Software Engineering Lecture-2

By: Usama Musharaf

What Is Project?

A project is an activity with specific goals which takes place over a finite period of time.

Examples:

- Developing a new product or service.
- Effecting a change in structure, staffing, or style of an organization.
- Designing a new transportation vehicle.
- Developing or acquiring a new or modified information system.
- Constructing a building or facility.
- Building a water system for a community in a developing country.
- Running a campaign for political office.
- Implementing a new business procedure or process.

What Is Management?

Basically, the 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 action to remedy hold-ups
- **Innovating**- coming up with new solutions
- **Representing**- liaising with users, etc.

What Is Project Management?

- Project Management is the art of maximizing the probability that a project delivers its goals on Time, to Budget and at the required Quality.
- Project management is the application of knowledge, skills, tools, and techniques to project activities to meet project requirements.

Goals of Project Management

- **Project management** is the discipline of defining and achieving a **set of goals** while optimizing the use of allocated resources (time, money, people, space, etc).

  This includes

  Planning,

  Scheduling

  Maintaining

  the progress of the activities that comprise the project.

Characteristics of Project

- Temporary
- Unique, Product or Service
- Aims/Tasks/Purpose
- Limited Time Scale

Why Project Fails?

- Management problems were more frequently dominant cause than technical problems

- Schedule overruns were more common (89%) than cost overruns (62%)

KPGM’s Survey in UK

What makes a Software Project success?

- User involvement – 20 points
- Executive Support – 15 points
- Clear Business Objectives – 15 points
- Experienced Project Manager – 15 points
- Small milestones – 10 points
- Firm basic requirements – 5 points
- Competent staff – 5 points
- Proper planning – 5 points
- Ownership – 5 points
- Others – 5 points

Chaos ten – Standish Group Report

Software Project Management

“Organizations that attempt to put software engineering discipline in place before putting project management discipline in place are domed to fail”

Software Engineering Institute

Four Project Dimensions (Four P’s)

Four Project Dimensions (Four P’s)

Four P's have a substantial influence on software project management- ***people***, ***product***, ***process***, and ***project***.

Software project management is an umbrella activity within software engineering.

It begins before any technical activity is initiated and continues throughout the definition, development, and support of computer software.

Four Project Dimensions (Four P’s)

People:

People must be organized into effective teams, motivated to do high-quality software work, and coordinated to achieve effective communication

Product:

The product requirements must be communicated from customer to developer, partitioned (decomposed) into their constituent parts, and positioned for work by the software team.

Four Project Dimensions (Four P’s)

Process:

The process must be adapted to the people and the problem. A common process framework is selected, an appropriate software engineering paradigm is applied, and a set of work tasks is chosen to get the job done.

Project:

The project must be organized in a manner that enables the software team to succeed.

(Four P’s)- People

1- Senior managers who define the business issues that often have significant influence on the project.

2- Project (technical) managers who must plan, motivate, organize, and control the practitioners who do software work.

3- Practitioners who deliver the technical skills that are necessary to engineer a product or application.

(Four P’s)- People

4- Customers who specify the requirements for the software to be engineered and other stakeholders who have a peripheral interest in the outcome.

5- End Users who interact with the software once it is released for production use.

(Four P’s)- Process

Software development is a social learning process. The process is a dialogue in which the knowledge that must become the software is brought together and embodied in the software. The process provides interaction between users and designers: between users and evolving tools, and between designers and evolving tools [technology] It is an iterative process in which the evolving tool itself serves as the medium for communication, with each new round of the dialogue eliciting more useful knowledge from the people involved.

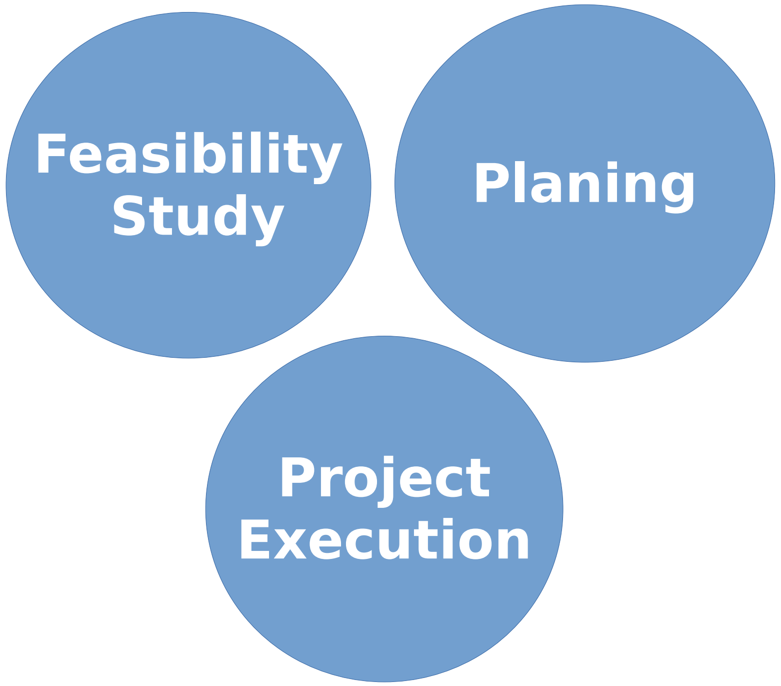
Howard Baetjer. Jr, comments on the software process

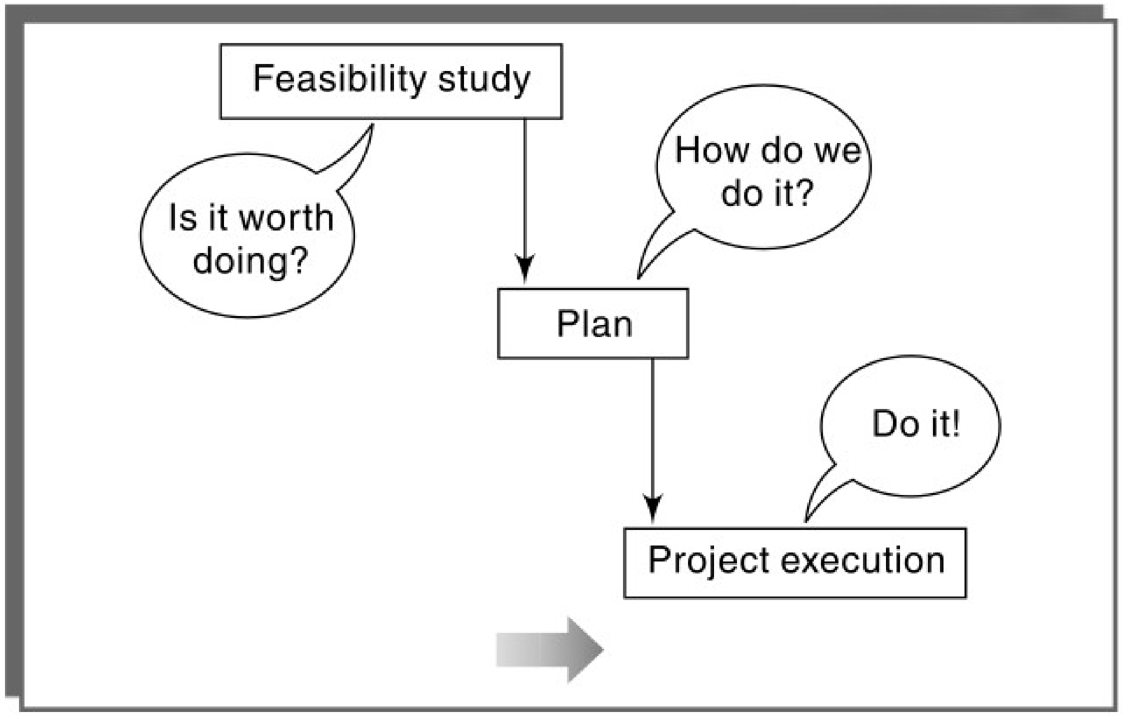
(Four P’s)- Process

When you build a product or system, it’s important to go through a series of predictable steps – a road map that helps you create a timely, high-quality result, The road map that you follow is called a ‘software process’

Activities Covered by Software Project Management

Activities Covered by Software Project Management





Contents List for a Project Plan

Contents List for a Project Plan

- 1. Introduction
- 2. Background
- 3. Project Objectives
- 4. Constraints
- 5. Methods
- 6. Project Products
- 7. Activities to be carried out
- 8. Resources to be used
- 9. Risks to the project
- 10. Management of the project, including
  - Organizational responsibilities
  - Management of quality
  - Configuration Management

Project Objectives

1- Define Project Scope

Objective:

Clearly outline what is included in the project and what is not.

Example:

For a project developing a mobile banking app, the scope might include features like account management, transaction history, and fund transfers, while excluding advanced analytics and third-party integrations.

Project Objectives

2- Set Clear Deliverables

Objective:

Identify the tangible outputs or products that will be delivered at various stages of the project.

Example:

Deliverables could include a requirements document, a working prototype, the final software application, and user documentation.

Project Objectives

3. Establish Timelines and Milestones

Objective:

Define the project schedule, including key milestones to track progress.

Example:

Milestones might include the completion of the design phase, the start of user acceptance testing, and the final release date.

Project Objectives

4. Allocate Resources

Objective:

Identify and assign the necessary resources, including team members, budget, tools, and equipment.

Example:

Assigning a team of five developers, two QA engineers, and a project manager, with a budget of $200,000 for a 6-month project.

Project Objectives

5. Risk Management

Objective:

Identify potential risks and develop strategies to mitigate them.

Example:

A risk could be the potential delay in obtaining third-party API access, with a mitigation plan that includes developing a backup solution using alternative APIs.

Project Objectives

6. Quality Assurance

Objective:

Define the quality standards and testing protocols to ensure the software meets the required specifications.

Example:

Implementing automated testing with coverage targets, and conducting regular code reviews and performance testing.

Project Objectives

7. Communication Plan

Objective:

Establish a communication strategy to ensure all stakeholders are informed and engaged throughout the project.

Example:

Weekly status meetings, monthly progress reports, and a dedicated project collaboration platform like Jira or Slack

Project Objectives

8. Budget Management

Objective:

Plan and monitor the project's financial resources to ensure it stays within the allocated budget.

Example:

Allocating $50,000 for development, $20,000 for testing, and $30,000 for contingency.

Project Objectives

9. Client and Stakeholder Expectations

Objective:

Ensure that the client and other stakeholders have a clear understanding of what will be delivered, including any limitations.

Example:

Setting expectations that the first version of the software will focus on core features, with additional features planned for future updates.

Project Objectives

10. Define Success Criteria

Objective:

Establish the criteria by which the project's success will be measured.

Example:

Success criteria could include completing the project on time and within budget, meeting all functional requirements, and achieving a user satisfaction score of 90% or higher in post-launch surveys.

Project Sub-Objectives

Measures of Effectiveness

Measures of Effectiveness (MoE) in software project management are metrics used to evaluate how well a software project is achieving its objectives and meeting stakeholder expectations.

These measures help determine whether the project is on track, whether it meets quality standards, and whether it delivers the expected value.

MoE Examples

Schedule Adherence

Tracks whether the project is being completed on time according to the planned schedule.

Metric Examples:

Percentage of milestones met on time.

Number of days ahead or behind schedule.

MoE Examples

Budget Adherence

Measures how well the project is staying within its allocated budget.

Metric Examples:

Percentage of budget spent vs. planned budget.

Cost variance (difference between planned and actual costs).

MoE Examples

Quality of Deliverables

Assesses the quality of the software deliverables against predefined standards.

Metric Examples:

Number of defects found during testing.

Defect density (defects per lines of code or function points).

Percentage of test cases passed.

MoE Examples

Team Productivity and Performance

Measures how effectively the project team is working and their output quality.

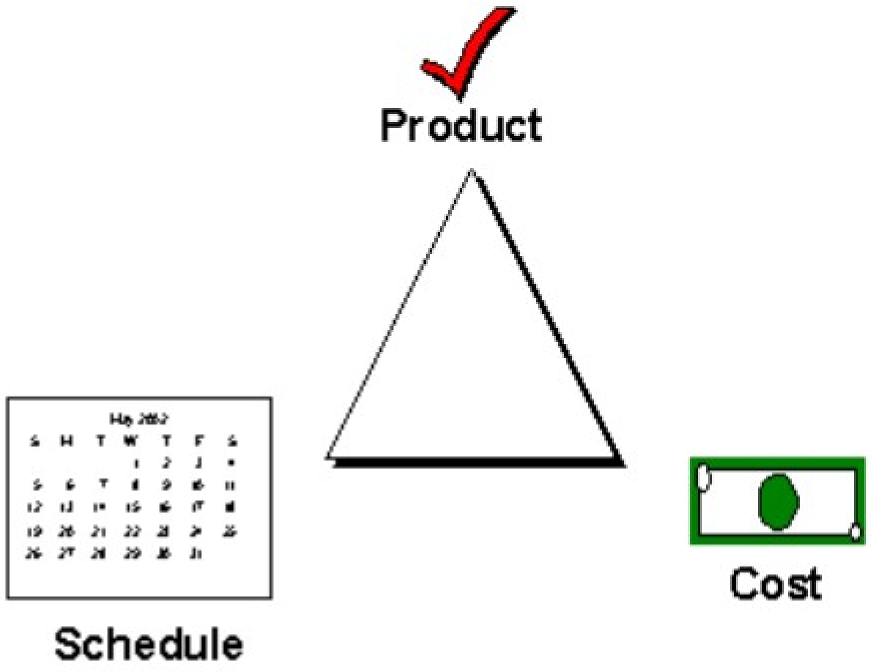
Metric Examples:

Velocity (in Agile projects, e.g., story points completed per sprint).

Number of tasks completed per team member.

Trade-off Triangle

Fast, Cheap, Good. Choose two.



Q & A

What is project success and failure according to you?

Evaluation of Projects

Technical Assesment

Technical assessment of a proposed system consists of evaluating whether the required functionality can be achieved with current affordable technologies.

Organizational policy, aimed at providing a consistent hardware/software infrastructure, is likely to limit the technical solutions considered.

The costs of the technology adopted must be taken into account in the cost–benefit analysis.

Cost Benefit Analysis

Even where the estimated benefits will exceed the estimated costs, it is often necessary to decide if the proposed project is the best of several options.

Not all projects can be undertaken at any one time and, in any case, the most valuable projects should get most resources.

Cost Benefit Analysis

Cost–benefit analysis comprises two steps:

1- Identifying all of the costs and benefits of carrying out the project and operating the delivered application.

These include

- the development costs,
- the operating costs,
- benefits expected from the new system.

Cost Benefit Analysis

2- Expressing these costs and benefits in common units. We must express each cost and benefit – and the net benefit which is the difference between the two – in money.

The proposed system is a replacement, these estimates should reflect the change in costs and benefits due to the new system.

A new sales order processing system, for example, could only claim to benefit an organization by the increase in sales due to the use of the new system.

Cost Benefit Analysis

Most direct costs are easy to quantify in monetary terms and can be categorized as:

- **Development costs**, including development staff costs;
- **Setup costs,** consisting of the costs of putting the system into place, mainly of any new hardware but also including the costs of file conversion, recruitment and staff training;
- **Operational costs** relating to operating the system after installation.

Cost Benefit Analysis

Example:

XYZ university is considering the replacement of the existing payroll service, operated by a third party, with a tailored, off-the-shelf computer-based system.

List some of the costs it might consider under the headings of:

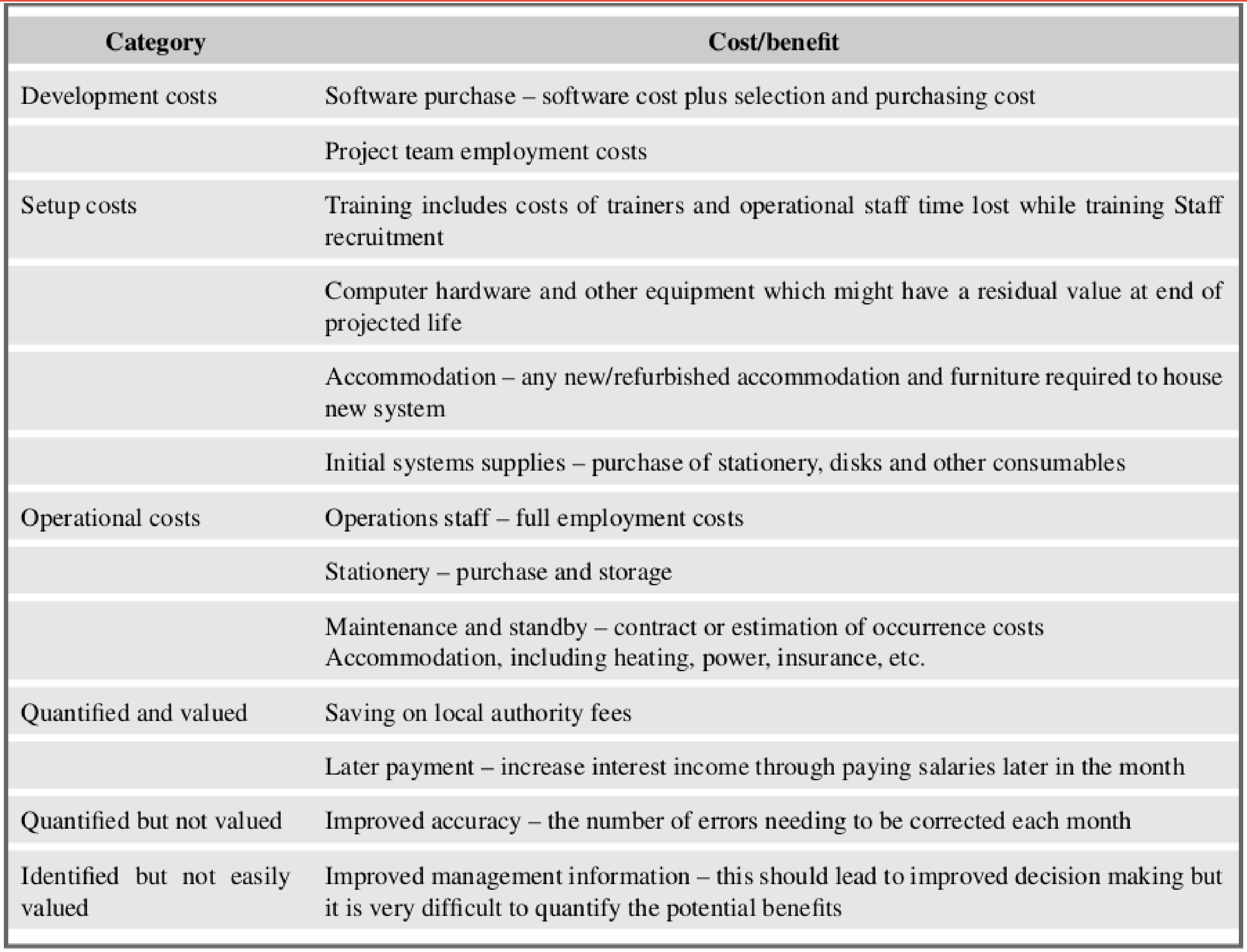
- Development costs
- Setup costs
- Operational costs

List some of the benefits under the headings:

- Quantified and valued benefits
- Quantified but not valued
- Identified but not easily valued

For each cost or benefit, explain how, in principle, it might be measured in monetary terms.

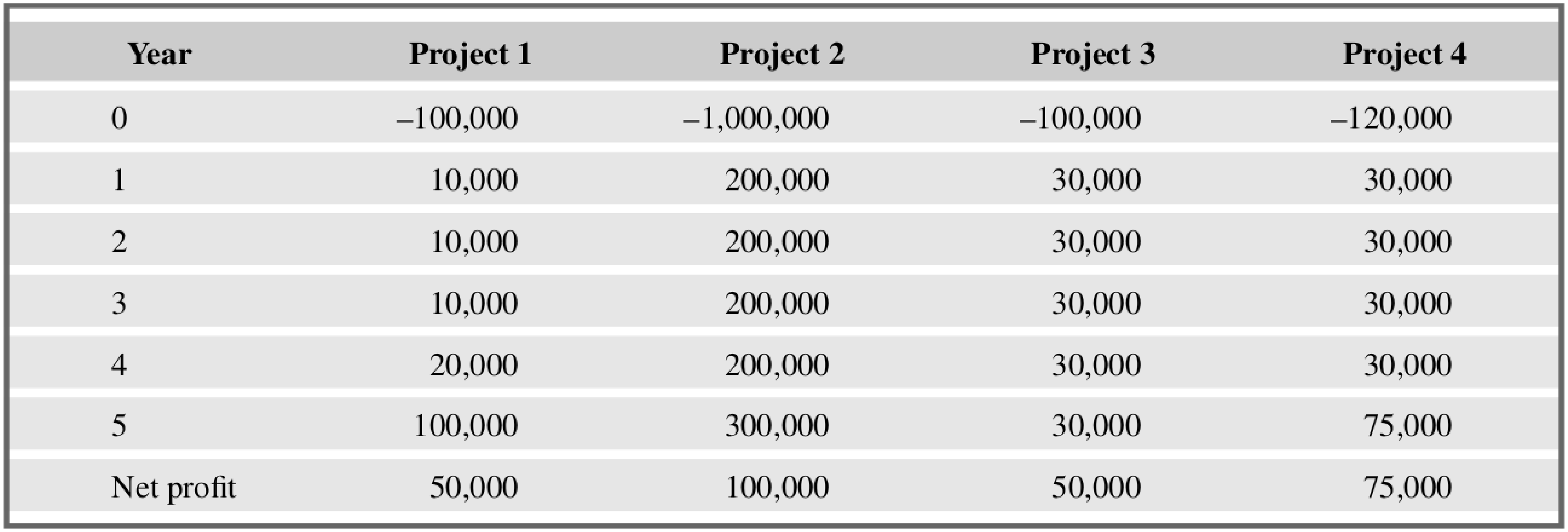
Cost Benefit Analysis



Category	Cost/benefit
Development costs	Software purchase – software cost plus selection and purchasing cost
	Project team employment costs
Setup costs	Training includes costs of trainers and operational staff time lost while training Staff recruitment
	Computer hardware and other equipment which might have a residual value at end of projected life
	Accommodation – any new/refurbished accommodation and furniture required to house new system
	Initial systems supplies – purchase of stationery, disks and other consumables
Operational costs	Operations staff – full employment costs
	Stationery – purchase and storage
	Maintenance and standby – contract or estimation of occurrence costs Accommodation, including heating, power, insurance, etc.
Quantified and valued	Saving on local authority fees
	Later payment – increase interest income through paying salaries later in the month
Quantified but not valued	Improved accuracy – the number of errors needing to be corrected each month
Identified but not easily valued	Improved management information – this should lead to improved decision making but it is very difficult to quantify the potential benefits

Cost–benefit Evaluation Techniques

We now take a look at some methods for comparing projects on the basis of their cash flow forecasts.

