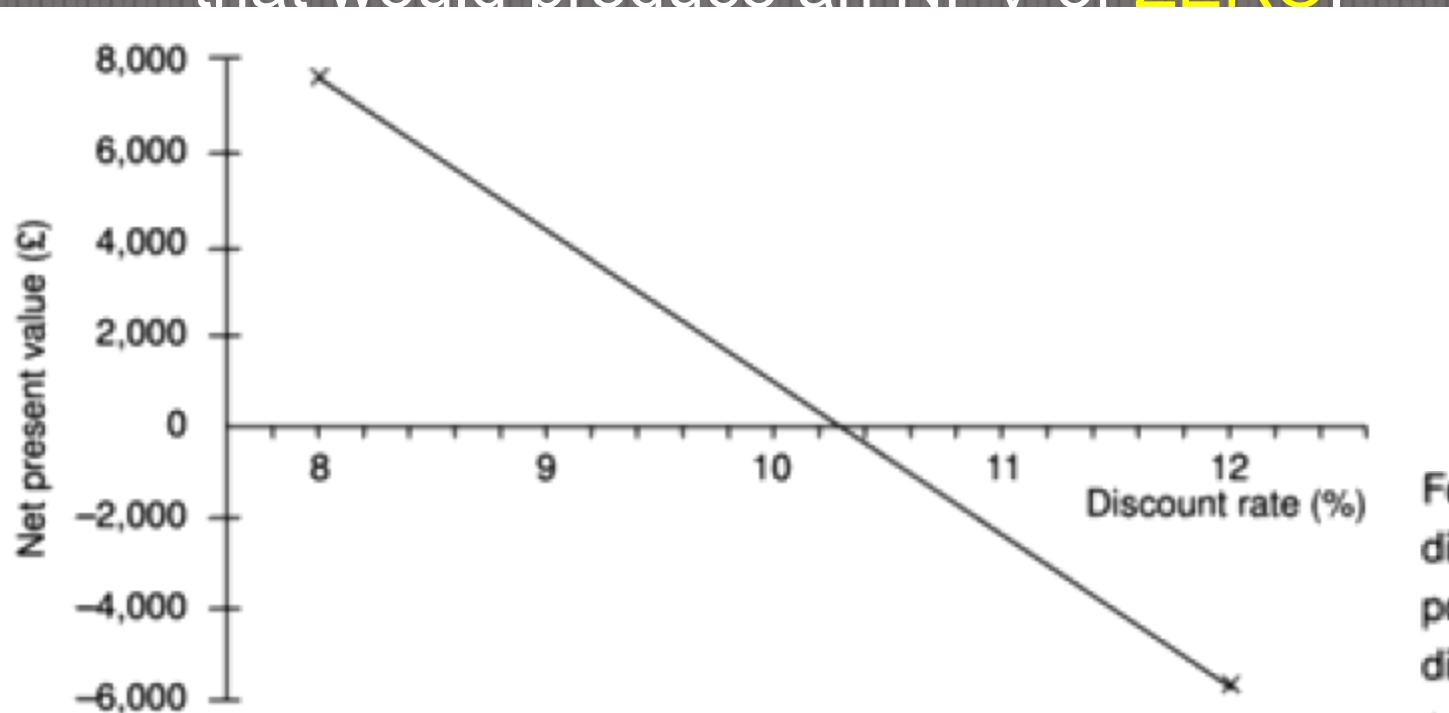
Year	Discount factor	Discounted cash flow (£)			
		Project 1	Project 2	Project 3	Project 4
0	1.0000	-100,000	-1,000,000	-100,000	-120,000
1	0.9091	9,091	181,820	27,273	27,273
2	0.8264	8,264	165,280	24,792	24,792
3	0.7513	7,513	150,260	22,539	22,539
4	0.6830	13,660	136,600	20,490	20,490
5	0.6209	62,090	186,270	18,627	46,568
NPV		£618	-179,770	13,721	21,662

Internal Rate of Return (IRR)

- It is a measure that compares the different rate of return of projects to analyze project profitability.

IRR

IRR is calculated as that percentage discount rate that would produce an NPV of **ZERO**.



Estimating the internal rate of return for project I.

For a particular project, a discount rate of 8% gives a positive NPV of £7,898; a discount rate of 12% gives a negative NPV of – £5,829. The IRR is therefore somewhere between these two values.

Table 3.5 *Three estimated project cash flows*

<i>Year</i>	<i>Project A (£)</i>	<i>Project B (£)</i>	<i>Project C (£)</i>
0	- 8,000	- 8,000	- 10,000
1	4,000	1,000	2,000
2	4,000	2,000	2,000
3	2,000	4,000	6,000
4	1,000	3,000	2,000
5	500	9,000	2,000
6	500	-6,000	2,000

Compute net profit, payback period, ROI, NPV with 8%, 10% and 12% discount rates and IRR .

For each of the discount rates, decide which is the best project. What you can conclude from these results?

Table F.3*The effect on net present value of varying the discount rate*

Year	Cash flow values (£)		
	Project A	Project B	Project C
0	-8,000	-8,000	-10,000
1	4,000	1,000	2,000
2	4,000	2,000	2,000
3	2,000	4,000	6,000
4	1,000	3,000	2,000
5	500	9,000	2,000
6	500	-6,000	2,000
Net Profit	£ 4,000	£ 5,000	£ 6,000
NPV @ 8%	£ 2,111	£ 2,365	£ 2,421
NPV @ 10%	£ 1,720	£ 1,818	£ 1,716
NPV @ 12%	£ 1,356	£ 1,308	£ 1,070

Payback Period**2 yrs****3.12 yrs****3 yrs****ROI****8.33%****10.42 %****10 %**

Risk evaluation ,identification and ranking.

- Attempt to identify the risks and their potential effects.
- Construct a risk matrix and classify each risk according to its relative importance and likelihood.
- Importance and likelihood is classified as high (H),medium (M), low (L) or exceeding unlikely (-).

Project Risk Matrix

Risk	Importance	Likelihood
Software never completed	H	-
Project cancelled after design	H	-
Maintenance costs higher than Expected.	L	L

Risk analysis using decision trees

- Decision Trees are excellent tools for helping you to choose between several courses of action
- Also help to form a balanced picture of the risks and rewards associated with each possible course of action.

Decision Trees and Expected Monetary Value (EMV)

- A **decision tree** is a diagramming analysis technique used to help select the best course of action in situations in which future outcomes are uncertain
- **Estimated monetary value (EMV)** is the product of a risk event probability and the risk event's monetary value
- You can draw a decision tree to help find the EMV

(Cont..)

- start a Decision Tree with a decision that you need to make.
- Draw a small square to represent this towards the left
- From this box draw out lines towards the right for each possible solution, and write that solution along the line
- Keep the lines apart as far as possible so that you can expand your thoughts.

(Cont..)

- At the end of each line, consider the results If the result of taking that decision is uncertain, draw a small circle.
- If the result is another decision that you need to make, draw another square.
- Squares represent decisions, and circles represent uncertain outcomes.



Figure 1:
Example Decision Tree:
Should we develop a new product or consolidate?

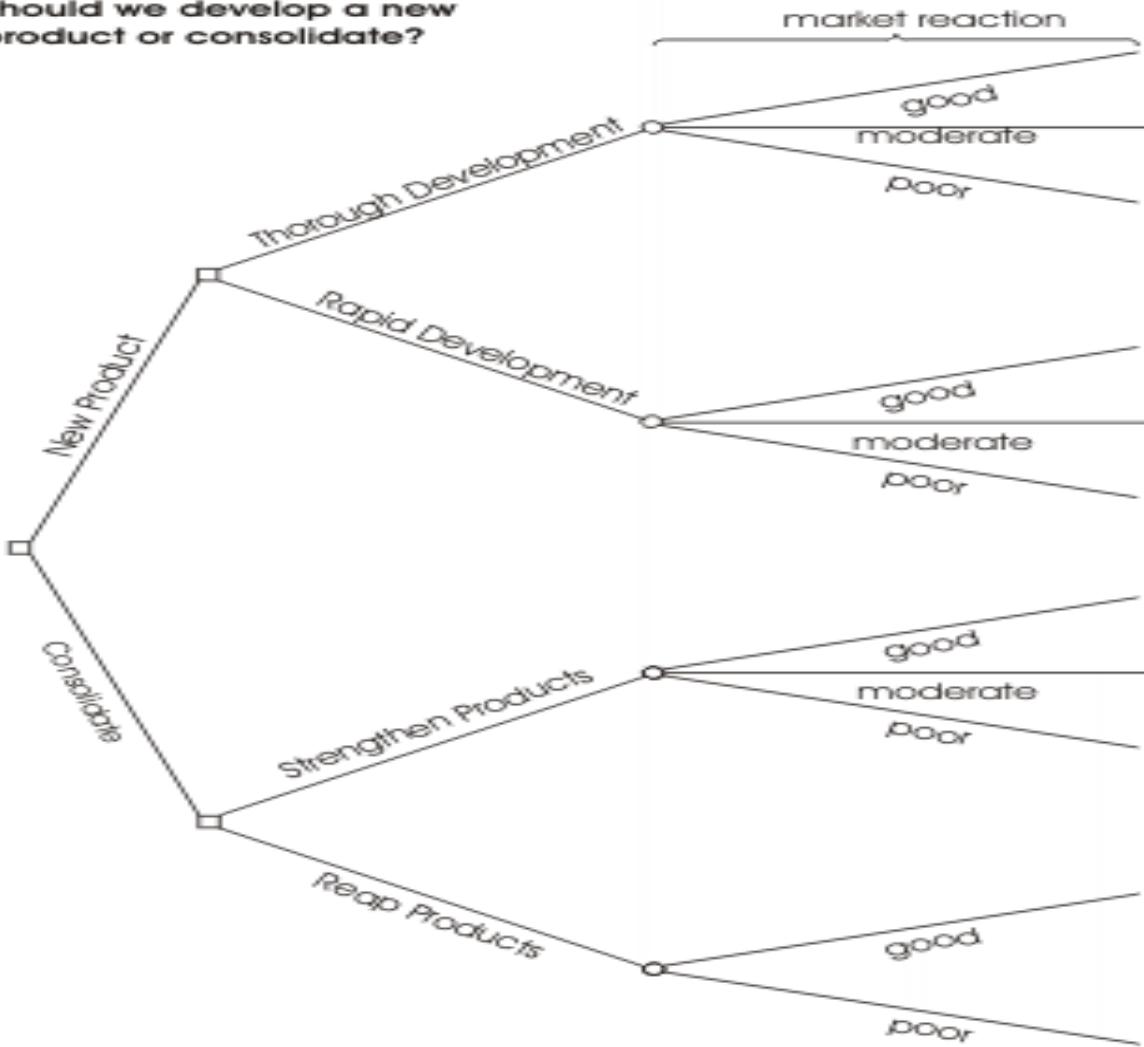
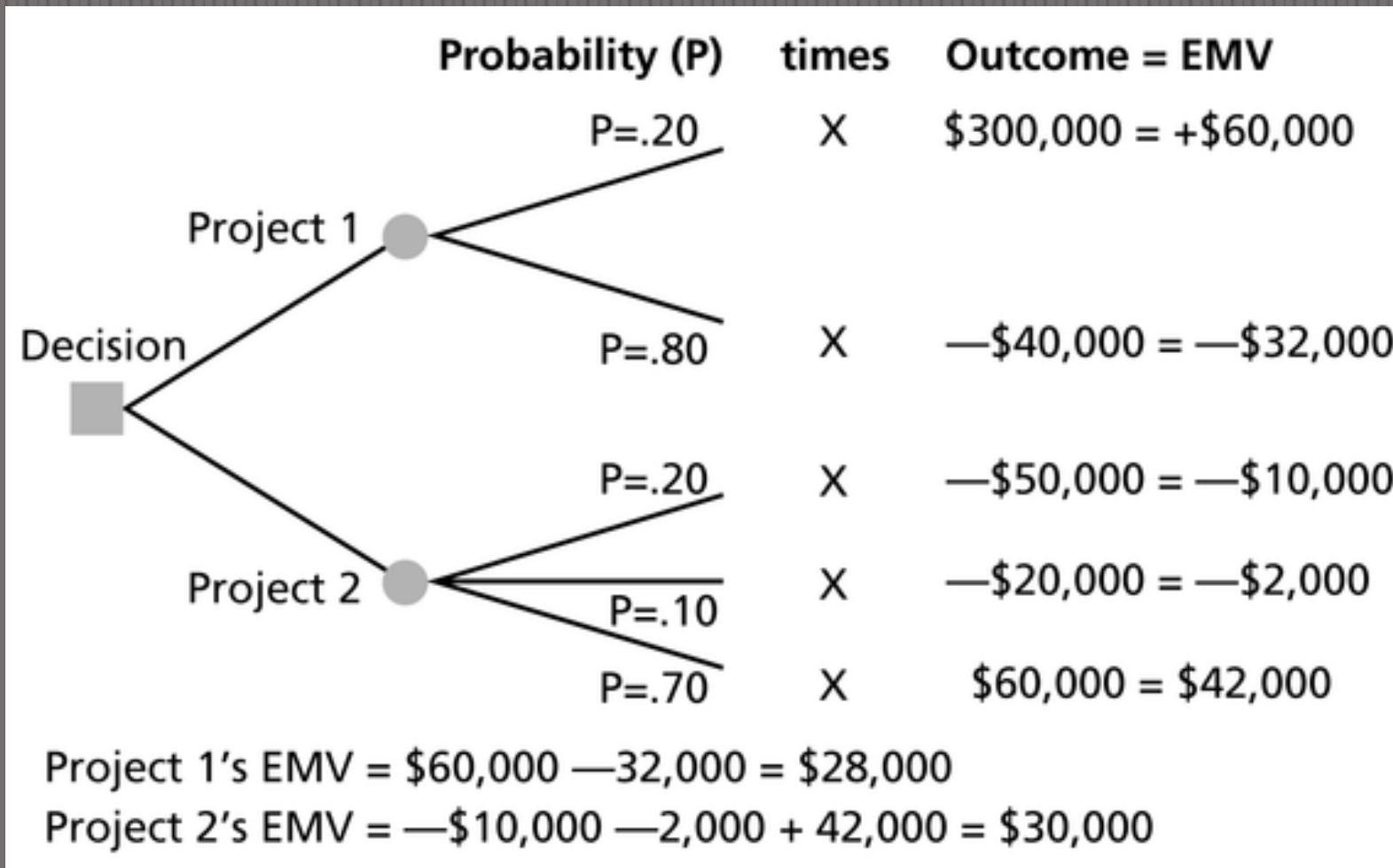
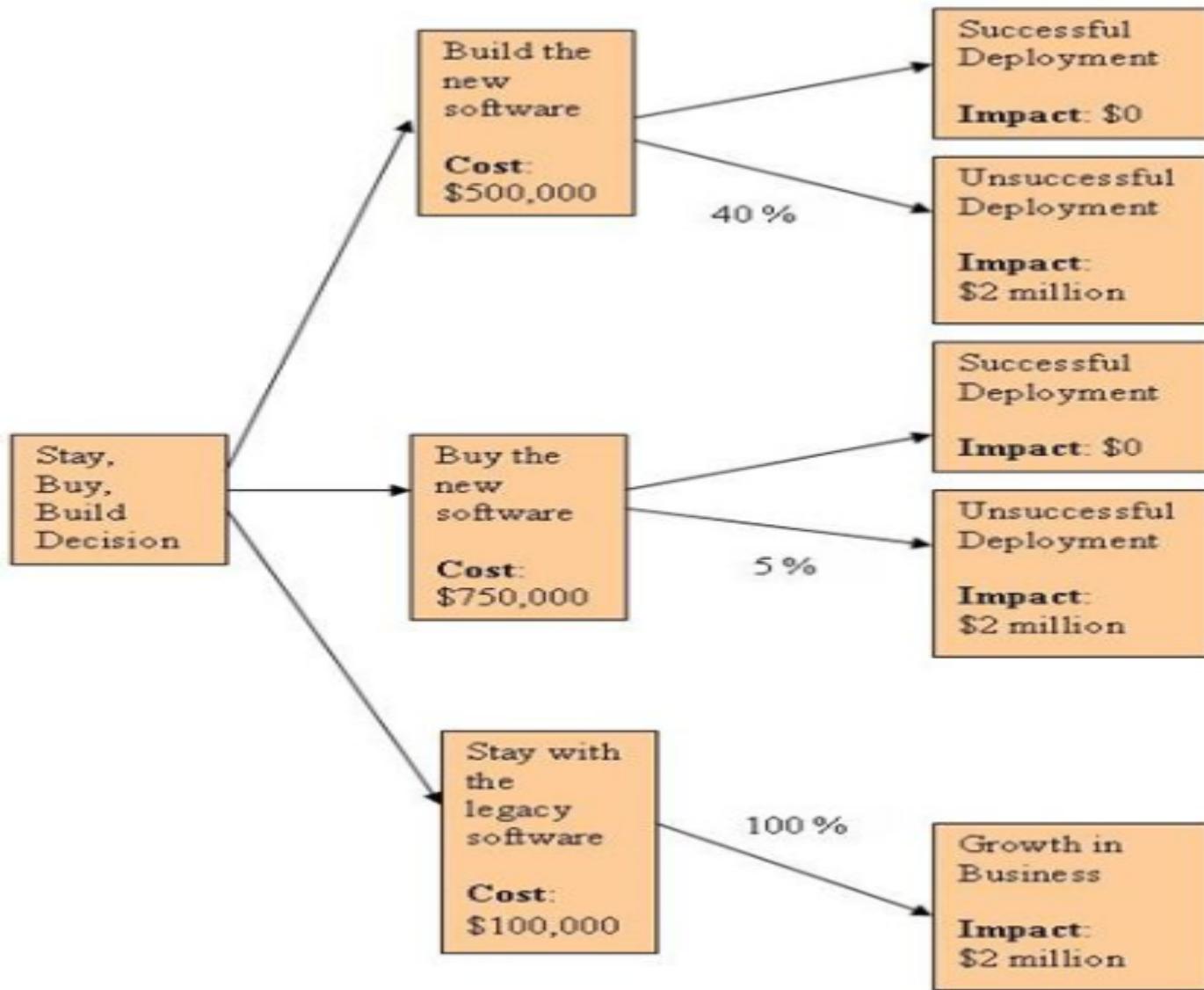


Figure 11-7: Expected Monetary Value (EMV) Example





Software Project Management

4th Edition



Chapter 4

Selection of an appropriate project approach

Selection of project approaches

- In-house development: most of these issues resolved by IS planning and standards
- Software houses: more applicable as different customers have different needs
- Selection of approach governed by:
 - uncertainties of the project
 - properties of application to be built

General approach

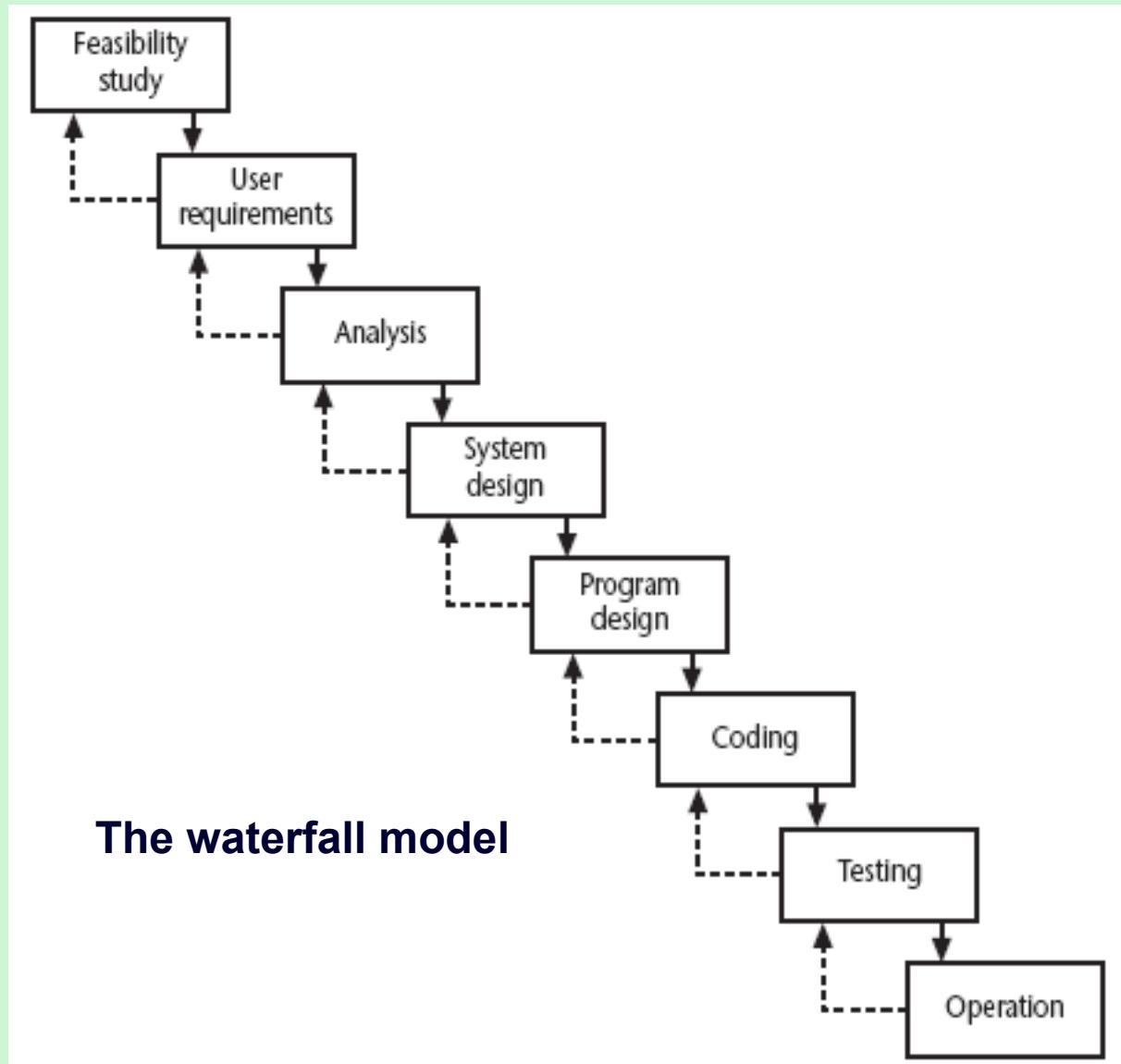
- Look at risks and uncertainties e.g.
 - are requirements well understood?
 - are technologies to be used well understood?
- Look at the type of application being built e.g.
 - information system? embedded system?
 - criticality? differences between target and development environments?
- Clients' own requirements
 - need to use a particular method

Choice of process models

- ‘waterfall’ also known as ‘one-shot’, ‘once-through’
- incremental delivery
- evolutionary development

Also use of ‘agile methods’ e.g. extreme programming

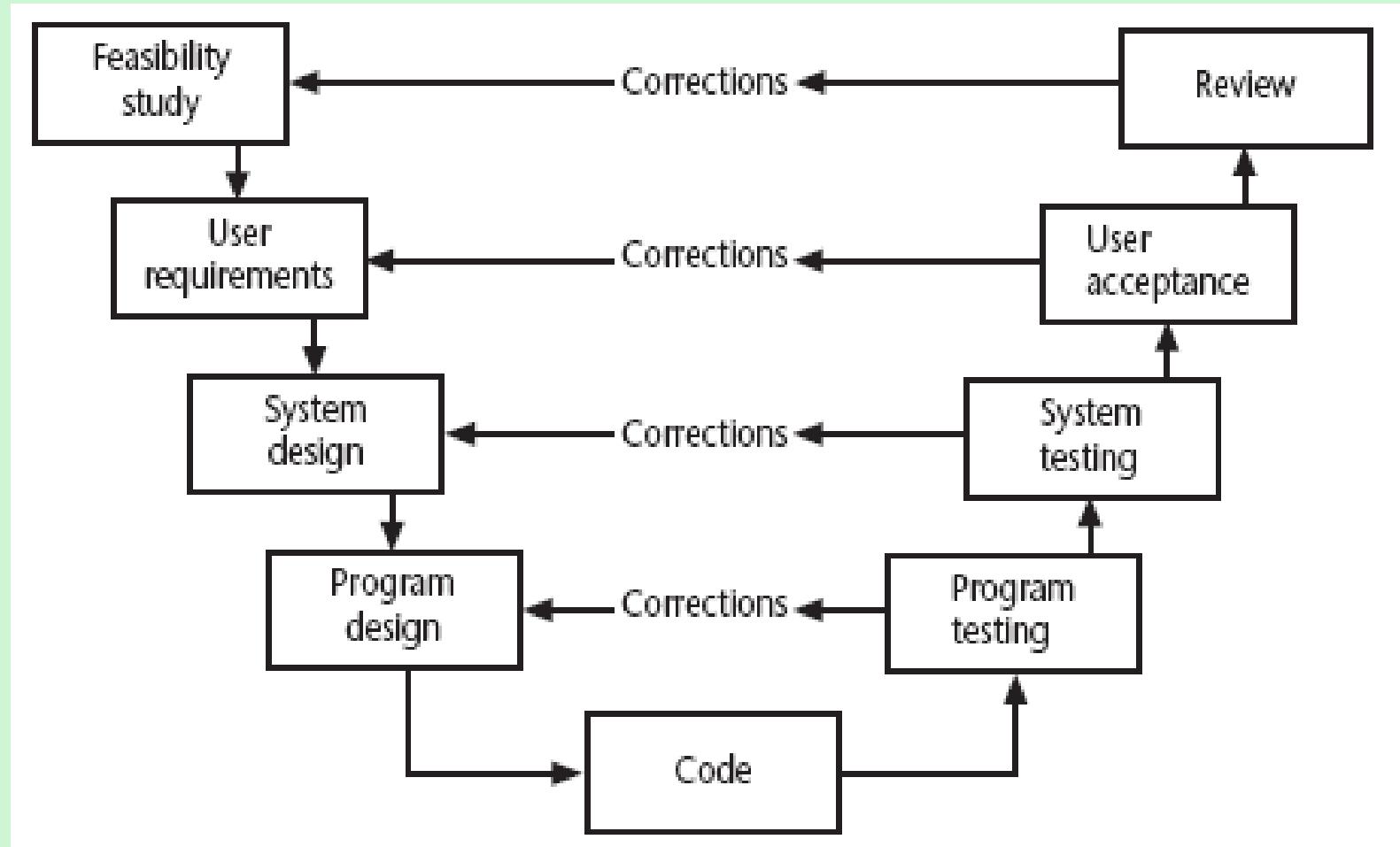
Waterfall



Waterfall

- the ‘classical’ model
- imposes structure on the project
- every stage needs to be checked and signed off
- BUT
 - limited scope for iteration

V-process model



Another way of looking at the waterfall model

Evolutionary delivery: prototyping

- ‘An iterative process of creating quickly and inexpensively live and working models to test out requirements and assumptions’

Sprague and McNurlin main types

- ‘throw away’ prototypes
- evolutionary prototypes

what is being prototyped?

- human-computer interface
- functionality

Reasons for prototyping

- learning by doing
- improved communication
- improved user involvement
- a feedback loop is established
- reduces the need for documentation
- reduces maintenance costs i.e. changes after the application goes live
- prototype can be used for producing expected results

Prototyping: some dangers

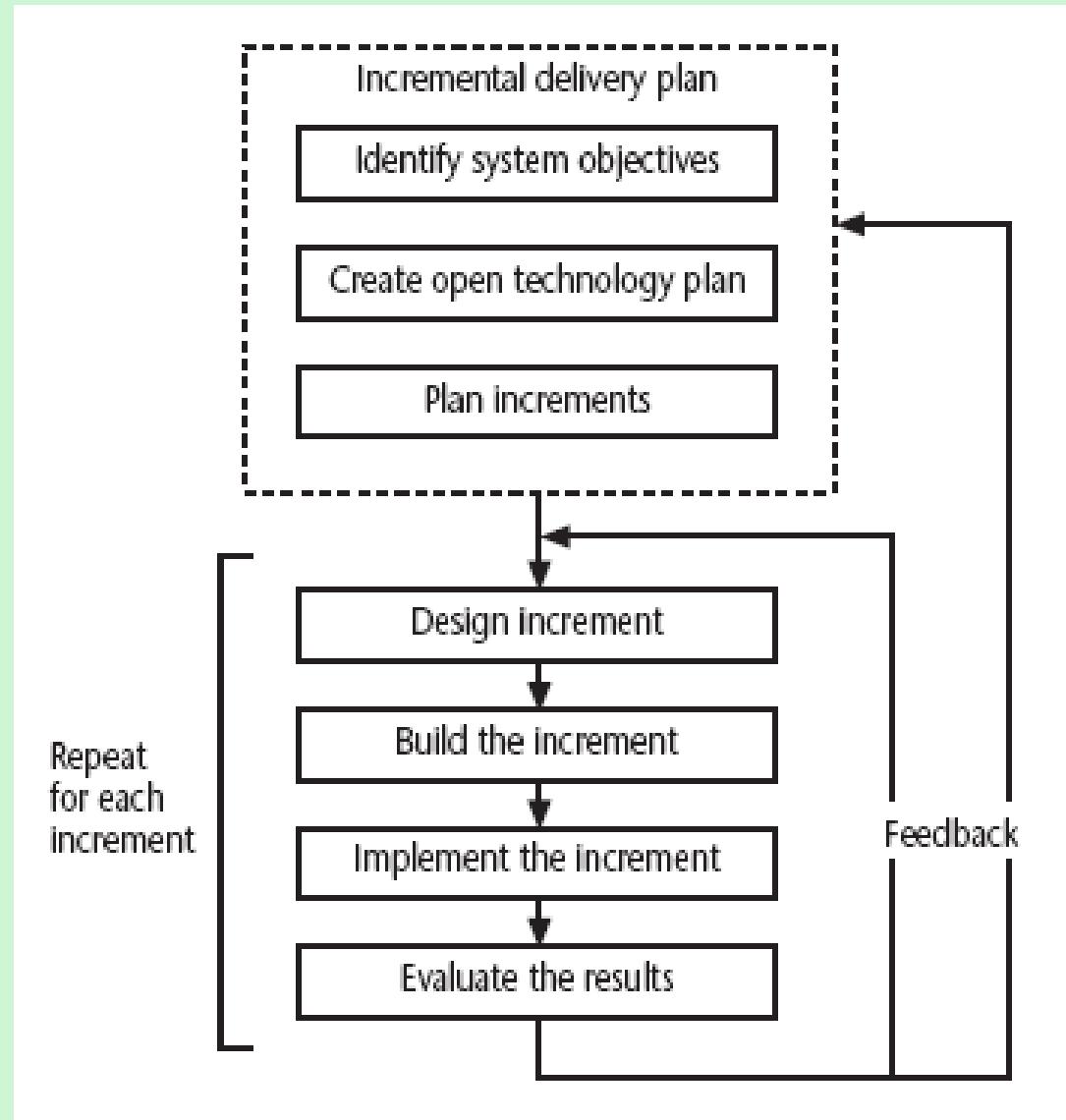
- users may misunderstand the role of the prototype
- lack of project control and standards possible
- additional expense of building prototype
- focus on user-friendly interface could be at expense of machine efficiency

Other ways of categorizing prototyping

- what is being learnt?
 - organizational prototype
 - hardware/software prototype ('experimental')
 - application prototype ('exploratory')
- to what extent
 - mock-ups
 - simulated interaction
 - partial working models: *vertical* versus *horizontal*

The incremental process

**Intentional
incremental
delivery**



Incremental approach:benefits

- feedback from early stages used in developing latter stages
- shorter development thresholds
- user gets some benefits earlier
- project may be put aside temporarily
- reduces ‘gold-plating’:

BUT there are some possible disadvantages

- loss of economy of scale
- ‘software breakage’

The outline incremental plan

- steps ideally 1% to 5% of the total project
- non-computer steps should be included
- ideal if a step takes one month or less:
 - not more than three months
- each step should deliver some benefit to the user
- some steps will be physically dependent on others

Which step first?

- some steps will be pre-requisite because of physical dependencies
- others may be in any order
- value to cost ratios may be used
 - V/C where
 - V is a score 1-10 representing value to customer
 - C is a score 0-10 representing value to developers

V/C ratios: an example

‘Agile’ methods

structured development methods have some perceived advantages

- produce large amounts of documentation which can be largely unread
- documentation has to be kept up to date
- division into specialist groups and need to follow procedures stifles communication
- users can be excluded from decision process
- long lead times to deliver anything etc. etc

The answer? ‘Agile’ methods?