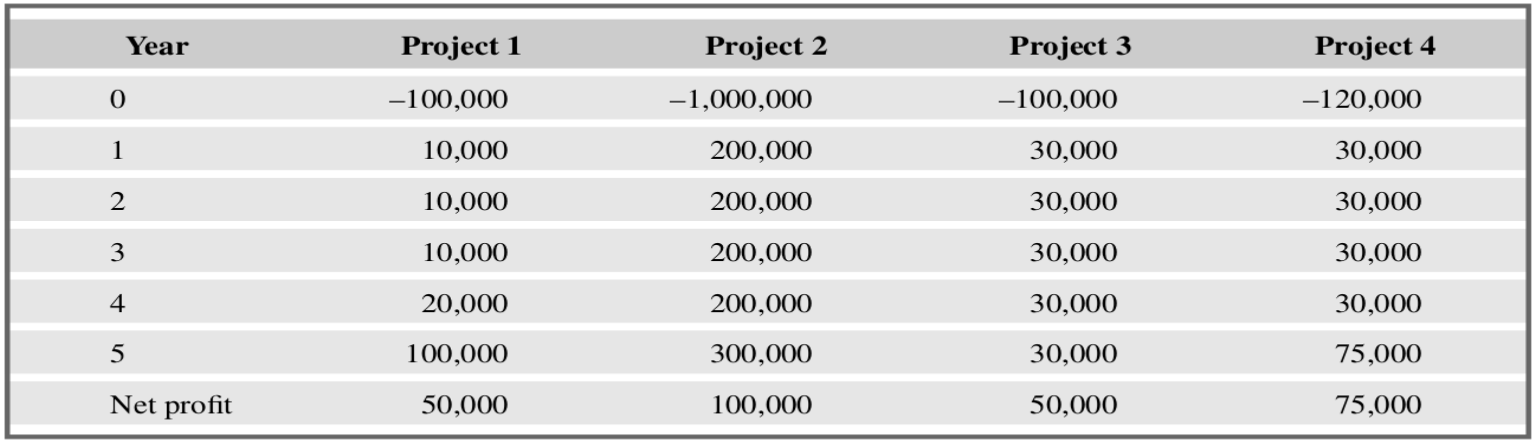


Year	Project 1	Project 2	Project 3	Project 4
0	–100,000	–1,000,000	–100,000	–120,000
1	10,000	200,000	30,000	30,000
2	10,000	200,000	30,000	30,000
3	10,000	200,000	30,000	30,000
4	20,000	200,000	30,000	30,000
5	100,000	300,000	30,000	75,000
Net profit	50,000	100,000	50,000	75,000

Cost–benefit Evaluation Techniques

Observation:

- Projects 1 and 3 each have a net profit of 50,000 and therefore, according to this selection criterion, would be equally preferable.
- The bulk of the income occurs late in the life of project 1, whereas project 3 returns a steady income throughout its life.
- Having to wait for a return has the disadvantage that the investment must be funded for longer.
- Estimates in the more distant future are less reliable than short-term estimates and we can see that the two projects are not equally preferable.

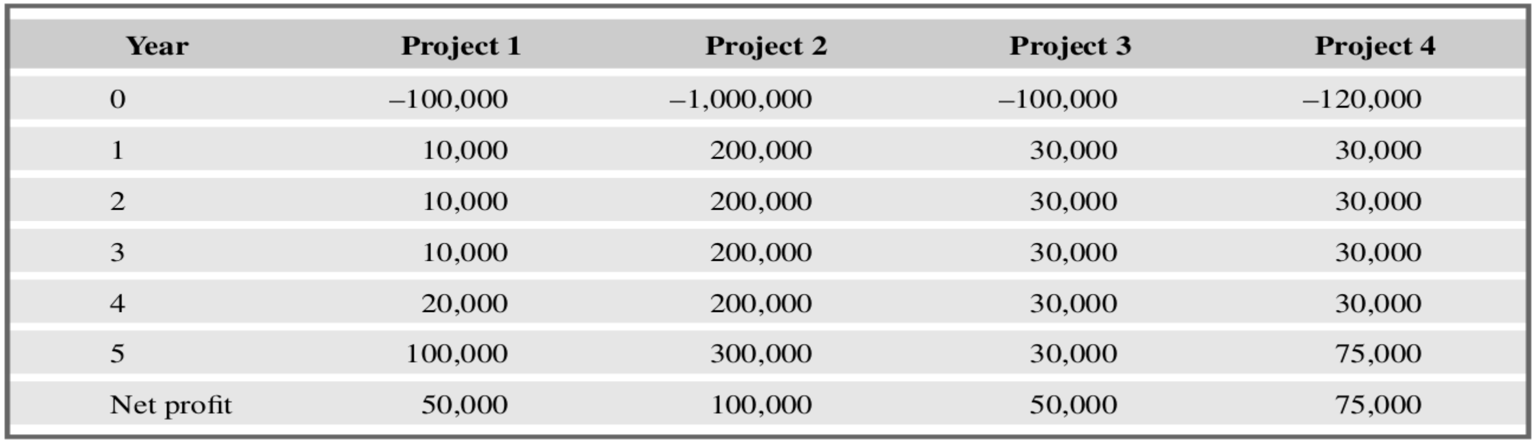


Year	Project 1	Project 2	Project 3	Project 4
0	–100,000	–1,000,000	–100,000	–120,000
1	10,000	200,000	30,000	30,000
2	10,000	200,000	30,000	30,000
3	10,000	200,000	30,000	30,000
4	20,000	200,000	30,000	30,000
5	100,000	300,000	30,000	75,000
Net profit	50,000	100,000	50,000	75,000

Cost–benefit Evaluation Techniques

Payback Period

- The payback period is the time taken to pay back the initial investment.
- Normally, the project with the shortest payback period will be chosen on the basis that an organization will wish to minimize the time that a project is 'in debt'
- The ***advantage*** of the payback period is that it is simple to calculate.
- Its ***disadvantage*** as a selection technique is that it ignores the overall profitability of the project – in fact, it totally ignores any income (or expenditure) once the project has broken even. Thus the fact that projects 2 and 4 are, overall, more profitable than project 3 is ignored.



Year	Project 1	Project 2	Project 3	Project 4
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

Cost–benefit Evaluation Techniques

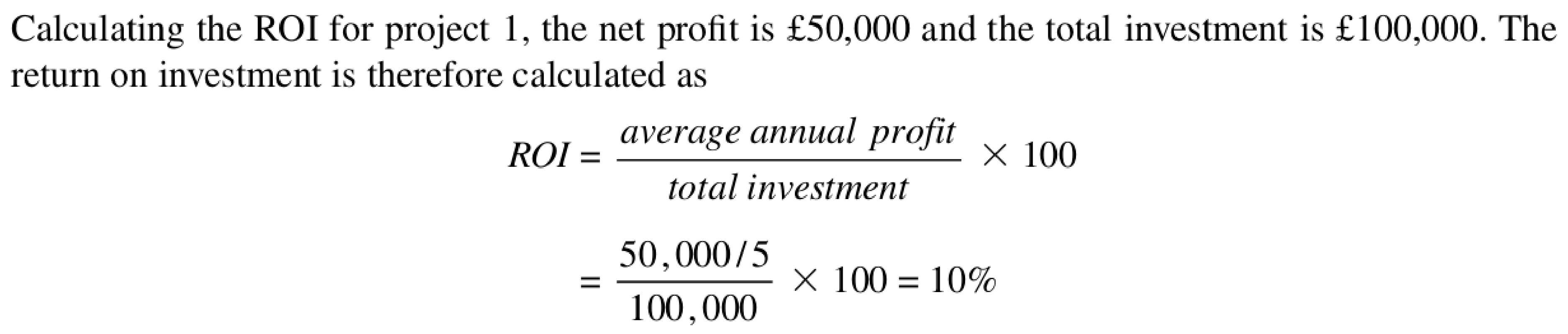
Return on Investment:

- The return on investment (ROI), also known as the accounting rate of return (ARR), provides a way of comparing the net profitability to the investment required.

$$\text{ROI} = \frac{\textit{average annual profit}}{\textit{total investment}} \times 100$$

Cost–benefit Evaluation Techniques

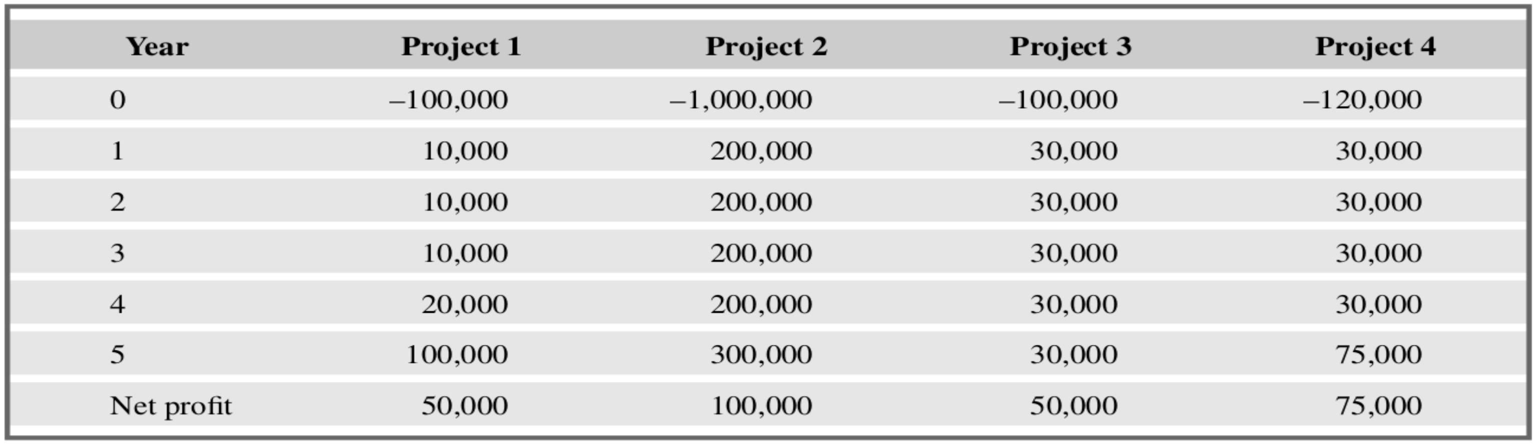
Return on Investment:



Calculating the ROI for project 1, the net profit is £50,000 and the total investment is £100,000. The return on investment is therefore calculated as

$$ROI = \frac{\textit{average annual profit}}{\textit{total investment}} \times 100$$

$$= \frac{50,000/5}{100,000} \times 100 = 10\%$$



Year	Project 1	Project 2	Project 3	Project 4
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

Cost–benefit Evaluation Techniques

Net Present Value:

- The calculation of net present value is a project evaluation technique that takes into account the profitability of a project and the timing of the cash flows that are produced.

- This is based on the view that receiving £100 today is better than having to wait until next year to receive it.

- The present value of £100 in a year’s time is £91, we mean that £100 in a year’s time is the equivalent of £91 now.

Cost–benefit Evaluation Techniques

Net Present Value:

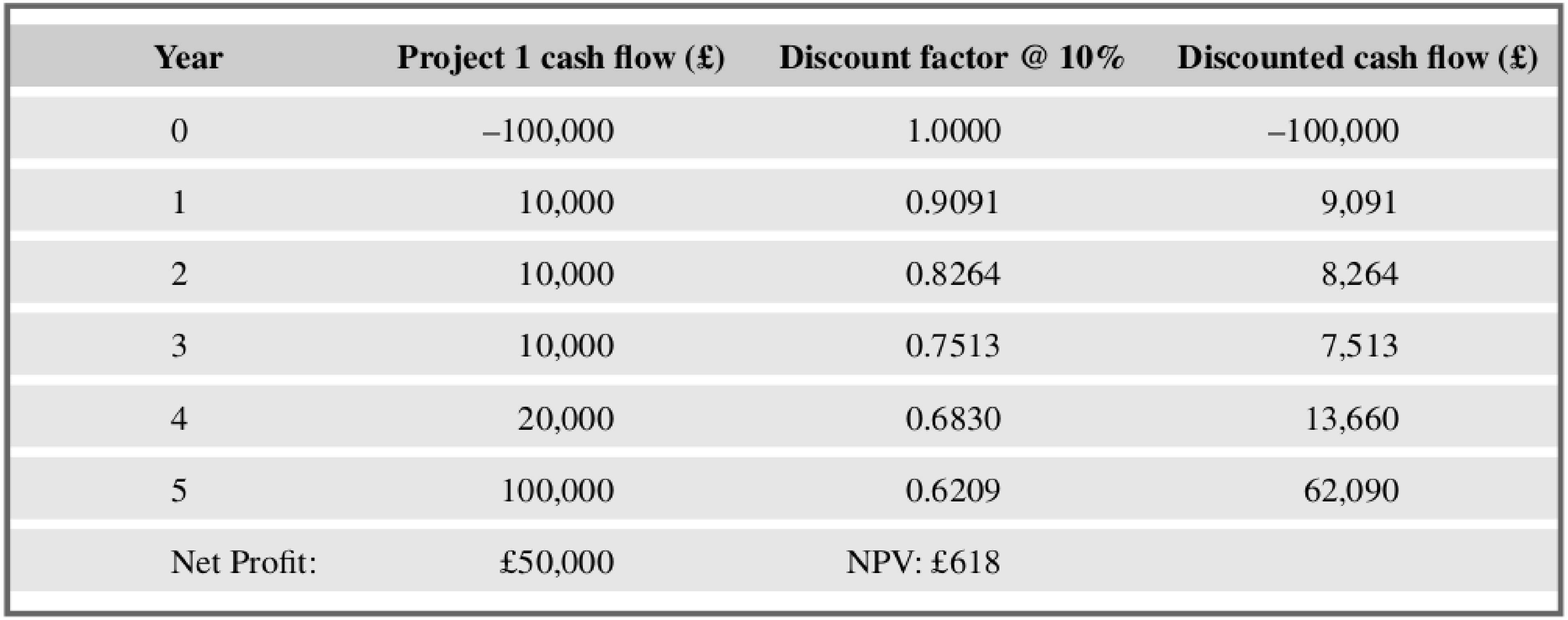
The present value of any future cash flow may be obtained by applying the following formula.

$$\text{Present value} = \frac{\text{value in year } t}{(1+r)^t}$$

where ***r*** is the discount rate, expressed as a decimal value, and ***t*** is the number of years into the future that the cash flow occurs.

Cost–benefit Evaluation Techniques

Example



Year	Project 1 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
Net Profit:	£50,000	NPV: £618	

Cost–benefit Evaluation Techniques

Internal rate of return:

One disadvantage of NPV as a measure of profitability is that, although it may be used to compare projects, it might not be directly comparable with earnings from other investments or the costs of borrowing capital. Such costs are usually quoted as a percentage interest rate.

The internal rate of return (IRR) attempts to provide a profitability measure as a percentage return that is directly comparable with interest rates.

A project that showed an estimated IRR of 10% would be worthwhile if the capital could be borrowed for less than 10%.

Cost–benefit Evaluation Techniques

Internal rate of return:

IRR is the actual rate of return from an investing activities.

The IRR is calculated as that percentage discount rate that would produce an NPV of zero.

If IRR > Cost of Capital, project is worthwhile

If IRR = Cost of Capital, project is neutral

If IRR < Cost of Capital, project is not good to invest at.

Example:

IRR is 15% > interest rate of 10% on loan from bank.

Cost–benefit Evaluation Techniques

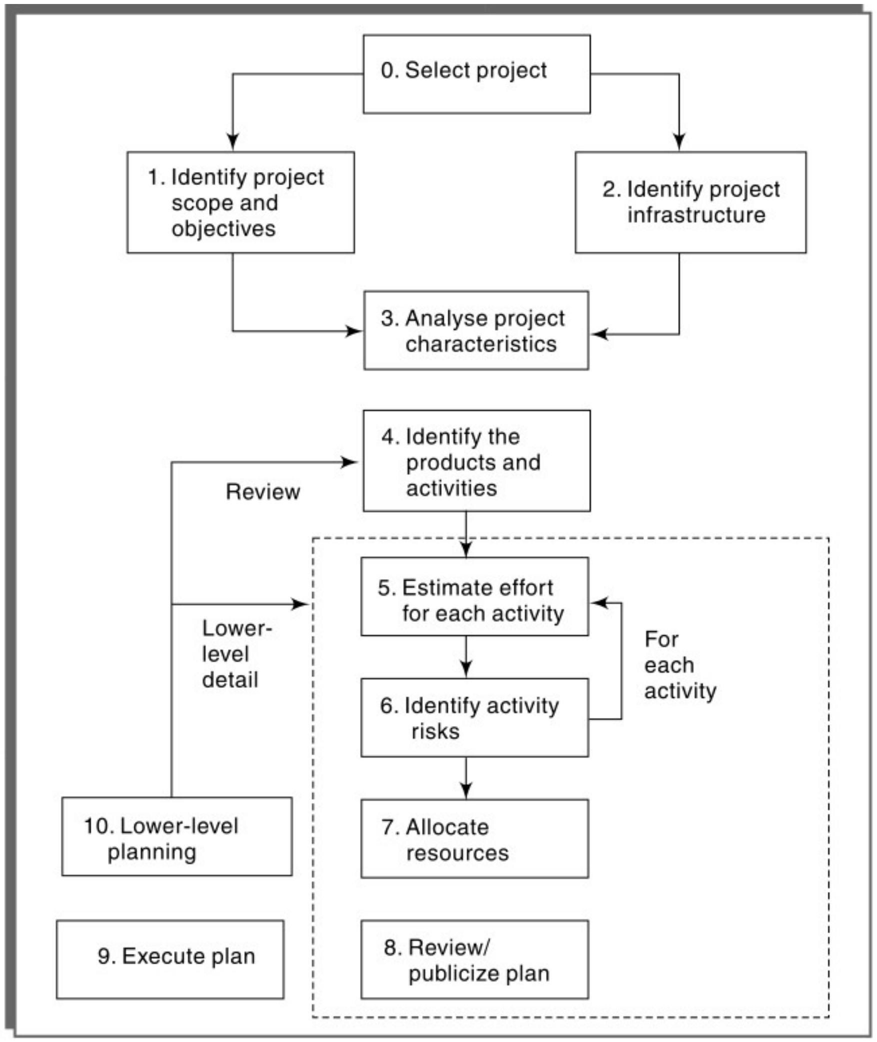
Year	Value
0	-10000
1	12000

NPV = Future value $_{(0)}$ / $(1+r)^0$ + Future value $_{(1)}$ / $(1+r)^1$.

0= Future value $_{(0)}$ / $(1+r)^0$ + Future value $_{(1)}$ / $(1+r)^1$.

Step-Wise Project Planing Framework

Step-Wise Project Planing Framework


0. Select project
1. Identify project scope and objectives
2. Identify project infrastructure
3. Analyse project characteristics
4. Identify the products and activities
Review
5. Estimate effort for each activity
Lower-level detail
For each activity
6. Identify activity risks
7. Allocate resources
10. Lower-level planning
9. Execute plan
8. Review/ publicize plan

An overview of Step Wise

Software Effort Estimation

Software Effort Estimation

Software effort estimation involves predicting the amount of effort (usually in person-hours, person-days, or person-months) required to complete a software development task or project.

It is crucial for project planning, resource allocation, and cost estimation.

Software Effort Estimation Techniques

Albrecht Function Point Analysis (FPA)

Albrecht Function Point Analysis (FPA)

Albrecht Function Point Analysis (FPA) is a structured technique for measuring the size and complexity of software systems based on their functional requirements.

It was introduced by Allan Albrecht in the late 1970s at IBM as a way to estimate the effort and resources needed for software development.

Instead of focusing on the lines of code, FPA evaluates the software based on the functionality it provides to its users.

Albrecht Function Point Analysis (FPA)

The fundamental idea behind **Function Point Analysis (FPA)** is that software is a collection of functions (inputs, outputs, inquiries, etc.), and the effort required to develop the software can be measured based on the complexity of these functions.

FPA helps in determining the size of the software in terms of function points (FPs), which can then be used to estimate project effort, cost, and duration.

Albrecht Function Point Analysis (FPA)

Components of FPA:

1. External Inputs (EI):
2. External Outputs (EO):
3. External Inquiries (EQ)
4. Internal Logical Files (ILF)
5. External Interface Files (EIF)

Albrecht Function Point Analysis (FPA)

1- External Inputs (EI):

These represent inputs into the system from external sources.

Examples include data entered by users, transactions submitted by other systems, and forms submitted via web interfaces.

Example:

Entering a customer’s details into a CRM system.

Albrecht Function Point Analysis (FPA)

2- External Outputs (EO):

These are outputs produced by the system that leave the system's boundary, such as reports, confirmations, and calculated results.

Example:

Generating a bank statement or transaction receipt.

Albrecht Function Point Analysis (FPA)

3- External Inquiries (EQ):

These are queries or retrievals of information from the system without any modifications to the data. It includes user-generated queries or lookups.

Example:

Retrieving a customer’s account balance.

Albrecht Function Point Analysis (FPA)

4- Internal Logic Files (ILF):

These are the logical data stores or databases maintained by the system. ILFs represent the internal data storage where the system maintains information.

Example:

A customer database or transaction history.

Albrecht Function Point Analysis (FPA)

5- External interface Files (EIF):

These are data stores that the system accesses but does not maintain, often belonging to other systems. An EIF is used to retrieve or query data from another system.

Example:

A bank’s CRM system accessing a credit bureau’s database for customer credit scores.

Albrecht Function Point Analysis (FPA)

Steps for FPA:

1. Count the Number of Each Function Type
2. Assign Complexity to Each Function
3. Apply the Weighting Factor
4. Calculate Unadjusted Function Points (UFP)
5. Adjust for Complexity (Value Adjustment Factor)
6. Calculate Final Adjusted Function Points

Albrecht Function Point Analysis (FPA)

Step-1. Count the Number of Each Function Type

For each of the five function types, count how many instances exist in the system.

For example, how many external inputs, external outputs, inquiries, internal logical files, and external interface files are present in the system.

Albrecht Function Point Analysis (FPA)

Step-2. Assign Complexity to Each Function

Each function type (EI, EO, EQ, ILF, EIF) is classified as Simple, Average, or Complex based on the number of data elements and the logical relationships involved.

This classification helps in determining the weighting factor for each function.

The classification is based on the number of **Data Element Types (DETs)** and **File Types Referenced (FTRs)** for inputs, outputs, and inquiries, or the **number of Record Element Types (RETs)** and **Data Element Types (DETs)** for files (ILFs and EIFs).

Albrecht Function Point Analysis (FPA)

Data Element Types (DETs): These are unique, user-recognizable fields (e.g., attributes like Customer Name, Customer ID, Address).

File Types Referenced (FTRs): Logical files accessed by a transaction (either ILFs or EIFs).

Record Element Types (RETs): Subgroups of data within an ILF or EIF (e.g., segments in a file like Customer Data and Transaction Data within a customer database).

Albrecht Function Point Analysis (FPA)

Guidelines for Assigning Complexity

1. External Inputs (EI)

An External Input (EI) is any input that updates data in the internal system (e.g., entering customer information, submitting a form).

Simple: Less than or equal to 4 DETs and 0 or 1 FTR.

Average: 5 to 15 DETs and 2 FTRs.

Complex: More than 15 DETs or 3 or more FTRs.

Example:

Simple: A form with three fields: Customer ID, Name, Address (3 DETs, 1 FTR - Customer Database).

Complex: A form with twenty fields that also checks three different files: Customer ID, Order History, Account Balance (20 DETs, 3 FTRs - Customer, Order, Account).

Albrecht Function Point Analysis (FPA)

2. External Outputs (EO)

An External Output (EO) represents the system-generated data provided to the user or another system (e.g., a report, invoice, or confirmation).

Simple: Less than or equal to 5 DETs and 0 or 1 FTR.

Average: 6 to 19 DETs and 2 or 3 FTRs.

Complex: More than 19 DETs or more than 3 FTRs.

Example:

Simple: A receipt with four fields: Transaction ID, Amount, Date, Customer Name (4 DETs, 1 FTR).

Complex: A detailed financial report pulling data from four different files and containing 25 fields (25 DETs, 4 FTRs).

Albrecht Function Point Analysis (FPA)

3. External Inquiries (EQ)

An External Inquiry (EQ) is a request to retrieve data without modifying it (e.g., viewing customer information, searching for a transaction).

Simple: Less than or equal to 5 DETs and 0 or 1 FTR.

Average: 6 to 19 DETs and 2 or 3 FTRs.

Complex: More than 19 DETs or more than 3 FTRs.

Example:

Simple: A search by Customer ID returning Name and Address (3 DETs, 1 FTR).

Complex: A query that retrieves customer history along with associated transactions, using multiple files (20 DETs, 3 FTRs).

Albrecht Function Point Analysis (FPA)

4. Internal Logical Files (ILF)

An Internal Logical File (ILF) is a logical grouping of data that the system maintains (e.g., customer database, transaction logs).

Simple: Less than or equal to 19 DETs and 1 RET.

Average: 20 to 50 DETs or 2 to 5 RETs.

Complex: More than 50 DETs or more than 5 RETs.

Example:

Simple: A customer database with one record type, storing basic data like ID, Name, Address (10 DETs, 1 RET).

Complex: A detailed customer file that stores several subgroups of information (contact details, transaction history, account preferences), having 6 subgroups and 100 fields (6 RETs, 100 DETs).

Albrecht Function Point Analysis (FPA)

5. External Interface Files (EIF)

An External Interface File (EIF) is a logical grouping of data maintained by an external system but accessed by the current system (e.g., a product catalog from another system).

Simple: Less than or equal to 19 DETs and 1 RET.

Average: 20 to 50 DETs or 2 to 5 RETs.

Complex: More than 50 DETs or more than 5 RETs.

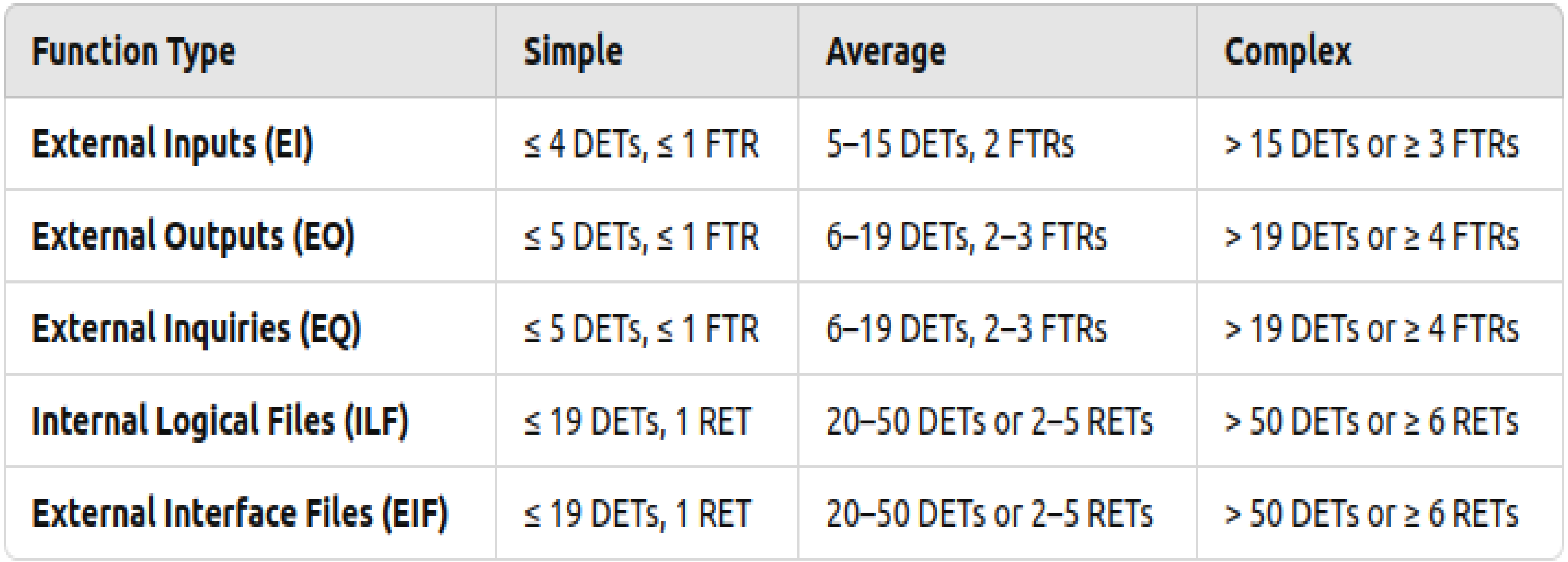
Example:

Simple: A reference to an external pricing file with a small number of fields and one logical grouping.

Complex: A detailed external product catalog with multiple groupings (product categories, price lists, stock levels) and hundreds of fields.

Albrecht Function Point Analysis (FPA)

Summary of Complexity Classification:

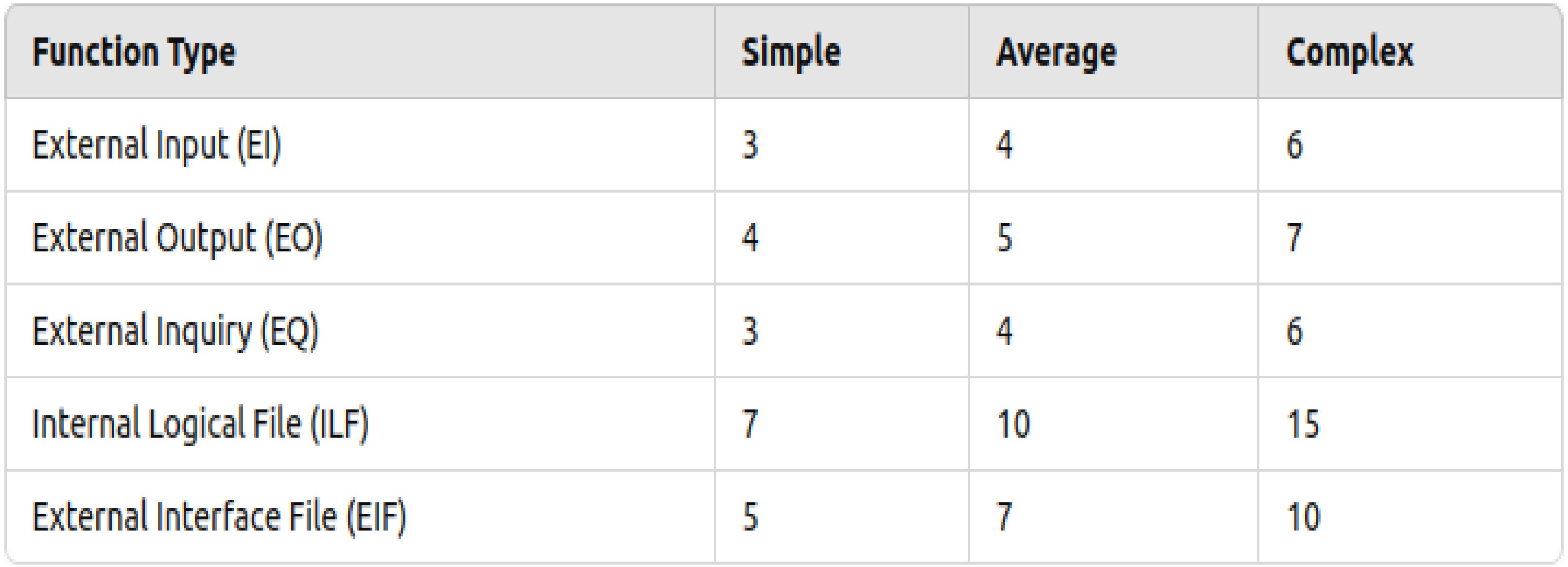


Function Type	Simple	Average	Complex
External Inputs (EI)	≤ 4 DETs, ≤ 1 FTR	5–15 DETs, 2 FTRs	> 15 DETs or ≥ 3 FTRs
External Outputs (EO)	≤ 5 DETs, ≤ 1 FTR	6–19 DETs, 2–3 FTRs	> 19 DETs or ≥ 4 FTRs
External Inquiries (EQ)	≤ 5 DETs, ≤ 1 FTR	6–19 DETs, 2–3 FTRs	> 19 DETs or ≥ 4 FTRs
Internal Logical Files (ILF)	≤ 19 DETs, 1 RET	20–50 DETs or 2–5 RETs	> 50 DETs or ≥ 6 RETs
External Interface Files (EIF)	≤ 19 DETs, 1 RET	20–50 DETs or 2–5 RETs	> 50 DETs or ≥ 6 RETs

Albrecht Function Point Analysis (FPA)

Step-3. Apply the Weighting Factor

Each function type has a specific weighting factor based on its complexity. Albrecht provided a table of standard weights to be used for each function type. The weights are shown below:



Function Type	Simple	Average	Complex
External Input (EI)	3	4	6
External Output (EO)	4	5	7
External Inquiry (EQ)	3	4	6
Internal Logical File (ILF)	7	10	15
External Interface File (EIF)	5	7	10

Albrecht Function Point Analysis (FPA)

Step-4. Calculate Unadjusted Function Points (UFP)

The Unadjusted Function Points (UFP) are calculated by multiplying the number of instances of each function type by its weighting factor and summing up the results.

$$\text{UFP}=\sum(\text{Number of Instances}\times\text{Weighting Factor})$$

For example, if a system has:

5 Simple External Inputs,, 3 Average External Outputs,, 2 Complex External Inquiries,

2 Average Internal Logical Files, 1 Complex External Interface File,

The calculation would be:

$$\text{UFP}=(5\times3)+(3\times5)+(2\times6)+(2\times10)+(1\times10)$$

$$\text{UFP}=15+15+12+20+10=72$$

Albrecht Function Point Analysis (FPA)

5. Adjust for Complexity (Value Adjustment Factor)

Once the unadjusted function points are calculated, they are adjusted for complexity based on 14 general system characteristics (GSCs). These characteristics account for the system's performance, usability, maintainability, and other environmental factors that affect development effort. The GSCs include factors like:

- Data communications
- Distributed Data Processing
- Performance considerations
- Heavily used configuration
- Transaction rate
- Online data entry
- End-user efficiency
- Online updates
- Complex processing
- Reusability
- Installation ease
- Operational ease
- Multiple sites
- Facilitate change

Each characteristic is rated on a scale from 0 to 5 (0 means no influence, 5 means strong influence), and the **Value Adjustment Factor (VAF)** is calculated using the formula:

$$\text{VAF} = 0.65 + (0.01 \times \text{Sum of the GSC ratings})$$

For example, if the sum of the GSC ratings is 40, the VAF would be:

$$\text{VAF} = 0.65 + (0.01 \times 40) = 0.65 + 0.4 = 1.05$$

Albrecht Function Point Analysis (FPA)

General System Characteristics:

1. Data Communications

Definition: This characteristic assesses how frequently and to what extent the system interacts with remote devices or other systems.

Examples: Systems with networked environments, multiple remote access points, or distributed databases.

Impact: The more complex the communication requirements, the higher the complexity rating. For example, systems that require constant communication over the internet or with remote sensors will score higher.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

2. Distributed Data Processing

Definition: This evaluates whether the system handles data processing at multiple locations.

Examples: A system that performs data processing on distributed nodes or across a cloud infrastructure.

Impact: If the system involves distributed processing (e.g., cloud systems, microservices), the complexity increases.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

3. Performance

Definition: This measures the need for system performance optimization, including response times, throughput, and transaction speed.

Examples: Real-time processing systems like stock trading platforms or gaming servers.

Impact: Systems requiring low-latency responses or high throughput will have higher performance requirements and thus score higher in complexity.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

4. Heavily Used Configuration

Definition: This considers how much of the system is under constant or heavy use and what the performance demands are on that configuration.

Examples: Mission-critical systems, such as those for airlines or banks, which must handle heavy loads with minimal downtime.

Impact: Systems subject to high usage (e.g., thousands of concurrent users) will be rated as more complex.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

5. Transaction Rate

Definition: This characteristic assesses how frequently transactions occur within the system.

Examples: High-frequency trading platforms, point-of-sale systems with thousands of transactions per minute.

Impact: The higher the transaction rate (number of transactions processed per second or minute), the greater the system’s complexity.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

6. On-Line Data Entry

Definition: This evaluates the extent and complexity of real-time data entry in the system.

Examples: E-commerce websites, online banking, or ticket booking systems where users continuously input data.

Impact: Systems with extensive online data entry, especially real-time input validation, will score higher in complexity.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

7. End-User Efficiency

Definition: This considers how user-friendly and efficient the system is for the end-users.

Examples: Systems with intuitive user interfaces, dashboards, or automation that reduce user effort.

Impact: The need for user-centric features, such as optimized navigation and ease of use, increases complexity, especially if the system must support a wide range of users with varying technical skills.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

8. On-Line Update

Definition: This characteristic measures the extent to which the system allows for updates in real-time.

Examples: A system that allows users to modify customer records in real-time or change pricing information in an e-commerce system.

Impact: Systems that require real-time updates to multiple data points will have higher complexity due to the need for synchronization and error handling.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

9. Complex Processing

Definition: This assesses the complexity of internal business logic, including algorithms, calculations, and data transformations.

Examples: Inventory management systems with complex re-ordering rules or financial systems with sophisticated tax calculations.

Impact: The more complex the business rules, calculations, or workflows, the higher the complexity rating.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

10. Reusability

Definition: This characteristic looks at how much of the system's components are designed to be reusable in other applications or systems.

Examples: Modular software systems, libraries, or APIs designed for reusability across different projects.

Impact: The higher the level of reusability of code or components, the greater the complexity due to the need for generalization and abstraction.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

11. Installation Ease

Definition: This measures how easy or complex it is to install the system in the production environment.

Examples: Systems that require extensive configuration, data migration, or compatibility checks during installation.

Impact: If the system has many dependencies or requires extensive setup, it will score higher on complexity. Systems with simple installations, like plug-and-play applications, score lower.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

12. Operational Ease

Definition: This considers how simple it is for operators to run and maintain the system.

Examples: Systems with intuitive control panels, automated backups, and straightforward error-handling mechanisms.

Impact: If the system includes tools that simplify operations (e.g., automation, system health monitoring), it will reduce the complexity. However, systems requiring extensive operator intervention will be rated as more complex

Albrecht Function Point Analysis (FPA)

General System Characteristics:

13. Multiple Sites

Definition: This characteristic evaluates how the system supports deployment and usage across multiple physical locations.

Examples: Retail POS systems used across different store locations or ERP systems deployed in different branches of an organization.

Impact: Systems deployed across multiple locations must handle issues like data synchronization, network reliability, and different local configurations, increasing complexity.

Albrecht Function Point Analysis (FPA)

General System Characteristics:

14. Facilitate Change

Definition: This measures how easily the system can be changed or enhanced to meet evolving business needs.

Examples: Systems designed with flexibility in mind, such as those with modular architecture or those that follow Agile development principles.

Impact: Systems designed for flexibility and future adaptability (like those using microservices or modular components) are more complex to develop but easier to modify, while rigid systems score lower.

Albrecht Function Point Analysis (FPA)

6. Calculate Final Adjusted Function Points

Finally, the adjusted function points (AFP) are calculated by multiplying the unadjusted function points (UFP) by the value adjustment factor (VAF):

$$\text{AFP} = \text{UFP} \times \text{VAF}$$

$$\text{AFP} = 72 \times 1.05 = 75.6$$

Albrecht Function Point Analysis (FPA)

How Function Points Are Used

Once you have calculated the number of function points, you can use them to:

Estimate Effort: Convert function points into person-hours or person-months by applying productivity rates (e.g., 10 function points per person-month).

Estimate Cost: Based on the effort, you can estimate the project cost using standard rates (e.g., cost per person-month).

Estimate Time: By knowing the effort and team size, you can estimate the project duration.

Compare Projects: Since function points are technology-agnostic, they can be used to compare the size and complexity of different projects regardless of the programming language or tools used.

Case Study (FPA)

Customer Relationship Management System

A financial institution wants to implement a ***Customer Relationship Management (CRM)*** system to streamline customer interactions, manage client information, and improve service delivery. The system will handle customer profiles, transactions, communications, and reporting, along with strong security measures.

The institution hires a software development team to estimate the effort, cost, and time required to build this system.

The team uses ***Function Point Analysis (FPA)*** to measure the size of the system and estimate the development effort.

Customer Relationship Management System

Step 1: Identifying Functions in the CRM System

The first step in Function Point Analysis (FPA) for the CRM system is to identify the types of functions in the system.

These include:

a. External Inputs (EI)

Functions where data is input into the system by the user or an external system.

Example in CRM:

Adding new customer information (e.g., name, address, contact details).

Updating customer information (e.g., changing phone number or address).

In function point terms, each data input screen is an External Input.

Customer Relationship Management System

Step 1: Identifying Functions in the CRM System

b. External Outputs (EO)

Functions where the system produces calculated or processed data for external use.

Example in CRM:

Generating customer reports, invoices, or summaries.

Outputting data for analysis or alerts, like overdue payments.

In function point terms, each report generated by the system is an External Output.

Customer Relationship Management System

Step 1: Identifying Functions in the CRM System

c. External Inquiries (EQ)

Simple data retrieval functions that display information without complex processing.

Example in CRM:

Searching for a customer’s contact details or their transaction history.

Querying the system to view information like sales records.

In function point terms, each query-based retrieval is an External Inquiry

Customer Relationship Management System

Step 1: Identifying Functions in the CRM System

d. Internal Logical Files (ILF)

Logical files that the system maintains and controls. These files are critical data stores within the system.

Example in CRM:

The Customer Database, which stores all customer-related information.

The Transaction Log, which keeps a record of customer purchases and interactions.

In function point terms, each major logical data store is an Internal Logical File.

Customer Relationship Management System

Step 1: Identifying Functions in the CRM System

e. External Interface Files (EIF)

Logical files that the system does not maintain but accesses from external systems.

Example in CRM:

Accessing product data from another system for invoicing purposes.

Accessing a third-party marketing database.

In function point terms, each external file the system interfaces with is an External Interface File.

Customer Relationship Management System

Step 2: Estimating the Function Point Count (Assigning Complexity)

Now that we have identified the functions, we can move to Step 2, where we calculate the function points based on the number of Data Element Types (DETs) and File Types Referenced (FTRs).

1. External Inputs (EI)

Function: Add new customer data (e.g., Name, Address, Phone Number, Email).

Data Elements (DETs): 4 fields (Name, Address, Phone Number, Email).

File Types Referenced (FTRs): 1 file (Customer Database).

Complexity: This is a Low Complexity EI (4 DETs and 1 FTR).

Function Point Count: For low complexity, the standard function point count is 3.

Function: Update customer details (similar DETs and FTRs as above).

Function Point Count: 3 (Low Complexity).

Customer Relationship Management System

Step 2: Estimating the Function Point Count (Assigning Complexity)

2. External Outputs (EO)

Function: Generate a customer invoice (involves summarizing purchases, taxes, and discounts).

Data Elements (DETs): 5 fields (Customer Name, Total Purchases, Discount, Tax, Final Amount).

File Types Referenced (FTRs): 2 files (Customer Database, Transaction Log).

Complexity: This is a Low Complexity EO (5 DETs and 2 FTRs).

Function Point Count: For low complexity, the standard function point count is 4.

Function: Generate customer transaction report (summary of purchase history).

Function Point Count: 4 (Low Complexity).

Customer Relationship Management System

Step 2: Estimating the Function Point Count (Assigning Complexity)

3. External Inquiries (EQ)

Function: Query customer details (e.g., search for customer by name).

Data Elements (DETs): 3 fields (Customer Name, Address, Phone Number).

File Types Referenced (FTRs): 1 file (Customer Database).

Complexity: This is a Low Complexity EQ (3 DETs and 1 FTR).

Function Point Count: 3.

Function: Query transaction history.

Function Point Count: 3 (Low Complexity).