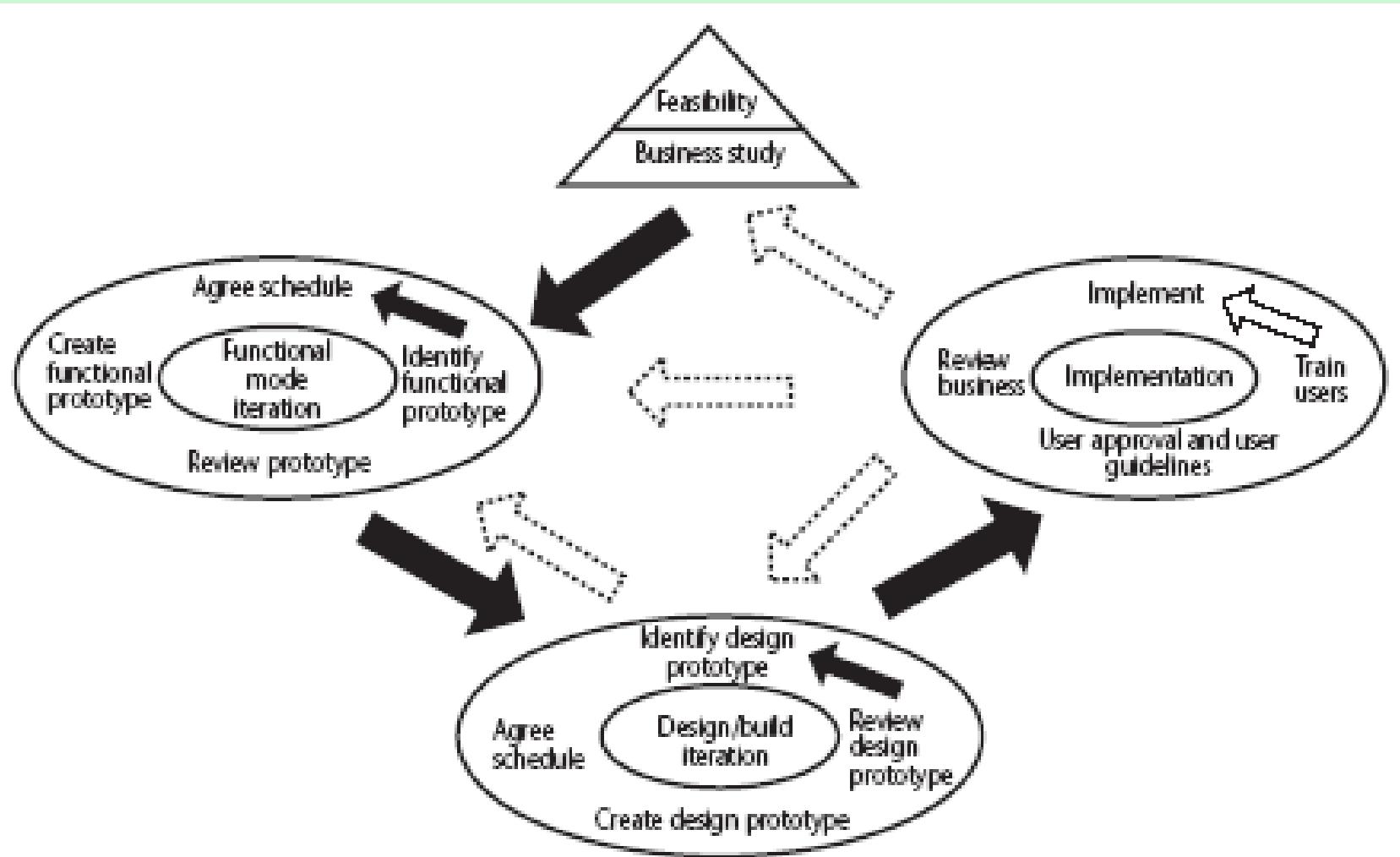
Dynamic system development method

- UK-based consortium
- *arguably* DSDM can be seen as replacement for SSADM
- DSDM is more a project management approach than a development approach
- Can still use DFDs, LDSs etc!

Nine core DSDM principles

1. Active user involvement
2. Teams empowered to make decisions
3. Frequent delivery of products
4. Fitness for *business purpose*
5. Iterative and incremental delivery
6. Changes are reversible
7. Requirements base-lined at a *high level*
8. Testing integrated with development
9. Collaborative and co-operative approach

DSDM framework



DSDM process model

DSDM: time-boxing

- *time-box* fixed deadline by which *something* has to be delivered
- typically two to six weeks
- MOSCOW priorities
 - Must have - essential
 - Should have - very important, but system could operate without
 - Could have
 - Want - but probably won't get!

Extreme programming

- increments of one to three weeks
 - customer can suggest improvement at any point
- argued that distinction between design and building of software are artificial
- code to be developed to meet current needs only
- frequent re-factoring to keep code structured

Extreme programming - contd

- developers work in pairs
- test cases and expected results devised before software design
- after testing of increment, test cases added to a consolidated set of test cases

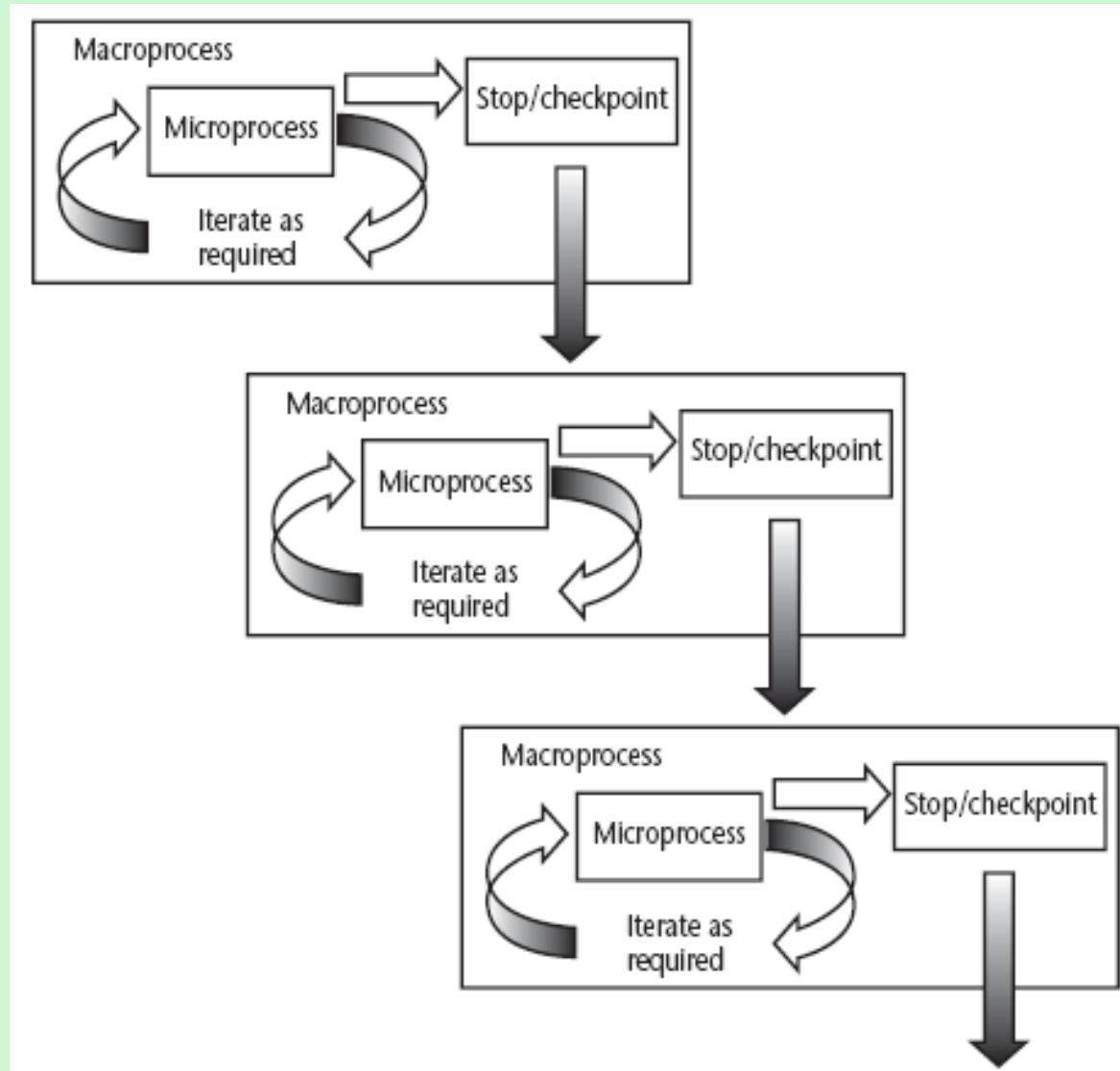
Grady Booch's concern

Booch, an OO authority, is concerned that with requirements driven projects:

'Conceptual integrity sometimes suffers because this is little motivation to deal with scalability, extensibility, portability, or reusability beyond what any vague requirement might imply'

Tendency towards a large number of discrete functions with little common infrastructure?

Macro and micro processes



A macro process containing three iterative micro processes

©The McGraw-Hill Companies

Combinations of approach

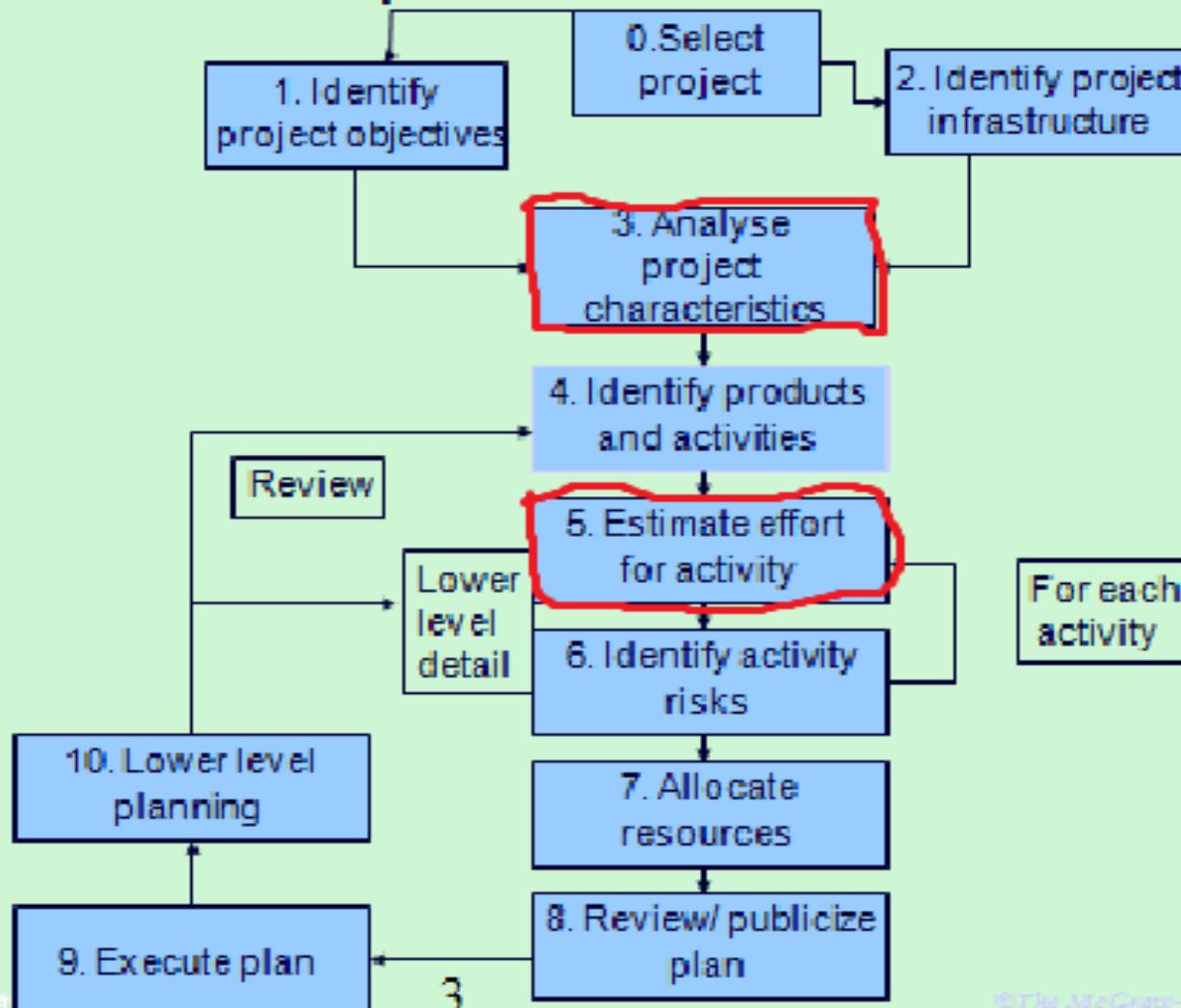
	one-shot	incremental	evolutionary
one-shot	yes	yes	no
incremental	yes	yes	no
evolutionary	yes	yes	yes

one-shot or incremental installation - any construction approach possible

evolutionary installation implies evolutionary construction

Software Effort Estimation

'Step Wise' - an overview



What makes a successful project?

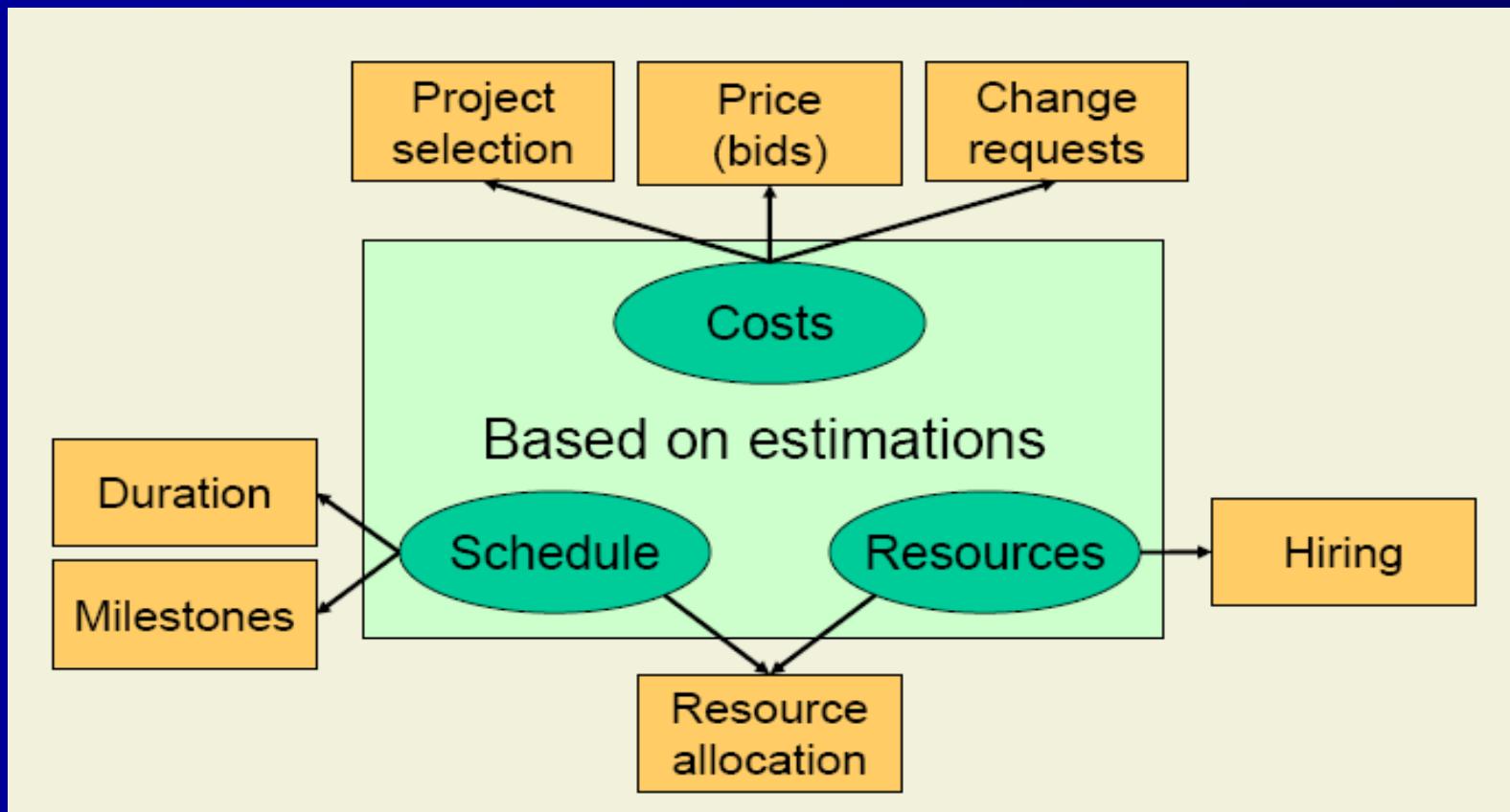
Delivering:

- n agreed functionality
- n on time
- n at the agreed cost
- n with the required quality

Stages:

1. set targets
2. Attempt to achieve targets

Why estimations?



What is estimation?

- n The project manager must set expectations about the time required to complete the software among the stakeholders, the team, and the organization's management.
- n If those expectations are not realistic from the beginning of the project, the stakeholders will not trust the team or the project manager.

Software effort estimation

- n Predicting the resources required for a software development process
- n i.e. Effort estimation consists in predict how many hours of work and how many workers are needed to develop a project.

Fundamental estimation questions

- „ How much effort is required to complete an activity?
- „ How much calendar time is needed to complete an activity?
- „ What is the total cost of an activity?

Software cost components

- n Hardware and software costs
- n Travel and training costs
- n Effort costs (the dominant factor in most projects)
 - salaries of engineers involved in the project
 - Social and insurance costs
- n Effort costs must take overheads into account
 - costs of building
 - costs of networking and communications
 - costs of shared facilities (e.g library, staff restaurant, etc.)

Where are estimation done?

- n Strategic planning
- n Feasibility Study
- n System Specification
- n Evaluation of supplier's proposals
- n Project Planning

Measure of Work

- n Time required vary with experience.
- n Effort is measured using a measure called SLOC.
- n Used to measure the size of a software program by counting the number of lines in the text of the program's source code
- n SLOC is typically used to predict the amount of effort that will be required to develop a program.

SLOC (Cont...)

- n To estimate programming **productivity** or **effort** once the software is produced.
- n Productivity= system size / Effort.
- n Effort=system size/productivity

KLOC

- n KLOC stands for Thousands (kilo) lines of code.
- n The size is determined by measuring the number of lines of source code a program has .

Problems with over and under-estimates

- n **Parkinson's Law:** 'work expands so as to fill the time available for its completion'.
- n Implies when given an easy target staff will work less hard.
- n **Brook's Law:** ' Putting more people on a late job makes it later'.
- n Danger with the under estimate is the effect on quality.

(cont..)

- n Under-estimating
 - Under- Staffing resulting in staff burnout
 - Setting too short schedule results in loss of credibility as deadlines are missed
- n Over-estimating
 - Projects cost more than they should resulting in unnecessary cost
 - Projects take longer to deliver resulting in lost opportunities

Calculate the productivity (that is, SLOC per work month) of each of the projects in Table 5.1 and also for the organization as a whole. If the project leaders for projects a and d had correctly estimated the source number of lines of code (SLOC) and then used the average productivity of the organization to calculate the effort needed to complete the projects, how far out would their estimates have been from the actual effort?

Table 5.1 *Some project data – effort in work months (as percentage of total effort in brackets)*

<i>Project</i>	<i>Design</i>		<i>Coding</i>		<i>Testing</i>		<i>Total</i>	
	<i>wm</i>	(%)	<i>wm</i>	(%)	<i>wm</i>	(%)	<i>wm</i>	<i>SLOC</i>
a	3.9	(23)	5.3	(32)	7.4	(44)	16.7	6050
b	2.7	(12)	13.4	(59)	6.5	(26)	22.6	8363
c	3.5	(11)	26.8	(83)	1.9	(6)	32.2	13334
d	0.8	(21)	2.4	(62)	0.7	(18)	3.9	5942
e	1.8	(10)	7.7	(44)	7.8	(45)	17.3	3315
f	19.0	(28)	29.7	(44)	19.0	(28)	67.7	38988
g	2.1	(21)	7.4	(74)	0.5	(5)	10.1	38614
h	1.3	(7)	12.7	(66)	5.3	(27)	19.3	12762
i	8.5	(14)	22.7	(38)	28.2	(47)	59.5	26500

<i>Project</i>	<i>Work-months</i>	<i>SLOC</i>	<i>Productivity</i> (<i>SLOC/month</i>)
a	16.7	6050	362
b	22.6	8363	370
c	32.2	13334	414
d	3.9	5942	1524
e	17.3	3315	192
f	67.7	38988	576
g	10.1	38614	3823
h	19.3	12762	661
i	59.5	26500	445
Overall	249.3	153868	617

Table F.5 *Estimated effort*

<i>Project</i>	<i>Estimated work-months</i>	<i>Actual</i>	<i>Difference</i>
a	$6050/617 = 9.80$	16.7	6.90
d	$5942/617 = 9.63$	3.9	-5.73

There would be an under-estimate of 6.9 work-months for project *a* and an over-estimate of 5.7 for project *d*.

Software Effort Estimation Techniques

- n Expert Judgement
- n Analogy
- n Parkinson
- n Price to win
- n Top-down estimating
- n Bottom-up estimating
- n Algorithmic models.

Expert Judgement

- n Estimate is based on experience.
- n Involve experts in
 - Development techniques
 - Application Domain
- n Asking someone who is knowledgeable.
- n Method used when estimating the effort needed to change an existing piece of software.

Price to win

- n The ‘estimate’ is a figure that appears to be sufficiently low to win a contract.

Parkinson

- n Identifies the staff effort available to do a project and use that as the estimate.

Estimating by Analogy

- n Analogy: where a similar completed project is identified and its actual effort forms the basis for new project.
- n Use of analogy also called **case-based reasoning**.
- n **Source cases** - project that already completed
- n **Target case** – the new project.
- n Calculate the base estimate (target case – source case)
- n Can be done by software named ANGEL.
- n Identifies the source that is nearest to target by measuring Euclidean Distance between cases.

Estimating by Analogy (cont..)

- n Distance=square-root of $[(\text{target_parameter}_1 - \text{source_parameter}_1)^2 + \dots + (\text{target_parameter}_n - \text{source_parameter}_n)^2]$
- n The cost of a project is computed by comparing the project to a similar project in the same application domain
- n Advantages: Accurate if project data available
- n Disadvantages: Impossible if no comparable project has been tackled.

Say that the cases are being matched on the basis of two parameters, the number of inputs to and the number of outputs from the system to be built. The new project is known to require 7 inputs and 15 outputs. One of the past cases, Project A, has 8 inputs and 17 outputs. The Euclidean distance between the source and the target is therefore the square-root of $((7-8)^2 + (17-15)^2)$, that is 2.24.

Project B has 5 inputs and 10 outputs. What would be the Euclidean distance between this project and the target new project being considered above? Is Project B a better analogy with the target than Project A?

The Euclidean distance between Project B and the target case

$$\begin{aligned} &= \sqrt{(7 - 5)^2 + (15 - 10)^2} \\ &= \sqrt{2^2 + 5^2} \\ &= 5.39 \end{aligned}$$

Project A is therefore a closer analogy.

Bottom up estimating

- n The estimator breaks the projects into component task
- n These component tasks are sized.
- n calculate the effort required for each task
- n Bottom up estimation is suitable when a project is completely novel or no historical data is available.
- n Otherwise, assumption have to be made about the characteristics of the final system like number and size of software modules.

Bottom-up estimating

1. Break project into smaller and smaller components
2. Stop when you get to what one person can do in one/two weeks
3. Estimate costs for the lowest level activities
4. At each higher level calculate estimate by adding estimates for lower levels

Top-down approach and parametric models

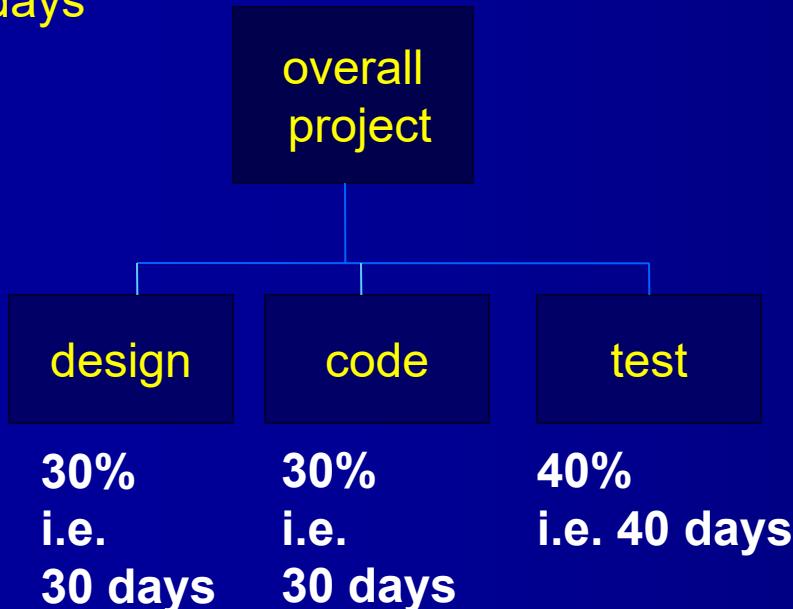
- n Associated with parametric models.
- n E.g. House-owner who need to rebuild the house with insurance coverage.
- n Insurance companies provide tables where owner can find an estimate based on parameters such as no. of storeys and the floor space.

(cont...)

- n Effort based on the characteristics of the final system.
- n Effort = system size * productivity rate
- n Once effort is calculated, allocate portion of that effort to various activities within that project.

Top-down estimates

Estimate
100 days



- Produce overall estimate using effort driver (s)
- distribute proportions of overall estimate to components

Bottom-up versus top-down

- **Bottom-up**
 - use when no past project data
 - identify all tasks that have to be done – so quite time-consuming
 - use when you have no data about similar past projects
- **Top-down**
 - produce overall estimate based on project cost drivers
 - based on past project data
 - divide overall estimate between jobs to be done

Albrecht function point analysis

- n Function point metric was devised by A.J. Albrecht.
- n Means of measuring software size and *productivity*.
- n FPA ,recognized method to measure the functional size of an information system.
- n The unit of measurement is "function points".

$$FP = \text{count total} [0.65 + 0.01 \Sigma(F_i)]$$

where count total is the sum of all FP entries

- n The F_i ($i = 1$ to 14) are "complexity adjustment values" based on responses to some questions

FPA's(Cont..)

- n Basis of FPA is that computer based information system consists five major components or external user types.
- n **External input types**: Input transactions that update internal computer files.
- n **External output types**: Where data is output to the user.E.g. printed reports.

(cont...)

- n Logical internal file type: standing files used by the system.

- n E.g. Purchase order.

record
types

Purchase order purchase
details order item

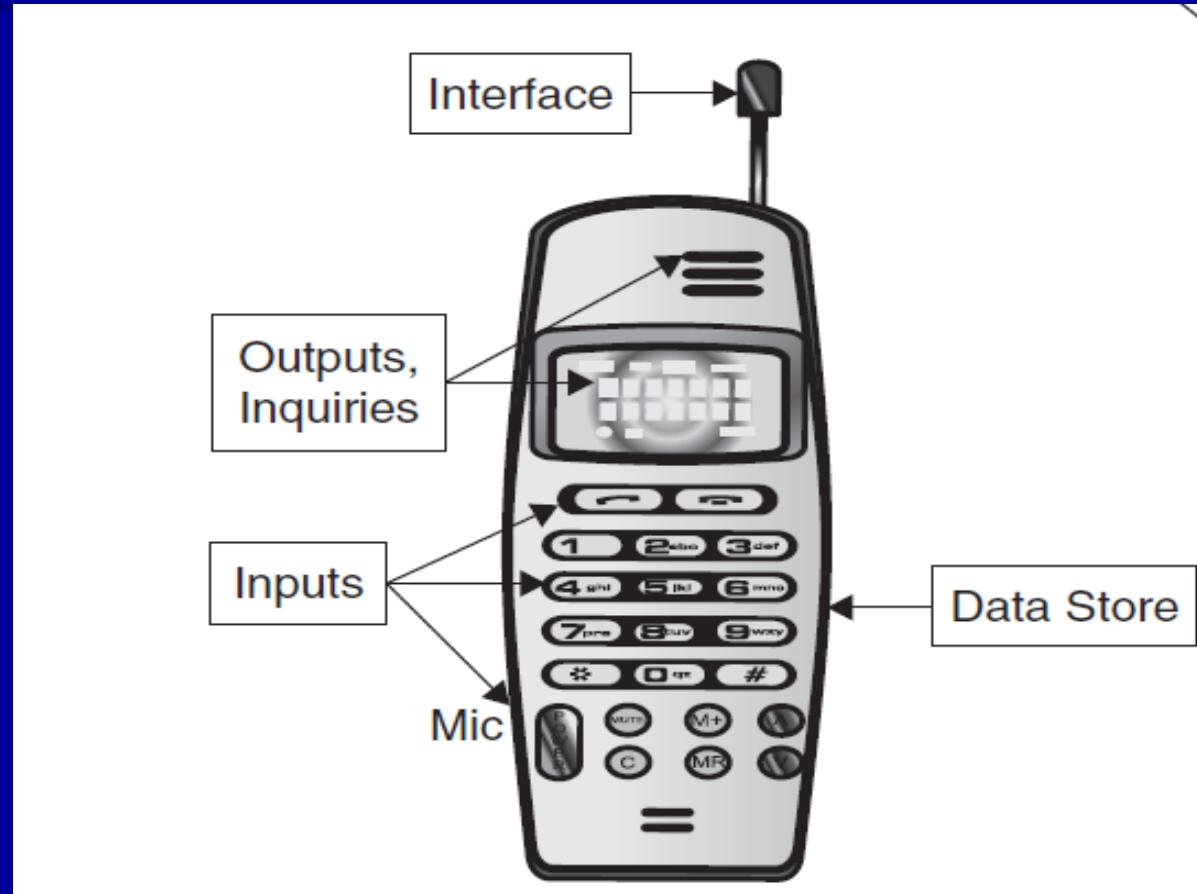
(cont...)

- n Data types in Purchase order details include: order number, supplier reference , order date.
- n Data types in Purchase order items include product code, unit price and number ordered.
- n **External interface file types**: Output and input that might pass to and from other computer application.

(Cont..)

- n **External Inquiry types**: transactions initiated by user that provide information but do not update the internal files.
- n After identifying External user types each component is classified as either high, low or average complexity.

Example.....



- n **Number of user inputs:** Each user input that provides distinct application oriented data to the software is counted.
- n **Number of user outputs:** Each user output that provides application oriented information to the user is counted. In this context "output" refers to reports, screens, error messages, etc. Individual data items within a report are not counted separately.
- n **Number of user inquiries:** An inquiry is defined as an on-line input that results in the generation of some immediate software response in the form of an on-line output. Each distinct inquiry is counted.
- n **Number of files:** Each logical master file is counted.
- n **Number of external interfaces:** All machine-readable interfaces that are used to transmit information to another system are counted.

Alternatively the following questionnaire could be utilized

- Does the system require reliable backup and recovery?
- Are data communications required?
- Are there distributed processing functions?
- Is performance critical?
- Will the system run in an existing, heavily utilized operational environment?
- Does the system require on-line data entry?
- Does the on-line data entry require the input transaction to be build over multiple screens or operations?
- Are the master files updated online?
- Are the input, outputs, files or inquiries complex?
- Is the internal processing complex?
- Is the code designed to be reusable?
- Are conversions and installation included in the design?
- Is the system designed for multiple installations in different organizations?
- Is the applications designed to facilitate change and ease of use?

Albrecht complexity multiplier

n	External user type	low	average	high
n	External input type	3	4	6
n	External output type	4	5	7
n	Logical internal file	7	10	15
n	External interface file	5	7	10
n	External inquiry type	3	4	6

Albrecht FP's now referred as IFPUG

(Cont..)

- n Count of external user type is multiplied by specified weights which are then summed to obtain overall FP count.
- n Lines of code for languages:
- n COBOL-91
- n C-128
- n QuickBasic- 64

Weighting factor

Measurement parameter	Count	Simple	Average	Complex	=	
Number of user inputs	<input type="text"/>	×	3	4	6	= <input type="text"/>
Number of user outputs	<input type="text"/>	×	4	5	7	= <input type="text"/>
Number of user inquiries	<input type="text"/>	×	3	4	6	= <input type="text"/>
Number of files	<input type="text"/>	×	7	10	15	= <input type="text"/>
Number of external interfaces	<input type="text"/>	×	5	7	10	= <input type="text"/>
Count total	<hr/>					<input type="text"/>

Example question

External user types	Count	Complexity
Number of user inputs	3	simple
Number of user outputs	10	average
Number of user inquiries	12	simple
Number of files	8	complex
Number of external interfaces	3	average

IFPUG file type complexity

Number of record types	Number of data types		
	<20	20-50	>50
1	low	low	average
2-5	low	average	high
>5	average	high	high

IFPUG external input type complexity

Number of file types	Number of data types accessed		
	<5	5-15	>15
n 0 or 1	low	low	average
n 2	Low	average	high
n >2	average	high	high

IFPUG external output type complexity

Number of file	Number of data types		
	<6	6-19	>19
n 0 or 1	low	low	average
n 2 or 3	low	average	high
n >3	average	high	high

Function Point Mark II

- n Mark II method recommended by CCTA(Central computer and Telecommunication Agency)
- n Here assumption is that any transaction comprises the basic structure.

The task for which Brigette has been made responsible in Exercise 5.2 needs a program that will extract yearly salaries from the payroll file, and the details of courses and hours taught on each course by each member of staff from two files maintained by the time-tabling system. The program will calculate the staff costs for each course and put the results into a file that will then be read by the main accounting system. The program will also produce a report showing for each course the hours taught by each member of staff and the cost of those hours.

Using the method described above, calculate the Albrecht function points for this subsystem, assuming that the report is of high complexity, but that all the other elements are of average complexity.

The function types are as follows.

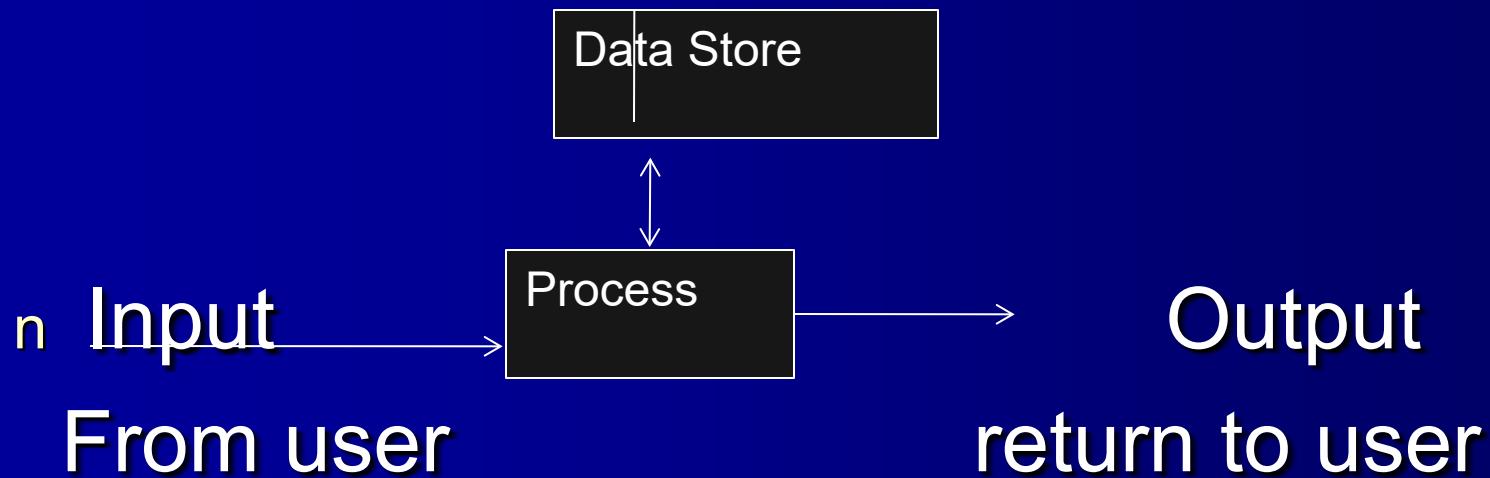
External input types	none
External output types	the report, 1
Logical internal file types	the accounting feeder file, 1
External interface file types	payroll file, staff file (timetabling,) courses file (timetabling,) accounting feeder file, 4
External inquiry types	none

Because the accounting feeder file is outgoing, it is counted once as a logical internal file type and once as an external interface file type.

External enquiry types	none
External output types	$1 \times 7 = 7$
Logical internal file types	$1 \times 10 = 10$
External interface types	$4 \times 7 = 28$
External inquiry types	none
Total	45

Estimated lines of Cobol = $45 \times 91 = 4095$

(cont..)



(Cont..)

- n **For each Transaction:**
- n $W_i * (\text{number of input data element types}) +$
- n $W_e * (\text{number of entity types referenced}) +$
- n $W_o * (\text{number of output data element types})$

Weightings

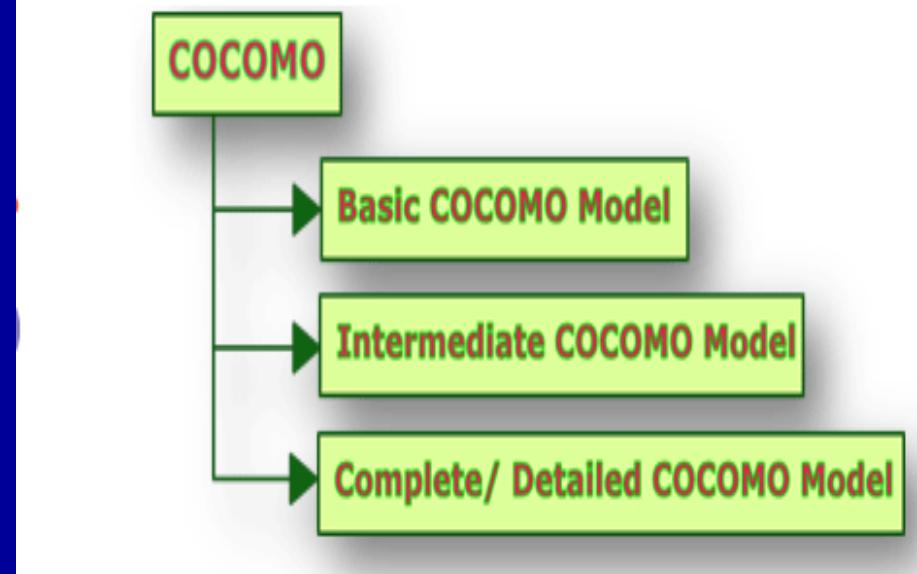
- n W_i, W_e, w_o are derived by asking developers what proportion of effort has been spent in previous projects.
Industry averages are available i.e.
- n 0.58 for W_i
- n 1.66 for W_e
- n 0.26 for w_o .

COCOMO : a parametric model

- n **Barry Boehm** designed **COCOMO**
- n It give an estimate of the number of man-months it will take to develop a software product.

COCOMO: The COCOMO (Constructive Cost Estimation Model) is proposed by **DR. Berry Boehm** in **1981** and that's why it is also known as COCOMO'81. It is a method for evaluating the cost of a software package. According to him software cost estimation should be done through three stages:

1. Basic COCOMO Model
2. Intermediate COCOMO Model
3. Complete/Detailed COCOMO Model



The hierarchy of COCOMO models takes the following form:

Model 1. The Basic COCOMO model is a static, single-valued model that computes software development effort (and cost) as a function of program size expressed in estimated lines of code (LOC).

Model 2. The Intermediate COCOMO model computes software development effort as a function of program size and a set of "cost drivers" that include subjective assessments of product, hardware, personnel and project attributes.

Model 3. The Advanced COCOMO model incorporates all characteristics of the intermediate version with an assessment of the cost driver's impact on each step (analysis, design, etc.) of the software engineering process

Basic COCOMO

- n The basic COCOMO estimation model is given by the following expression:
$$\text{Effort} = a * \text{size}^b$$
- n Effort is measured in person-month.
- n Size is measured in ‘kdsi’, thousands of delivered source code instruction.
- n **a** and **b** are constants.
- n Their value depends on whether the system is ‘organic’, ‘semi-detached’, ‘embedded’.

(cont..)

- n Organic mode :when small teams develop software that is very familiar.
- n Embedded mode: product being delivered within very tight constraints and unfamiliar.
- n Semi-detached – combines both.

Organic

The organic mode is typified by systems such as payroll, inventory, and scientific calculation. Other characterizations are that the project team is small, little innovation is required, constraints and deadlines are few, and the development environment is stable.

Semidetached

The semidetached mode is typified by utility systems such as compilers, database systems, and editors. Other characterizations are that the project team is medium-size, some innovation is required, constraints and deadlines are moderate, and the development environment is somewhat fluid.

Embedded

The embedded mode is typified by real-time systems such as those for air traffic control, ATMs, or weapon systems. Other characterizations are that the project team is large, a great deal of innovation is required, constraints and deadlines are tight, and the development environment consists of many complex interfaces, including those with hardware and with customers

- n The basic COCOMO estimation model is given by the following expressions:

$$\text{Effort} = ax (\text{KLOC})^b \text{ PM}$$

$$T_{\text{dev}} = 2.5x (\text{Effort})^c \text{ Months}$$

People Required=effort/Development time

- n KLOC is the estimated size of the software product expressed in Kilo Lines of Code,
- n a, b, c are constants for each category of software products,
- n Tdev (nominal development time) is the estimated time to develop the software, expressed in months,
- n Effort is the total effort required to develop the software product, expressed in person months (PMs).

COCOMO constants

System Type	a	b	c
n Organic	2.4	1.05	0.38
n Semi-detached	3.0	1.12	0.35
n Embedded	3.6	1.20	0.32

COCOMO cont...

n **Example:**

Assume that the size of an organic type software product has been estimated to be 32,000 lines of source code. Assume that the average salary of software engineers be Rs. 15,000/- per month. Determine the effort required to develop the software product and the nominal development time.

From the basic COCOMO estimation formula for organic software:

$$\text{Effort} = 2.4 \times (32)^{1.05} = 91 \text{ PM}$$

$$\text{Nominal development time} = 2.5 \times (91)^{0.38} = \\ \mathbf{14 \text{ months}}$$

$$\text{Cost required to develop the product} = 14 \times \\ \mathbf{15,000 = Rs. 210,000/-}$$

COCOMO Intermediate

- n Is an extension of the Basic COCOMO model
- n Considers a set of four "cost driver attributes", each with a number of subsidiary attributes:
 - Product attributes
 - Computer attributes
 - Personnel attributes
 - Project attributes

(cont..)

- n **Product attributes**
 - Required software reliability
 - Size of application database
 - Complexity of the product
- n **Hardware attributes**
 - Run-time performance constraints
 - Memory constraints
 - Volatility of the virtual machine environment
 - Required turnabout time

Cont..

n **Personnel attributes**

- Analyst capability
- Software engineer capability
- Applications experience
- Virtual machine experience
- Programming language experience

n **Project attributes**

- Use of software tools
- Application of software engineering methods
- Required development schedule

(cont..)

- n In COCOMO Intermediate , a nominal effort estimate (pm_{nom}) is calculated same as the basic model.
- n Finally it is adjusted by development effort multiplier(dem)

$$pm_{est} = pm_{nom} * dem$$

- n *dem* is calculated by multiplying the multiplier selected for each cost drivers.

COCOMO III

- n COCOMO II addresses the following three phases of the spiral life cycle:
- n applications development, early design and post architecture
- n Most significant input to the COCOMO II model is size. Size is treated as a special cost driver in that it has an exponential factor, E. This exponent is an aggregation of five scale factors

- n The five Scale Factors are:
 - n PREC Precedentedness (how novel the project is for the organization)
 - n FLEX Development Flexibility
 - n RESL Architecture / Risk Resolution
 - n TEAM Team Cohesion
 - n PMAT Process Maturity

Estimate of person months

$$pm = A(\text{size})^{(sf)} \times (em_1) \times (em_2) \times (em_3) \dots (em_n)$$

Scale factor is driven as

$$sf = 1.01 + 0.01 \times \sum (\text{exponent driver ratings})$$



Activity Planning



Introduction

- q A detailed plan should include a **schedule** indicating the **start and completion time** for each activity. This enable us to:
 - ø Ensure **appropriate resources** are available when required.
 - ø Avoid different activities **competing for the same resources at the same time**.
 - ø Produce a **detailed schedule** showing which staff carry out each activity.
 - ø Produce **time cash flow forecast**.
 - ø **Replan the project** during its correct drift from the target.



When to plan?

- q Planning is an **ongoing process** of refinement.
- q Each **iteration** become **more accurate and detailed** than the last.
- q Purpose of planning during feasibility study and project start up:
 - ø To estimate timescales
 - ø Risk for not achieving target completion dates
 - ø Keeping within budget.
 - ø Ensuring resource availability.

Objectives of Activity Planning

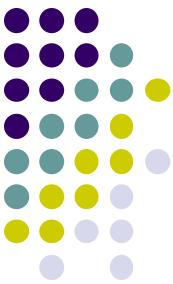


- q **Feasibility assessment**: Is the project possible within required timescales and resource constraint ?
- q **Resource allocation**: What are most effective ways of allocating resources and when they should be available.
- q **Detailed costing**: How much will the project cost and when that expenditure will take place?
 - Ø After activity plan and allocating resources we can obtain detailed estimates of costs and timing.
- q **Motivation**: Providing targets and monitor the achievement.
- q **Co-ordination**: When do the staff need to be available to work on a particular project.
 - Ø When staff need to be transferred between projects?
 - Ø Co-ordination among teams when involved in large projects.

Project Schedules



- q Project plan must be developed to the level of **showing dates** when each activity should **start and finish** and how much of each **resource** will be required.
- q Once plan has been refined in detail it become the ***project schedule***.
- q Project schedule comprises of four main stages:
 1. Produce an ideal activity plan (when & in what order).
 2. Ideal activity plan then subject to activity risk analysis.
 3. Resource allocation, availability of resources.
 4. Schedule production, after allocation we can draw up and publish a project schedule indicating the start and end dates



Identifying activities

- q Project composed of a number of interrelated activities.
- q Three approaches to identify the activities:
 - ø Activity based approach
 - ø Product based approach
 - ø Hybrid approach

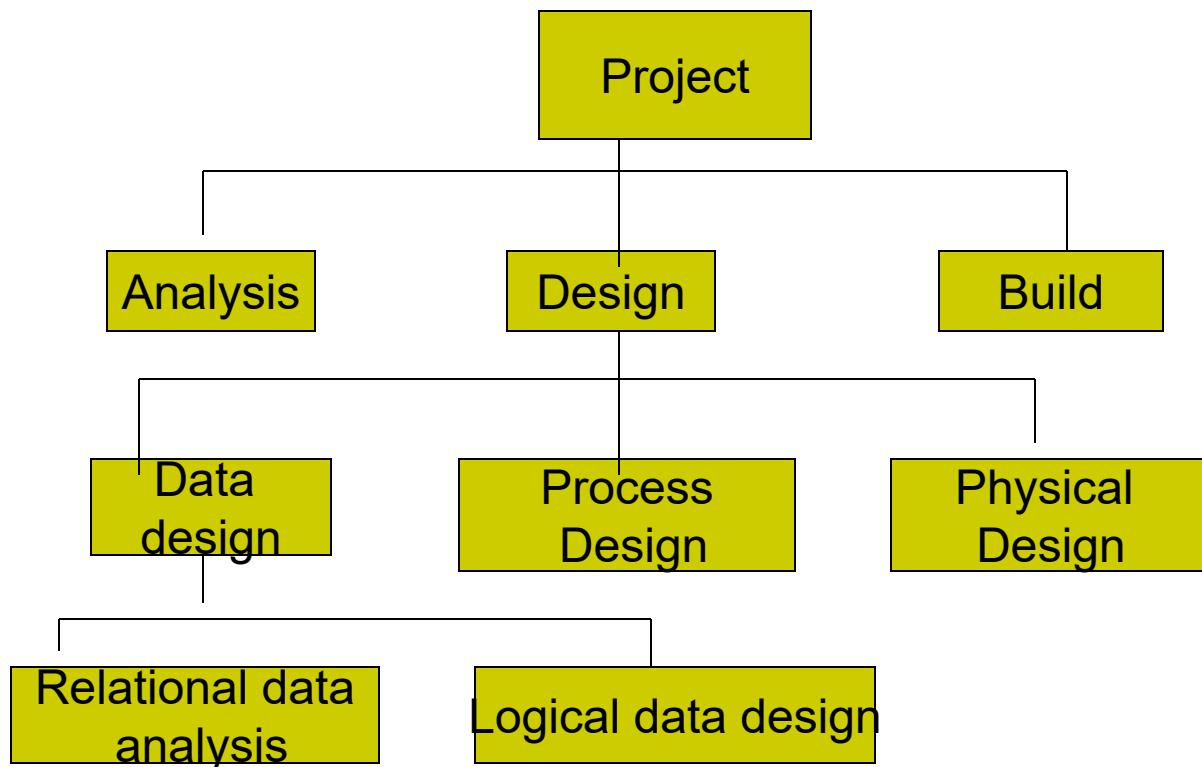


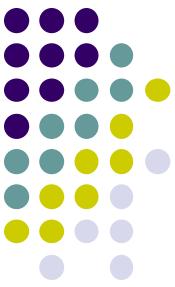
Activity Based Approach

- q Creating a **list of all activities** that the project involve.
- q Common way of generating a task list is to create a **Work Breakdown Structure** (WBS).
- q **WBS** is an exhaustive, hierarchical (from general to specific) tree structure of **deliverables and tasks** that need to be performed to complete a project.
- q I.e. identify the **main task** and breaking each of these into set of **lower level tasks**.
- q A well-designed WBS describes **planned outcomes** instead of planned actions.
- q Outcomes are the desired ends of the project, and can be predicted accurately;
- q actions comprise the project plan and may be difficult to predict accurately.



Work Breakdown Structure





WBS Outline Example

- 0.0 Retail Web Site
- 1.0 Project Management
- 2.0 Requirements Gathering
- 3.0 Analysis & Design
- 4.0 Site Software Development
 - 4.1 HTML Design and Creation
 - 4.2 Backend Software
 - 4.2.1 Database Implementation
 - 4.2.2 Middleware Development
 - 4.2.3 Security Subsystems
 - 4.2.4 Catalog Engine
 - 4.2.5 Transaction Processing
 - 4.3 Graphics and Interface
 - 4.4 Content Creation
- 5.0 Testing and Production



Product Based Approach

- q It consists of producing PBS and PFD.
- q PFD indicates for each product, which other products are required as input.

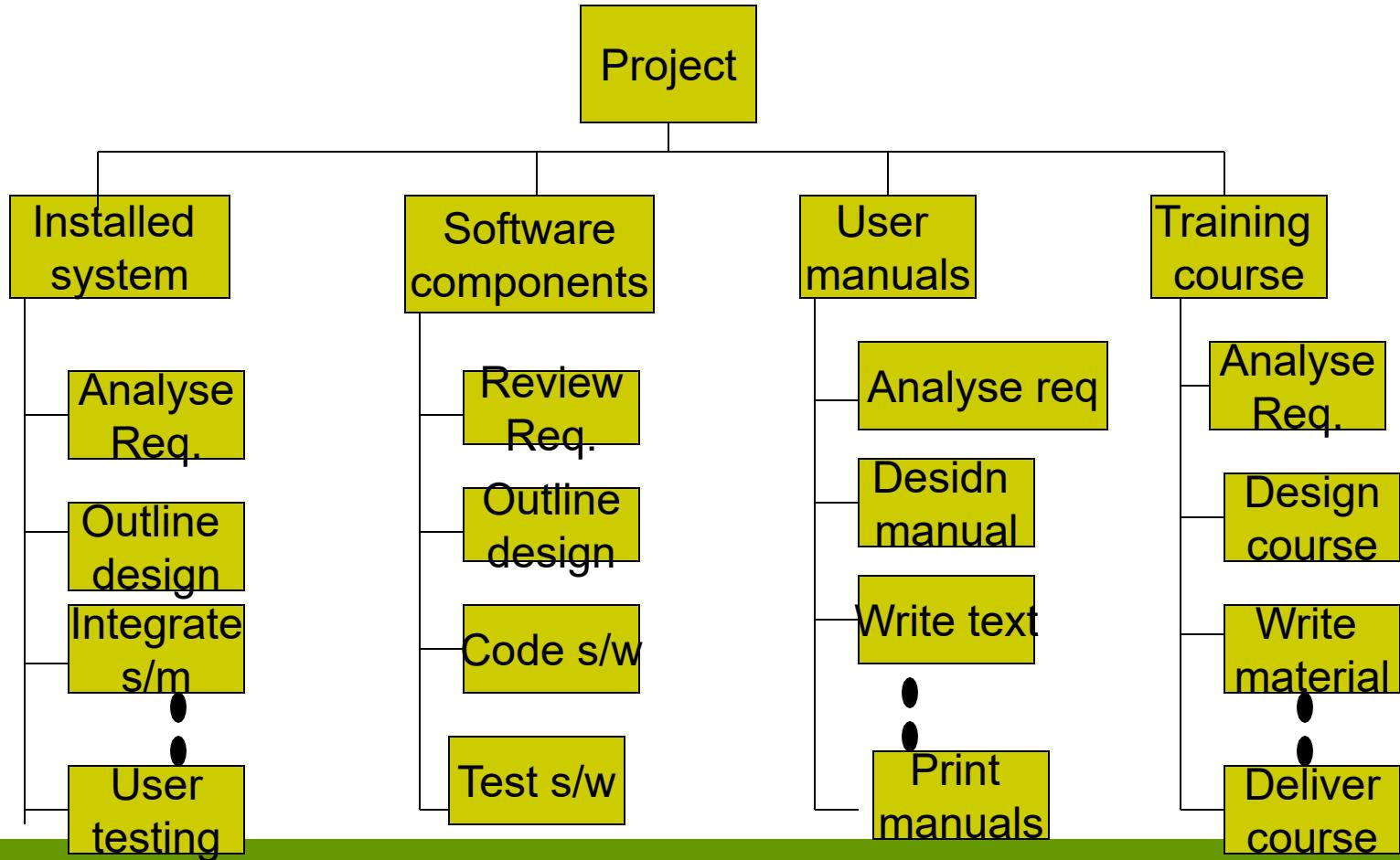


Hybrid Based Approach

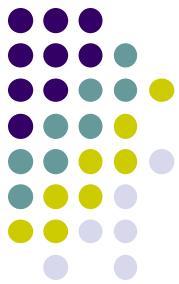
- q WBS discussed based on structuring of activities.
- q WBS can also be based on simple list of final deliverables.
- q Consists of five levels:
 - q Level 1: **Project**
 - q Level 2: **Deliverables** such as software, manuals, training courses.
 - q Level 3: **Components**, key items needed to produce deliverables.
 - q Level 4; **Work packages**, major work items for making components.
 - q Level 5: **Task**, responsibility of a single person.



WBS based on deliverables



Project Management Methods



- q Bar Chart / Gantt Chart
- q Milestone Chart
- q Pert
- q CPM

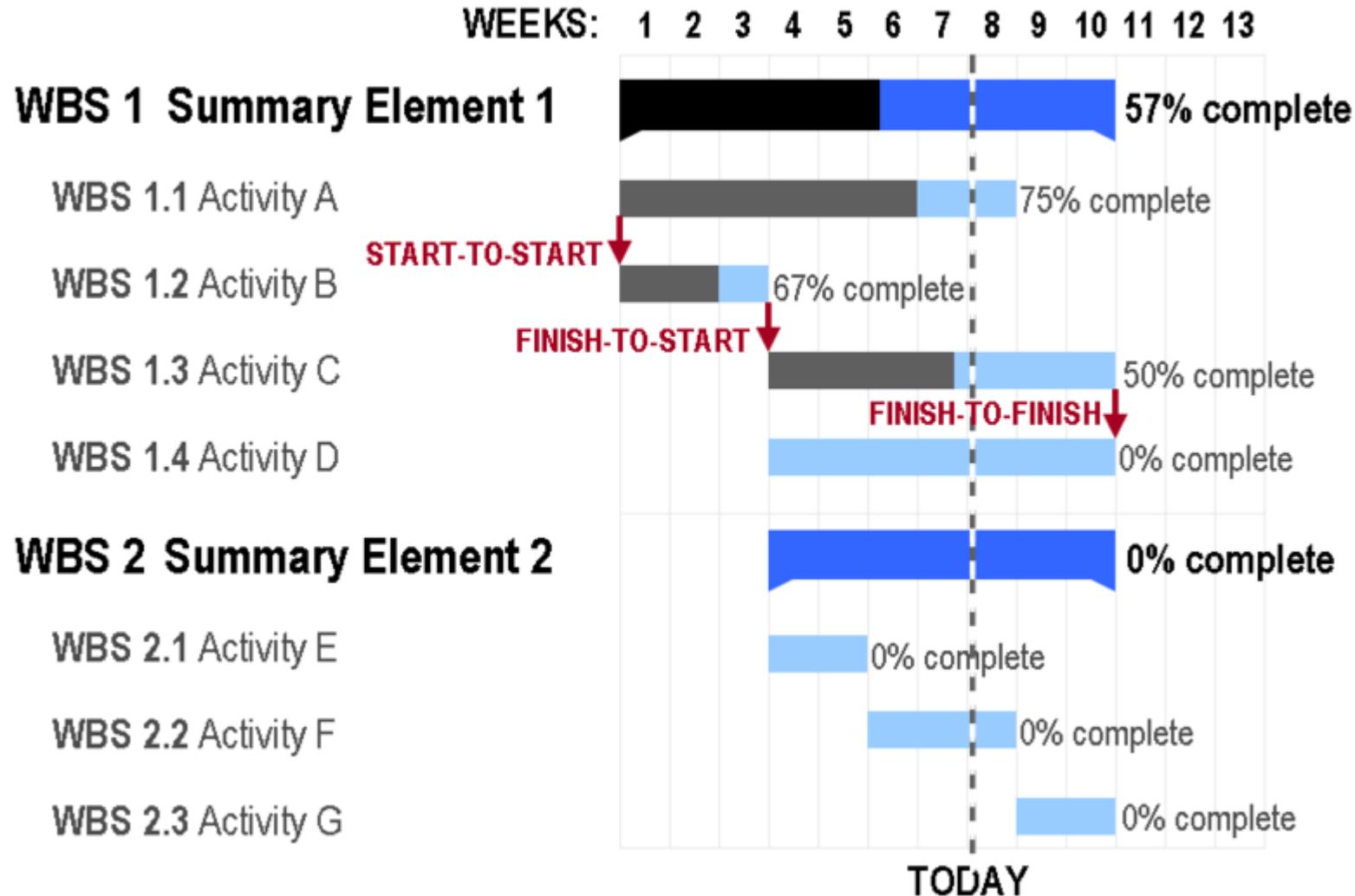
Gantt Chart



- q A popular type of bar chart that illustrates a ***project schedule***.
- q Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project.
- q Terminal elements and summary elements comprise the WBS of the project.
- q Also show the dependency (i.e., precedence network) relationships between activities.



Gantt Chart





Bar chart

Task : Person \ Weeks	1	2	3	4	5	6	7	8	9	10	11	12	13
A : Andy													
B : Andy													
C : Andy													
D : Andy													
E : Bill													
F : Bill													
G : Charlie													
H : Charlie													
I : Dave													

Activity key

- | | |
|----------------------|-------------------------|
| A : Overall design | F : Code module 3 |
| B : Specify module 1 | G : Code module 2 |
| C : Specify module 2 | H : Integration testing |
| D : Specify module 3 | I : System testing |
| E : Code module 1 | |

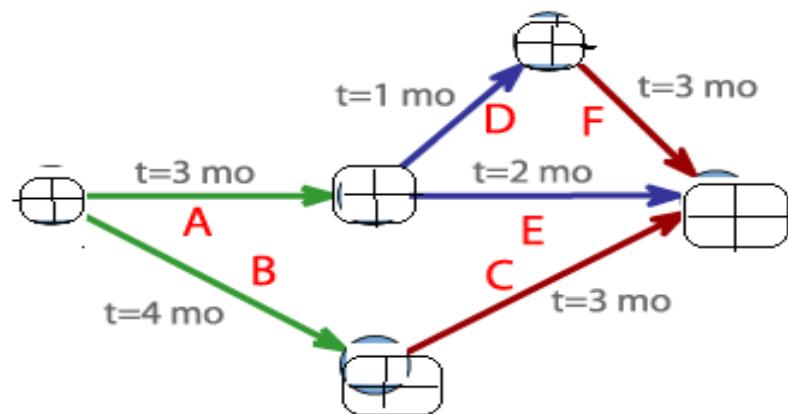


PERT Chart

- q ***Program Evaluation and Review Technique***, a model for project management to **analyze and represent** the tasks involved in completing a given project.
- q Especially the **time needed** to **complete** each task, and identifying the **minimum time** needed to complete the **total project**.



PERT chart

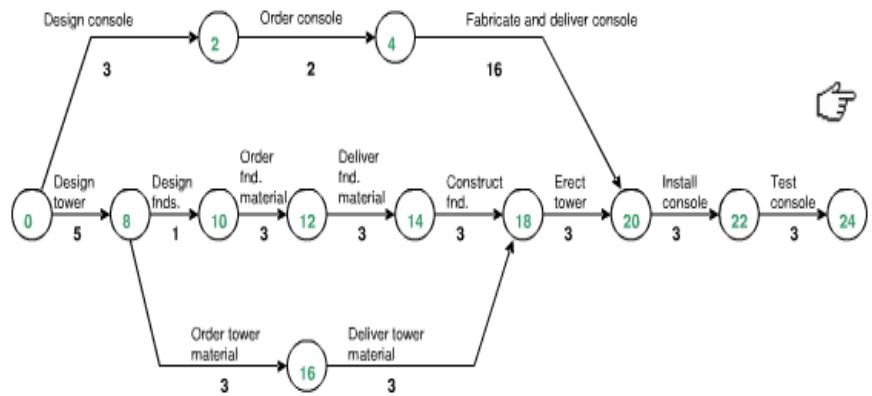
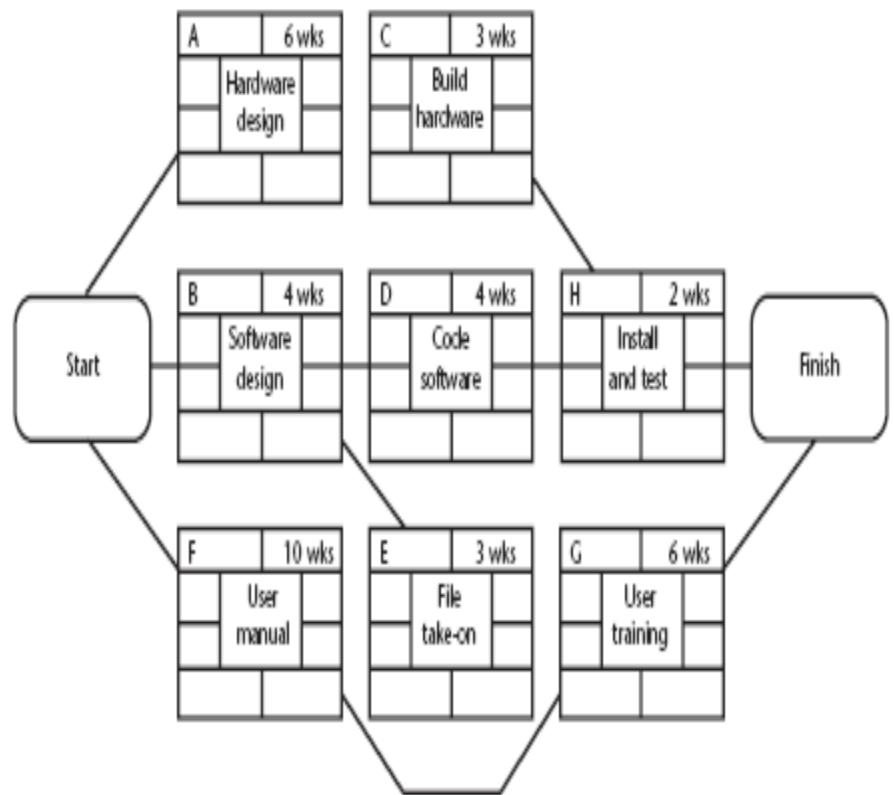
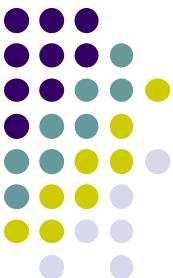




Activity Conventions

- q Activity On Node Convention (AON)
 - ø Nodes are the project activities
 - ø links between nodes represent precedence relationships

- q Activity On Arrow Convention (AOA)
 - ø Nodes are events: beginning or endings.
 - ø Arrows represents the activities.
 - ø CPM and PERT developers used AOA.

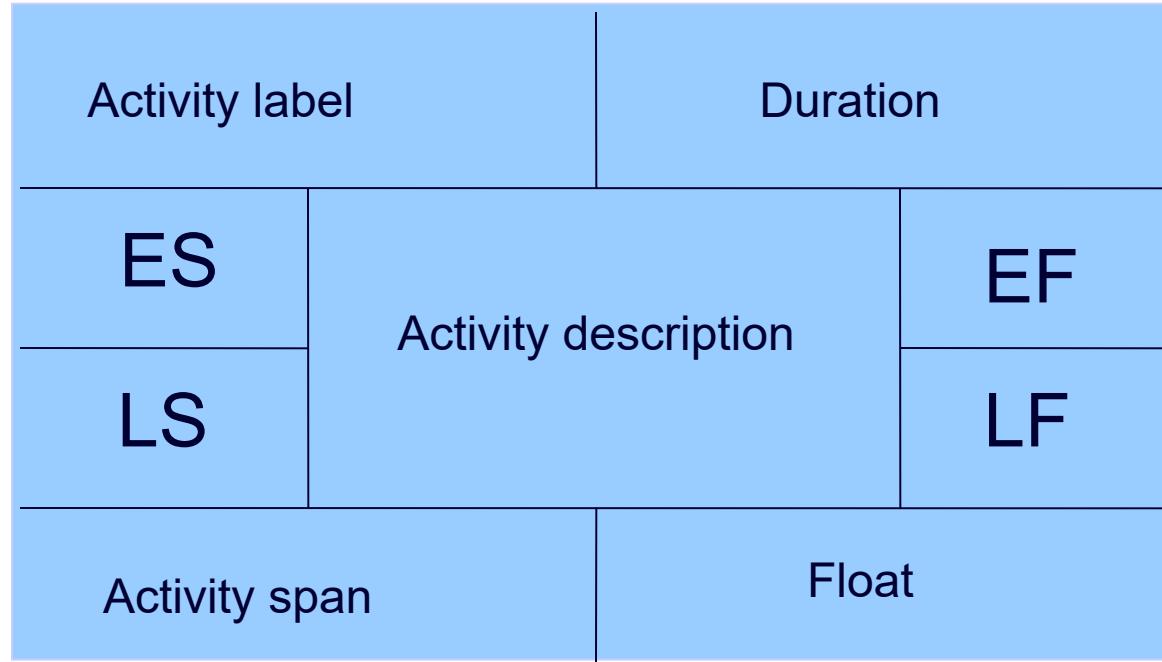


Constructing precedence network

Some rules for the construction of precedence network.