Customer Relationship Management System

Step 2: Estimating the Function Point Count (Assigning Complexity)

4. Internal Logical Files (ILF)

Function: Maintain the customer database.

Data Elements (DETs): 4 fields (Customer Name, Address, Phone Number, Email).

Complexity: Based on its size (number of fields), this is considered a Low Complexity ILF.

Function Point Count: For low complexity, the standard function point count is 7.

Function: Maintain the transaction log.

Function Point Count: 7 (Low Complexity).

Customer Relationship Management System

Step 2: Estimating the Function Point Count (Assigning Complexity)

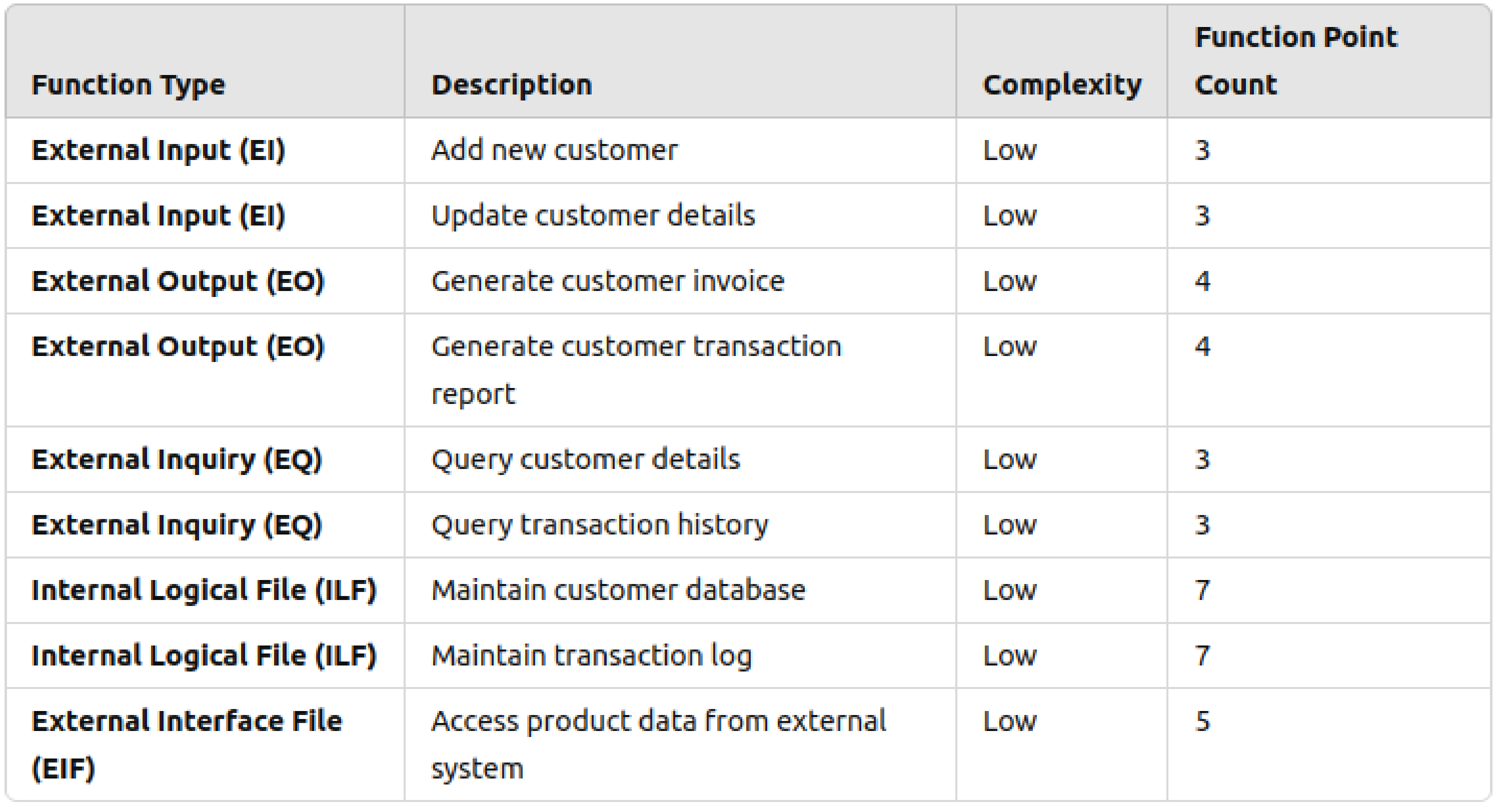
5. External Interface Files (EIF)

Function: Access product data from external system.

Complexity: This is a Low Complexity EIF.

Function Point Count: 5.

Customer Relationship Management System

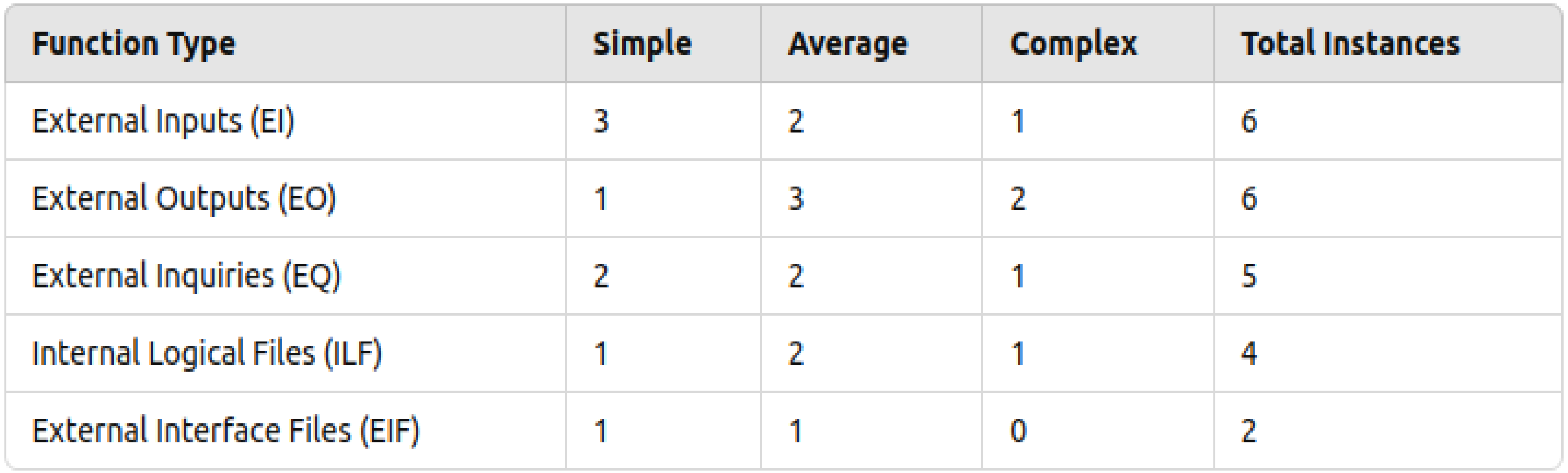


Function Type	Description	Complexity	Function Point Count
External Input (EI)	Add new customer	Low	3
External Input (EI)	Update customer details	Low	3
External Output (EO)	Generate customer invoice	Low	4
External Output (EO)	Generate customer transaction report	Low	4
External Inquiry (EQ)	Query customer details	Low	3
External Inquiry (EQ)	Query transaction history	Low	3
Internal Logical File (ILF)	Maintain customer database	Low	7
Internal Logical File (ILF)	Maintain transaction log	Low	7
External Interface File (EIF)	Access product data from external system	Low	5

3+3+4+4+3+3+7+7+5=39 function points

Customer Relationship Management System

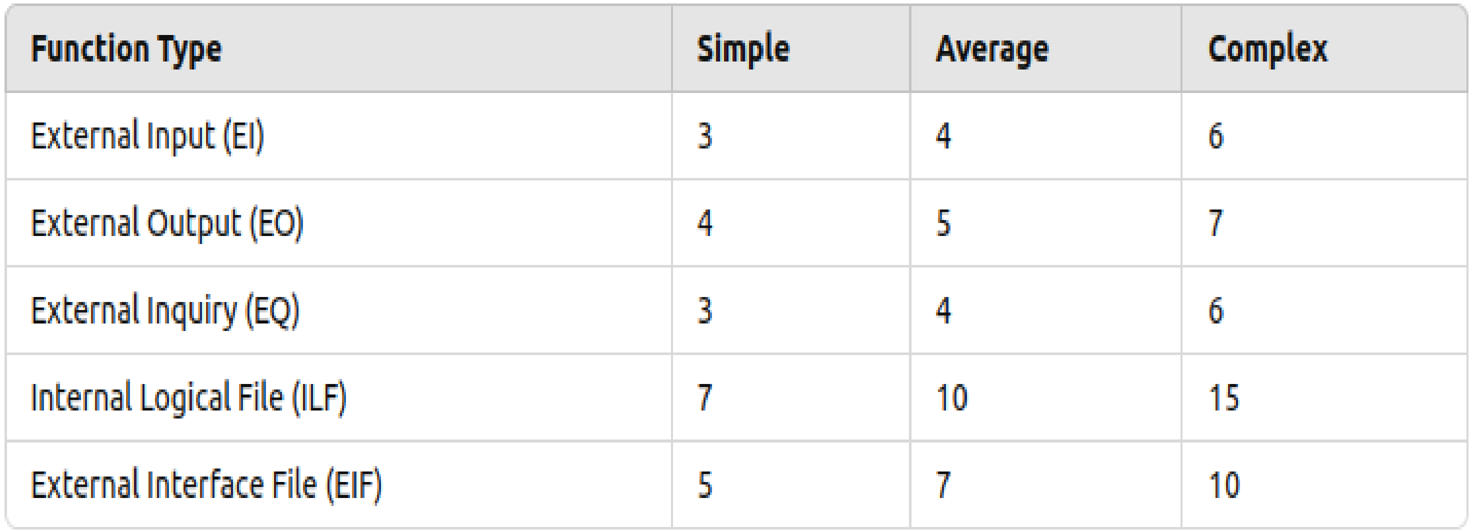
Lets Consider another table.



Function Type	Simple	Average	Complex	Total Instances
External Inputs (EI)	3	2	1	6
External Outputs (EO)	1	3	2	6
External Inquiries (EQ)	2	2	1	5
Internal Logical Files (ILF)	1	2	1	4
External Interface Files (EIF)	1	1	0	2

Customer Relationship Management System

Step 3: Apply the Weighting Factor



Function Type	Simple	Average	Complex
External Input (EI)	3	4	6
External Output (EO)	4	5	7
External Inquiry (EQ)	3	4	6
Internal Logical File (ILF)	7	10	15
External Interface File (EIF)	5	7	10

Customer Relationship Management System

Step 4: Calculate Unadjusted Function Points (UFP)

External Inputs (EI):

$(3\times3)+(2\times4)+(1\times6)=9+8+6=23$ FP

External Outputs (EO)

$(1\times4)+(3\times5)+(2\times7)=4+15+14=33$ FP

External Inquiries (EQ):

$(2\times3)+(2\times4)+(1\times6)=6+8+6=20$ FP

Internal Logical Files (ILF):

$(1\times7)+(2\times10)+(1\times15)=7+20+15=42$ FP

External Interface Files (EIF):

$(1\times5)+(1\times7)=5+7=12$ FP

UFP$=23+33+20+42+12=130$

Customer Relationship Management System

Step 5: Apply the Value Adjustment Factor (VAF)

Suppose the sum of GSC’s ratings is 45.

The Value Adjustment Factor (VAF) is calculated as:

$$\mathbf{VAF} = 0.65 + (0.01 \times 45) = 0.65 + 0.45 = 1.10$$

Customer Relationship Management System

Step 6: Calculate Adjusted Function Points (AFP)

$$AFP = UFP \times VAF$$

$$AFP = 130 \times 1.10 = 143 \text{ FP}$$

Customer Relationship Management System

Step 7: Estimating Effort, Cost, and Time

Effort Estimation

Typically, effort is estimated in person-hours or person-months. Suppose the organization has a historical productivity rate of 8 function points per person-month.

Effort (person-months) = AFP / Productivity rate **= 143 / 8 = 17.88 person-months**

Cost Estimation

Assuming the cost of one person-month is 10,000, the total cost of the project would be:

Cost=18 person-months×10,000=180,000

Time Estimation

If the development team consists of 3 full-time developers, the time to complete the project would be:

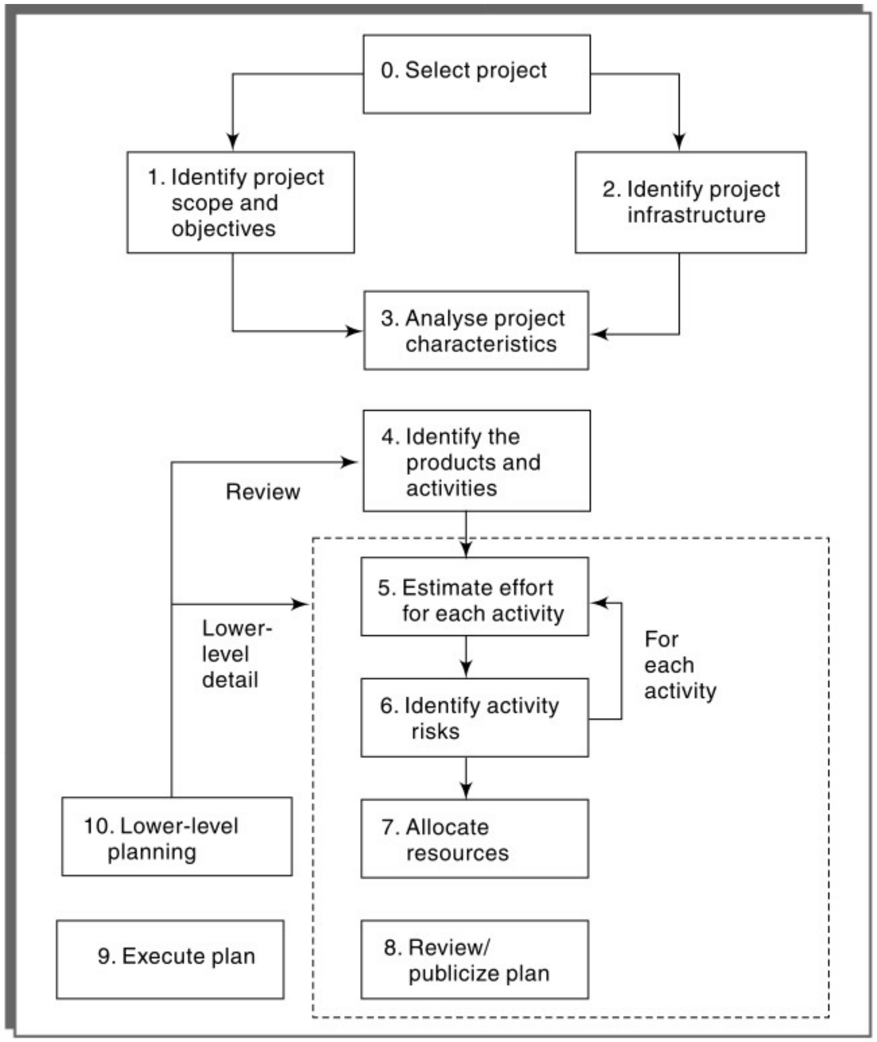
Time (months) = Effort (person-months)

Number of developers =

18 / 3 = 6 months

Activity Planing

Step-Wise Project Planing Framework


0. Select project
1. Identify project scope and objectives
2. Identify project infrastructure
3. Analyse project characteristics
4. Identify the products and activities
Review
5. Estimate effort for each activity
Lower-level detail
For each activity
6. Identify activity risks
7. Allocate resources
10. Lower-level planning
9. Execute plan
8. Review/ publicize plan

An overview of Step Wise

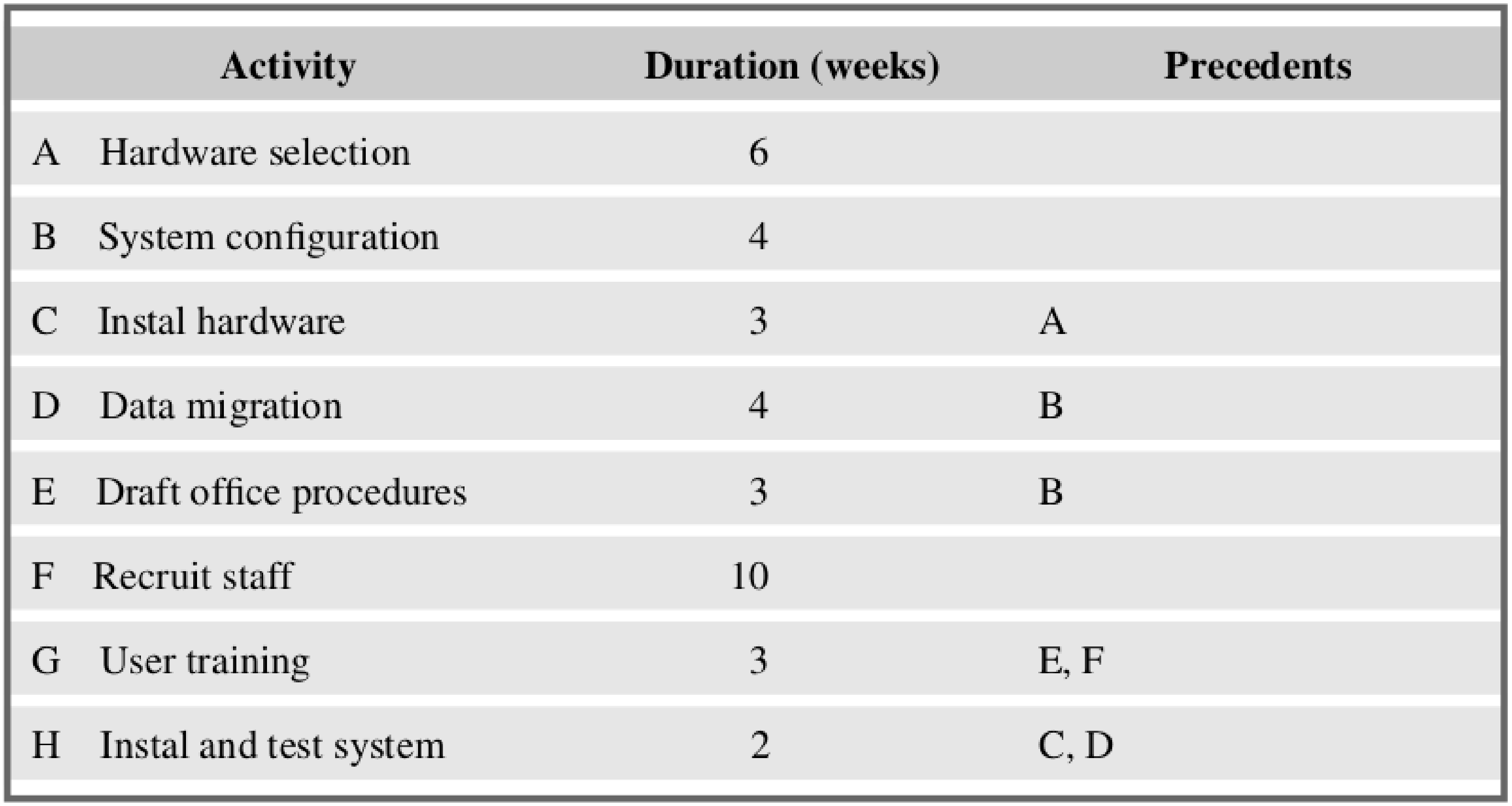
Activity Planning

Activity planning and scheduling techniques place an emphasis on completing the project in a minimum time at an acceptable cost or, alternatively, meeting a set target date at minimum cost.

Network Planing Models

Network Planning Models

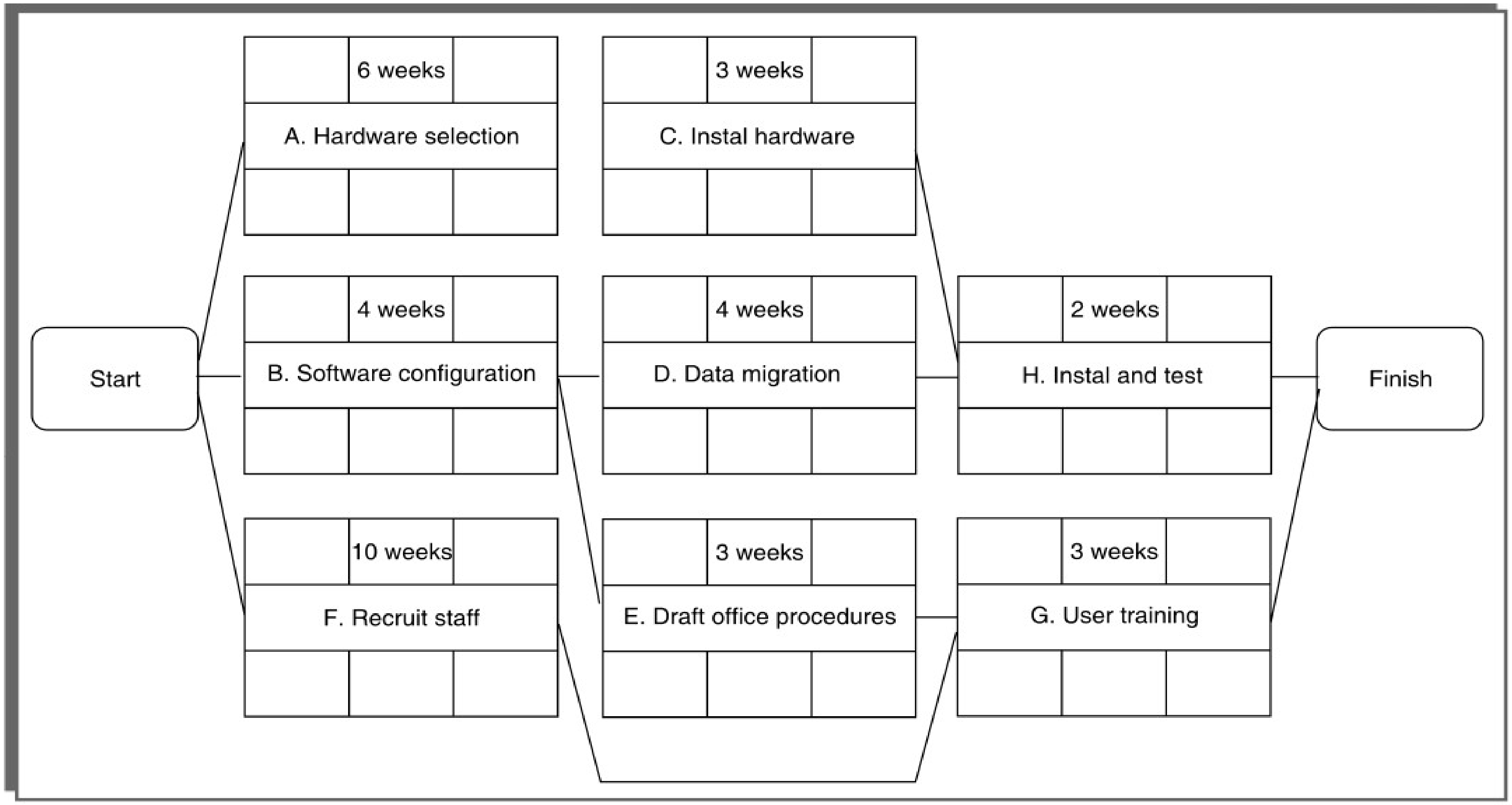
An example project specification with estimated activity duration and precedence requirements.



Activity	Duration (weeks)	Precedents
A Hardware selection	6	
B System configuration	4	
C Instal hardware	3	A
D Data migration	4	B
E Draft office procedures	3	B
F Recruit staff	10	
G User training	3	E, F
H Instal and test system	2	C, D

Network Planning Models

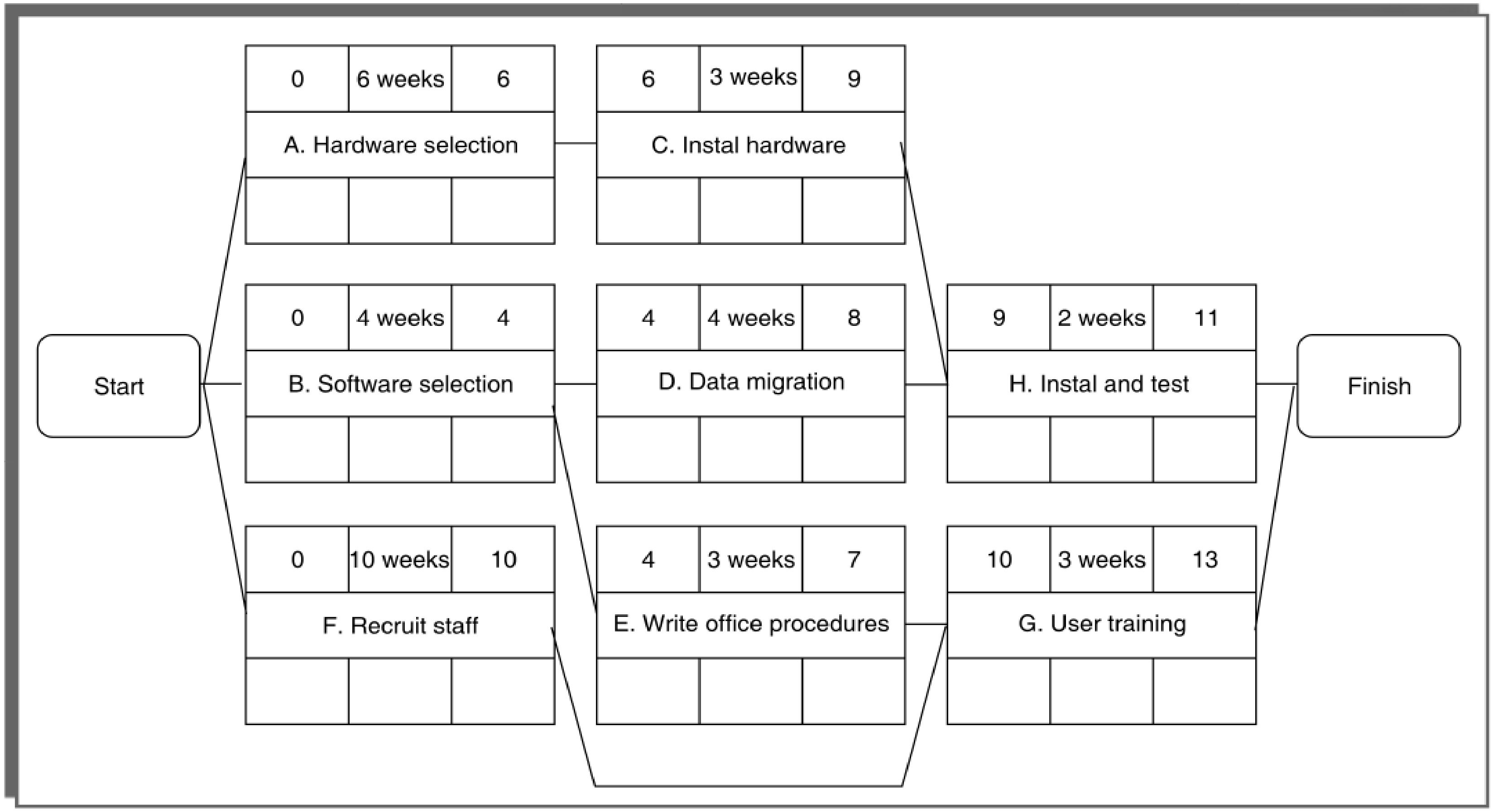
Precedence Network.



Network Planning Models

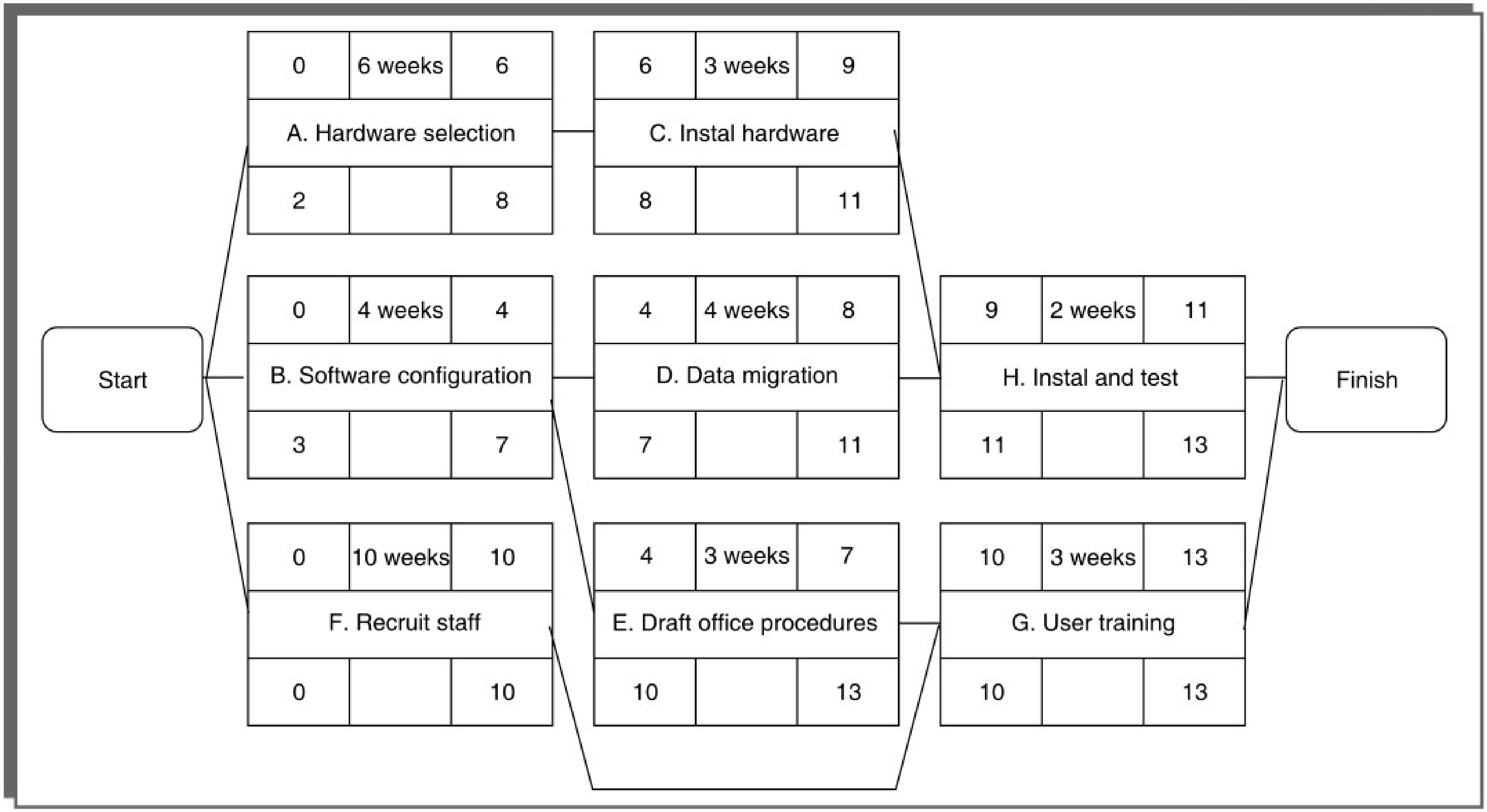
Forward Pass:

The forward pass is carried out to calculate the earliest dates on which each activity may be started and completed.



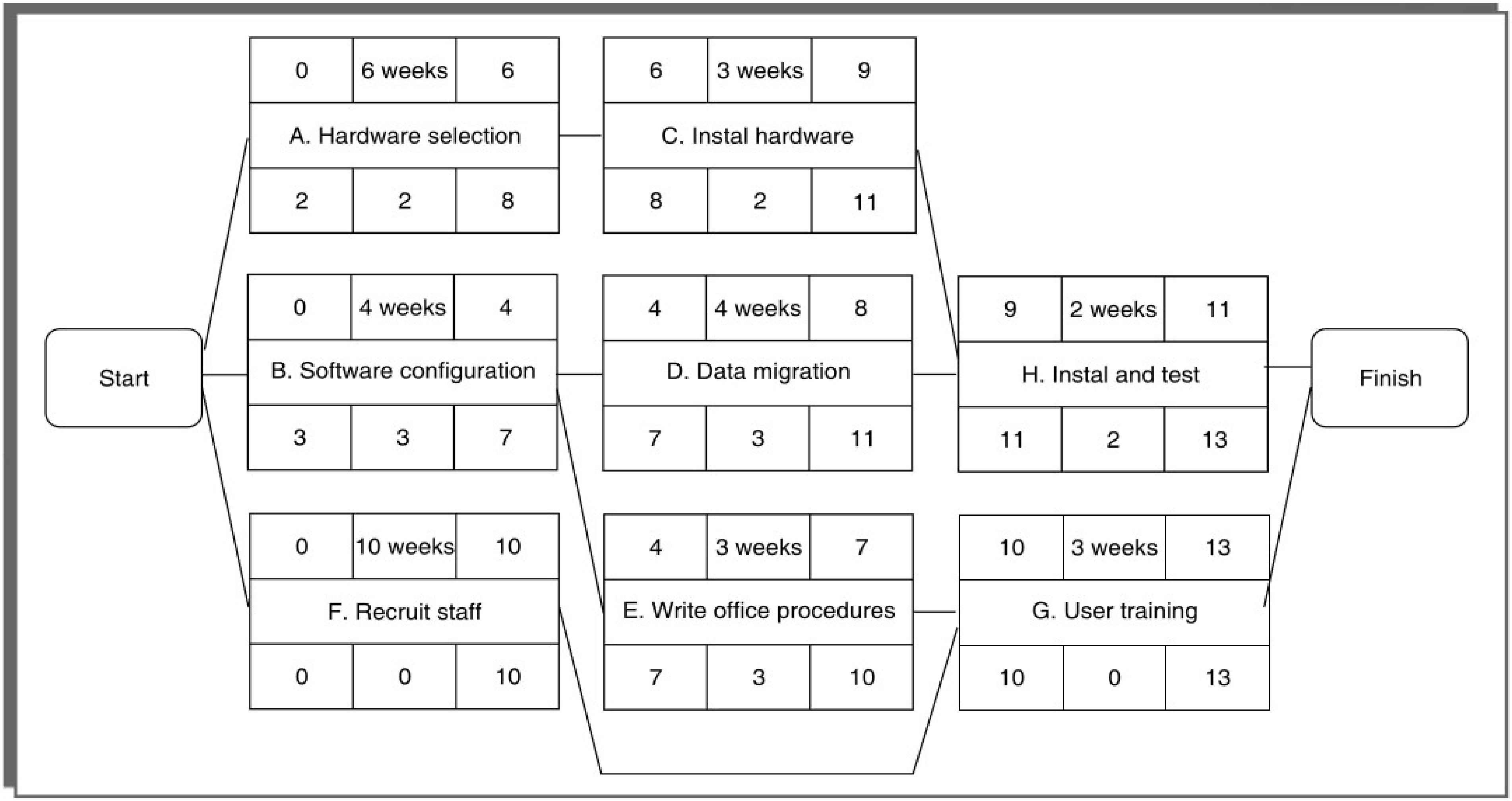
Network Planning Models

Backward Pass: There is some mistake in the network



Network Planning Models

Critical Path:



Network Planning Models

Activity Float:

Activities A and C each have 2 weeks’ total float. If,however, activity A uses up its float (that is, it is not completed until week 8) then activity B will have zero float (it will have become critical).

In such circumstances it may be misleading and detrimental to the project’s success to publicize total float!

Network Planning Models

There are a number of other measures of activity float, including the following:

Total Float:

The maximum amount of time an activity can be delayed without delaying the project completion date.

Free Float:

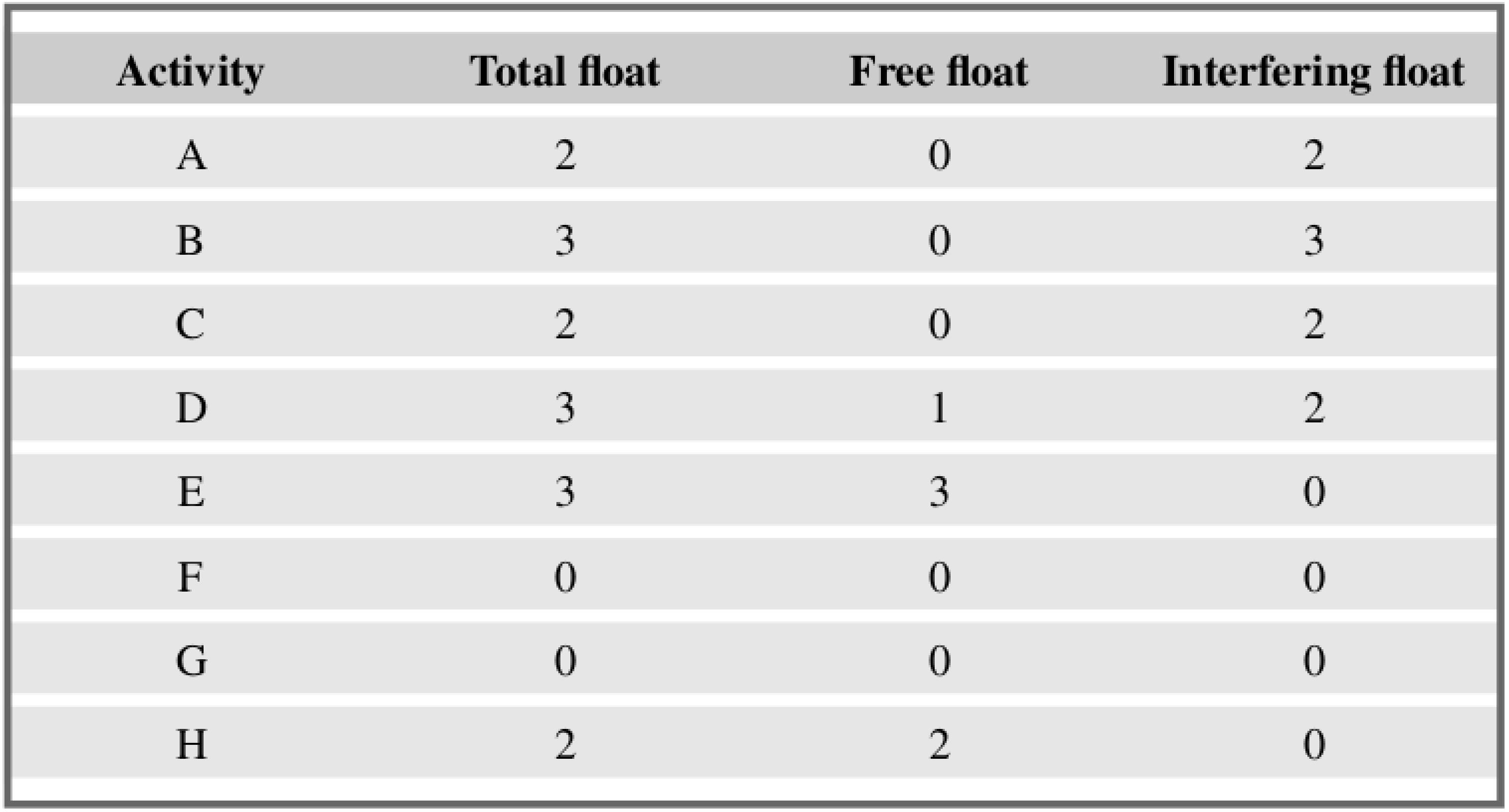
The amount of time an activity can be delayed without delaying the start of any succeeding activity.

Interfering Float:

The part of total float that, if used, would delay the start of some succeeding activities but not the project completion.

Network Planning Models

Activity Float:



Activity	Total float	Free float	Interfering float
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

Risk Management

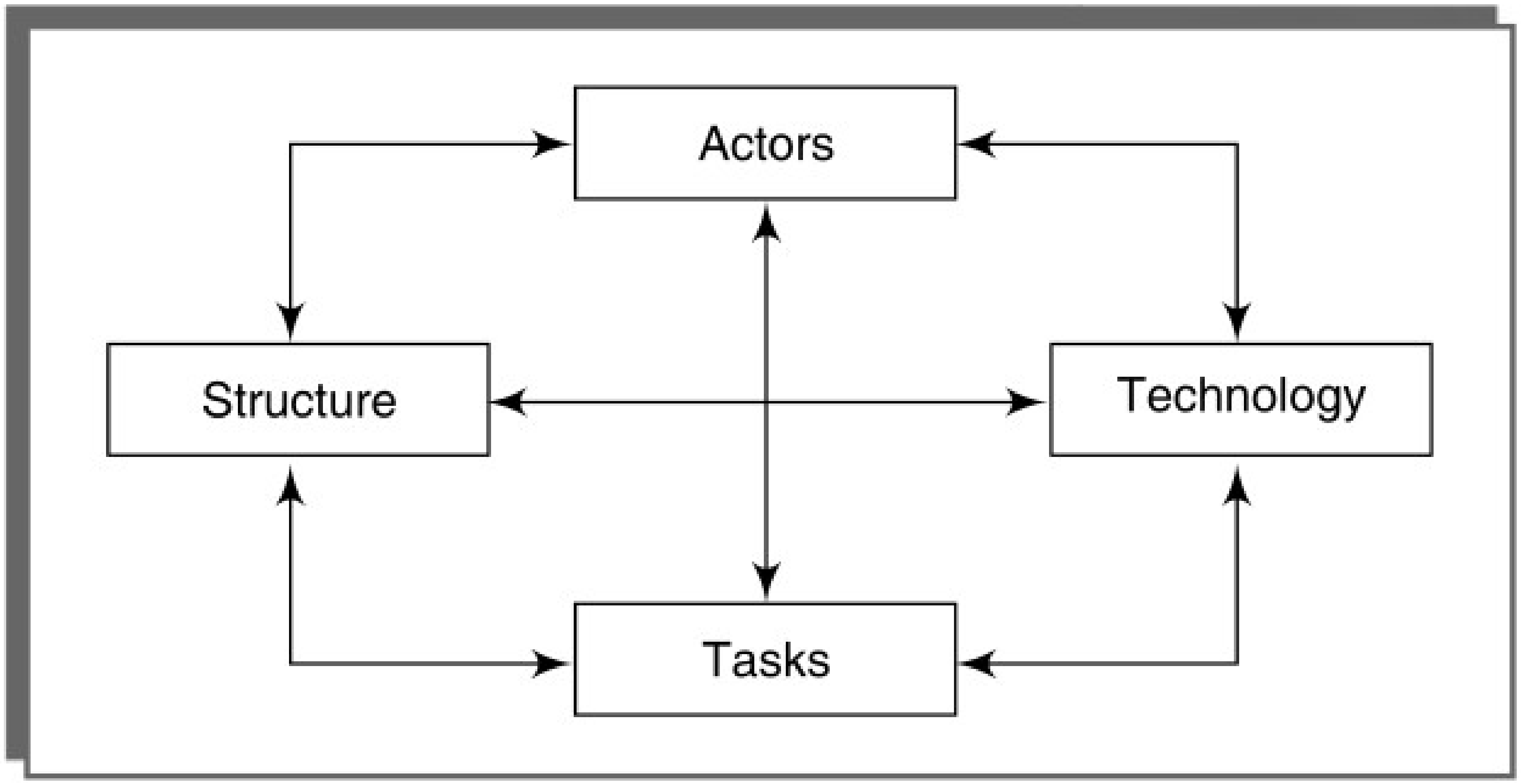
Risk Management

An uncertain event or condition that, if it occurs, has a positive or negative effect on a project’s objectives.

The key elements of a risk follow.

- It relates to the future.
- The future is inherently uncertain.
- Some things which seem obvious when a project is over, for example that the costs were under estimated or that a new technology was overly difficult to use, might not have been so obvious during planning.

Risk Framework



Risk Management

Structure describes the management structures and systems, including those affecting planning and control.

For example, the implementation might need user participation in some tasks, but the responsibility for managing the users’ contribution might not be clearly allocated.