- ø A project network should have only one start node
- ø A project network should have only one end node
- ø A node has duration
- ø Links normally have no duration
- ø Precedents are the immediate preceding activities
- ø Times moves from left to right
- ø A network may not contain loops
- ø A network should not contain dangles

Notation



Activity label	Duration
Earliest start	Activity description
Latest start	Earliest finish
Activity span	Latest finish
Activity span	Float

Start and finish times



Latest
finish



- Activity 'write report software'
- Earliest start (ES)
- Earliest finish (EF) = ES + duration
- Latest finish (LF) = latest task can be completed without affecting project end
 $\text{Latest start} = \text{LF} - \text{duration}$

Example

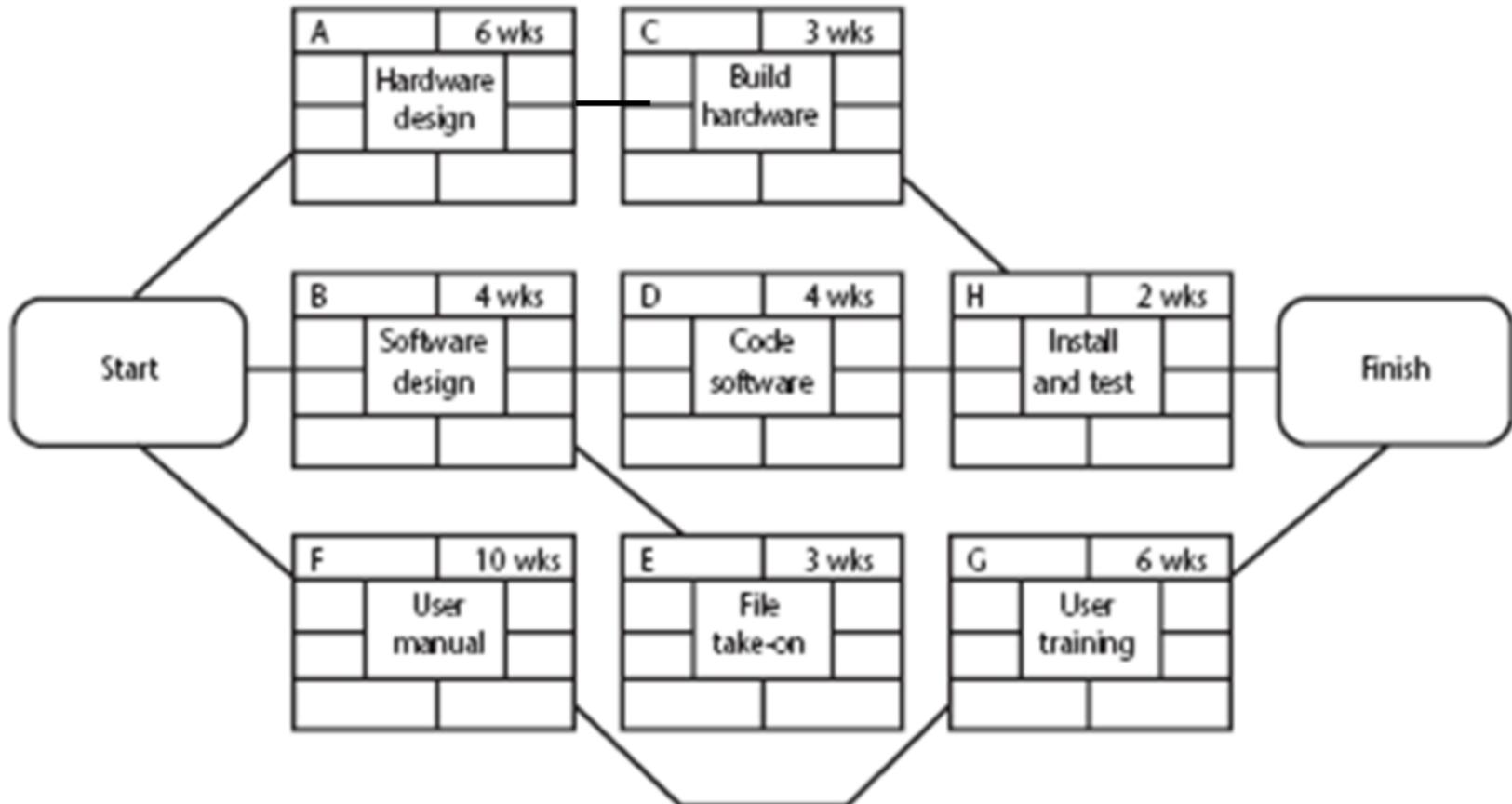


- earliest start = day 5
 - latest finish = day 30
 - duration = 10 days
- earliest finish = ?
 - latest start = ?

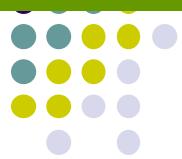
Float = LF - ES - duration

What is it in this case?

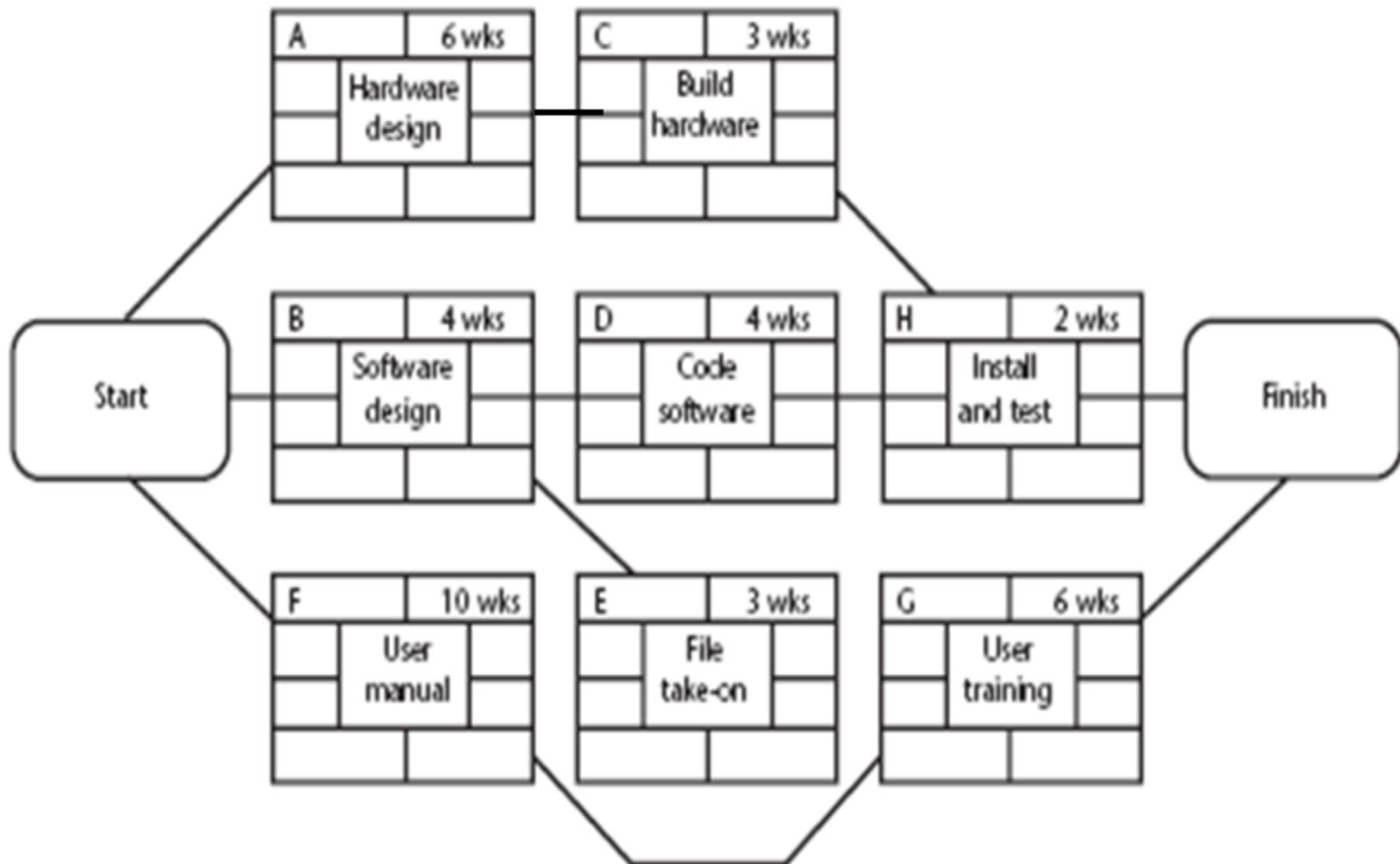
Example of an activity network



Example



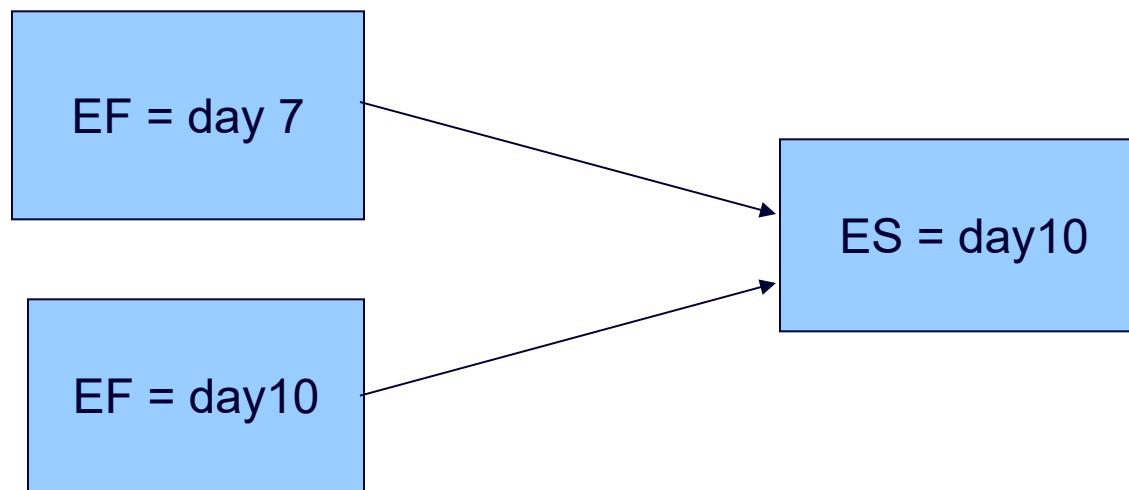
Activity	Activity Description	Duration(weeks)	Precedents
A	Hardware selection	6	
B	Software design	4	
C	Install hardware	3	A
D	Code & test software	4	B
E	File take-on	3	B
F	Write user manuals	10	
G	User training	3	E,F
H	Install & test system	2	C,D



Earliest start date



- Earliest start date for the *current* activity = earliest finish date for the *previous*
- When there is more than one previous activity, take the *latest* earliest finish
- Note ‘day 7’ = end of work on day 7



Latest start dates



- Start from the *last* activity
- Latest finish (LF) for last activity = earliest finish (EF)
- work backwards
- Latest start (LS) = LF for activity – duration
- Latest finish for *current* activity = Latest start for the *following*
- More than one following activity - take the *earliest* LS



L4 Clip

Forward Pass

Begin at starting event and work forward

Earliest Start Time Rule:

- ◆ If an activity has only a single immediate predecessor, its ES equals the EF of the predecessor
- ◆ If an activity has multiple immediate predecessors, its ES is the maximum of all the EF values of its predecessors

$$\text{ES} = \text{Max } \{\text{EF of all immediate predecessors}\}$$

Backward Pass

Begin with the last event and work backwards

Latest Finish Time Rule:

- ◆ If an activity is an immediate predecessor for just a single activity, its LF equals the LS of the activity that immediately follows it
- ◆ If an activity is an immediate predecessor to more than one activity, its LF is the minimum of all LS values of all activities that immediately follow it

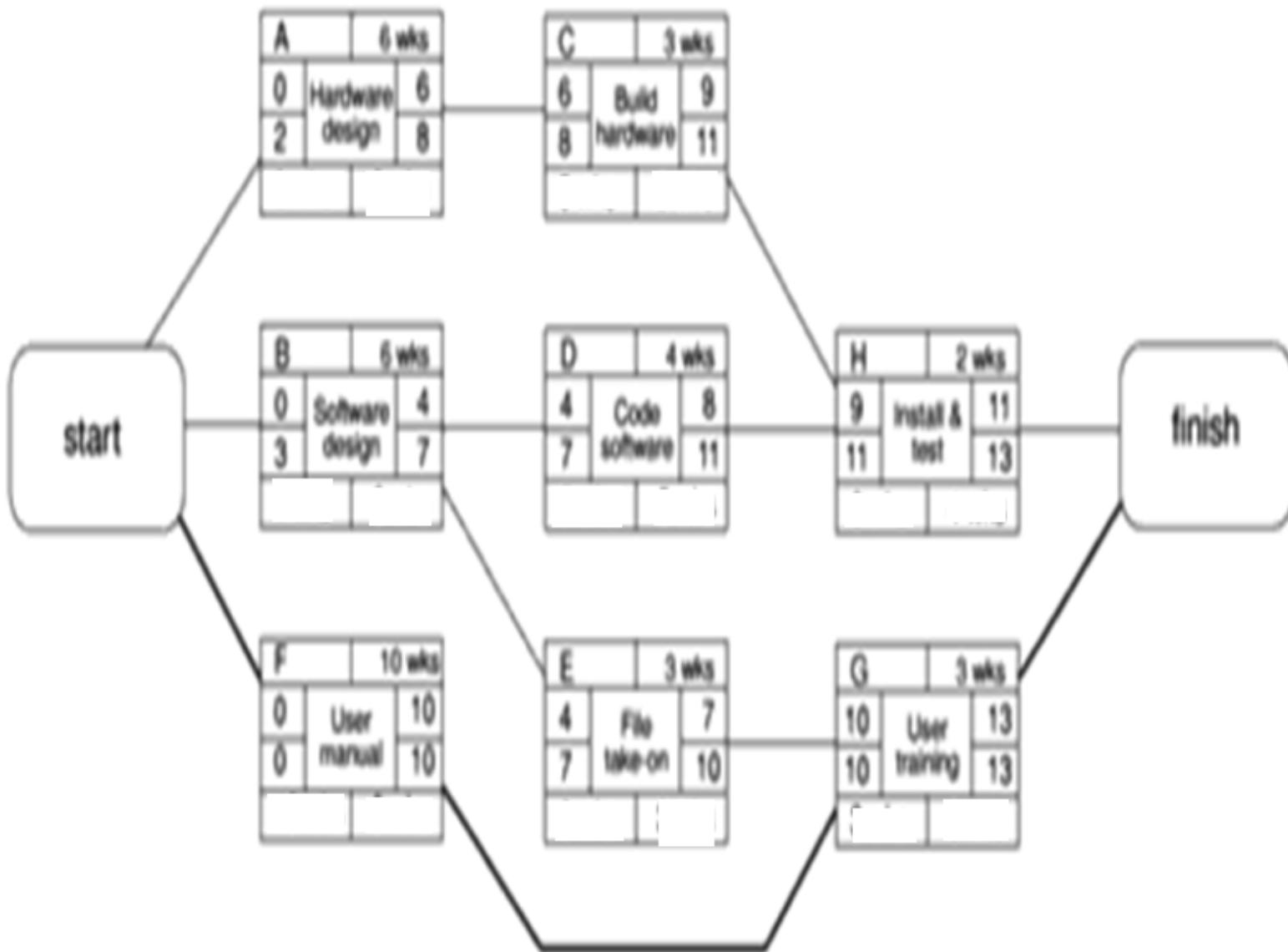
$$\text{LF} = \text{Min } \{\text{LS of all immediate following activities}\}$$

Clip



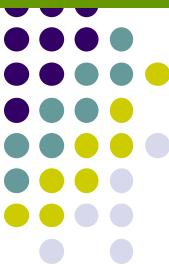
Table 6.4 The activity schedule showing total float for each activity

Activity	Precedents	Duration (weeks)	Earliest start date	Latest start date	Earliest finish date	Latest finish date	Total float
A		6	0	2	6	8	2
B		4	0	3	4	7	3
C	A	3	6	8	9	11	2
D	B	4	4	7	8	11	3
E	B	3	4	7	7	10	3
F		10	0	0	10	10	0
G	E,F	3	10	10	13	13	0
H	C,D	2	9	11	11	13	2



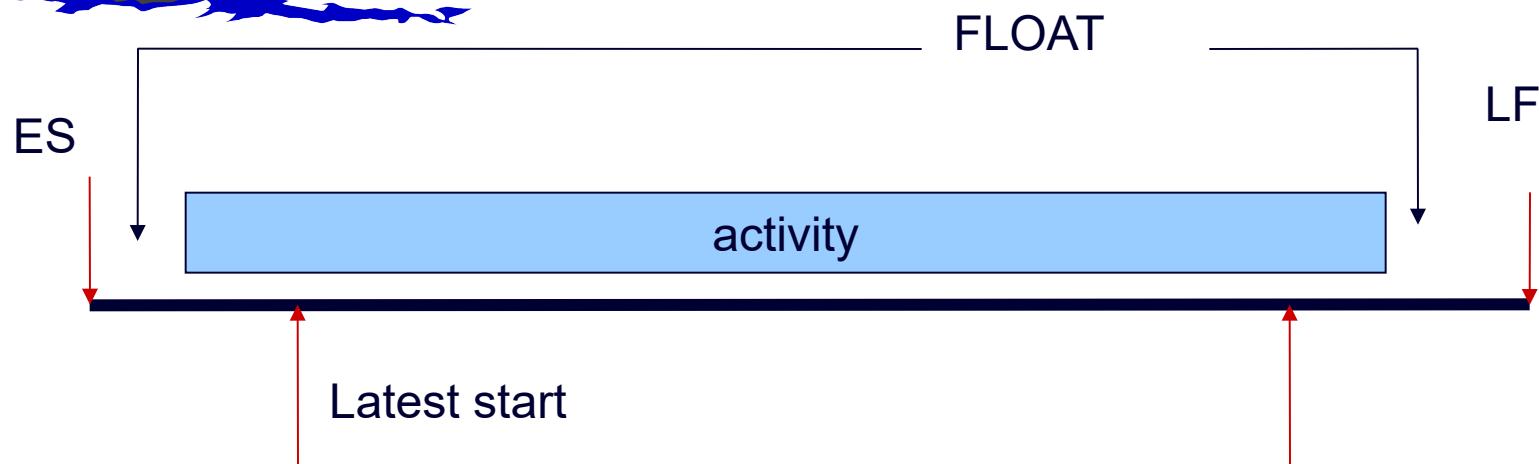


Float



Float = Latest finish -Earliest start - Duration

Float can also be calculated as the difference between the earliest and latest start dates for an activity or the difference between the earliest and latest finish dates.





Activity Float =difference between the earliest and latest start dates for an activity *or* the difference between the earliest and latest finish dates.

Its a measure of how much the start or completion of an activity may be delayed without affecting the end date of the project.

Activity span= difference between the earliest start and latest finish dates.

Its a measure of the maximum time allowable for the activity.

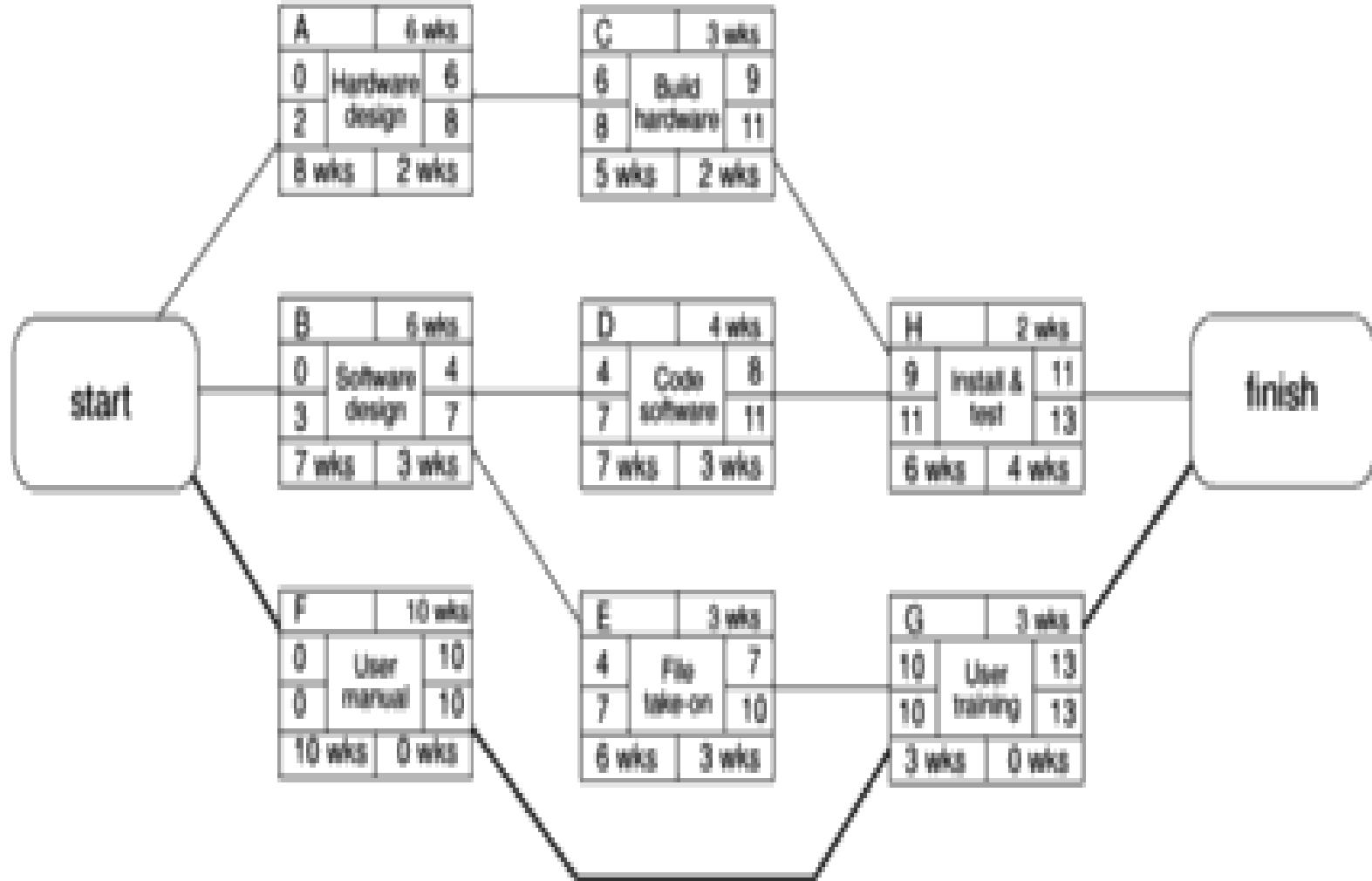


Figure 6.21 A precedence network.

Critical path: F-G

Free and interfering float



A	7w
0	7
2	9
	2

B can be up to 3 days late
and not affect any
other activity = **free float**

B	4w
0	4
5	9
	5

D	1w
7	8
9	10
	2

E	2w
10	12
10	12
	0

C	10w
0	10
0	10
	0

B can be a further 2 days late – affects
D but not the project end date =
interfering float



Free float: the time by which an activity may be delayed without affecting any subsequent activity.

It is calculated as the **difference between the earliest completion date for the activity and earliest start date** of the succeeding activity.

Interfering float: how much the activity may be delayed without delaying the project date-even though it will delay the start of subsequent activities.

It is calculated as the **difference between total float and the free float**

Critical path



- Note the path through network with zero floats
- Critical path: any delay in an activity on this path will delay whole project
- Can there be more than one critical path?
- Can there be no critical path?
- Sub-critical paths



Table E.9 *Activity floats*

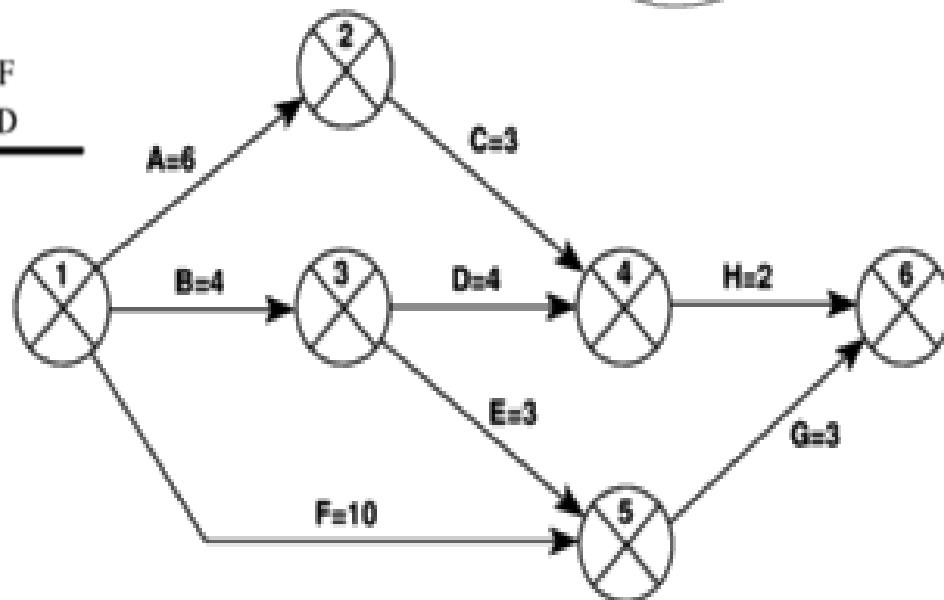
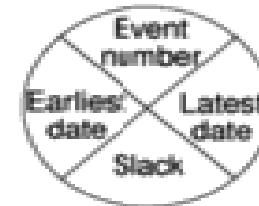
<i>Activity</i>	<i>Total float</i>	<i>Free float</i>	<i>Interfering float</i>
A	2	0	2
B	3	0	3
C	2	0	2
D	3	1	2
E	3	3	0
F	0	0	0
G	0	0	0
H	2	2	0

Activity on arrow network(CPM NETWORK)



Activity	Duration (weeks)	Precedents
A Hardware selection	6	
B Software design	4	
C Install hardware	3	A
D Code & test software	4	B
E File take-on	3	B
F Write user manuals	10	
G User training	3	E, F
H Install & test system	2	C, D

Labelling Convention





Forward Pass rule

During the forward pass, earliest dates are recorded as they are calculated. For events, they are recorded on the network diagram and for activities they are recorded on the activity table.

The forward pass rule: the earliest date for an event is the earliest finish date for all the activities terminating at that event. Where more than one activity terminates at a common event we take the latest of the earliest finish dates for those activities.

Table 6.2 *The activity table after the forward pass*

Activity	Duration (weeks)	Earliest start date	Latest start date	Earliest finish date	Latest finish date	Total float
A	6	0		6		
B	4	0		4		
C	3	6		9		
D	4	4		8		
E	3	4		7		
F	10	0		10		
G	3	10		13		
H	2	9		11		

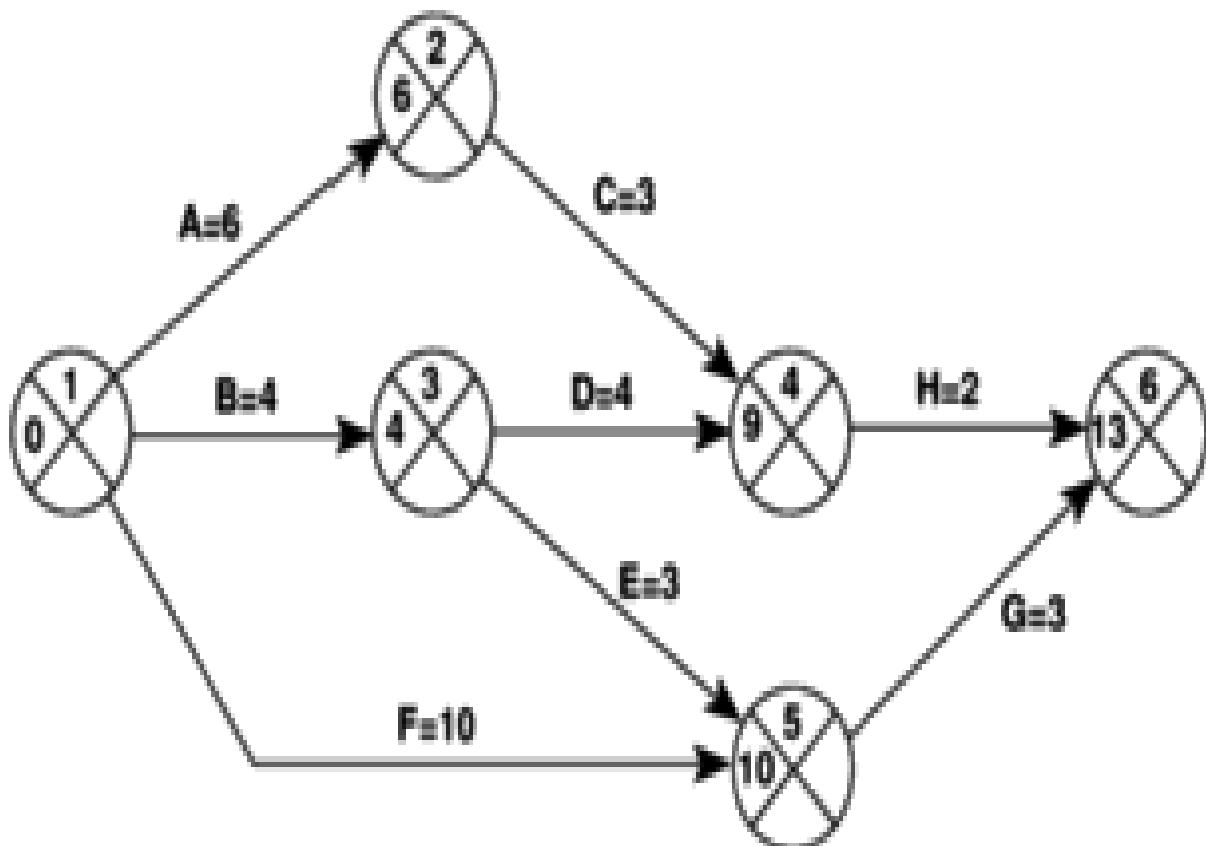


Figure 6.18 A CPM network after the forward pass.



Backward Pass Rule

The backward pass rule: the latest date for an event is the latest start date for all the activities that may commence from that event. Where more than one activity commences at a common event we take the earliest of the latest start dates for those activities.

Table 6.3 The activity table following the backward pass

Activity	Duration (weeks)	Earliest start date	Latest start date	Earliest finish date	Latest finish date	Total float
A	6	0	2	6	8	
B	4	0	3	4	7	
C	3	6	8	9	11	
D	4	4	7	8	11	
E	3	4	7	7	10	
F	10	0	0	10	10	
G	3	10	10	13	13	
H	2	9	11	11	13	

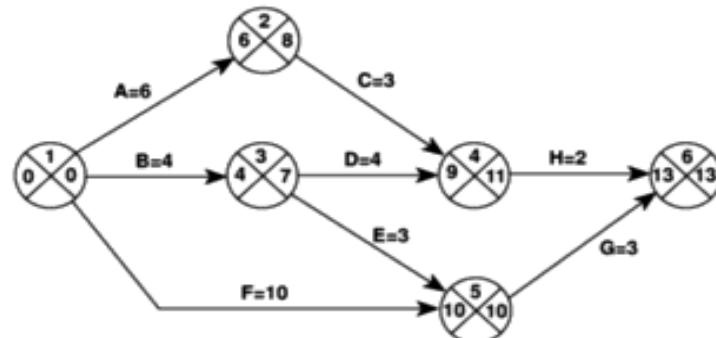


Figure 6.19 The CPM network after the backward pass.



Critical Path

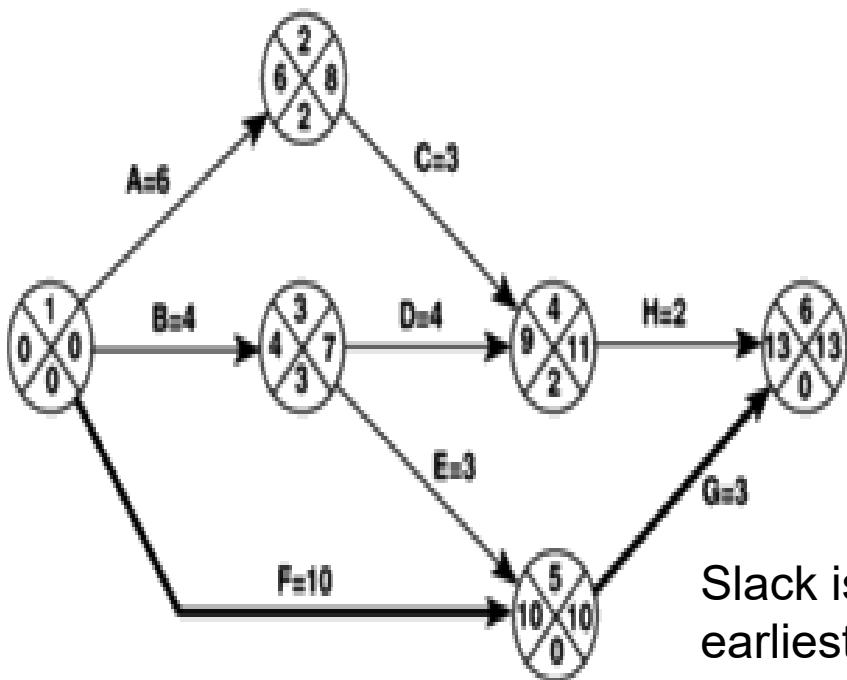


Figure 6.20 The critical path.