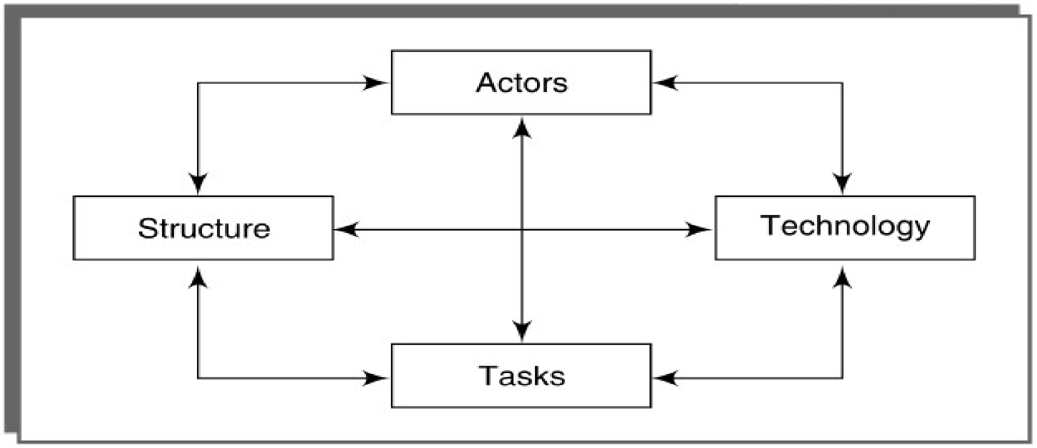
Tasks’ relates to the work planned. For instance, the complexity of the work might lead to delays because of the additional time required integrate the large number of components.

Risk Management

Risks often arise from the relationships between factors.

– for example between technology and people.

If a development technology is novel then the developers might not be experienced in its use and delay results.



Risk Management Framework

Risk Management Framework

Planning for risk includes these steps:

(i) risk identification;

(ii) risk analysis and prioritization;

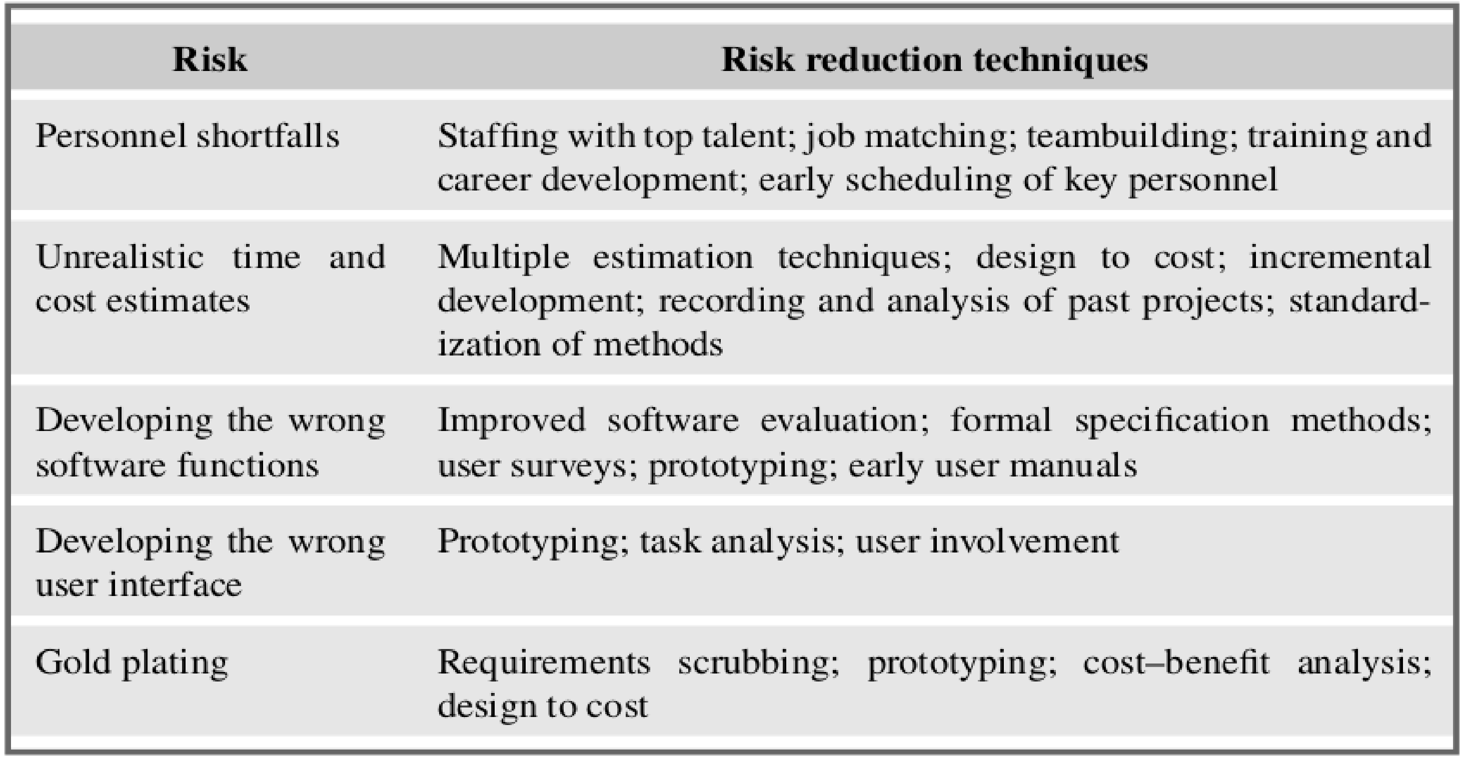
(iii) risk planning;

(iv) risk monitoring.

Risk Management Framework

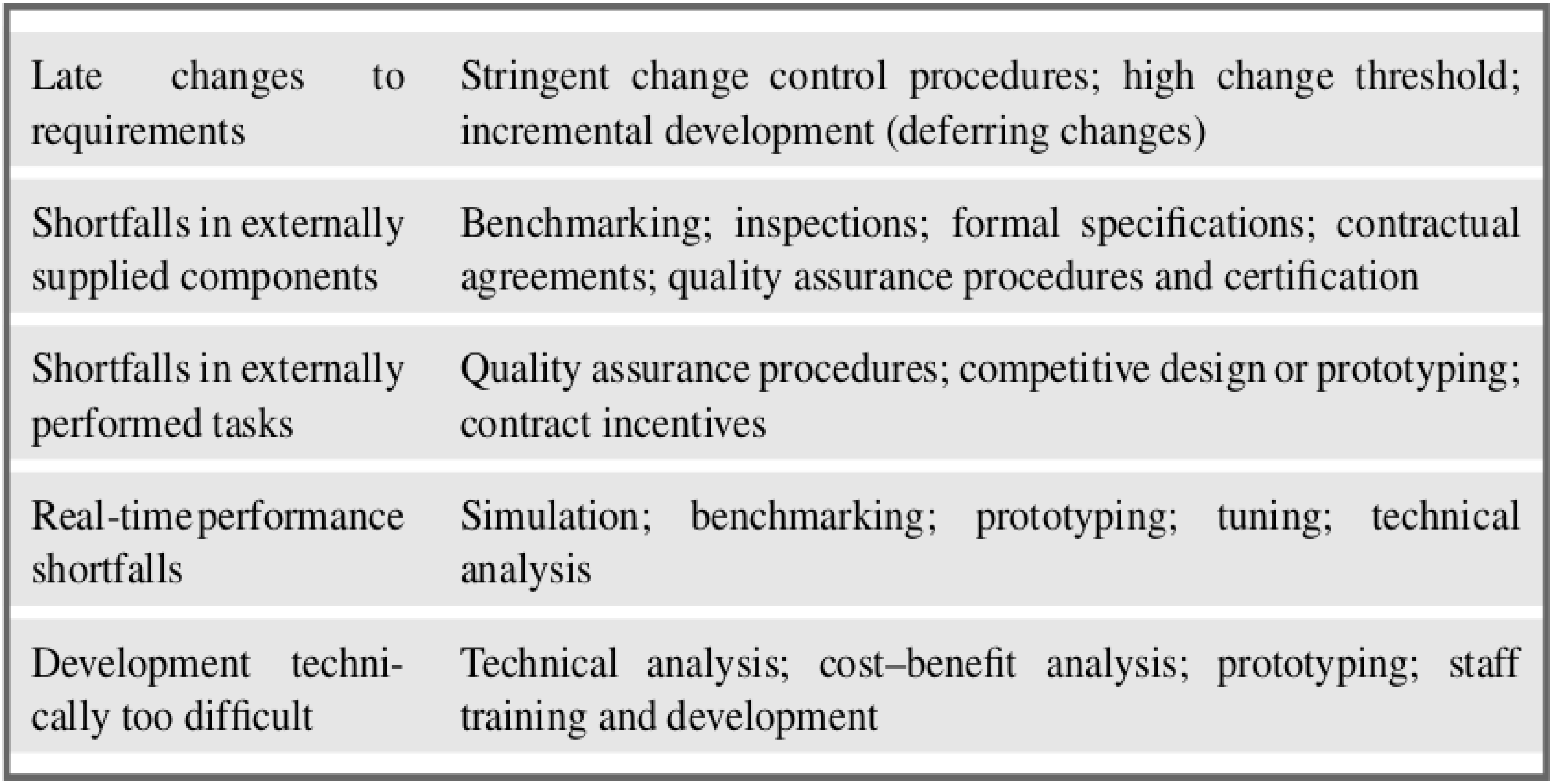
(i) Risk Identification:

The two main approaches to the identification of risks are the use of ***checklists*** and ***brainstorming.***



Risk	Risk reduction techniques
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
Gold plating	Requirements scrubbing; prototyping; cost–benefit analysis; design to cost

Risk Management Framework



Late changes to requirements	Stringent change control procedures; high change threshold; incremental development (deferring changes)
Shortfalls in externally supplied components	Benchmarking; inspections; formal specifications; contractual agreements; quality assurance procedures and certification
Shortfalls in externally performed tasks	Quality assurance procedures; competitive design or prototyping; contract incentives
Real-time performance shortfalls	Simulation; benchmarking; prototyping; tuning; technical analysis
Development technically too difficult	Technical analysis; cost–benefit analysis; prototyping; staff training and development

Risk Management Framework

(ii) Risk Assessment

A common problem with risk identification is that a list of risks is potentially endless. A way is needed of distinguishing the damaging and likely risks.

This can be done by estimating the risk exposure for each risk using the formula:

risk exposure = (potential damage) * (probability of occurrence)

Risk Management Framework

Example:

A project depended on a data center vulnerable to fire.

It might be estimated that if a fire occurred a new computer configuration could be established for £500,000.

It might also be estimated that where the computer is located there is a 1 in 1000 chance of a fire actually happening, that is a probability of 0.001.

The risk exposure in this case would be:

$$£500{,}000 * 0.001 = £500$$

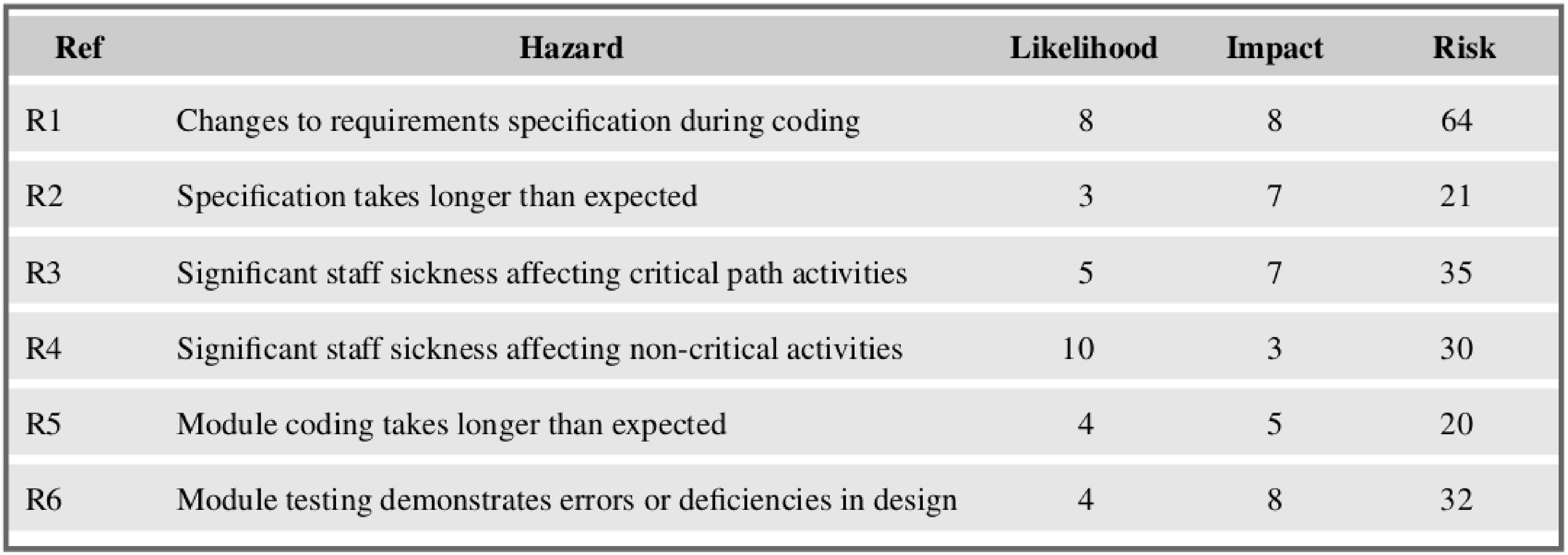
Risk Management Framework

Justification:

A way of understanding this value is as the minimum sum an insurance company would require as a premium.

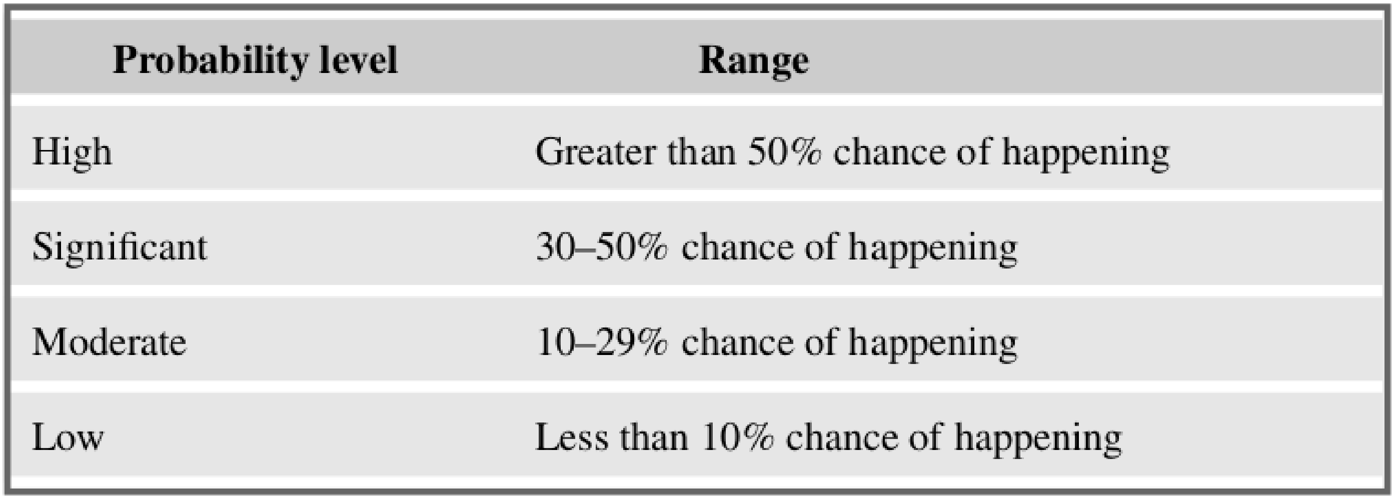
If 1000 companies, all in the same position, each contributed £500 to a fund then, when the 1 in 1000 chance of the fire actually occurred, there would be enough money to cover the cost of recovery.

Risk Management Framework

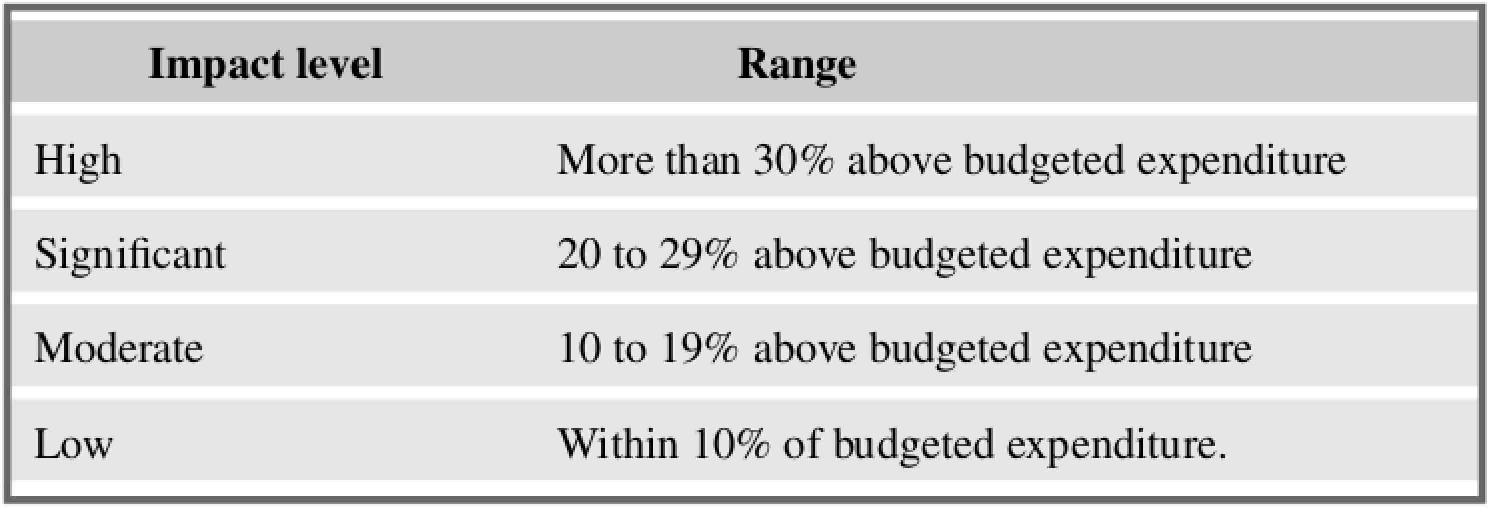


Ref	Hazard	Likelihood	Impact	Risk
R1	Changes to requirements specification during coding	8	8	64
R2	Specification takes longer than expected	3	7	21
R3	Significant staff sickness affecting critical path activities	5	7	35
R4	Significant staff sickness affecting non-critical activities	10	3	30
R5	Module coding takes longer than expected	4	5	20
R6	Module testing demonstrates errors or deficiencies in design	4	8	32

Risk Management Framework

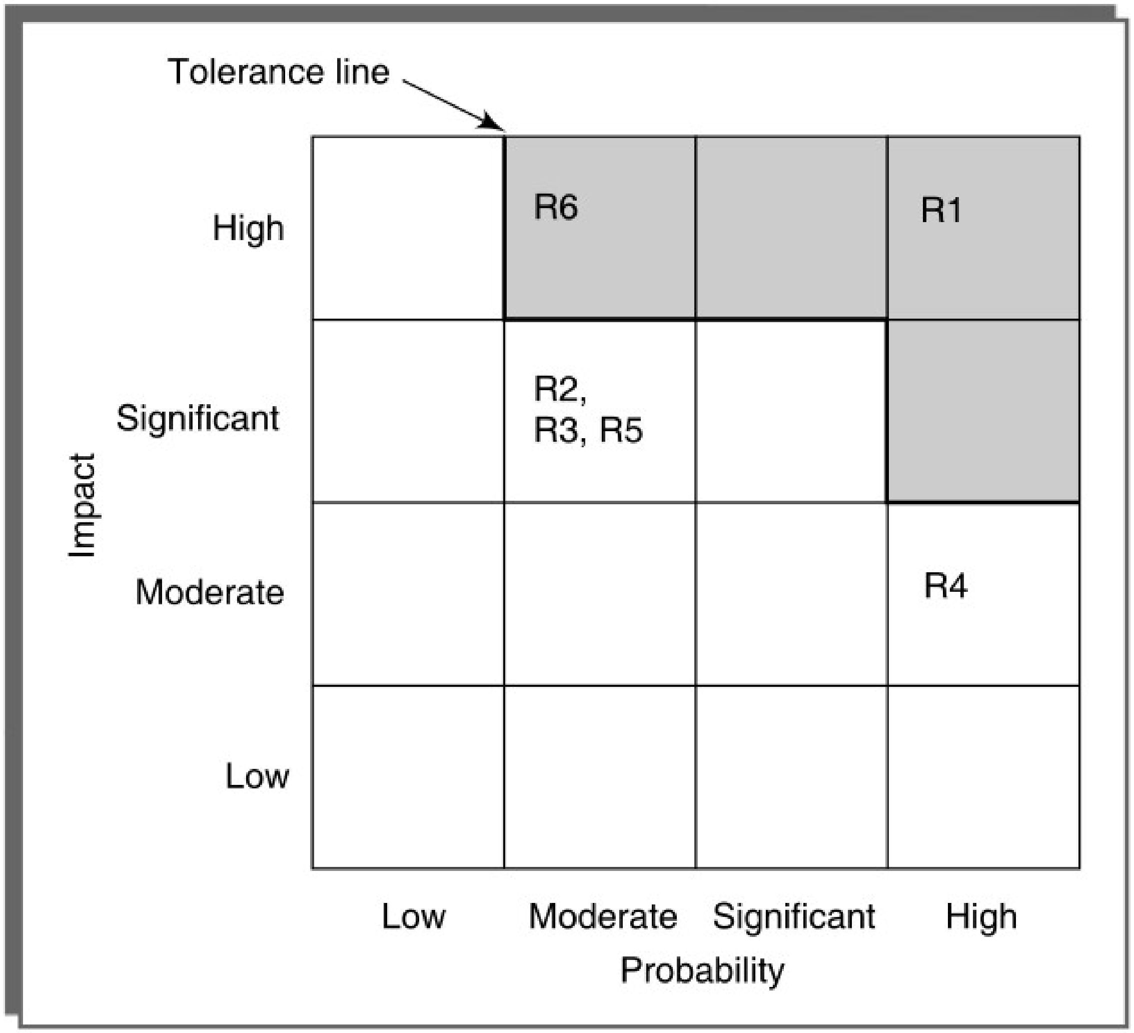


Probability level	Range
High	Greater than 50% chance of happening
Significant	30–50% chance of happening
Moderate	10–29% chance of happening
Low	Less than 10% chance of happening



Impact level	Range
High	More 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.

Risk Management Framework



Cost Benefit Analysis

Example:

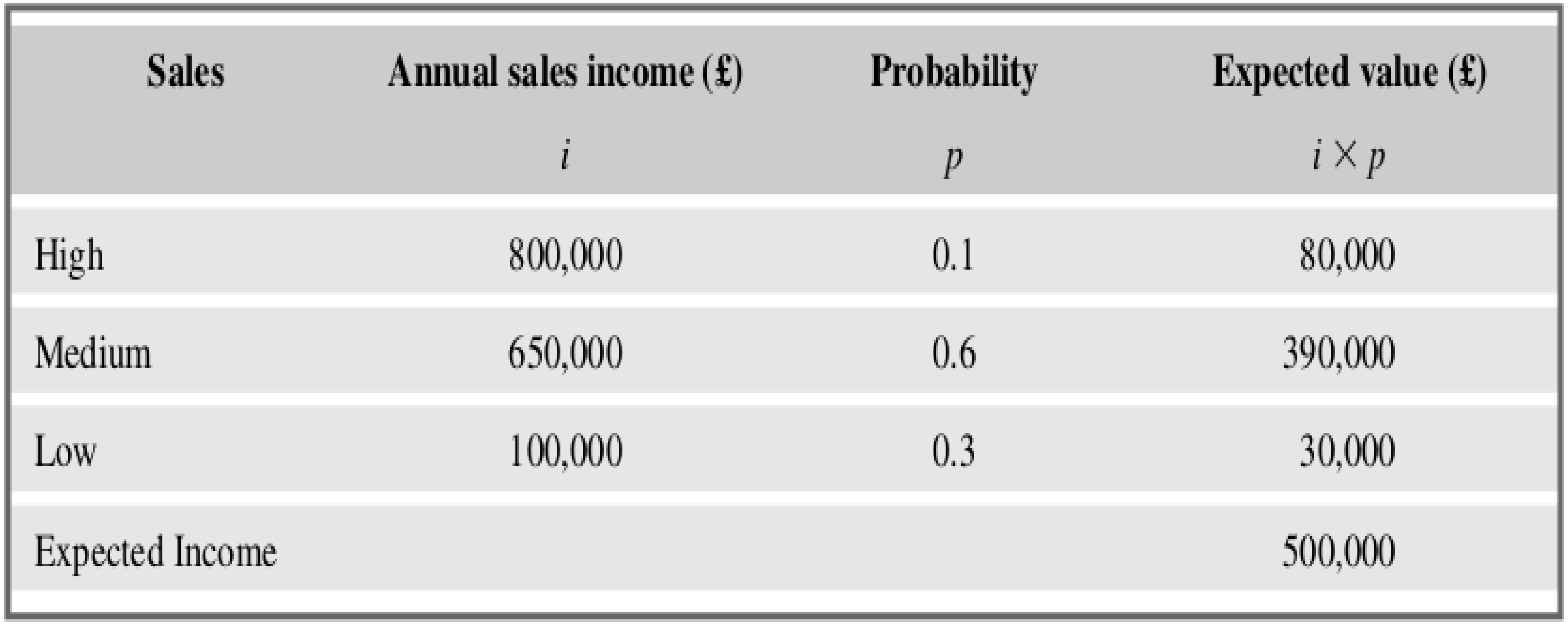
BuyRight, a software house, is considering developing a payroll application for use in academic institutions and is currently engaged in a cost–benefit analysis.

Study of the market has shown that, if BuyRight can target it efficiently and no competing products become available, it will obtain a high level of sales generating an annual income of £800,000. It estimates that there is a 1 in 10 chances of this happening.

However, a competitor might launch a competing application before its own launch date and then sales might generate only £100,000 per year. It estimates that there is a 30% chance of this happening.

The most likely outcome, it believes, is somewhere in between these two extremes – it will gain a market lead by launching before any competing product becomes available and achieve an annual income of £650,000.

Cost Benefit Analysis



Sales	Annual sales income (£)	Probability	Expected value (£)
	i	p	$i \times p$
High	800,000	0.1	80,000
Medium	650,000	0.6	390,000
Low	100,000	0.3	30,000
Expected Income			500,000

Risk Profile Analysis

An approach which attempts to overcome some of the objections to cost–benefit averaging is the construction of risk profiles using sensitivity analysis.

This involves varying each of the parameters that affect the project’s cost or benefits to as certain how sensitive the project’s profitability is to each factor.

We might, for example, vary one of our original estimates by plus or minus 5% and recalculate the expected costs and benefits for the project.

Risk Profile Analysis

By repeating this exercise for each of our estimates in turn we can evaluate the sensitivity of the project to each factor.

By studying the results of a sensitivity analysis we can identify those factors that are most important to the success of the project.

We then need to decide whether we can exercise greater control over them or otherwise mitigate their effects. If neither is the case, then we must live with the risk or abandon the project.

Risk Profile Analysis Using Decision Trees

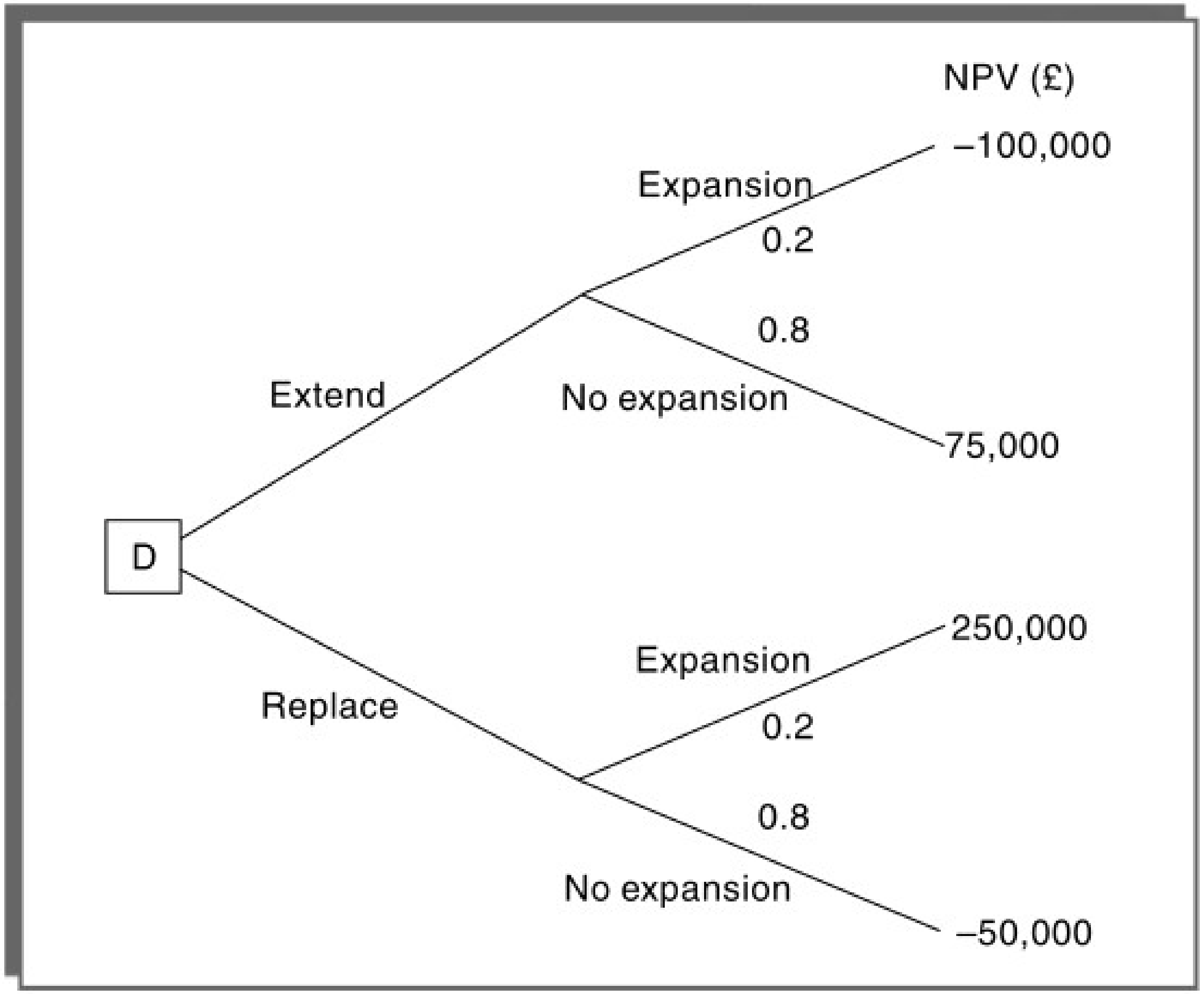
A company “xyz” is considering when to replace its sales order processing system.

The decision largely rests upon the rate at which its business expands – if its market share significantly increases (which it believes will happen if rumours of a competitor’s imminent bankruptcy are fulfilled) the existing system might need to be replaced within two years.

Not replacing the system in time could be an expensive option as it could lead to lost revenue if it cannot cope with increased sales.

Replacing the system immediately will, however, be expensive as it will mean deferring other projects already scheduled.

Risk Profile Analysis Using Decision Trees



Risk Profile Analysis Using Decision Trees

The expected value of each path is the sum of the value of each possible outcome multiplied by its probability of occurrence.

The expected value of extending the system is therefore
£40,000 (75,000 * 0.8 – 100,000 * 0.2)

and the expected value of replacing the system
£10,000 (250,000 * 0.2 – 50,000 * 0.8).

The company should therefore choose the option of extending the existing system.

Risk Management Framework

(iii) Risk Planing:

- risk acceptance;

- risk avoidance;

  (For example, given all the problems with developing software solutions from scratch, managers might decide to retain existing clerical methods, or to buy an off-the-shelf solution.)

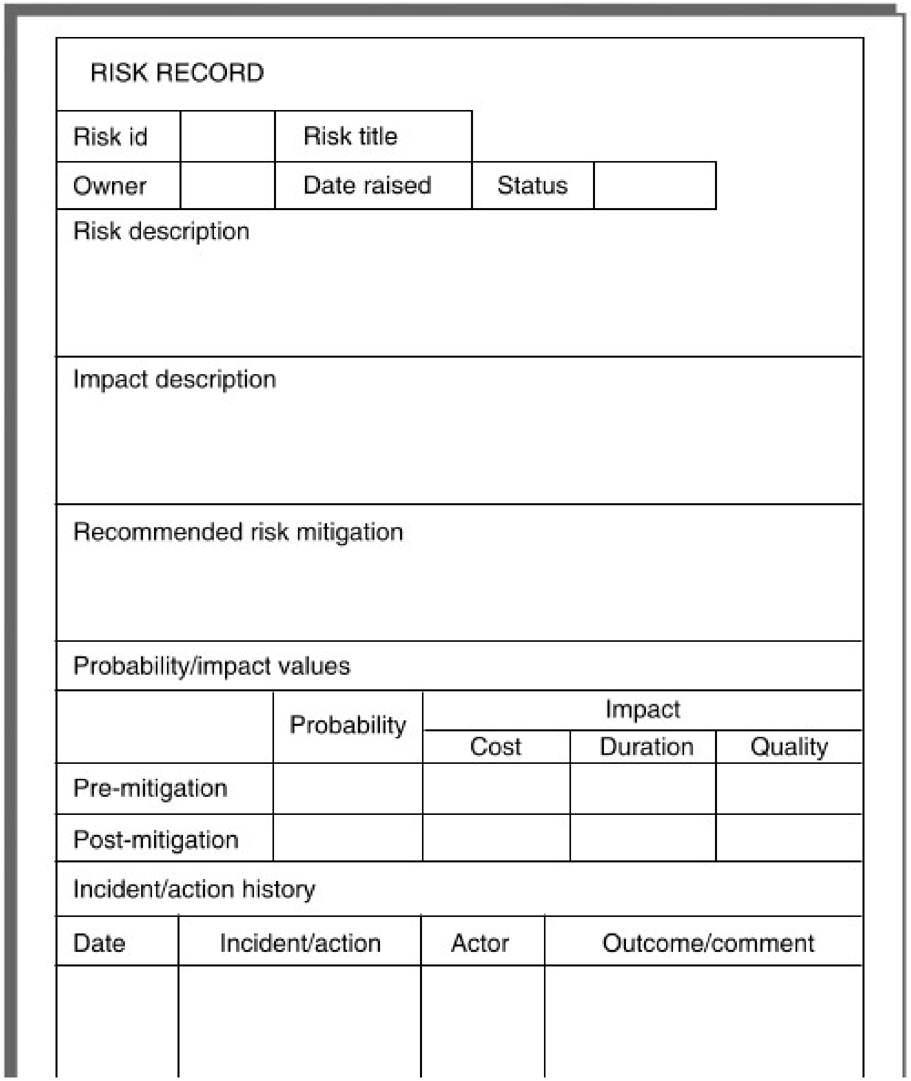
- risk reduction and mitigation;

  (For example, taking regular back-ups of data storage would reduce the impact of data corruption)

- risk transfer;

  (outsource project/ activity)

Risk Management Framework



RISK RECORD

Risk id		Risk title			
Owner		Date raised	Status		

Risk description

Impact description

Recommended risk mitigation

Probability/impact values

	Probability	Impact		
		Cost	Duration	Quality
Pre-mitigation				
Post-mitigation				

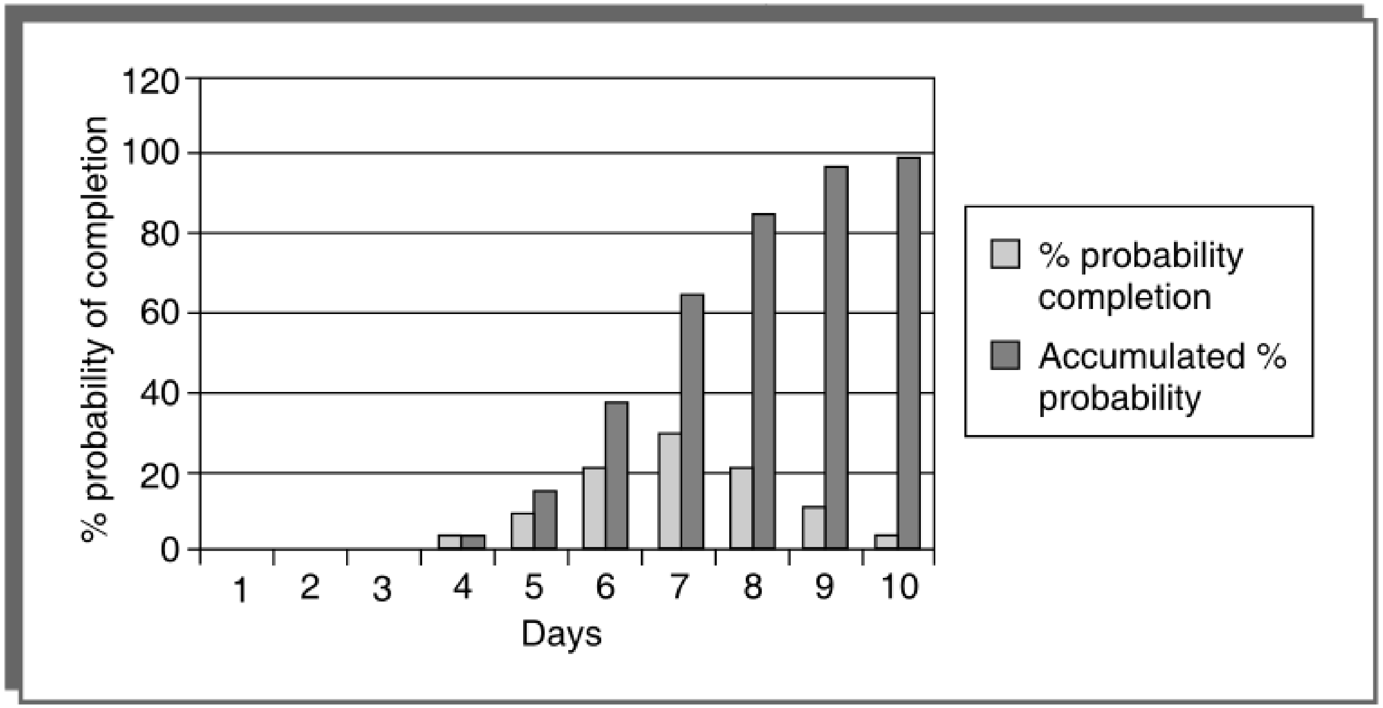
Incident/action history

Date	Incident/action	Actor	Outcome/comment

Risk Management Framework

PERT (Program Evaluation Review Technique)

- To evaluate the effects of uncertainty.



Risk Management Framework

PERT (Program Evaluation Review Technique)

PERT requires three estimates:

Most likely time:

The time we would expect the task to take under normal circumstances. We shall identify this by the letter m.

Optimistic time:

The shortest time in which we could expect to complete the activity, barring outright miracles. We shall use the letter a for this.

Pessimistic time:

The worst possible time, allowing for all reasonable eventualities.

We shall call this b.

Risk Management Framework

PERT (Program Evaluation Review Technique)

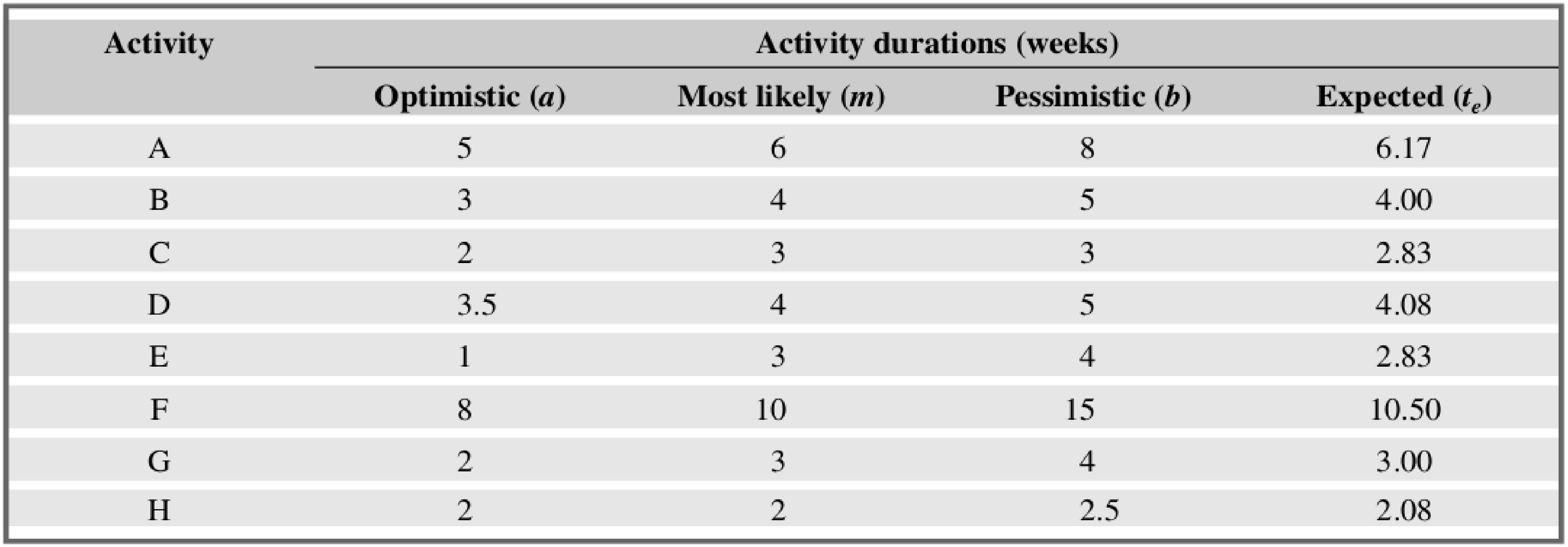
PERT then combines these three estimates to form a single expected duration, t_e , using the formula

$$t_e = \frac{a + 4m + b}{6}$$

Risk Management Framework

PERT (Program Evaluation Review Technique)