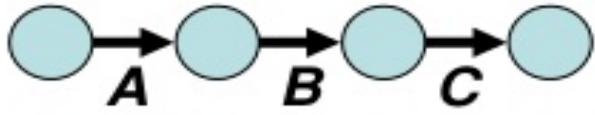
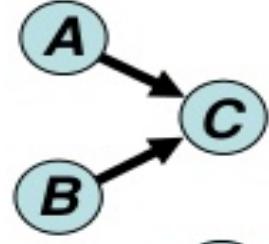
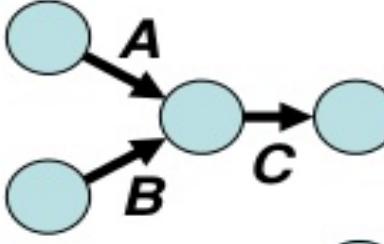
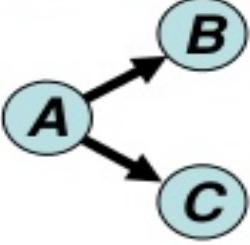
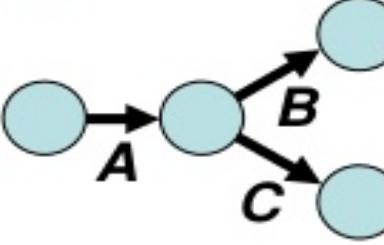
Critical path: F-G

The critical path is the longest path through the network.

Slack is the difference between earliest date and latest date of an event. Its a measure of how late an event may be without delaying end date of the project



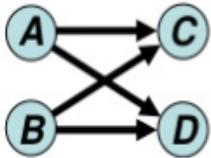
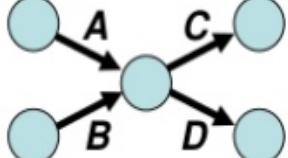
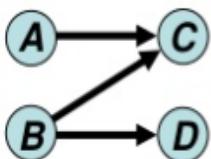
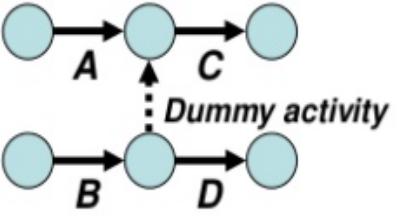
A Comparison of AON and AOA Network Conventions

	<i>Activity on Node (AON)</i>	<i>Activity Meaning</i>	<i>Activity on Arrow (AOA)</i>
(i)		<i>A comes before B, which comes before C</i>	
(ii)		<i>A and B must both be completed before C can start</i>	
(iii)		<i>B and C cannot begin until A is completed</i>	

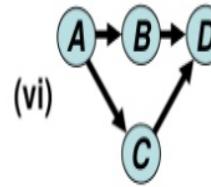
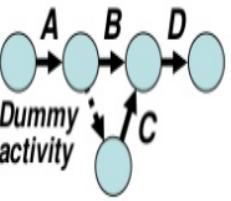


Clip slide

A Comparison of AON and AOA Network Conventions

Activity on Node (AON)	Activity Meaning	Activity on Arrow (AOA)
(iv) 	<i>C and D cannot begin until both A and B are completed</i>	
(v) 	<i>C cannot begin until both A and B are completed; D cannot begin until B is completed. A dummy activity is introduced in AOA</i>	

A Comparison of AON and AOA Network Conventions

Activity on Node (AON)	Activity Meaning	Activity on Arrow (AOA)
(vi) 	<i>B and C cannot begin until A is completed. D cannot begin until both B and C are completed. A dummy activity is again introduced in AOA.</i>	



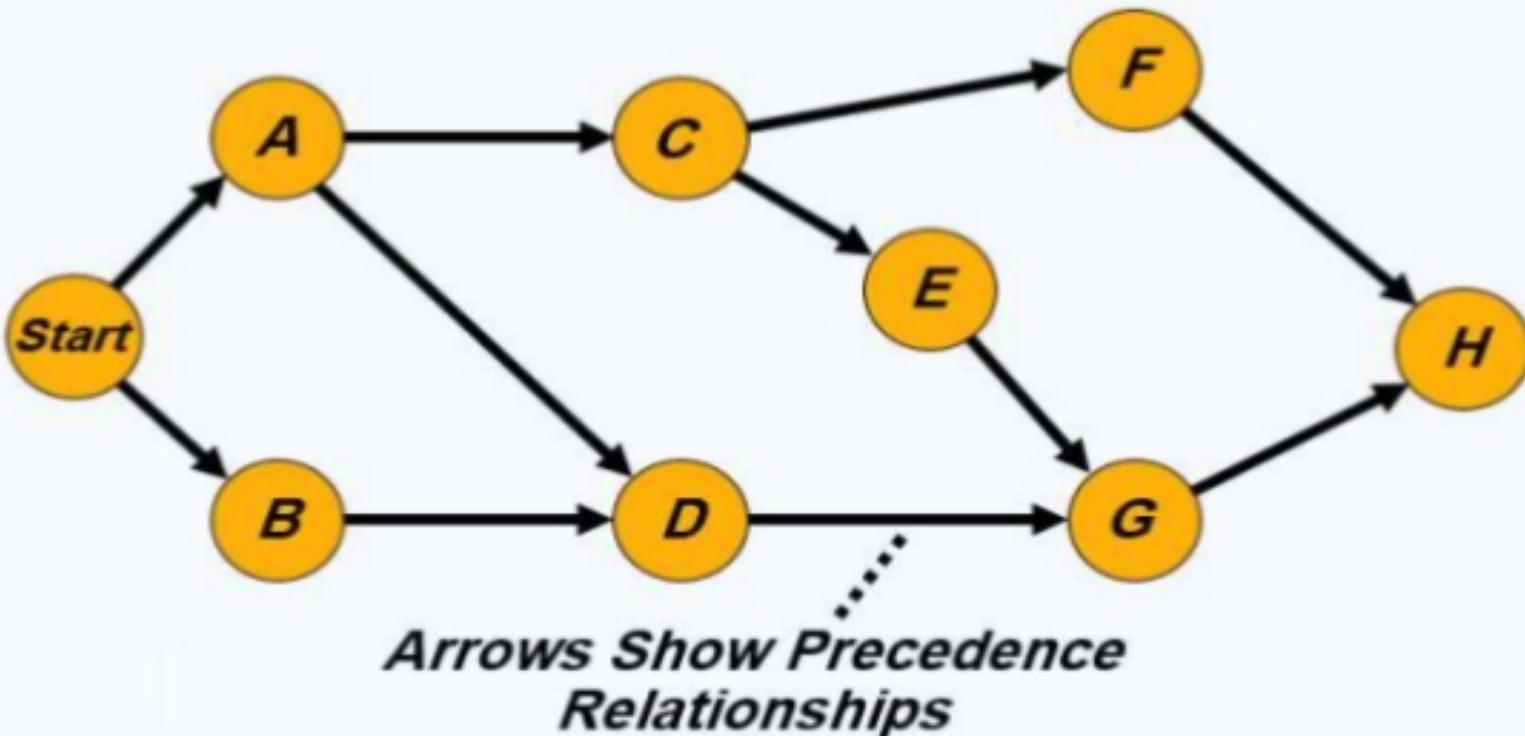
Example:

Activity	Immediate Predecessors
A	—
B	—
C	A
D	A, B
E	C
F	C
G	D, E
H	F, G

ACTIVITY	DESCRIPTION	IMMEDIATE PREDECESSORS	TIME (WEEKS)
A	Build internal components	—	2
B	Modify roof and floor	—	3
C	Construct collection stack	A	2
D	Pour concrete and install frame	A, B	4
E	Build high-temperature burner	C	4
F	Install pollution control system	C	3
G	Install air pollution device	D, E	5
H	Inspect and test	F, G	2

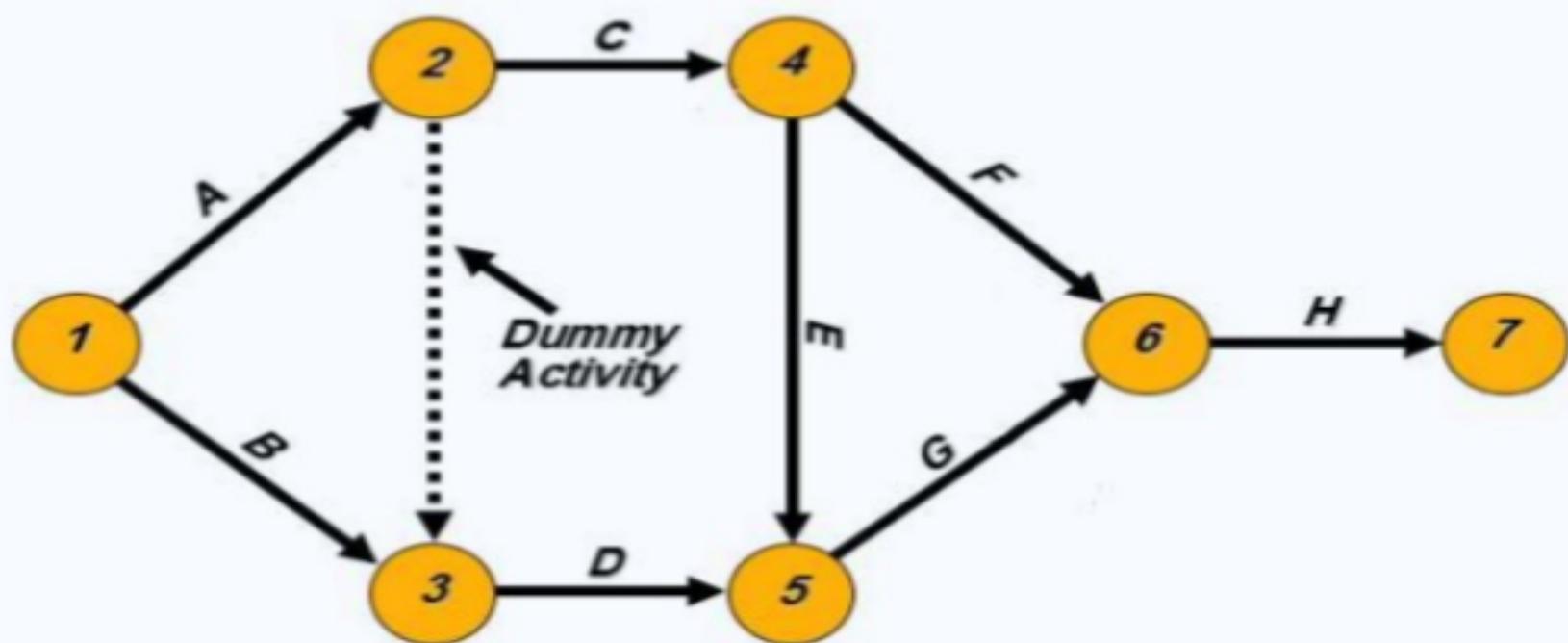


AON Network





AOA Network

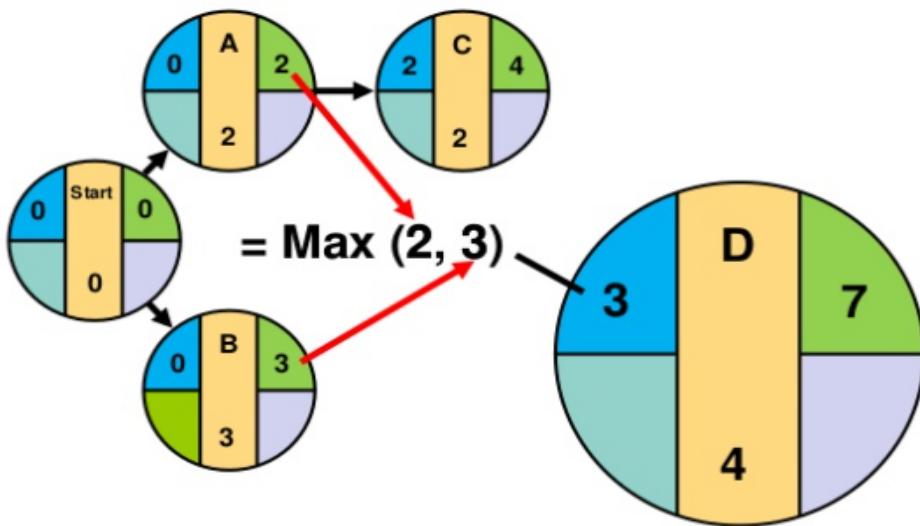




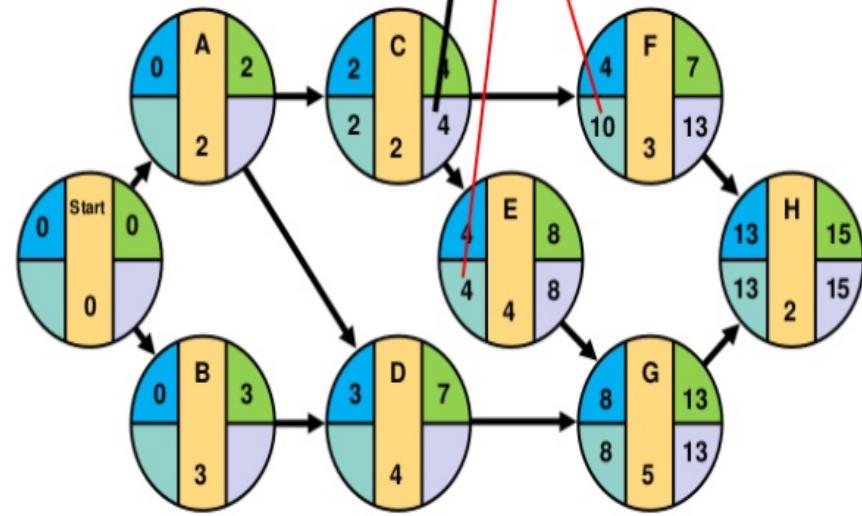
Computing Slack Time

Activity	Earliest Start ES	Earliest Finish EF	Latest Start LS	Latest Finish LF	Slack = $(LS - ES) = (LF - EF)$	On Critical Path
A	0	2	0	2	0	Yes
B	0	3	1	4	1	No
C	2	4	2	4	0	Yes
D	3	7	4	8	1	No
E	4	8	4	8	0	Yes
F	4	7	10	13	6	No
G	8	13	8	13	0	Yes
H	13	15	13	15	0	Yes

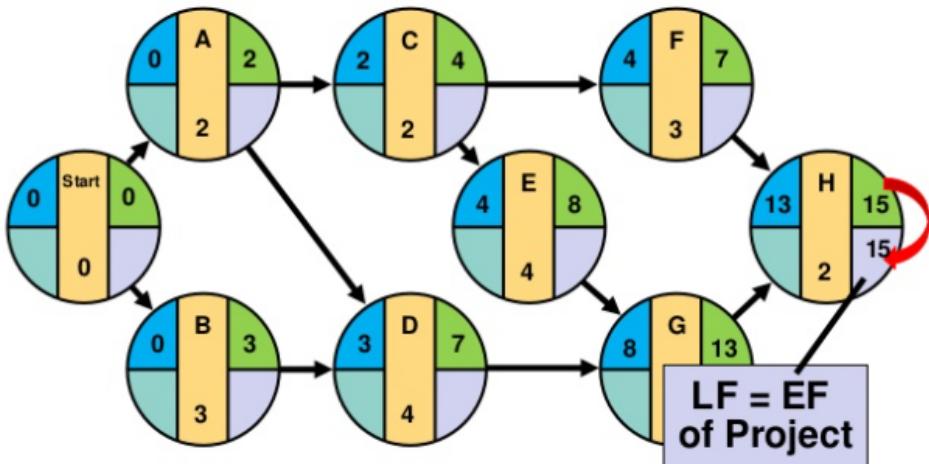
ES / EF Calculations



LS / LF Calculations
LF = Min(4, 10) ns

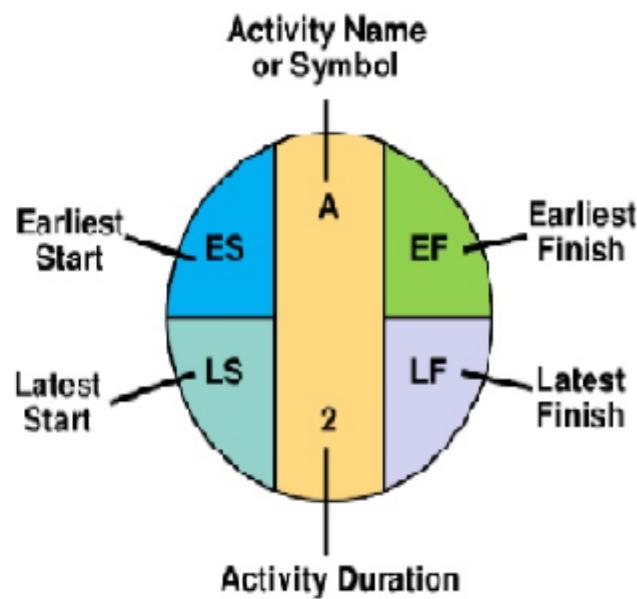
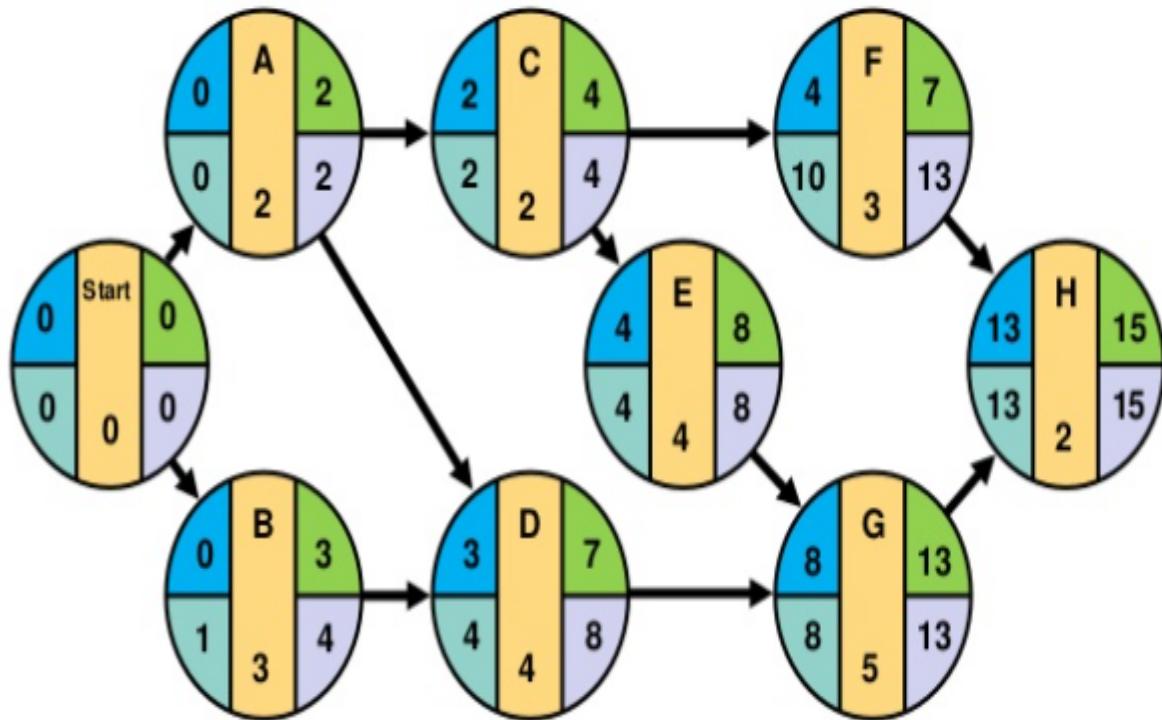


LS / LF Calculations





LS / LF Calculations



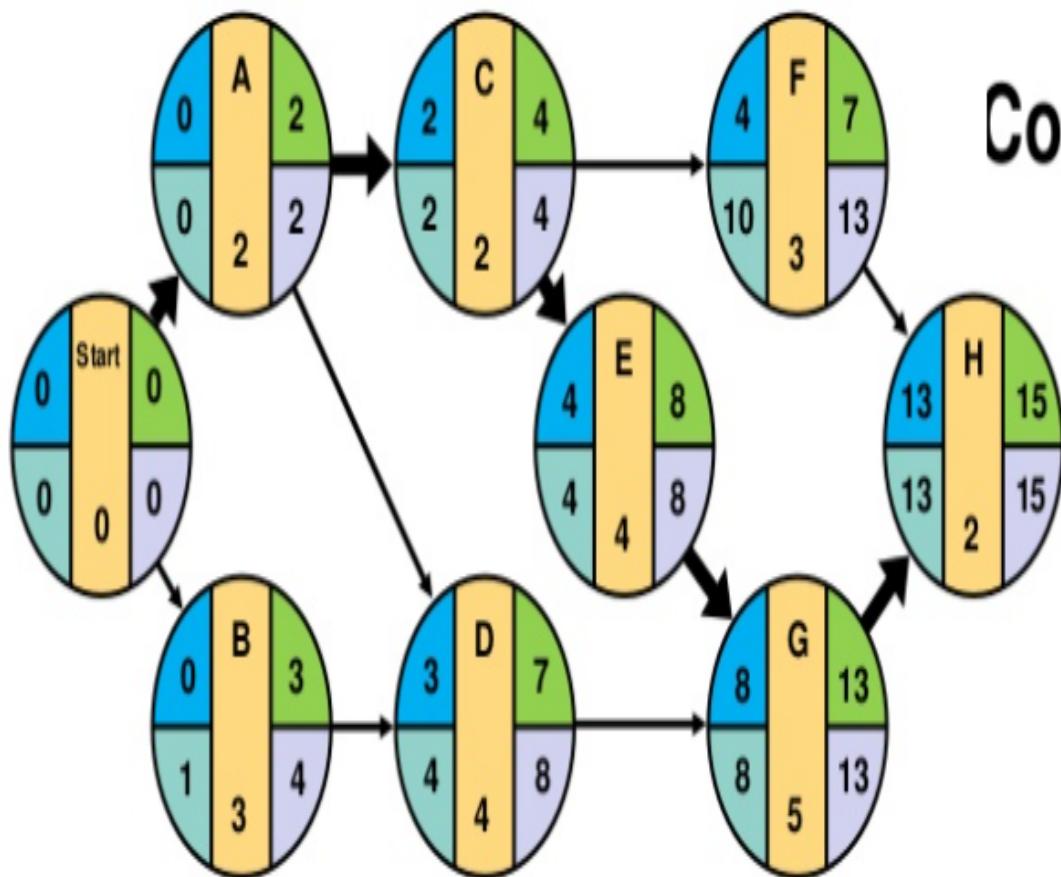


Critical Path

Critical Path

A - C - E - G - H

Completion Time = 15 weeks



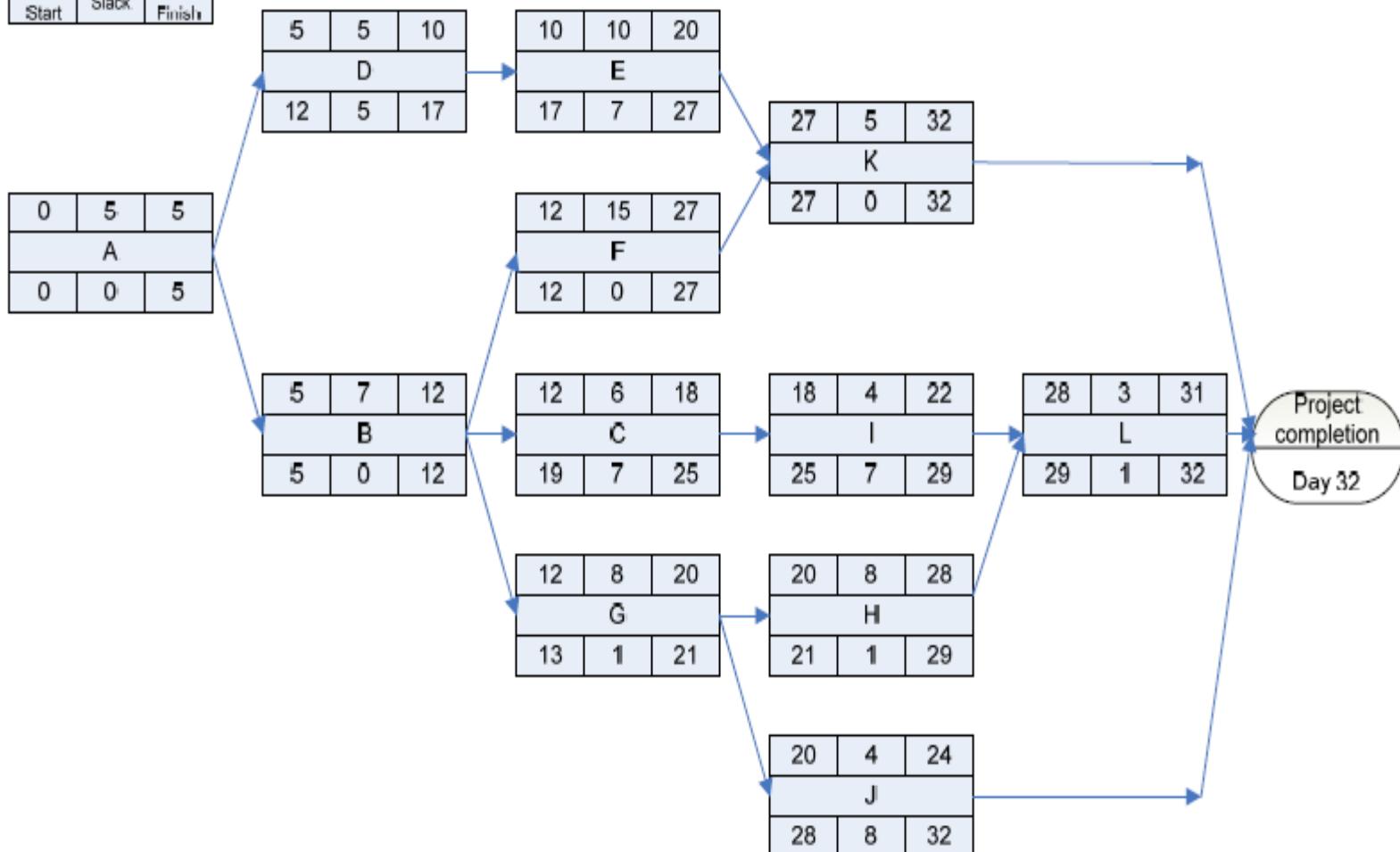


Create a precedence activity network using the following details:

<i>Activity</i>	<i>Depends on</i>	<i>Duration (days)</i>
A		5
B	A	7
C	B	6
D	A	5
E	D	10
F	B	15
G	B	8
H	G	8
I	C	4
J	G	4
K	E,F	5
L	I,H	3



Early Start	Duration	Early Finish
Task Name		
Late Start	Slack	Late Finish



The critical path is A-B-F-K where all the floats are 0.

Software Project Management

Chapter Seven

Risk management

Risk management

This lecture will touch upon:

- Definition of ‘risk’ and ‘risk management’
- Some ways of categorizing risk
- Risk management
 - Risk identification – what are the risks to a project?
 - Risk analysis – which ones are really serious?
 - Risk planning – what shall we do?
 - Risk monitoring – has the planning worked?
- We will also look at PERT risk and critical chains

Some definitions of risk

'the chance of exposure to the adverse consequences of future events'

PRINCE2

'an uncertain event or condition that, if it occurs, has a positive or negative effect on a project's objectives' PM-BOK

- Risks relate to **possible future** problems, not current ones
- They involve a possible cause and its effect(s) e.g. developer leaves
> task delayed

A framework for dealing with risk

The planning for risk includes these steps:

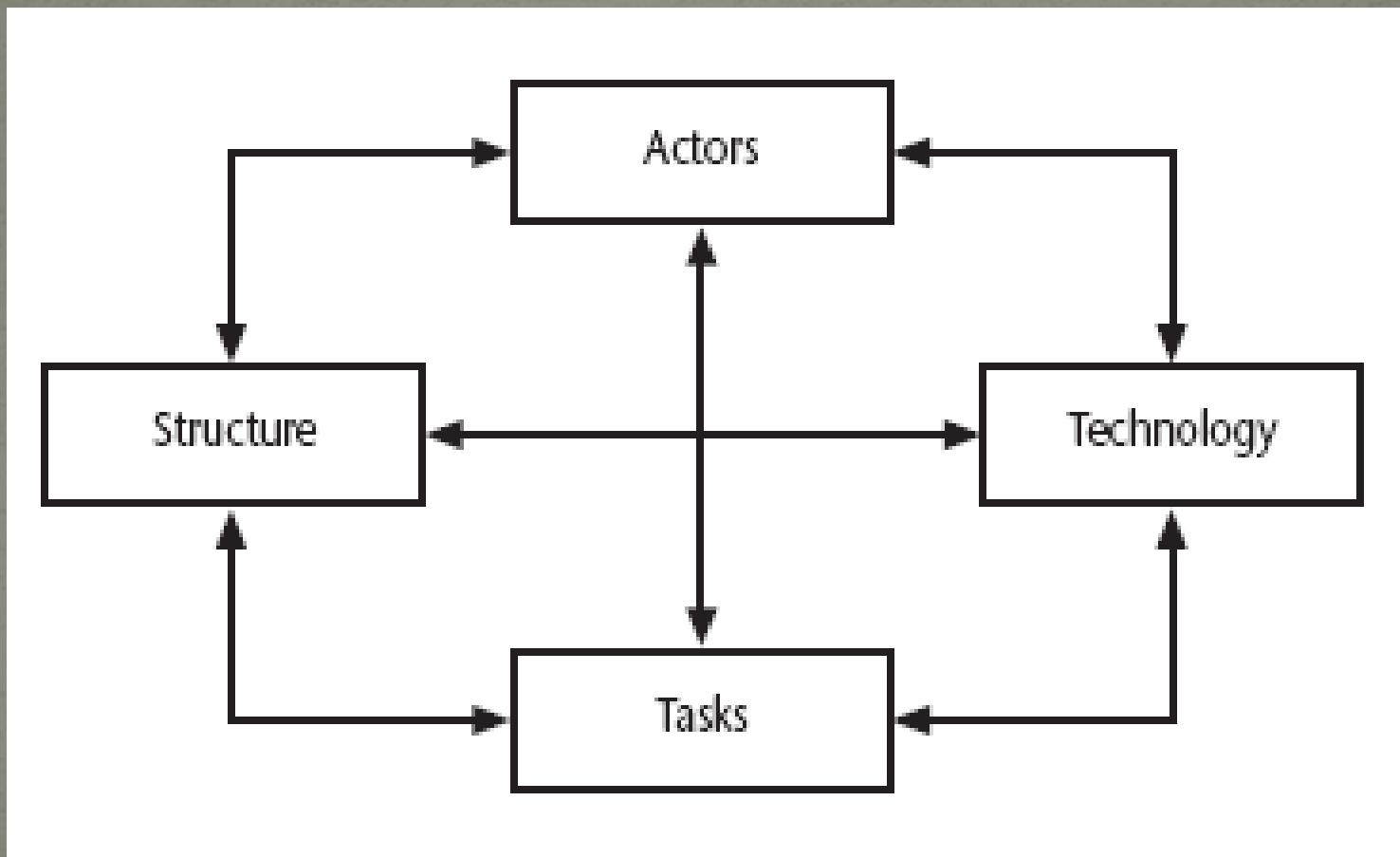
- Risk identification – what risks might there be?
- Risk analysis and prioritization – which are the most serious risks?
- Risk planning – what are we going to do about them?
- Risk monitoring – what is the current state of the risk?

Risk identification

Approaches to identifying risks include:

- Use of checklists – usually based on the experience of past projects
- Brainstorming – getting knowledgeable stakeholders together to pool concerns
- Causal mapping – identifying possible chains of cause and effect

Causal mapping - interventions



Boehm's top 10 development risks

<i>Risk</i>	<i>Risk reduction techniques</i>
Personnel shortfalls	Staffing with top talent; job matching; teambuilding; training and career development; early scheduling of key personnel
Unrealistic time and cost estimates	Multiple estimation techniques; design to cost; incremental development; recording and analysis of past projects; standardization of methods
Developing the wrong software functions	Improved software evaluation; formal specification methods; user surveys; prototyping; early user manuals
Developing the wrong user interface	Prototyping; task analysis; user involvement

Boehm's top ten risk - continued

Gold plating	Requirements scrubbing, prototyping, design to cost
Late changes to requirements	Change control, incremental development
Shortfalls in externally supplied components	Benchmarking, inspections, formal specifications, contractual agreements, quality controls
Shortfalls in externally performed tasks	Quality assurance procedures, competitive design etc
Real time performance problems	Simulation, prototyping, tuning
Development technically too difficult	Technical analysis, cost-benefit analysis, prototyping , training

Risk prioritization

Risk exposure (RE)

$$= (\text{potential damage}) \times (\text{probability of occurrence})$$

Ideally

Potential damage: a money value e.g. a flood would cause £0.5 millions of damage

Probability 0.00 (absolutely no chance) to 1.00 (absolutely certain)
e.g. 0.01 (one in hundred chance)

$$\text{RE} = £0.5\text{m} \times 0.01 = £5,000$$

Crudely analogous to the amount needed for an insurance premium

Risk Reduction Leverage (RRL)

- RRL is used to determine whether it is worthwhile to carry out the risk reduction plan.
- The higher is the RRL value, the more worthwhile is to carry out the risk reduction plan.

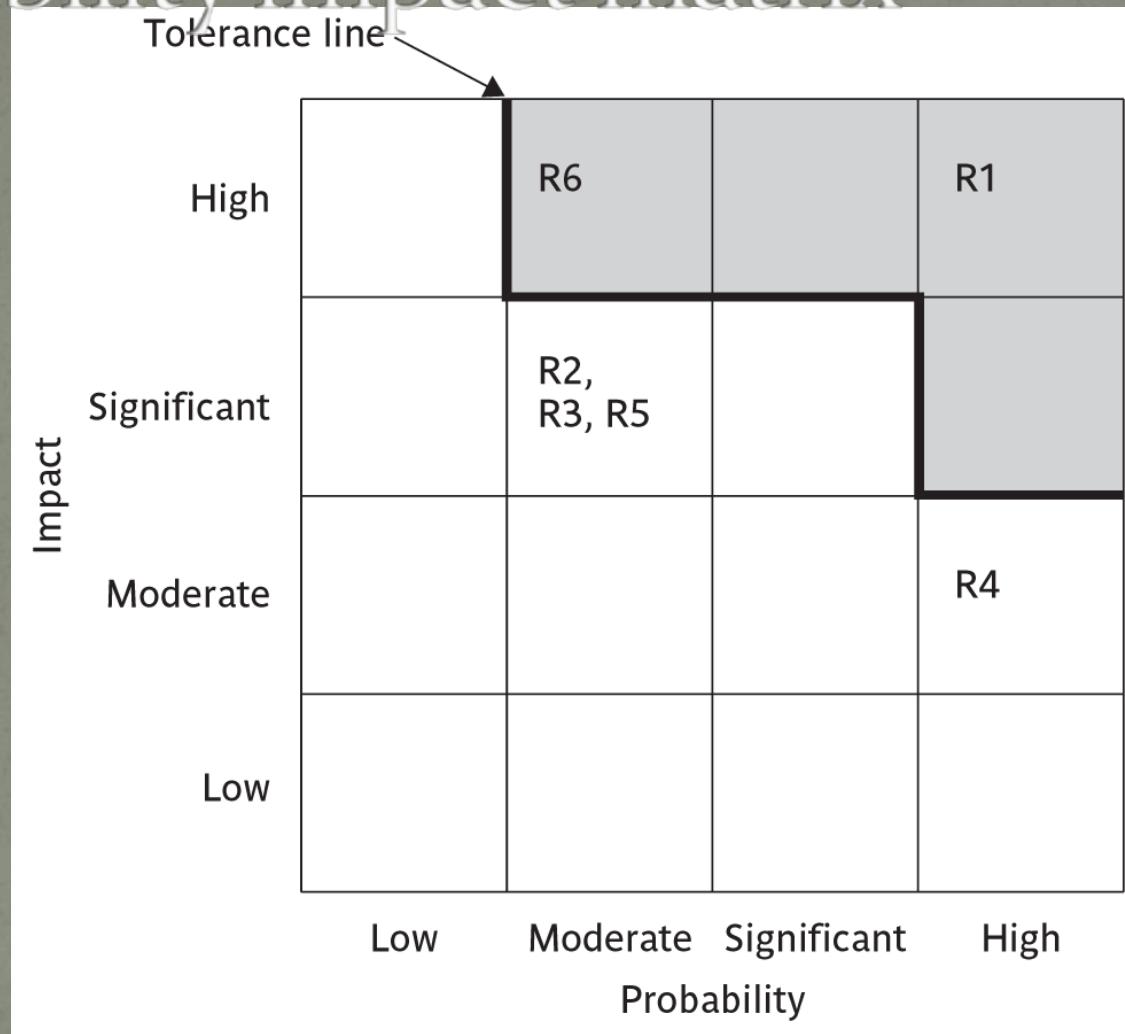
Risk probability: qualitative descriptors

<i>Probability level</i>	<i>Range</i>
High	Greater than 50% chance of happening
Significant	30-50% chance of happening
Moderate	10-29% chance of happening
Low	Less than 10% chance of happening

Qualitative descriptors of impact on cost and associated range values

<i>Impact level</i>	<i>Range</i>
High	Greater than 30% above budgeted expenditure
Significant	20 to 29% above budgeted expenditure
Moderate	10 to 19% above budgeted expenditure
Low	Within 10% of budgeted expenditure.

Probability impact matrix



Risk planning

Risks can be dealt with by:

- Risk acceptance
- Risk avoidance
- Risk reduction
- Risk transfer
- Risk mitigation/contingency measures

Risk acceptance – do nothing option. The cost of avoiding the risk may be greater than the actual cost of the damage that might be inflicted

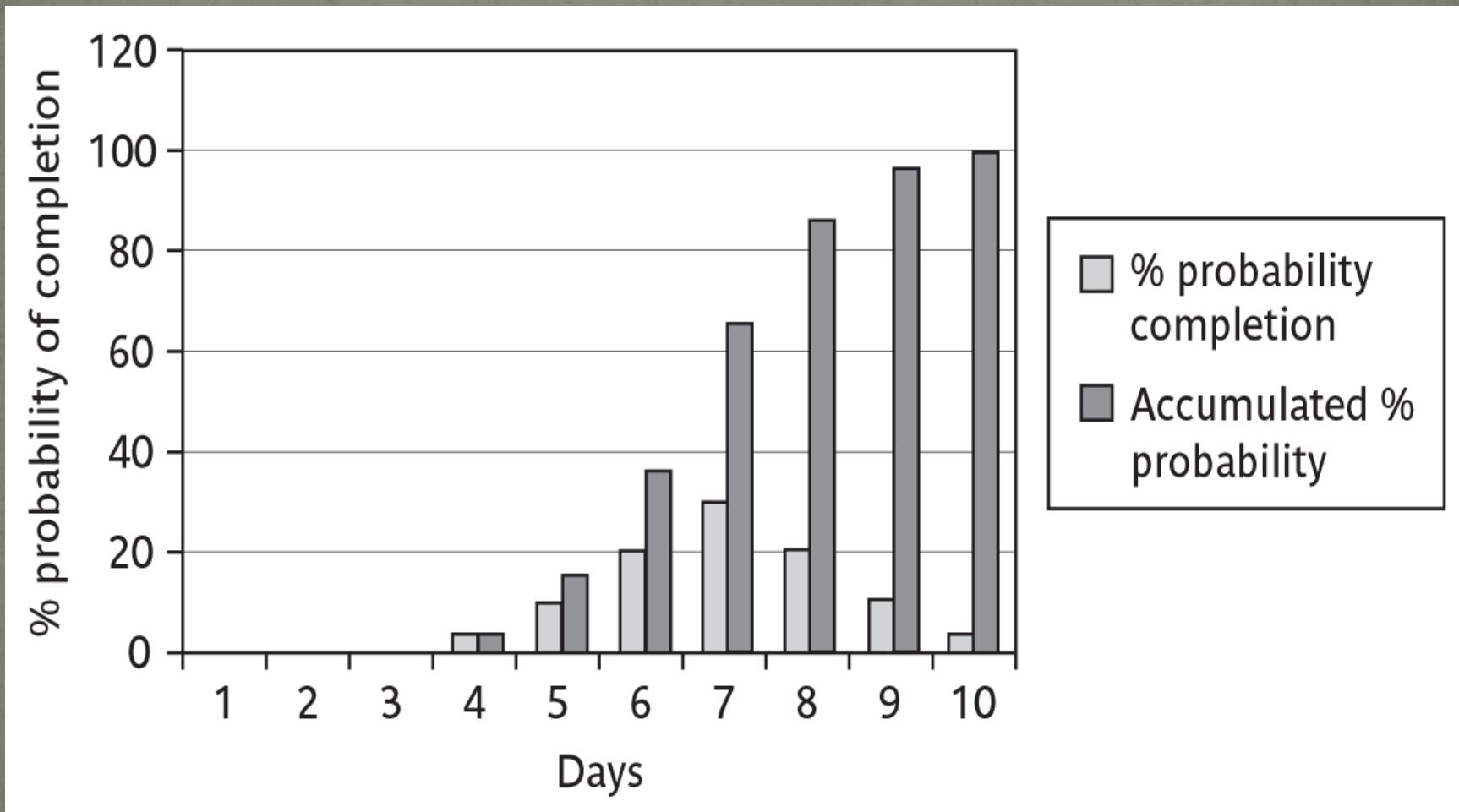
Risk avoidance – avoid the environment in which the risk occurs e.g. buying an OTS application would avoid a lot of the risks associated with software development e.g. poor estimates of effort.

Risk reduction – the risk is accepted but actions are taken to reduce its likelihood e.g. prototypes ought to reduce the risk of incorrect requirements

Risk transfer – the risk is transferred to another person or organization. The risk of incorrect development estimates can be transferred by negotiating a fixed price contract with an outside software supplier.

Risk mitigation – tries to reduce the impact if the risk does occur e.g. taking backups to allow rapid recovery in the case of data corruption

Probability chart

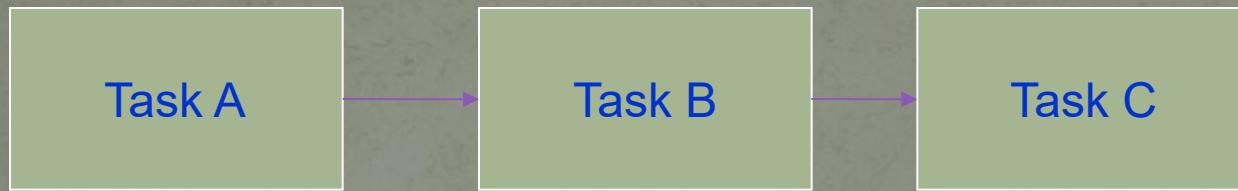


Using PERT to evaluate the effects of uncertainty

Three estimates are produced for each activity

- Most likely time (m)*
- Optimistic time (a)*
- Pessimistic (b)*
- ‘expected time’ $t_e = (a + 4m + b) / 6$
- ‘activity standard deviation’ $S = (b-a)/6$

A chain of activities



Task	a	m	b	t_e	s
A	10	12	16	?	?
B	8	10	14	?	?
C	20	24	38	?	?

A chain of activities

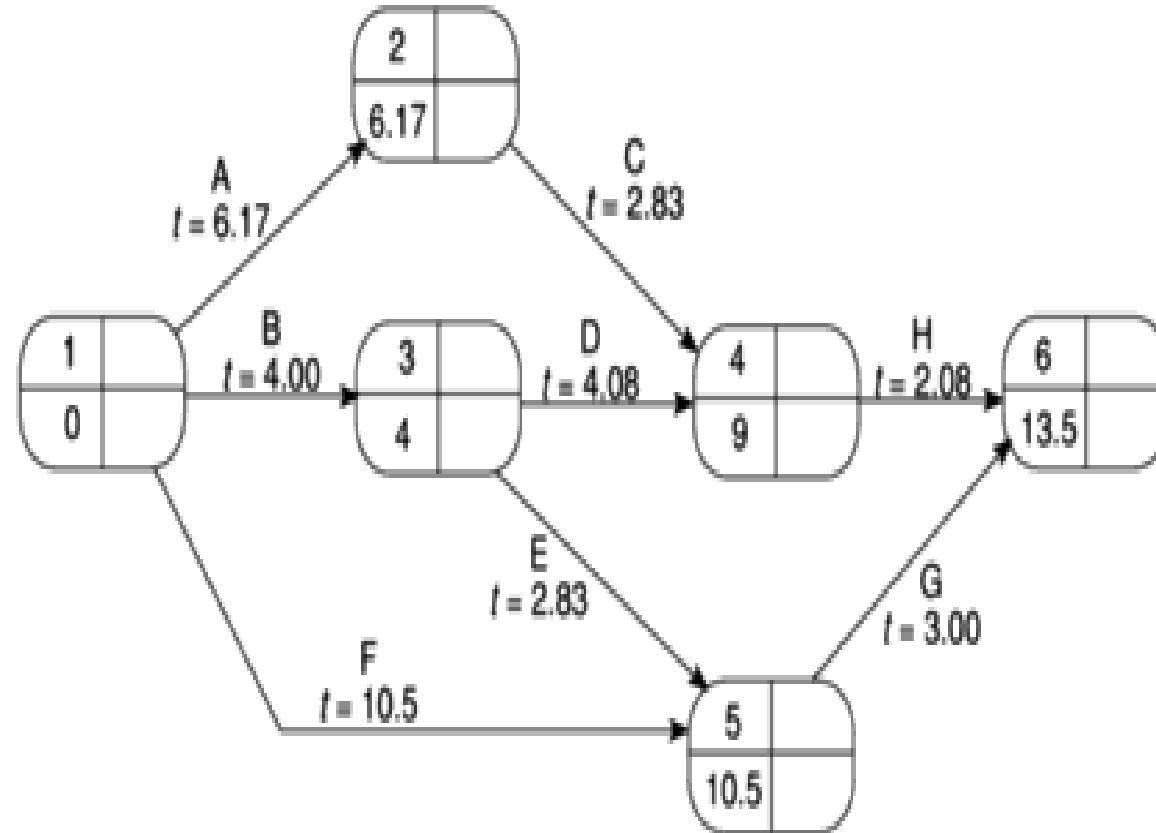
- What would be the expected duration of the chain A + B + C?
- Answer: $12.66 + 10.33 + 25.66$ i.e. 48.65
- What would be the standard deviation for A + B+ C?
- Answer: square root of $(1^2 + 1^2 + 3^2)$ i.e.
3.32

PERT activity time estimates

Activity	Precedents	Activity durations (weeks)		
		Optimistic (a)	Most likely (m)	Pessimistic (b)
A		5	6	8
B		3	4	5
C	A	2	3	3
D	B	3.5	4	5
E	B	1	3	4
F		8	10	15
G	E, F	2	3	4
H	C, D	2	2	2.5

Table 7.4 *Expected times and standard deviations*

Activity	Activity durations (weeks)				
	Optimistic (a)	Most likely (m)	Pessimistic (b)	Expected (t_e)	Standard deviation (s)
A	5	6	8	6.17	0.50
B	3	4	5	4.00	0.33
C	2	3	3	2.83	0.17
D	3.5	4	5	4.08	0.25
E	1	3	4	2.83	0.50
F	8	10	15	10.50	1.17
G	2	3	4	3.00	0.33
H	2	2	2.5	2.08	0.08



Event number	Target date
Expected date	Standard deviation

The PERT event labelling convention adopted here indicates event number and its target date along with the calculated values for expected time and standard deviation.

Figure 7.3 The PERT network after the forward pass.

The PERT technique uses the following three-step method for calculating the probability of meeting or missing a target date:

- calculate the standard deviation of each project event;
- calculate the z value for each event that has a target date;
- convert z values to a probabilities.

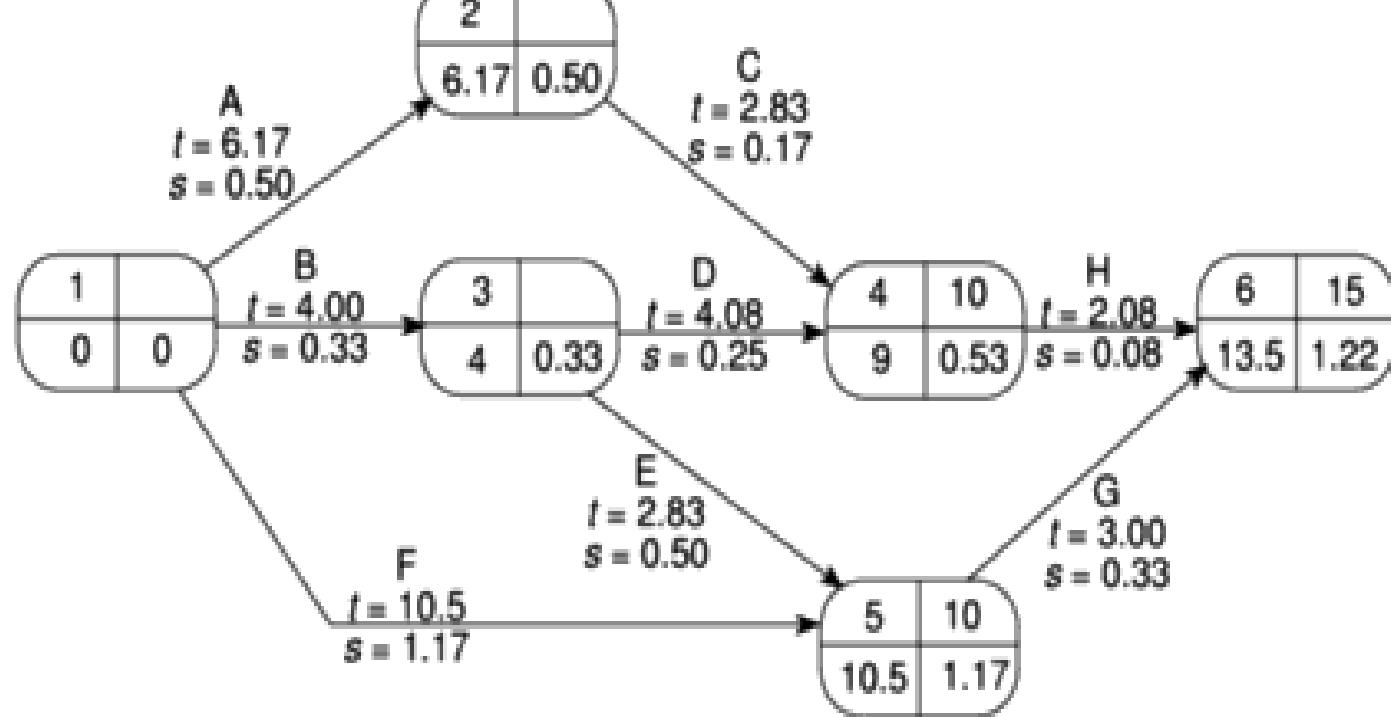


Figure 7.4 The PERT network with three target dates and calculated event standard deviations.

Calculating the standard deviation of each project event

Standard deviations for the project events can be calculated by carrying out a forward pass using the activity standard deviations in a manner similar to that used with expected durations. There is, however, one small difference – to add two standard deviations we must add their squares and then find the square root of the sum.

The standard deviation for event 3 depends solely on that of activity B. The standard deviation for event 3 is therefore 0.33.

For event 5 there are two possible paths, B + E or F. The total standard deviation for path B + E is $\sqrt{(0.33^2 + 0.50^2)} = 0.6$ and that for path F is 1.17; the standard deviation for event 5 is therefore the greater of the two, 1.17.

Calculating the z values

The z value is calculated for each node that has a target date. It is equivalent to the number of standard deviations between the node's expected and target dates. It is calculated using the formula

$$z = \frac{T - t_e}{s}$$

where t_e is the expected date and T the target date.

The z value for event 4 is $(10 - 9.00)/0.53 = 1.8867$.

The z value for the project completion (event 6) is 1.23. Using Figure 7.5 we can see that this equates to a probability of approximately 11%, that is, there is an 11% risk of not meeting the target date of the end of week 15.

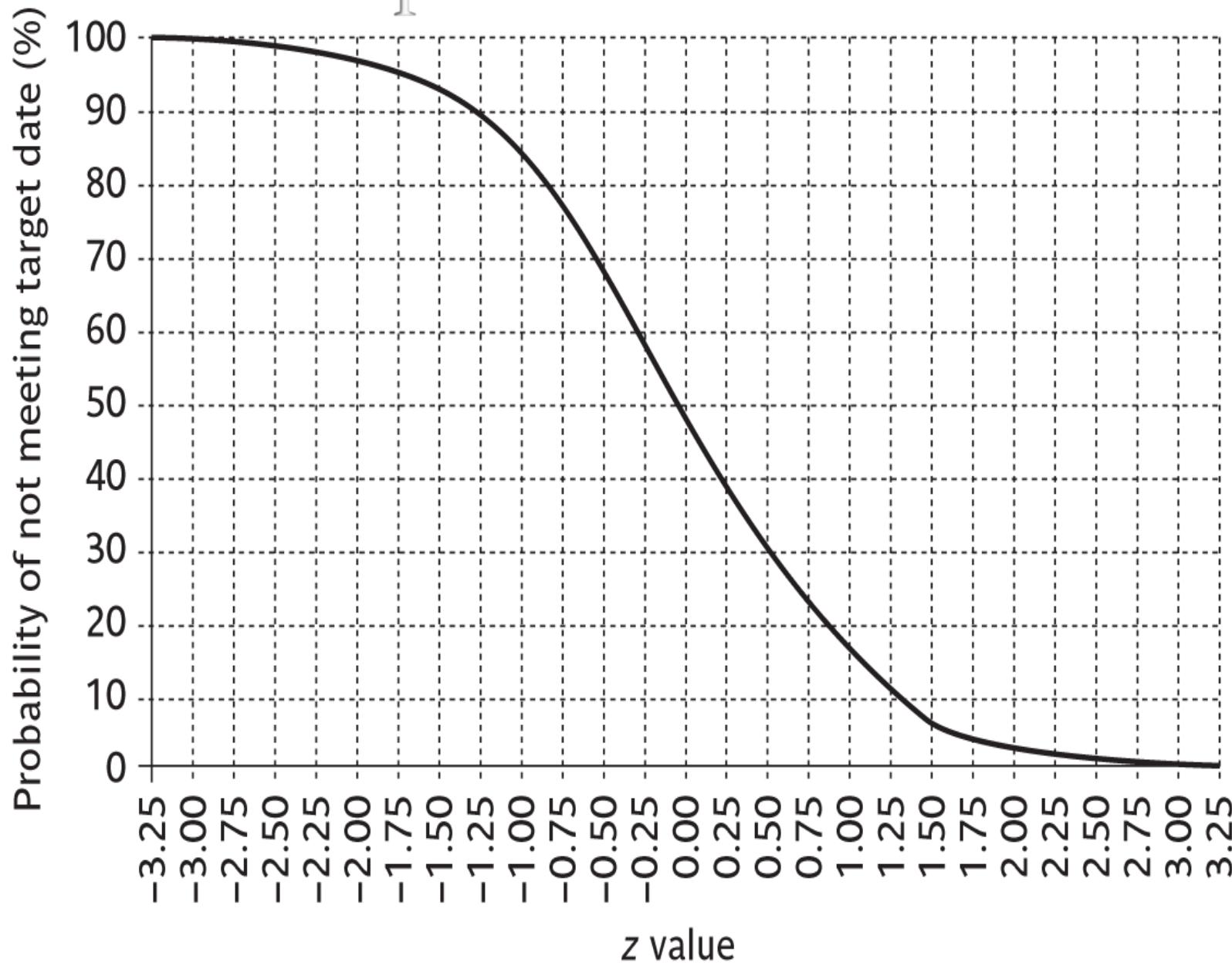
Find the probabilities of not achieving events 4 or 5 by their target dates of the end of week 10.

What is the likelihood of completing the project by week 14?

Assessing the likelihood of meeting a target

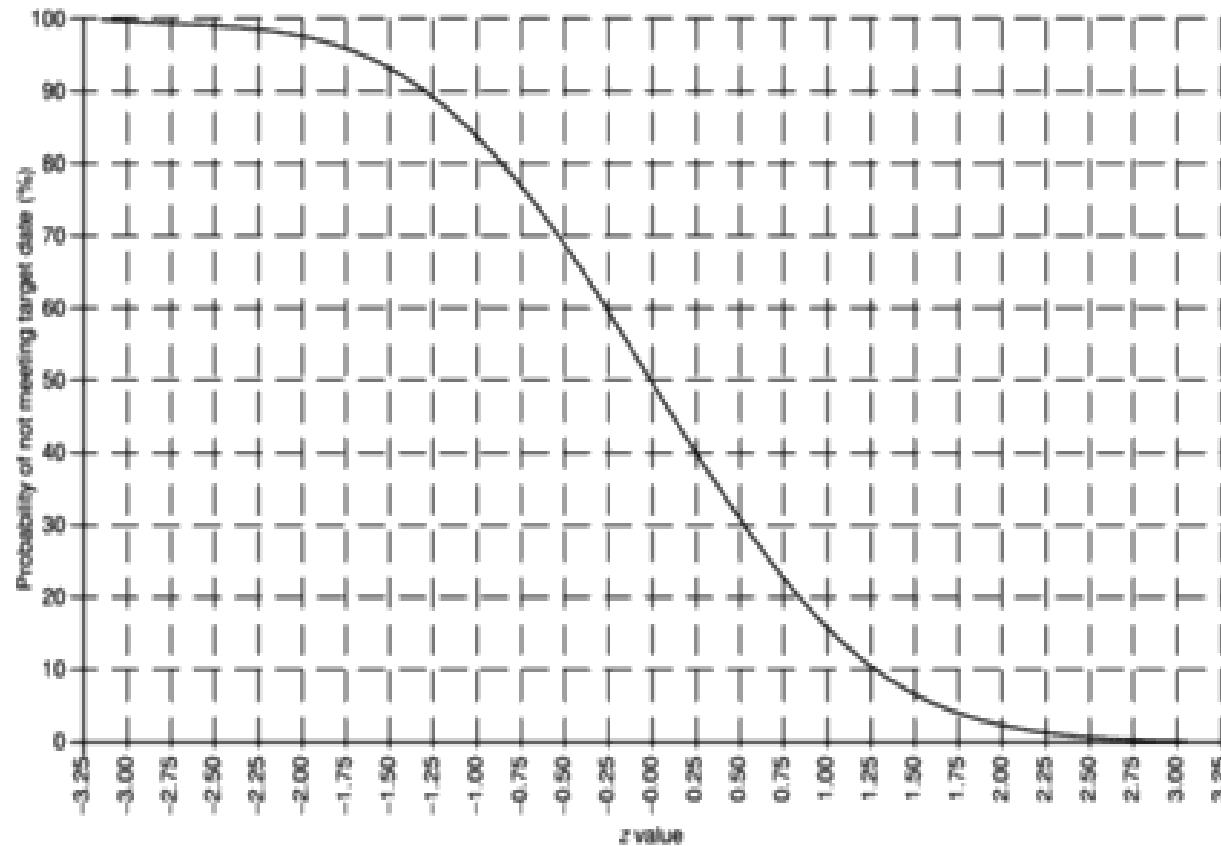
- Say the target for completing A+B+C was 52 days (T)
- Calculate the z value thus
$$z = (T - t_e)/s$$
- In this example $z = (52-48.33)/3.32$ i.e. 1.01
- Look up in table of z values – see next overhead

Graph of z values



Converting z values to probabilities

A z value may be converted to the probability of not meeting the target date by using the graph in Figure 7.5.



This graph is the equivalent of tables of z values, also known as standard normal deviates, which may be found in most statistics textbooks

Figure 7.5 *The probability of obtaining a value within z standard deviations of the mean for a normal distribution.*

Monte Carlo Simulation

- An alternative to PERT.
- A class of general analysis techniques:
 - Valuable to solve any problem that is complex, nonlinear, or involves more than just a couple of uncertain parameters.
- Monte Carlo simulations involve repeated random sampling to compute the results.
- Gives more realistic results as compared to manual approaches.

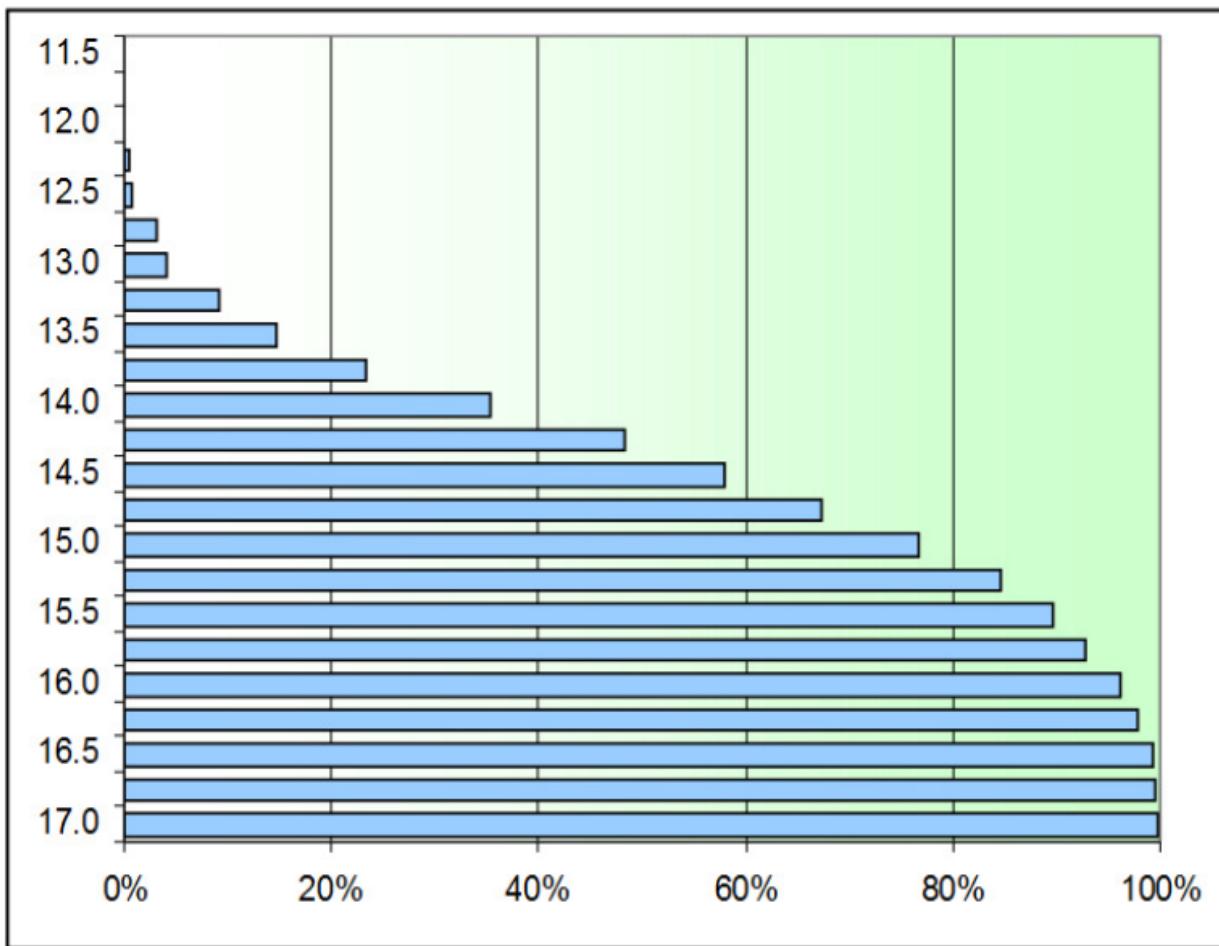


Figure 1: Probability of Completion Within Specified Time (Months)

Critical chain approach

One problem with estimates of task duration:

- Estimators add a safety zone to estimate to take account of possible difficulties
- Developers work to the estimate + safety zone, so time is lost
- No advantage is taken of opportunities where tasks can finish early – and provide a buffer for later activities

Critical chain approach

One answer to this:

1. Ask the estimators for two estimates
 - Most likely duration: 50% chance of meeting this
 - Comfort zone: additional time needed to have 95% chance
2. Schedule all activities using most likely values and starting all activities on latest start dates

Critical chain approach

Originally this concept is developed by Eliyahu Goldratt. CCPM helps to overcome the following phenomenon

- - Parkinson's Law: Work expands to fill the available time.
- - Student Syndrome: People start to work in full fledge only when deadline is near.
- - Murphy's Law: What can go wrong will go wrong.
- - Bad Multi Tasking: Bad multitasking can delay start of the successor tasks

General steps in critical chain concept

- The target date is given to the developer is one where it is estimated that there is a 50% chance of success.
- Working towards from the target completion date, each activity is scheduled to start as late as possible. Among other things, this should reduce the chance of staff being pulled off the project on to other work.