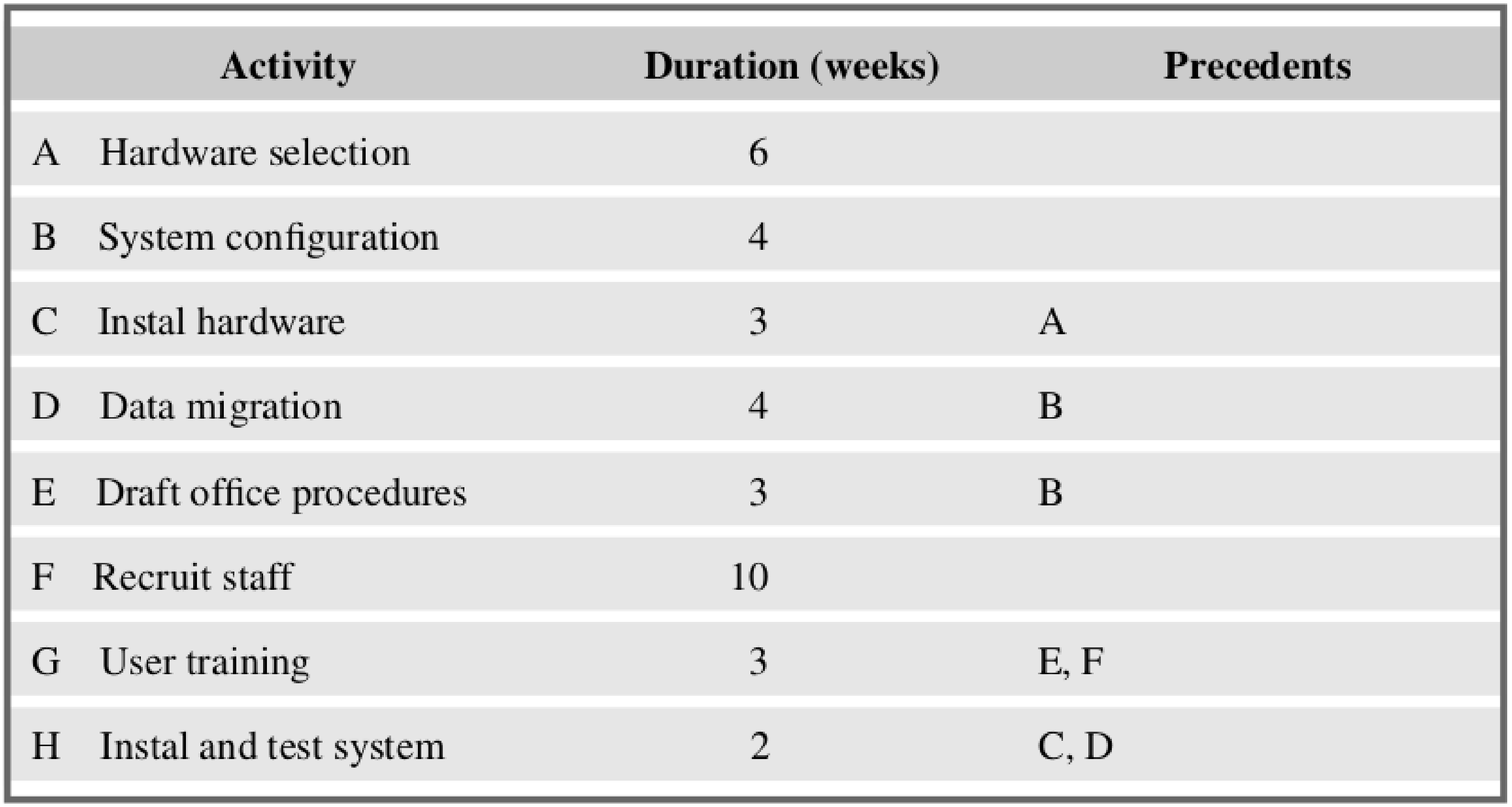
$$t_e = \frac{a + 4m + b}{6}$$



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

Risk Management Framework

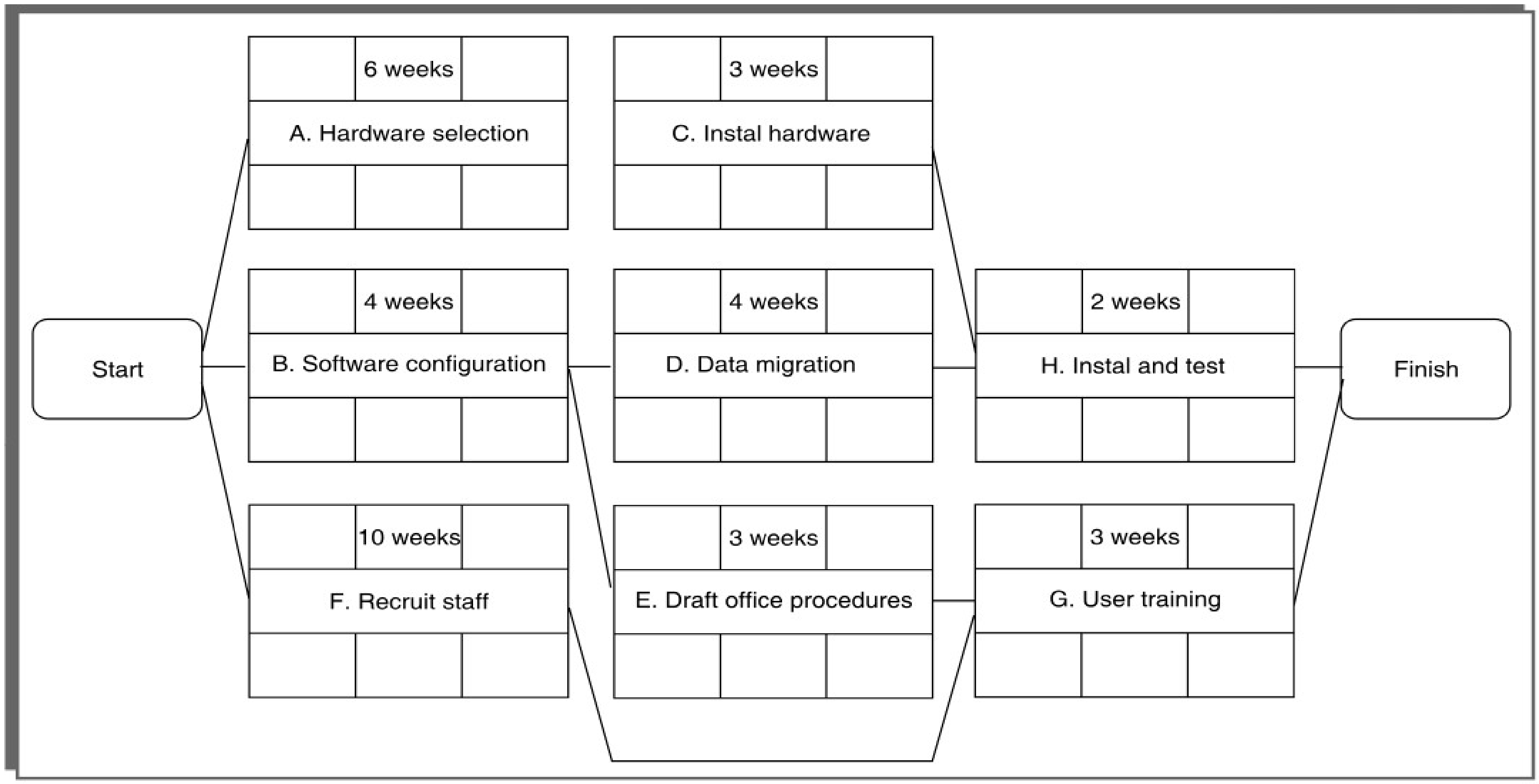
PERT (Program Evaluation Review Technique)



Activity		Duration (weeks)	Precedents
A	Hardware selection	6	
B	System configuration	4	
C	Instal hardware	3	A
D	Data migration	4	B
E	Draft office procedures	3	B
F	Recruit staff	10	
G	User training	3	E, F
H	Instal and test system	2	C, D

Risk Management Framework

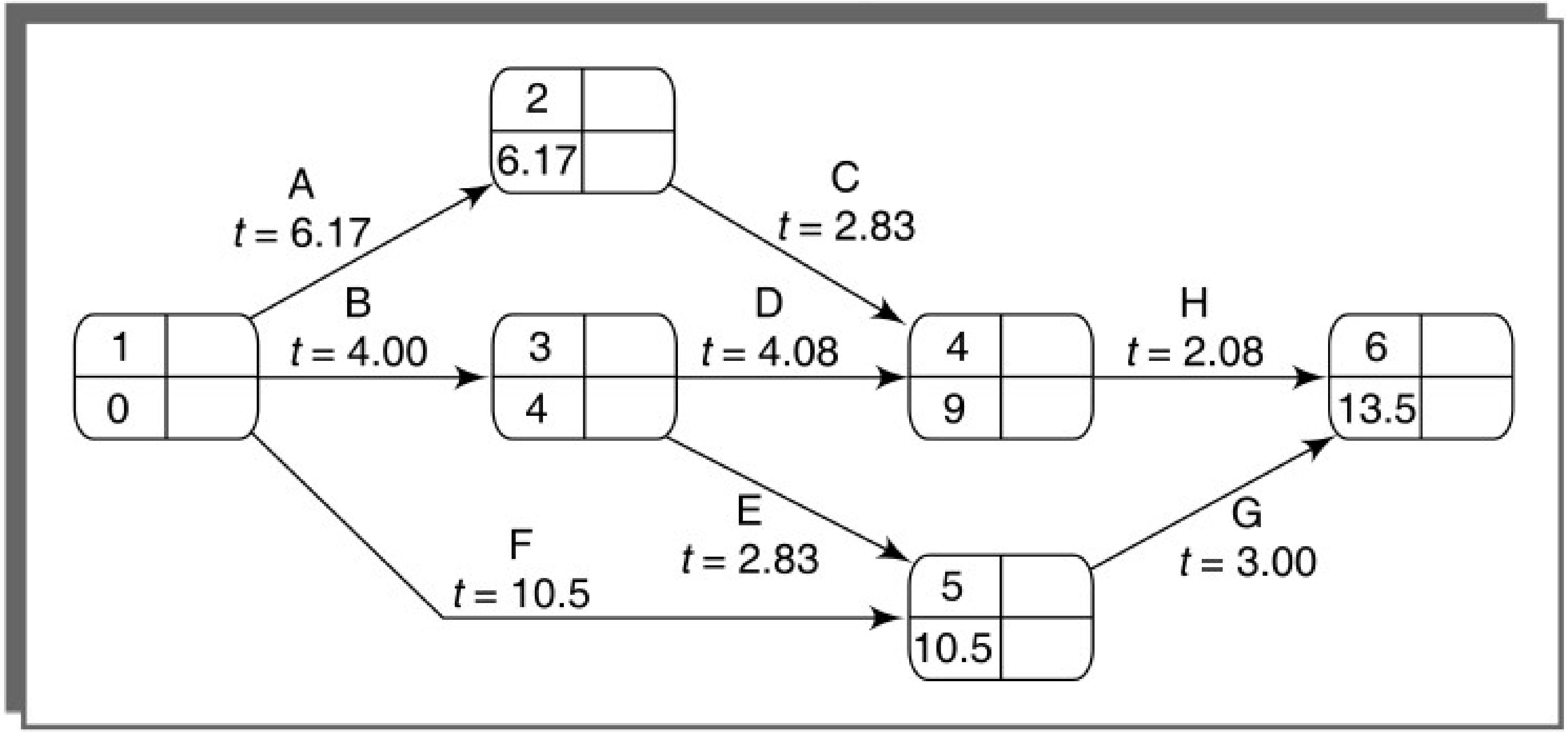
PERT (Program Evaluation Review Technique)



Risk Management Framework

PERT (Program Evaluation Review Technique)

Forward pass:



Risk Management Framework

PERT (Program Evaluation Review Technique)

Activity standard deviation: The activity standard deviation is proportional to the difference between the optimistic and pessimistic estimates, and can be used as a ranking measure of the degree of uncertainty or risk for each activity.

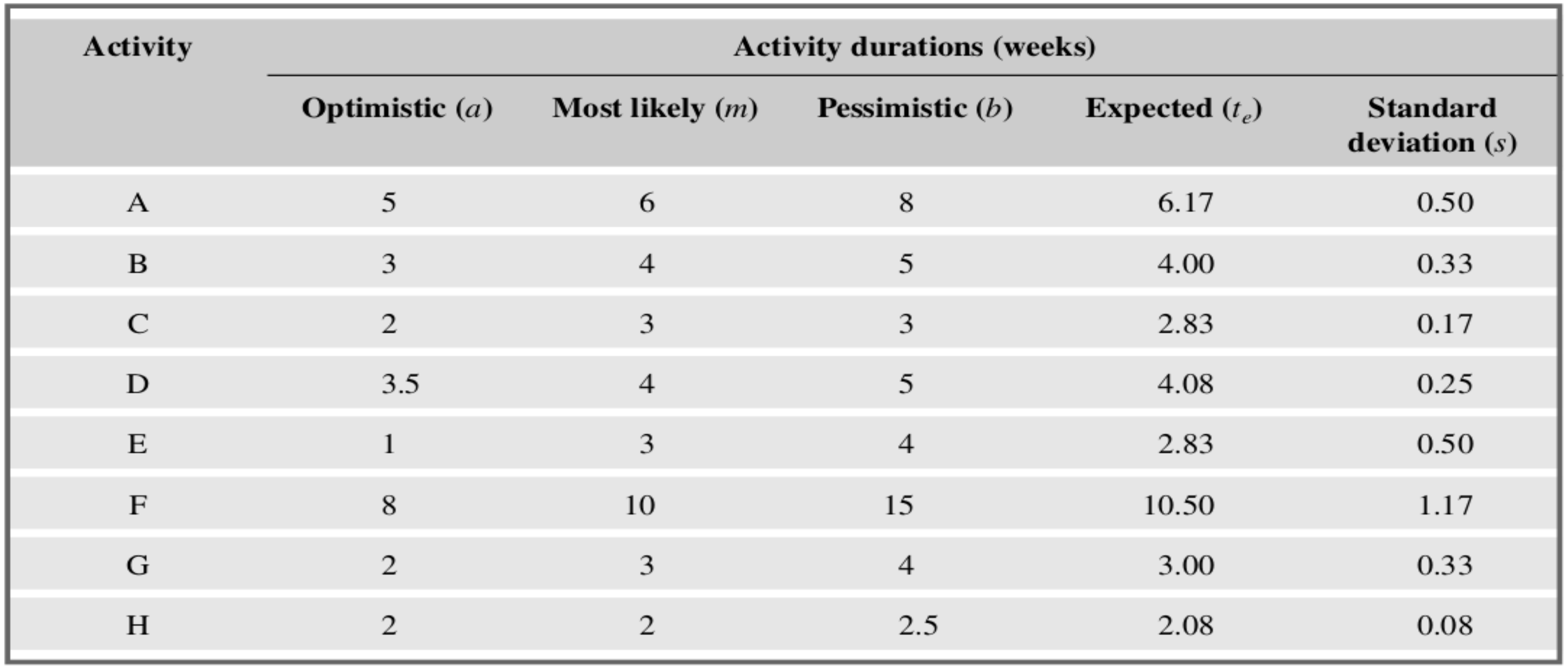
$$s = \frac{b - a}{6}$$

Risk Management Framework

PERT (Program Evaluation Review Technique)

Activity standard deviation:

$$s = \frac{b - a}{6}$$



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

Risk Management Framework

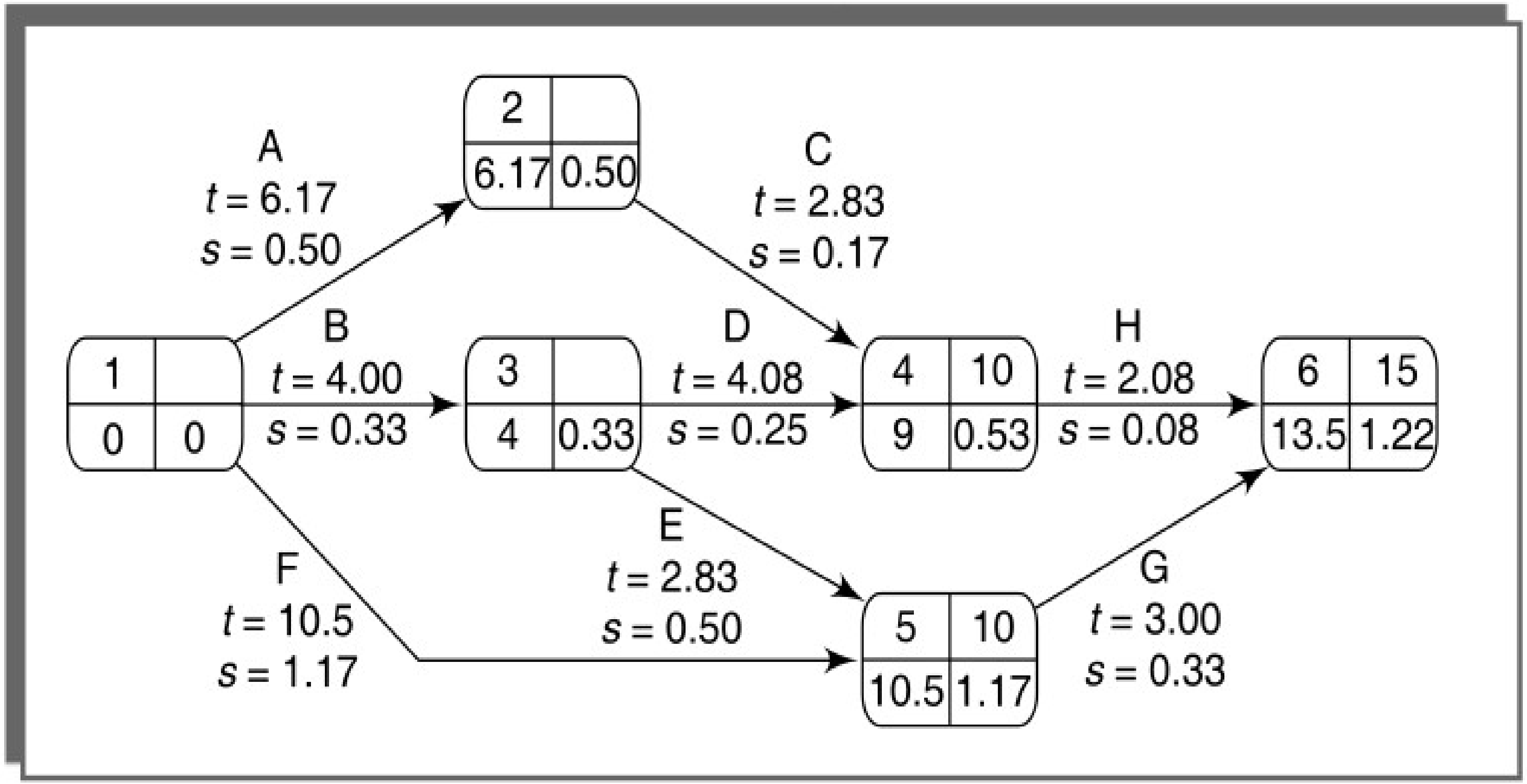
PERT (Program Evaluation Review Technique)

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

Risk Management Framework

PERT (Program Evaluation Review Technique)



Risk Management Framework

PERT (Program Evaluation Review Technique)

Calculate the standard deviation of each project event;

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.

Risk Management Framework

PERT (Program Evaluation Review Technique)

Standard deviation:

For Event 4:

Path A + C has a standard deviation of $\sqrt{(0.50^2 + 0.17^2)} = 0.53$

Path B + D has a standard deviation of $\sqrt{(0.33^2 + 0.25^2)} = 0.41$

For Event 6:

Path 4 + H has a standard deviation of $\sqrt{(0.53^2 + 0.08^2)} = 0.54$

Path 5 + G has a standard deviation of $\sqrt{(1.17^2 + 0.33^2)} = 1.22$

Node 6 therefore has a standard deviation of 1.22.

Risk Management Framework

PERT (Program Evaluation Review Technique)

Calculate 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 5** is $\frac{10 - 10.5}{1.17} = -0.43$

The z value for **event 6** is $\frac{15 - 13.5}{1.22} = 1.23$

Risk Management Framework

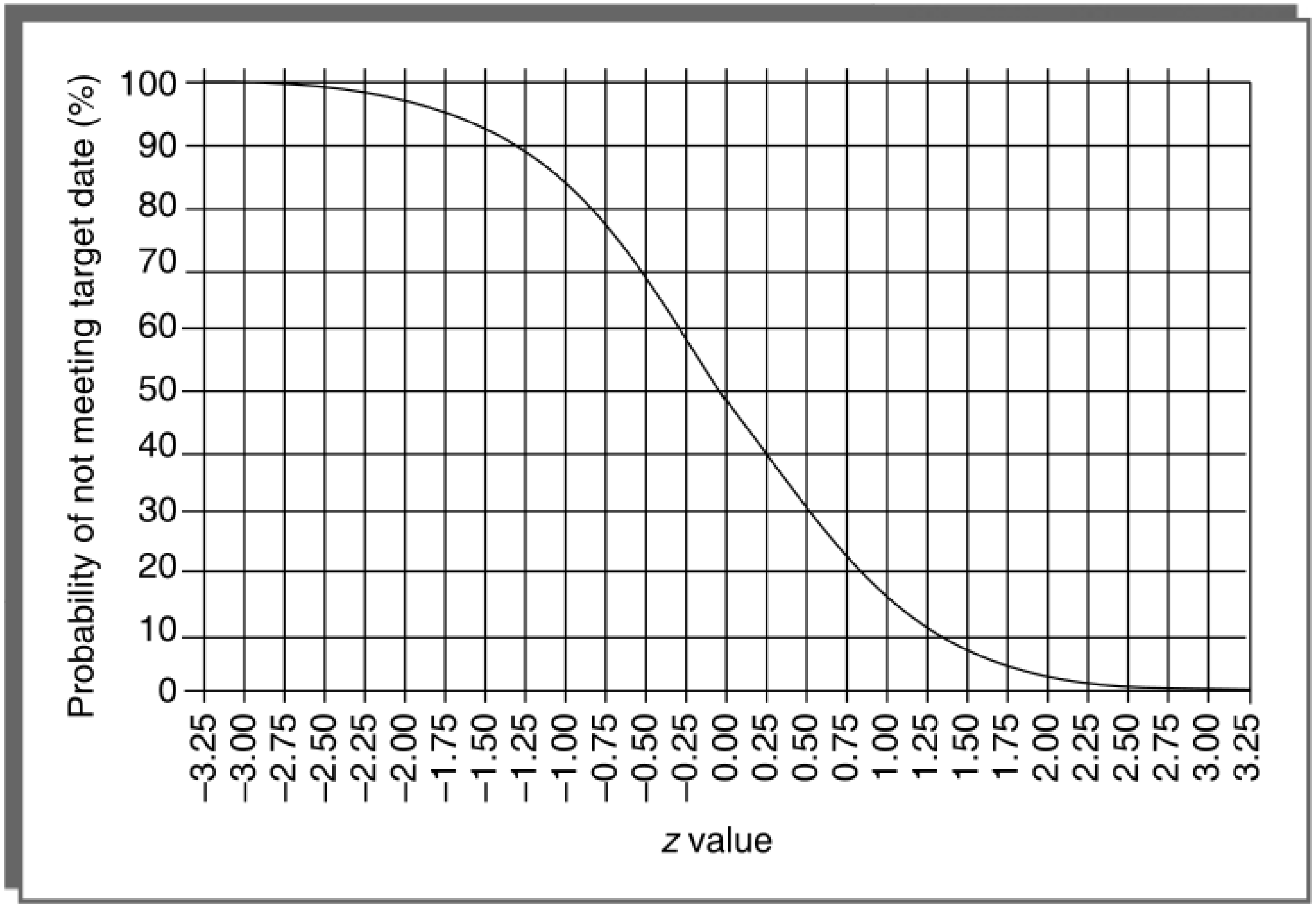
PERT (Program Evaluation Review Technique)

Obtaining Probabilities:

Event 4: The z value is 1.89 which equates to a probability of approximately 3%. There is therefore only a 3% chance that we will not achieve this event by the target date of the end of week 10.

Event 5: The z value is −0.43 which equates to a probability of approximately 67%. There is therefore a 67% chance that we will not achieve this event by the target date of the end of week 10.

Risk Management Framework


Probability of not meeting target date (%)
z value