Scheduling Buffers

- In traditional estimates, people often add a buffer to each task and use it if it's needed or not
- Critical chain scheduling removes buffers from individual tasks and instead creates:
 - **Project buffers** or additional time added before the project's due date
 - **Feeding buffers** or additional time added before tasks on the critical path

Project buffer

- The ***project buffer*** protects the project from missing its scheduled end date due to variations along the critical chain. It places a portion of the safety margin time that was removed from each task estimate into a buffer task, thus moving the times of uncertainty from individual tasks to a pooled buffer task

Project buffer

- The project buffer is inserted between the final scheduled task and the scheduled project end date. The critical chain starts at the beginning of the project and ends at the start of the project buffer, not at the end of the project.
- Time is added to or subtracted from the project buffer as the actual time required to complete each task changes.
- The project buffer will be calculated as
half the sum of the comfort zones of the activities on the critical chain

Feeding buffer

- The **feeding buffer** minimizes the risk that late completion of a non-critical chain task will affect the critical chain. The project manager inserts an amount of time at those points in the schedule where inputs from non-critical chain tasks merge with critical chain tasks. The result is very similar to a relay race where the speed of the race, in general, is able to be maintained by the overlap in runners at the hand-off point.
- Insert feeding buffers at points **where non-critical chain paths intersect the critical chain**
It can be calculated as ...**half the size of the comfort time taken out of the feeding path.**

Comfort zone

- Comfort zone is the difference between the pessimistic and the most likely durations

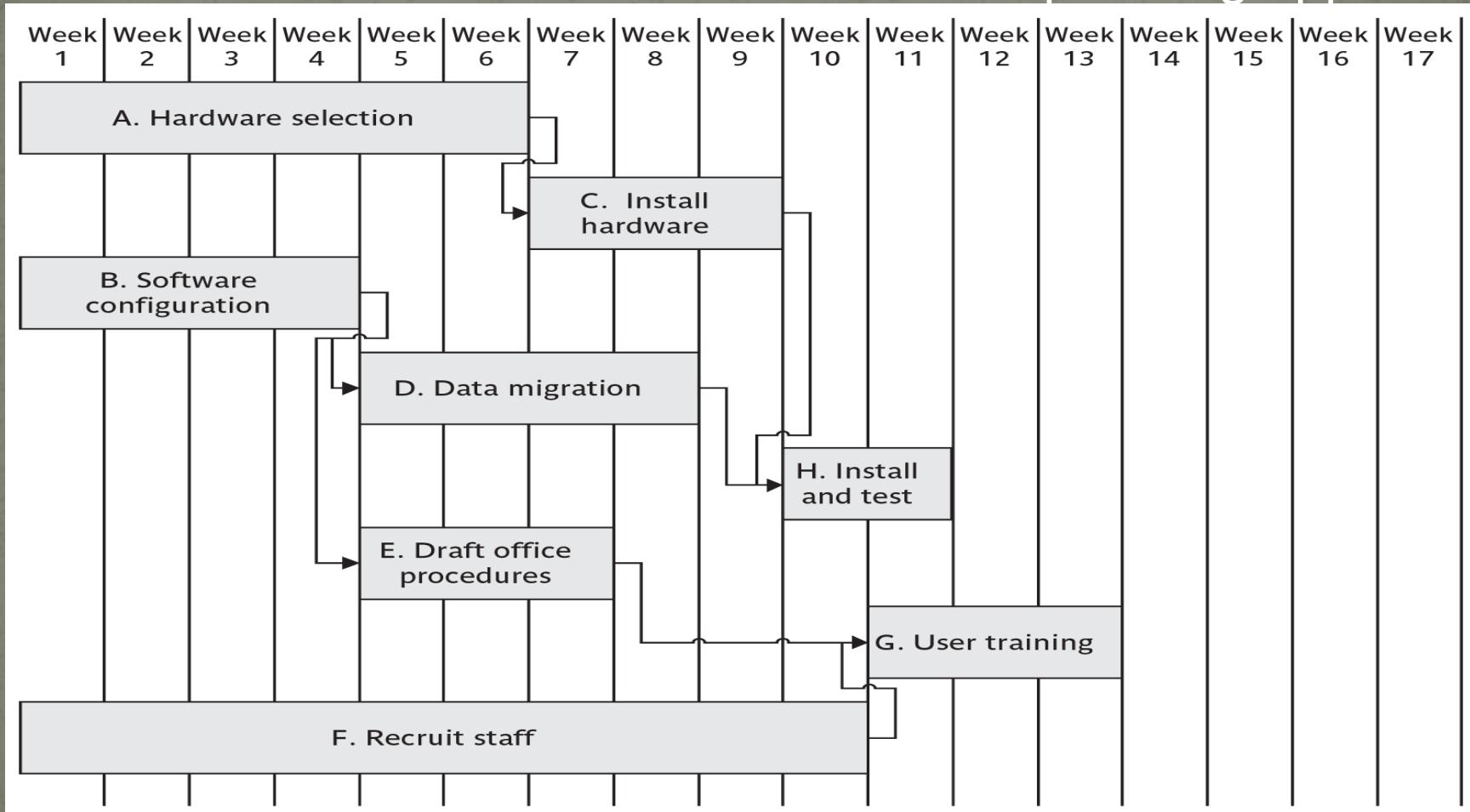
Most likely and comfort zone estimates

Activity	Most likely	Plus comfort zone	Comfort zone
A	6	8	2
B	4	5	1
C	3	3	0
D	4	5	1
E	3	4	1
F	10	15	5
G	3	4	1
H	2	2.5	0.5

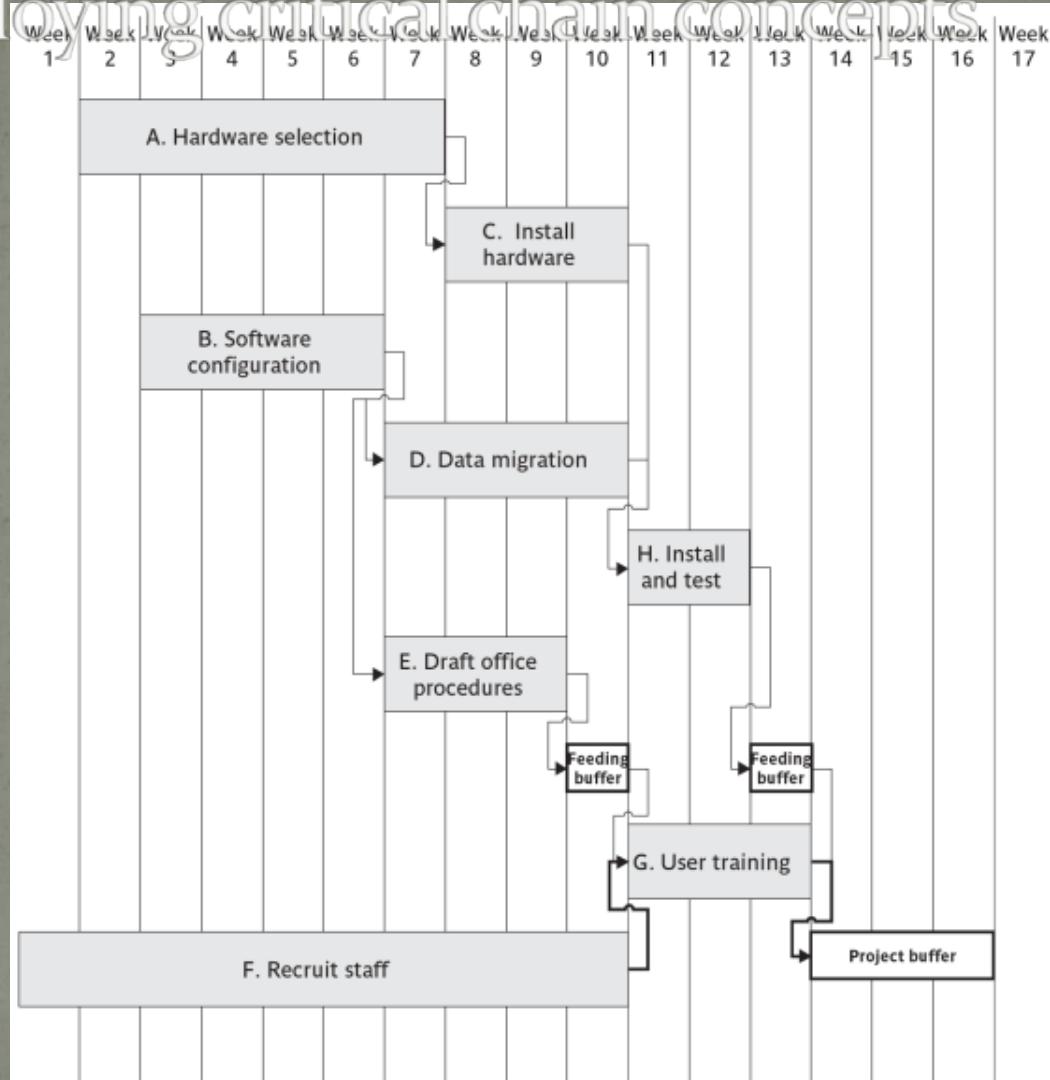
TABLE 7.8 Most likely and comfort zone estimates (days)

Critical chain concept

Traditional planning approach



Plan employing critical chain concepts



<i>Activity</i>	<i>Depends on</i>	<i>Optimistic time</i>	<i>Most likely</i>	<i>Pessimistic</i>
A	-	8	10	12
B	A	10	15	20
C	B	5	7	9
D	-	8	10	12
E	D,C	3	6	9

Using the activity table

- Calculate the expected duration and standard deviation for each activity
- Identify the critical path
- Draw up an activity diagram applying critical chain principles for this project
- Locate the places where the buffers will need to be located
- Assess the size of the buffers
- Start all activities as late as possible

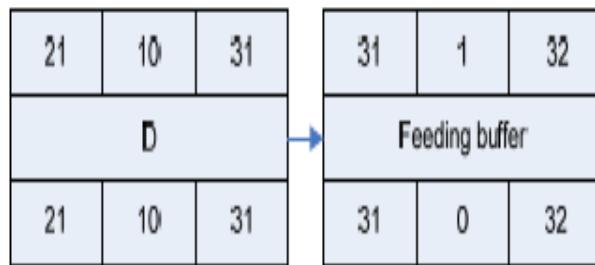
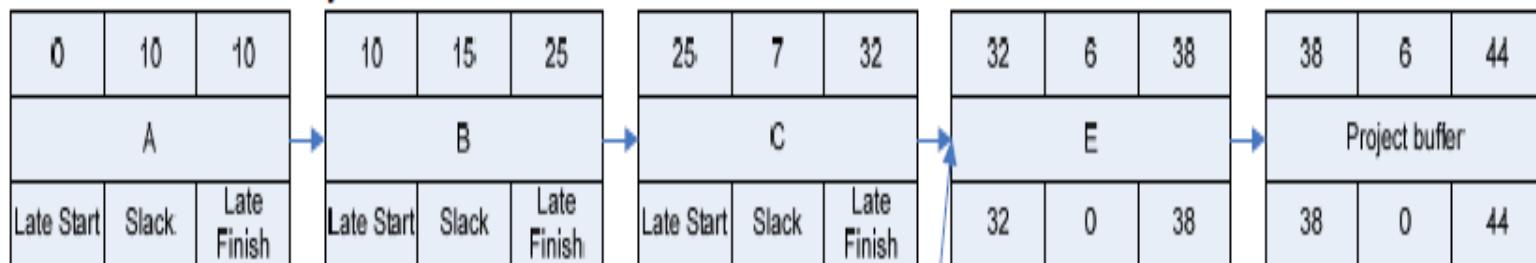
					activity duration		start		end	
activity	Depends on	Optimistic time	Most likely	Pessimistic	te	s	expected time	stdev	expected time	stdev
A	-	8	10	12	10	0.67	0	0.00	10	0.67
B	A	10	15	20	15	1.67	10	0.67	25	1.80
C	B	5	7	9	7	0.67	25	1.80	32	1.91
D	-	8	10	12	10	0.67	0.00	0.00	10	0.67
E	D,C	3	6	9	6	1.00	32	1.91	38	2.16

The critical path would be A,B,C,E.

Note: project buffer is half the sum of the differences between the pessimistic and most likely durations of the activities on the critical chain: i.e.

$$((12-10) + (20 - 15) + (9-7) + (9-6))/2 = (2+5+2+3)/2 = 6 \text{ days}$$

The feeding buffer is half the difference between the pessimistic and the most likely durations for Activity D



<i>Activity</i>	<i>Depends on</i>	<i>Most likely</i>	<i>Plus safety</i>
A		10	14
B	A	5	7
C	B	15	21
D	A	3	5
E	A	8	12
F	E	20	22
G	D	6	8
H	C,F,G	10	14

Using the activity table

- Calculate the expected duration and standard deviation for each activity
- Identify the critical path
- Draw up an activity diagram applying critical chain principles for this project
- Locate the places where the buffers will need to be located

ES = Earliest start EF = Earliest finish LS = Latest start LF = Latest finish

The critical path (with all floats zero) is A,E,F,H

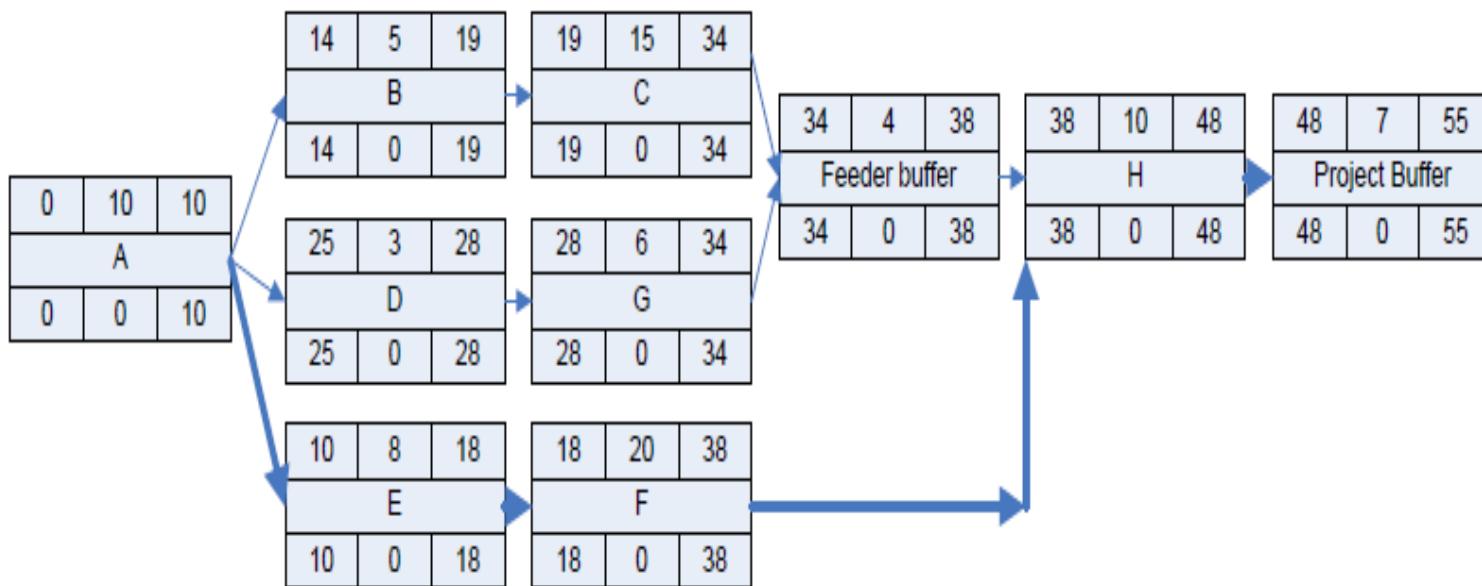
Based on most likely durations

Activity	Depends on	Most likely	ES	Duration	EF	LS	LF	Float
A		10	0	10	10	0	10	0
B	A	5	10	5	15	18	23	8
C	B	15	15	15	30	23	38	8
D	A	3	10	3	13	29	32	19
E	A	8	10	8	18	10	18	0
F	E	20	18	20	38	18	38	0
G	D	6	13	6	19	32	38	19
H	C,F,G	10	38	10	48	38	48	0

Based on durations with a safety factor

Activity	Depends on	Plus safety	ES	dur	EF	LS	LF	float
A		14	0	14	14	0	14	0
B	A	7	14	7	21	20	27	6
C	B	21	21	21	42	27	48	6
D	A	5	14	5	19	35	40	21
E	A	12	14	12	26	14	26	0
F	E	22	26	22	48	26	48	0
G	D	8	19	8	27	40	48	21
H	C,F,G	14	48	14	62	48	62	0

Note that there is only one feeder buffer for B+C and D+G, as these are running in parallel. In this case they share a buffer with the size based on whichever feeding buffer would have been bigger. For B+C this would have been 4 days (50% of 2+6 days) as opposed to 2 days for D+G (50% of 2+2 days).



Executing the critical chain-based plan

- No **chain** of tasks is started earlier than scheduled, but once it has started is finished as soon as possible
- This means the activity following the current one starts as soon as the current one is completed, even if this is early – the relay race principle

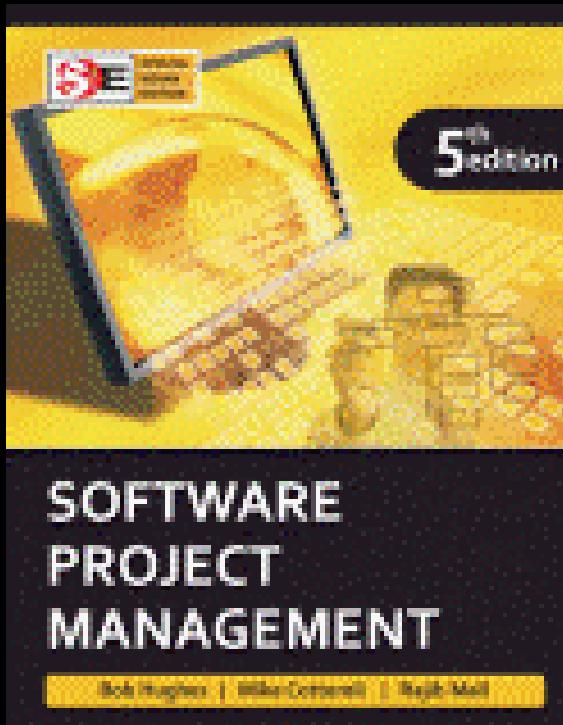
Executing the critical chain-based plan

Buffers are divided into three zones:

- **Green**: the first 33%. No action required
- **Amber** : the next 33%. Plan is formulated
- **Red** : last 33%. Plan is executed.

Software Project Management

Fifth Edition



Chapter 8

Resource allocation

Introduction

- Resources are the people, equipment and materials needed to complete the task in a project.
- In resource allocation we will see how to match the activity plan to available resources and assess the efficacy of changing the plan to fit the resources.

Schedules

- **Activity schedule** - indicating start and completion dates for each activity
- **Resource schedule** - indicating dates when resources needed + level of resources
- **Cost schedule** showing accumulative expenditure

Resources

- These include
 - ◆ labour
 - ◆ equipment (e.g. workstations)
 - ◆ materials
 - ◆ space
 - ◆ Services
 - ◆ Time:
 - ◆ Money

Nature of Resources

- Resources can be any item from paper clip to key personnel.
- Resources fall into one of seven categories:
 - Labour: Members of the development team and any employees participate in specific activities. e.g. Project Manager, Analysts
 - Equipment: Include workstations or office equipment.

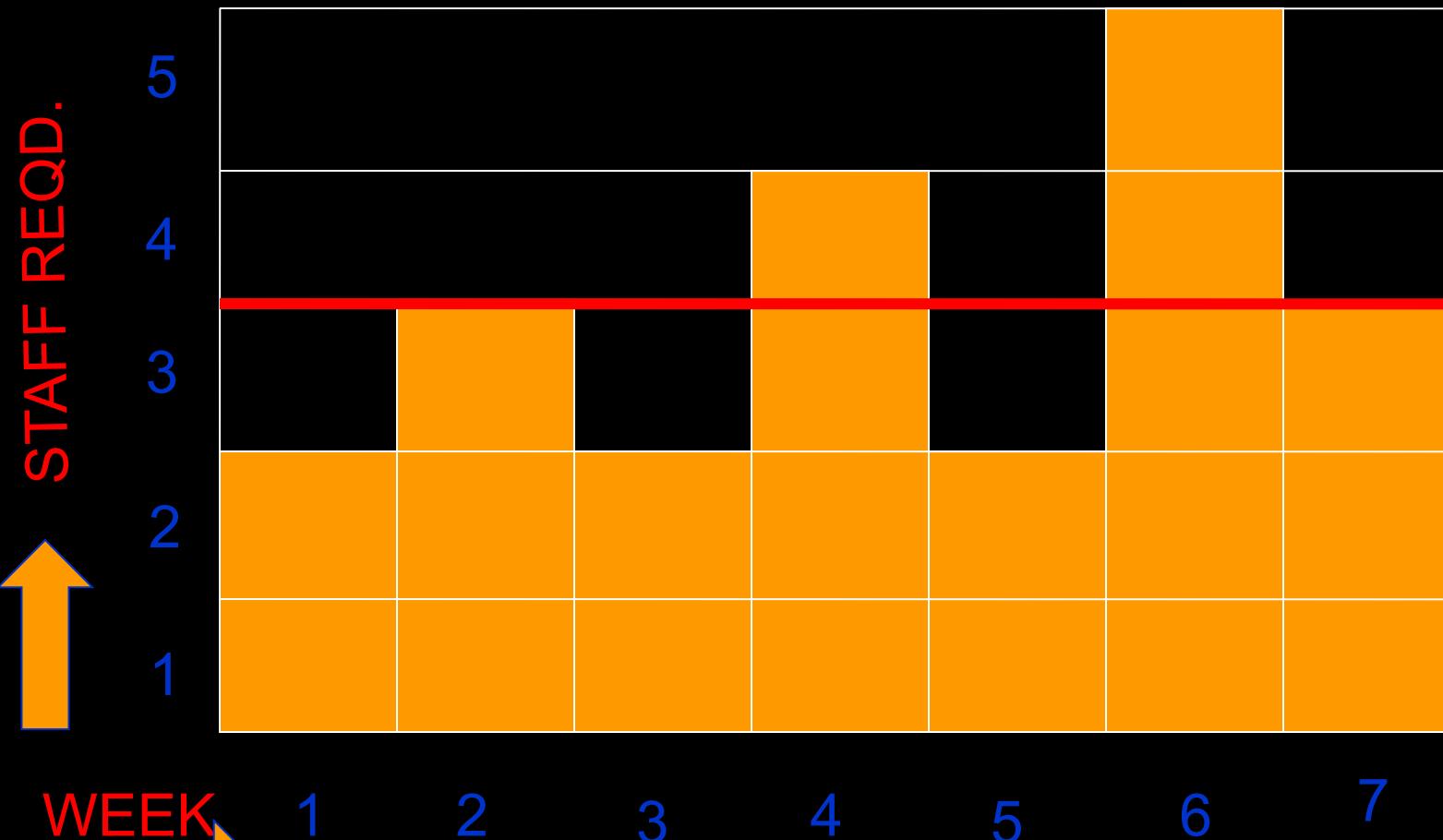
Nature of Resources (Cont..)

- **Materials**: Resources that we consume as the project proceeds. e.g floppy disk.
- **Space**: Office space is needed for existing staff and for contracted or recruited one.
- **Services**: Require procurement of services e.g. telecommunication
- **Time**: Primary resources, project timescales can be reduced by increasing resources. Elapsed time can often be reduced by adding more staff
- **Money**: Secondary resources ,used to buy other resources.

Resource allocation

- Identify the resources needed for each activity and create a *resource requirement list*
- Identify *resource types* - individuals are interchangeable within the group (e.g. 'VB programmers' as opposed to 'software developers')
- Allocate resource types to activities and examine the *resource histogram*

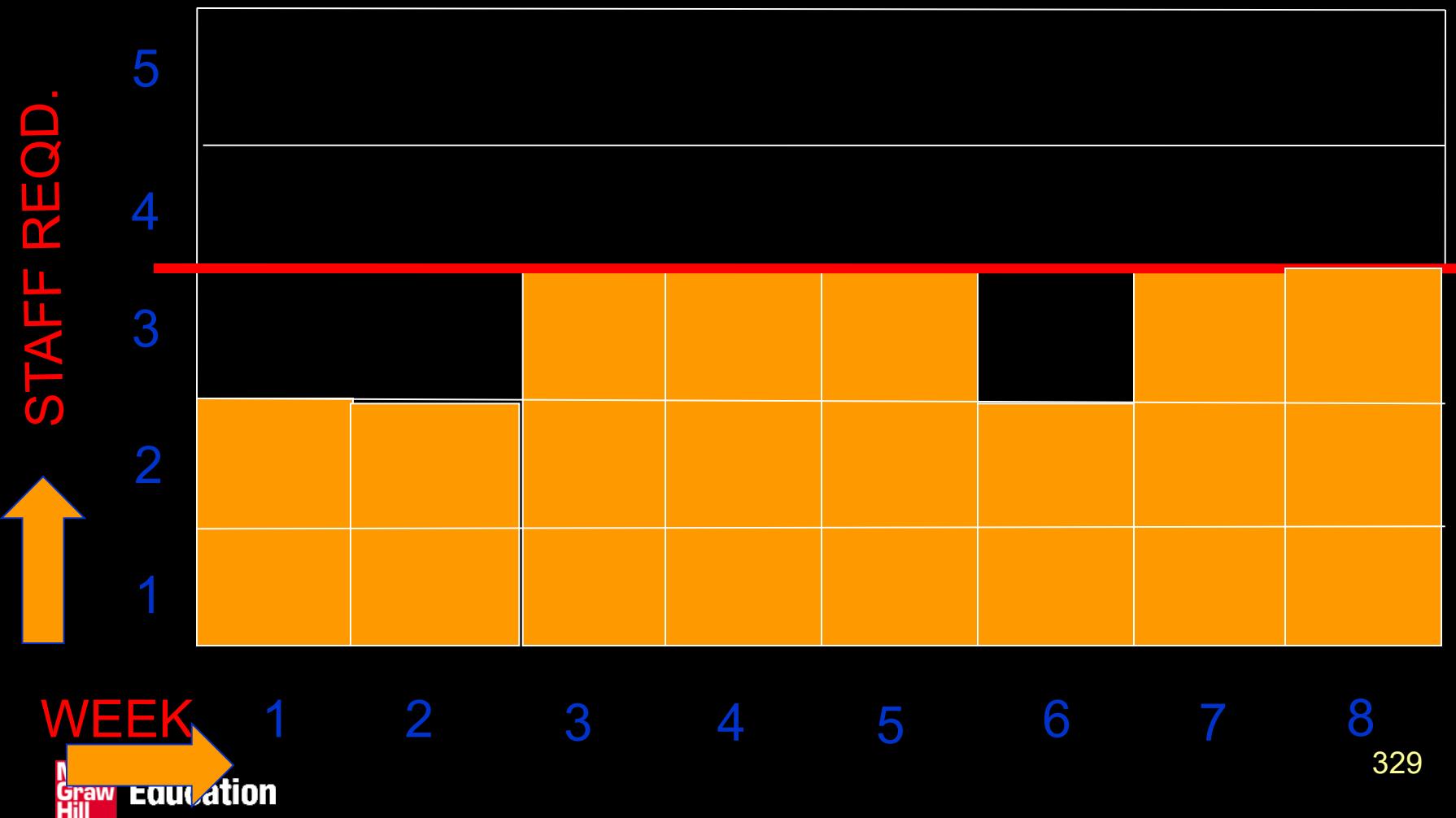
Resource histogram: systems analysts



Resource smoothing

- It is usually difficult to get specialist staff who will work odd days to fill in gaps – need for staff to learn about application etc
- Staff often have to be employed for a continuous block of time
- Therefore desirable to employ a constant number of staff on a project – who as far as possible are fully employed
- Hence need for **resource smoothing**

Resource smoothing



Resource clashes

- Where same resource needed in more than one place at the same time
- can be resolved by:
 - ◆ delaying one of the activities
 - taking advantage of float to change start date
 - delaying start of one activity until finish of the other activity that resource is being used on - *puts back project completion*
 - ◆ moving resource from a non-critical activity
 - ◆ bringing in additional resource - *increases costs*

Prioritizing activities

There are two main ways of doing this:

- *Total float priority* – those with the smallest float have the highest priority
- *Ordered list priority* – this takes account of the duration of the activity as well as the float – see next overhead

Burman's priority list

Give priority to:

- Shortest critical activities
- Other critical activities
- Shortest non-critical activities
- Non-critical activities with least float
- Non-critical activities

Resource usage

- need to maximise %usage of resources i.e. reduce idle periods between tasks
- need to balance costs against early completion date
- need to allow for contingency

Allocating individuals to activities

The initial ‘resource types’ for a task have to be replaced by actual individuals.

Factors to be considered:

- Availability
- Criticality
- Risk
- Training
- Team building – and motivation

Cost schedules

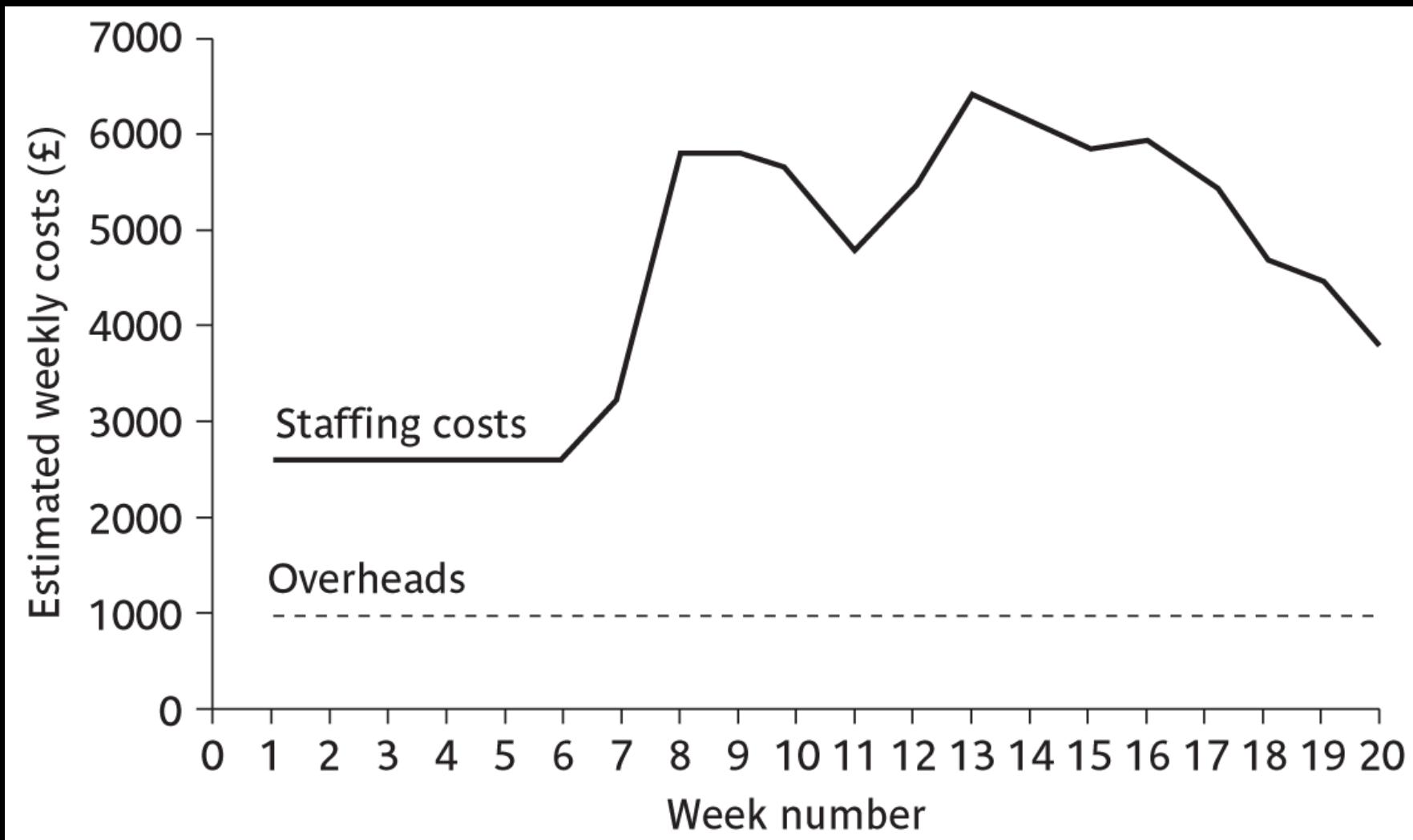
Cost schedules can now be produced:

Costs include:

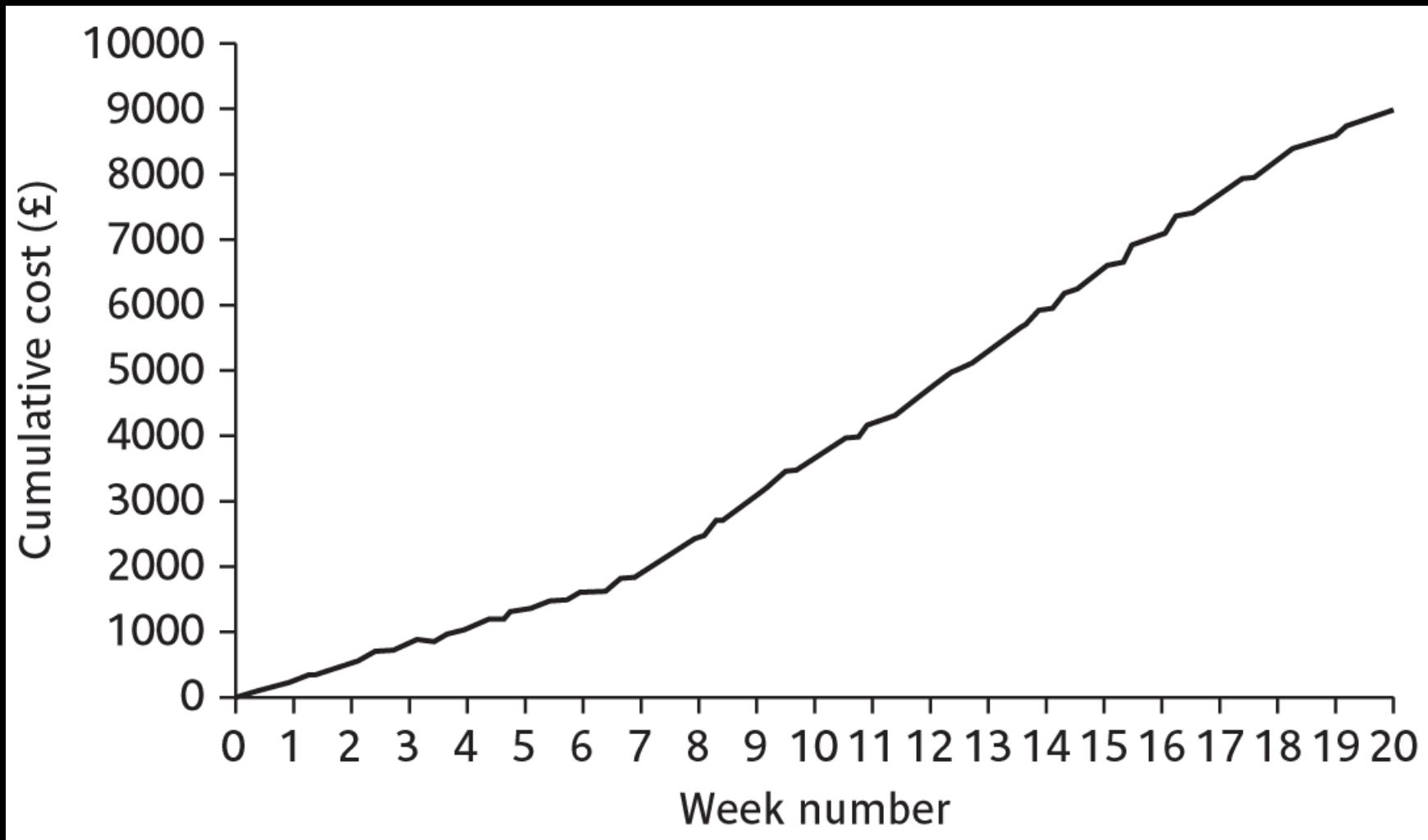
- Staff costs
- Overheads
- Usage charges

- **Staff costs** – includes not just salary, but also social security contributions by the employer, holiday pay etc. Timesheets are often used to record actual hours spent on each project by an individual. One issue can be how time when a staff member is allocated and available to the project, but is not actually working on the project, is dealt with.
- **Overheads** e.g. space rental, service charges etc. Some overheads might be directly attributable to the project; in other cases a percentage of departmental overheads may be allocated to project costs.
- **Usage charges** – some charges can be on a ‘pay as you go’ basis e.g. telephone charges, postage, car mileage – at the planning stage an estimate of these may have to be made

Cost profile

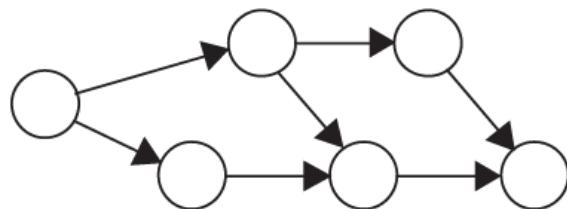


Accumulative costs



Balancing concerns

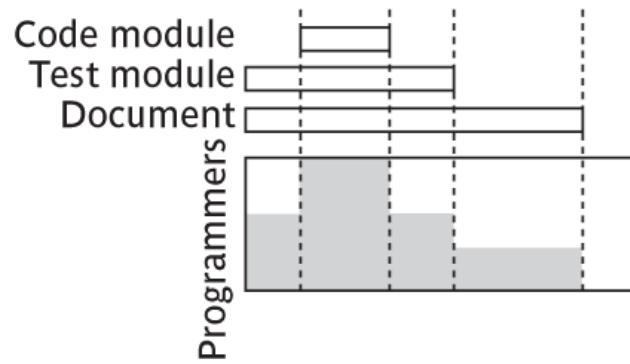
Activity plan



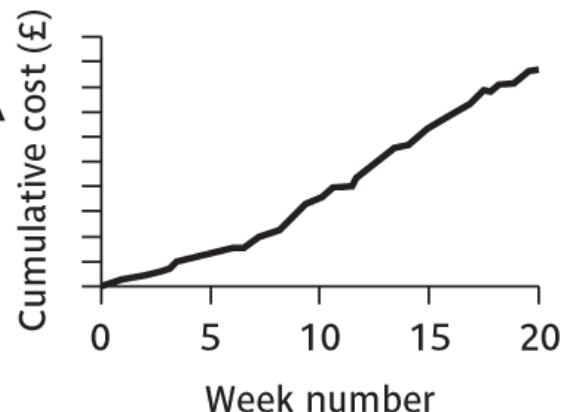
Risk assessment



Resource allocation



Cost schedule



Software Project Management

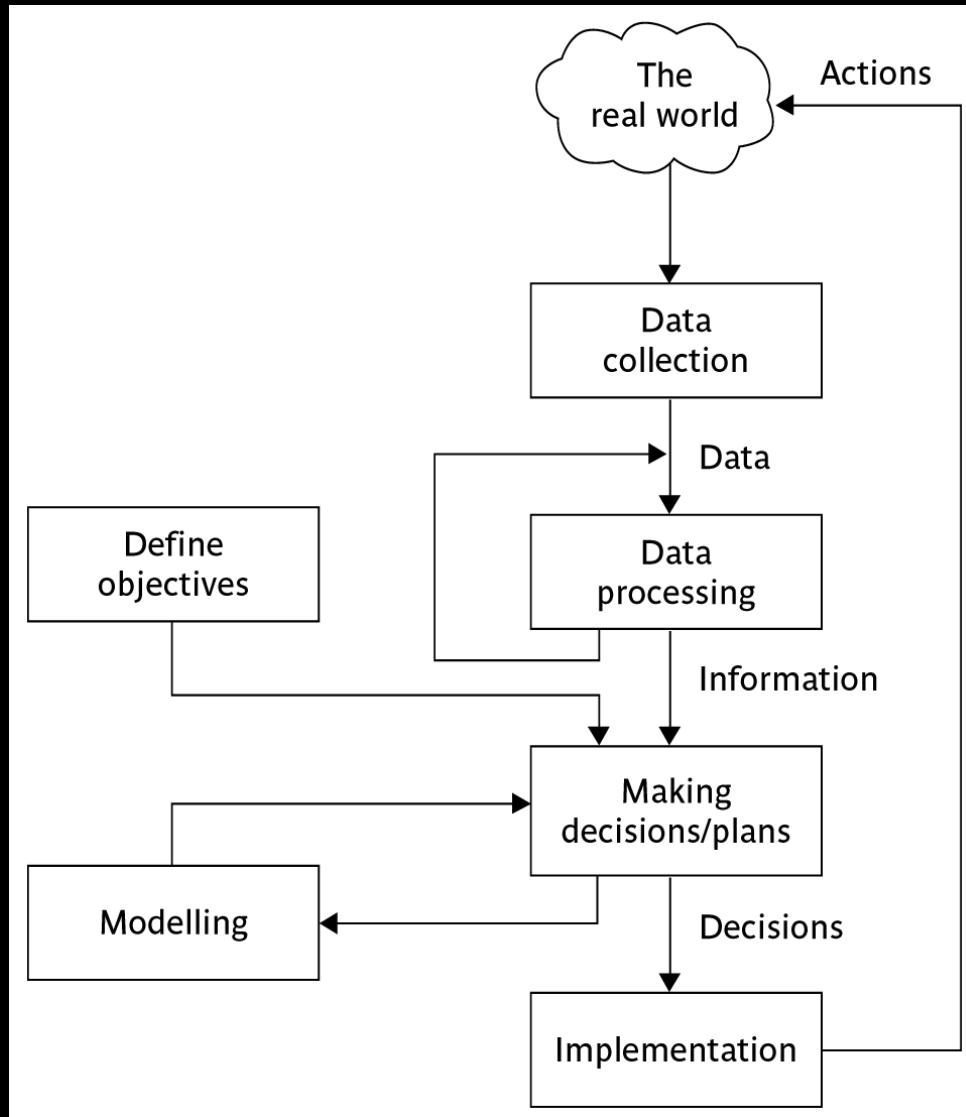
Fifth Edition



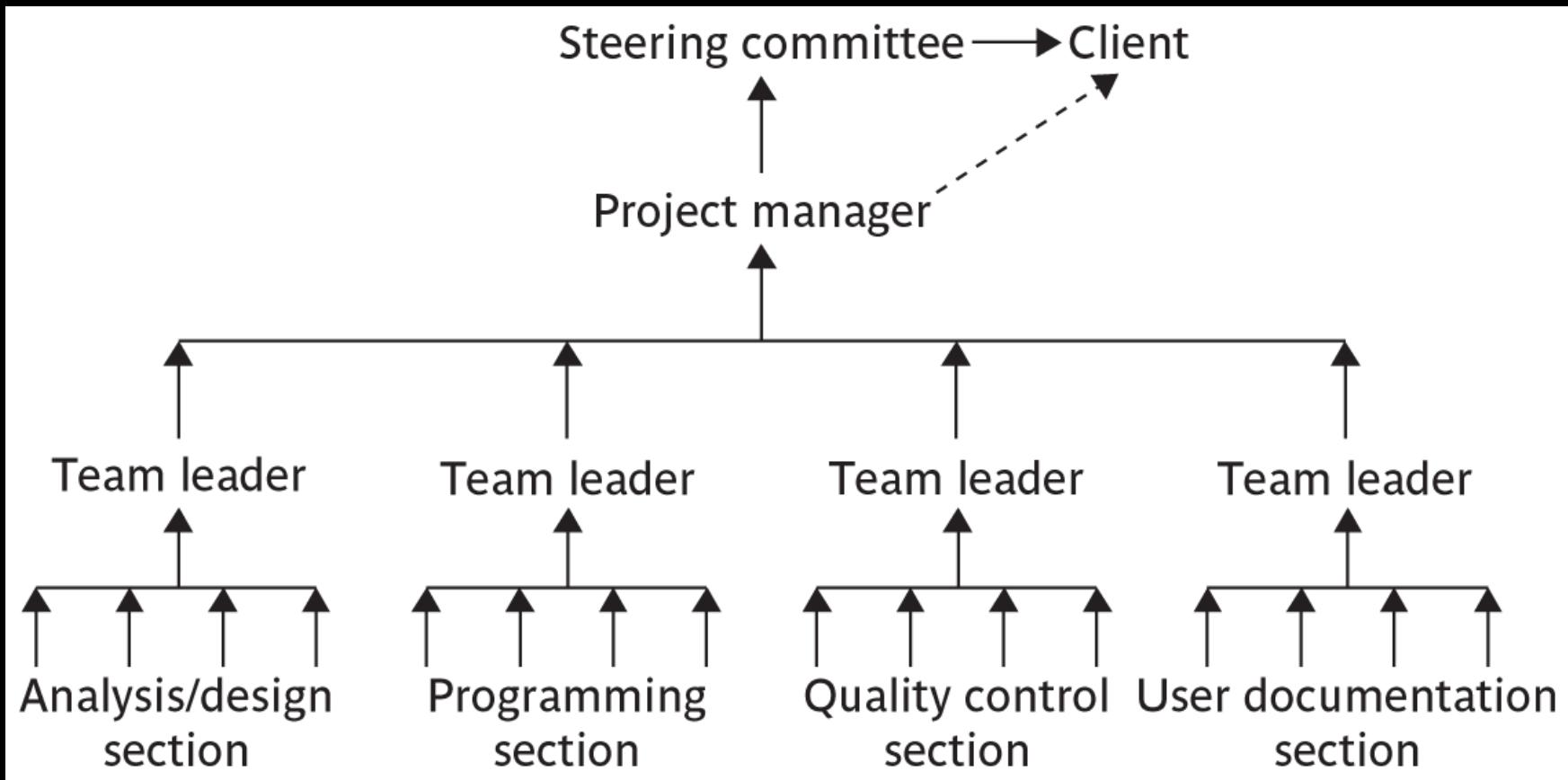
Chapter 9

Monitoring and control

The control cycle



Responsibilities



Assessing progress



Checkpoints – predetermined times when progress is checked

- ◆ Event driven: check takes place when a particular event has been achieved
- ◆ Time driven: date of the check is pre-determined

Frequency of reporting

The higher the management level then generally the longer the gaps between checkpoints

Collecting progress details

Need to collect data about:

- Achievements
- Costs

A big problem: how to deal with *partial completions*

99% completion syndrome

Possible solutions:

- Control of products, not activities
- Subdivide into lots of sub-activities

Red/Amber/Green reporting

- Identify key tasks
- Break down into sub-tasks
- Assess subtasks as:
 - Green – ‘on target’
 - Amber – ‘not on target but recoverable’
 - Red – ‘not on target and recoverable only with difficulty’
- Status of ‘critical’ tasks is particularly important

Review

- Review of work products is an important mechanism for monitoring the progress of a project and ensuring the quality of the work products.
- Testing is an effective defect removal mechanism.
 - ◆ However, testing is applicable to only executable code.
 - ◆ Review is applicable to all work products.

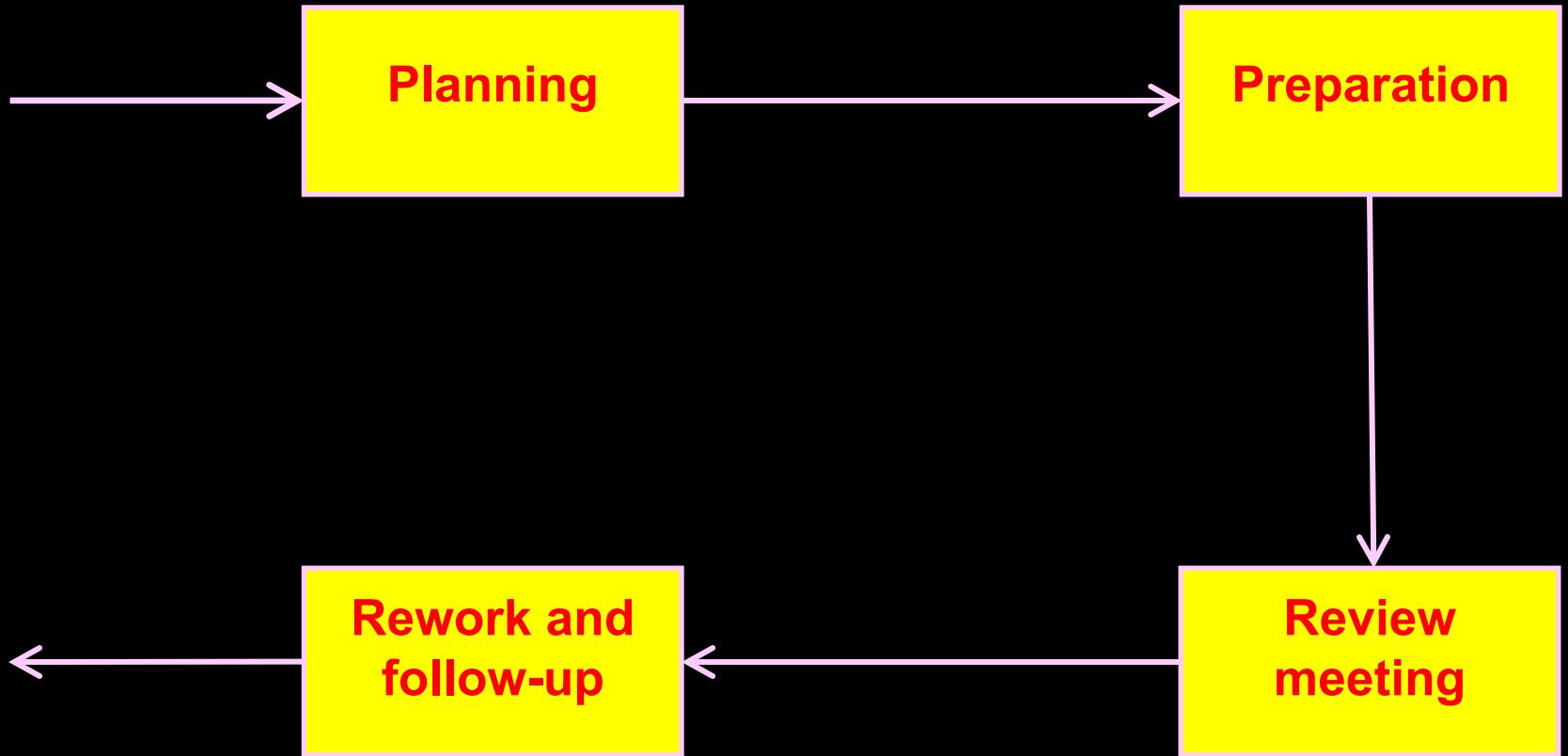
Utility of Review

- A cost-effective defect removal mechanism.
- Review usually helps to identify any deviation from standards.
- Reviewers suggest ways to improve the work product
- a review meeting often provides learning opportunities to not only the author of a work product, but also the other participants of the review meeting.
- The review participants gain a good understanding of the work product under review, making it easier for them to interface or use the work product in their work.

Review Roles

- Moderator:
 - ◆ Schedules and convenes meetings, distributes review materials, leads and moderates review sessions.
- Recorder:
 - ◆ Records the defects found and the time and effort data.
- Reviewers.

Review Process



Project Termination Review

- Project termination reviews provide important opportunities to learn from past mistakes as well as successes.
- Project termination need not necessarily mean project failure or premature abandonment.
 - ◆ A project may be terminated on successful completion

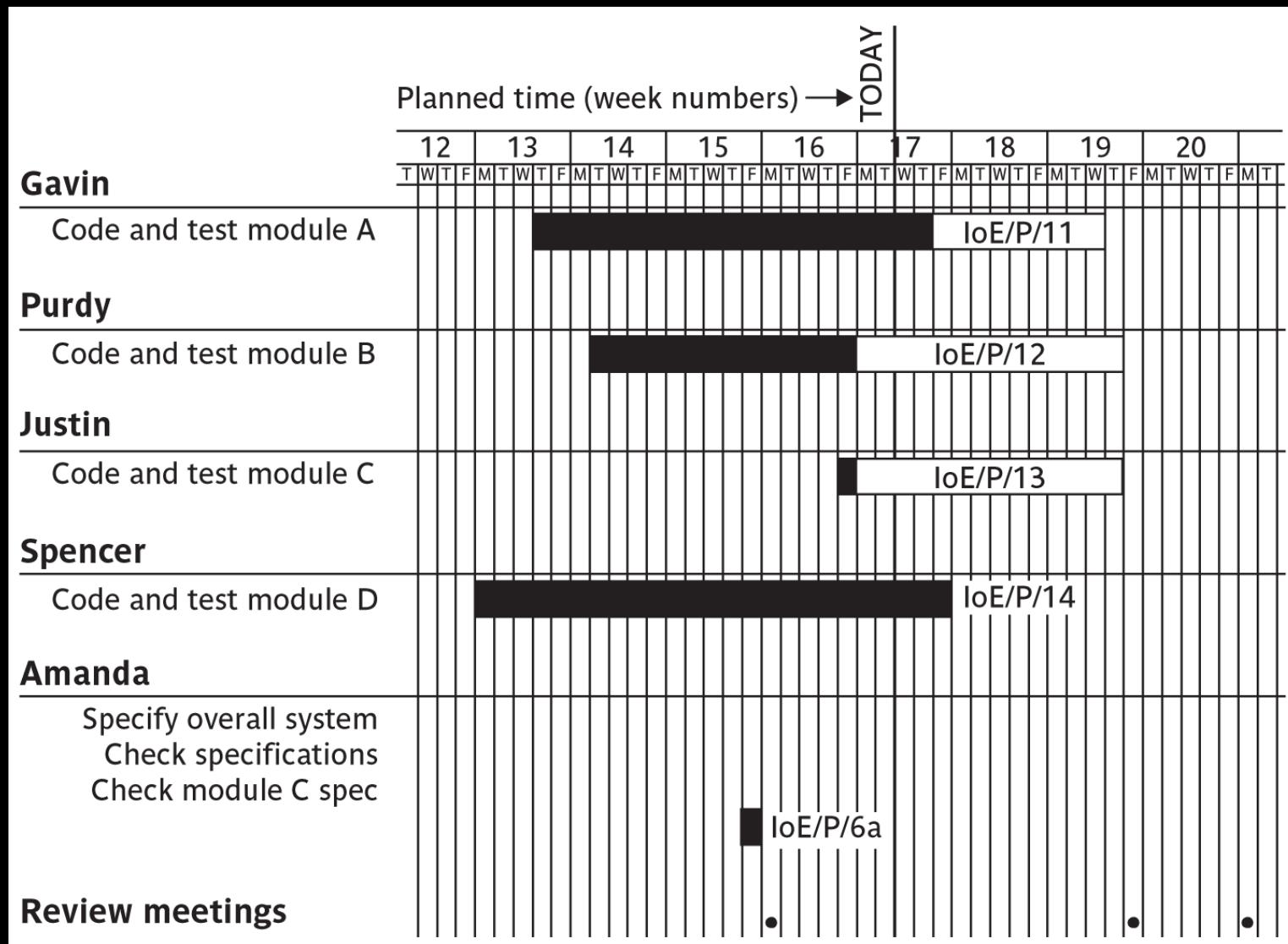
Reasons for Project Termination

- Project is completed successfully handed over to the customer.
- Incomplete requirements
- Lack of resources
- Some key technologies used in the project have become obsolete during project execution
- Economics of the project has changed, for example because many competing product may have become available in the market.

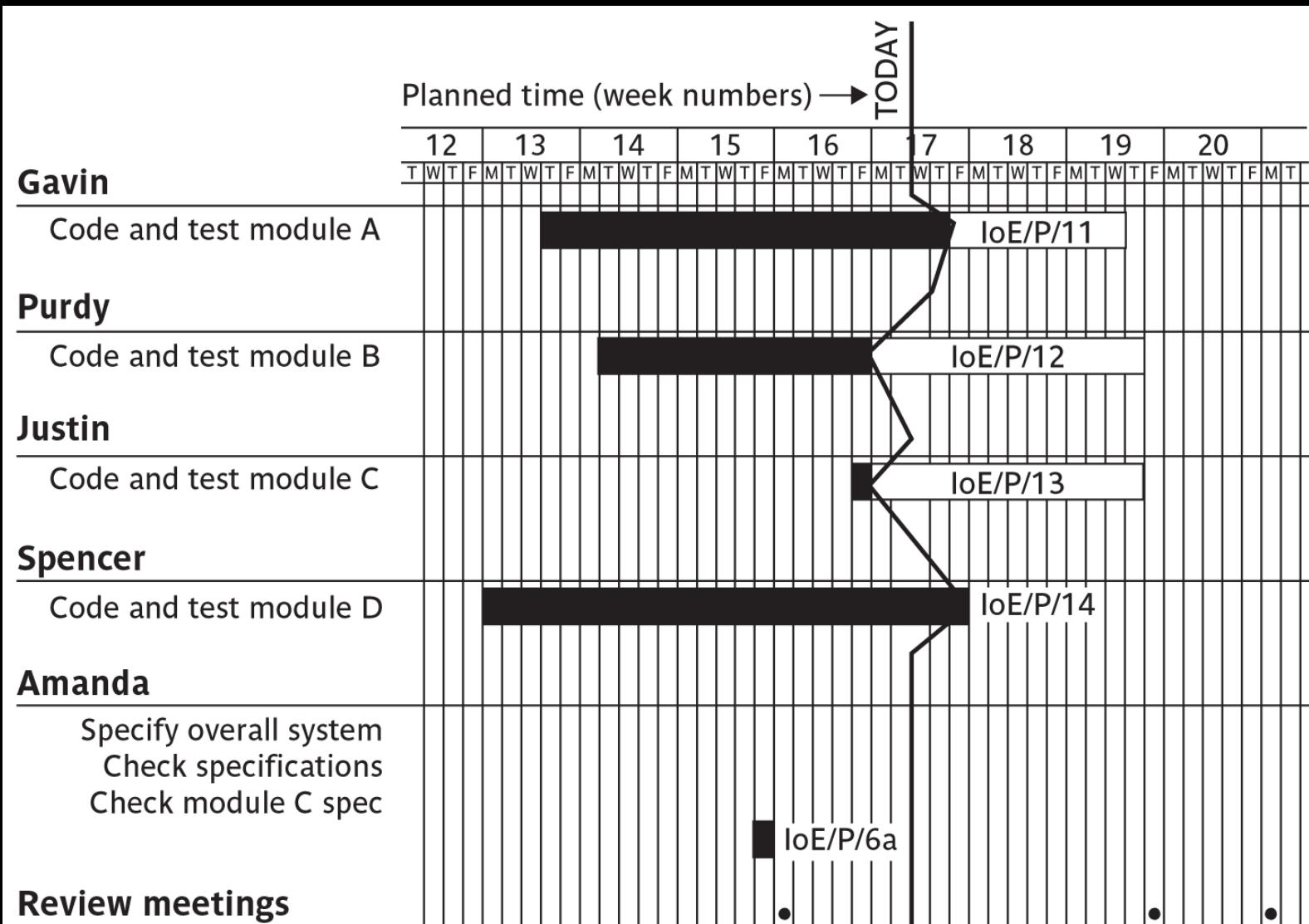
Project Termination Process

- Project survey
- Collection of objective information
- Debriefing meeting
- Final project review
- Result publication

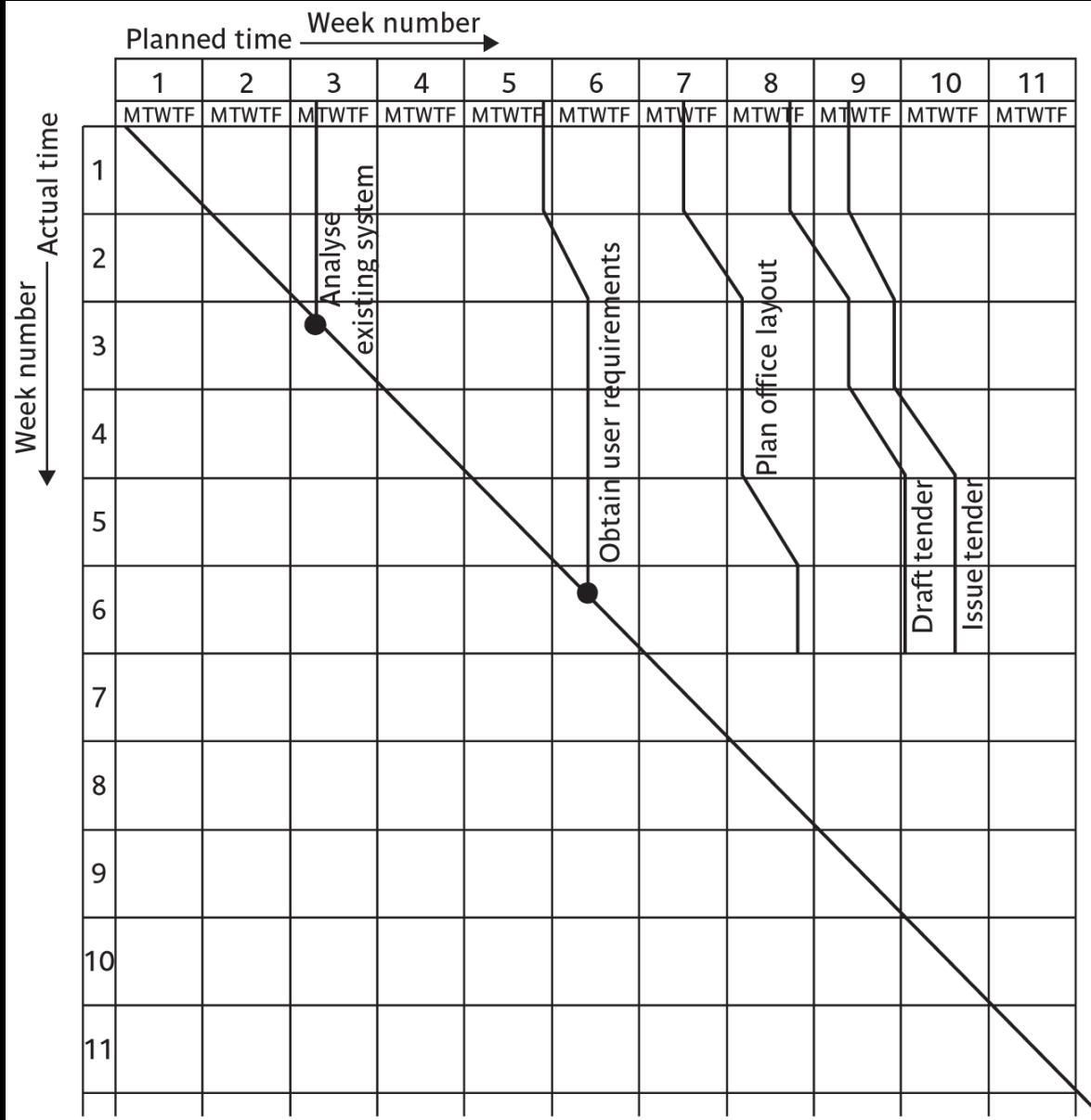
Gantt charts



Slip charts



The timeline



Cost monitoring

- A project could be late because the staff originally committed have not been deployed
- In this case the project will be *behind time* but *under budget*
- A project could be on time but only because additional resources have been added and so be *over budget*
- Need to monitor both achievements and costs

Earned value analysis

- *Planned value (PV)* or *Budgeted cost of work scheduled (BCWS)* – original estimate of the effort/cost to complete a task (compare with idea of a ‘price’)
- *Earned value (EV)* or *Budgeted cost of work performed (BCWP)* – total of PVs for the work completed at this time

Accounting conventions

- Work completed allocated on the basis
 - ◆ 50/50 half allocated at start, the other half on completion. These proportions can vary e.g. 0/100, 75/25 etc
 - ◆ *Milestone* current value depends on the milestones achieved
 - ◆ *Units processed*
- Can use money values, or staff effort as a surrogate

Earned value – an example

- Tasks
 - ◆ Specify module 5 days
 - ◆ Code module 8 days
 - ◆ Test module 6 days
- At the beginning of day 20, PV = 19 days
- If everything but testing not completed EV = 13 days
- Schedule variance = EV-PV i.e. $13-19 = -6$
- Schedule performance indicator (SPI) = $13/19 = 0.68$
- SV negative or SPI <1.00, project behind schedule

Earned value analysis – actual cost

- Actual cost (AC) is also known as Actual cost of work performed (ACWP)
- In previous example, if
 - ◆ ‘Specify module’ actually took 3 days
 - ◆ ‘Code module’ actually took 4 days
- Actual cost = 7 days
- Cost variance (CV) = EV-AC i.e. $13-7 = 6$ days
- Cost performance indicator = $13/7 = 1.86$
- Positive CV or CPI > 1.00 means project within budget

Earned value analysis – actual costs

- CPI can be used to produce new cost estimate
- Budget at completion (BAC) – current budget allocated to total costs of project
- Estimate at completion (EAC) – updated estimate = BAC/CPI
 - ◆ e.g. say budget at completion is £19,000 and CPI is 1.86
 - ◆ $EAC = BAC/CPI = £10,215$ (projected costs reduced because work being completed in less time)

Time variance

- Time variance (TV) – difference between time when specified EV should have been reached and time it actually was
- For example say an EV of £19000 was supposed to have been reached on 1st April and it was actually reached on 1st July then $TV = - 3$ months

Earned value chart with revised forecasts

Activity Assessment Sheet

Staff Justin

Ref: IoE/P/13

Activity: Code and test module C

Week number	13	14	15	16	17	18	
Activity summary	G	A	A	R			
Component							Comments
Screen handling procedures	G	A	A	G			
File update procedures	G	G	R	A			
Housekeeping procedures	G	G	G	A			
Compilation	G	G	G	R			
Test data runs	G	G	G	A			
Program documentation	G	G	A	R			

Prioritizing monitoring

We might focus more on monitoring certain types of activity
e.g.

- Critical path activities
- Activities with no free float – if delayed later dependent activities are delayed
- Activities with less than a specified float
- High risk activities
- Activities using critical resources

Getting back on track: options

- Renegotiate the deadline – if not possible then
- Try to shorten critical path e.g.
 - ◆ Work overtime
 - ◆ Re-allocate staff from less pressing work
 - ◆ Buy in more staff
- Reconsider activity dependencies
 - ◆ Over-lap the activities so that the start of one activity does not have to wait for completion of another
 - ◆ Split activities

Exception planning

- Some changes could affect
 - ◆ Users
 - ◆ The business case (e.g. costs increase reducing the potential profits of delivered software product)
- These changes could be to
 - ◆ Delivery date
 - ◆ Scope
 - ◆ Cost
- In these cases an **exception report** is needed

Exception planning - continued

- First stage
 - ◆ Write an **exception report** for sponsors (perhaps through project board)
 - Explaining problems
 - Setting out options for resolution
- Second stage
 - ◆ Sponsor selects an option (or identifies another option)
 - ◆ Project manager produces an **exception plan** implementing selected option
 - ◆ Exception plan is reviewed and accepted/rejected by sponsors/Project Board

Change control

The role of configuration librarian:

- Identifying items that need to be subject to change control
- Management of a central repository of the master copies of software and documentation
- Administering change procedures
- Maintenance of access records

Typical change control process

One or more users might perceive the need for a change

User management decide that the change is valid and worthwhile and pass it to development management

A developer is assigned to assess the practicality and cost of making the change

4. Development management report back to user management on the cost of the change; user management decide whether to go ahead

Change control process contd.

5. One or more developers are authorized to make copies of components to be modified
6. Copies modified. After initial testing, a test version might be released to users for acceptance testing
7. When users are satisfied then operational release authorized – master configuration items updated

Software Configuration Management (SCM)

- SCM is concerned with tracking and controlling changes to a software.
- Development and maintenance environment:
 - ◆ Various work products associated with the software continually change.
 - ◆ Unless a proper configuration management system is deployed, several problems can appear.

Why Use SCM?

- Problems associated with concurrent access
- Undoing Changes
- System accounting
- Handling variants
- Accurate determination project status
- Preventing unauthorized access to the work products

Configuration Control

- Two main operations:
 - ◆ Reserve
 - ◆ Restore

