



T.Y.B.Sc. (I.T)
SEMESTER - V

**SOFTWARE PROJECT
MANAGEMENT**

SUBJECT CODE: USIT501

Dr. Suhas Pednekar

Vice Chancellor
University of Mumbai, Mumbai

Prof. Ravindra D. Kulkarni

Pro Vice-Chancellor,
University of Mumbai

Prof. Prakash Mahanwar

Director,
IDOL, University of Mumbai

Programme Co-ordinator

: Shri Mandar Bhanushe

Head, Faculty of Science and Technology
IDOL, University of Mumbai – 400098.

Course Co-ordinator

: Gouri S.Sawant

Assistant Professor B.Sc.I.T, IDOL,
University of Mumbai- 400098.

Editor

: Dr Sujatha Iyer

Assistant Professor,
Satish Pradhan Dnyanasadhana College,
Thane

Course Writers

: Dr. Veera Talukdar

Principal, Shri Ram College of
Commerce and Science, Bhandup West.

: Dr. M. Krishna Sudha

Assistant Professor, Sri Vasavi College,
Erode.

: Mr Vinay Vilas Shahapurkar

Assistant Professor,
Bunts Sangha's S.M.Shetty College
of Science, Powai,.

: Ms Jyotika D. Chheda

Assistant Professor
Mulund College of Commerce, S. N. Road,
Mulund West

: Ms Pinky Sadashiv Rane

Assistant Professor
New Horizon College of Commerce Airoli

January 2022, Print - 1

Published by : Director,

Institute of Distance and Open Learning,
University of Mumbai,
Vidyanagari, Mumbai - 400 098.

DTP Composed and Printed by:

Mumbai University Press
Vidyanagari, Santacruz (E), Mumbai-400098.

CONTENTS

Unit/Chapter No.	Title	Page No.
Unit 1		
1.	Introduction to Software Project Management.....	01
2.	Project Evaluation and Programme Management.....	26
3.	Introduction to Stepwise Project Planning	53
Unit 2		
4.	Selection of An Appropriate Project Approach	77
5.	Software Prototyping.....	101
6.	Software Effort Estimation	131
Unit 3		
7.	Activity Planning	148
8.	Risk Management	177
9.	Resource Allocation	197
Unit 4		
10.	Architectural Design.....	206
11.	Managing Contracts.....	230
12.	Managing People in Software Environment	246
Unit 5		
13.	Working in Teams.....	263
14.	Verification And Validation	290
15.	Project Closeout.....	318

T.Y.B.SC. (I.T)

SEMESTER -V

**SOFTWARE PROJECT
MANAGEMENT**

SYLLABUS

Unit	Details	Lectures
I	<p>Introduction to Software Project Management:Introduction, Why is Software Project Management Important? What is a Project? Software Projects versus Other Types of Project, Contract Management and Technical Project Management, Activities Covered by Software Project Management, Plans, Methods and Methodologies, Some Ways of Categorizing Software Projects, Project Charter, Stakeholders, Setting Objectives, The Business Case, Project Success and Failure, What is Management? Management Control, Project Management Life Cycle, Traditional versus Modern Project Management Practices.</p> <p>Project Evaluation and Programme Management: Introduction, Business Case, Project Portfolio Management, Evaluation of Individual Projects, Cost–benefit Evaluation Techniques, Risk Evaluation, Programme Management, Managing the Allocation of Resources within Programmes, Strategic Programme Management, Creating a Programme, Aids to Programme Management, Some Reservations about Programme Management, Benefits Management.</p> <p>An Overview of Project Planning:Introduction to Step Wise Project Planning, Step 0: Select Project, Step 1: Identify Project Scope and Objectives, Step 2: Identify Project Infrastructure, Step 3: Analyse Project Characteristics, Step 4: Identify Project Products and Activities, Step 5: Estimate Effort for Each Activity, Step 6: Identify Activity Risks, Step 7: Allocate Resources, Step 8: Review/Publicize Plan, Steps 9 and 10: Execute Plan/Lower Levels of Planning</p>	12
II	<p>Selection of an Appropriate Project Approach:Introduction, Build or Buy? Choosing Methodologies and Technologies, Software Processes and Process Models, Choice of Process Models, Structure versus Speed of Delivery, The Waterfall Model, The Spiral Model, Software Prototyping, Other Ways of Categorizing Prototypes, Incremental Delivery, Atern/Dynamic Systems Development Method, Rapid Application Development, Agile Methods, Extreme Programming (XP), Scrum, Lean Software Development, Managing Iterative Processes, Selecting the Most Appropriate Process Model.</p> <p>Software Effort Estimation:Introduction, Where are the Estimates Done? Problems with Over- and Under-Estimates, The Basis for Software Estimating, Software Effort Estimation Techniques, Bottom-up Estimating, The Top-down Approach and Parametric Models, Expert Judgement, Estimating by Analogy, Albrecht Function Point</p>	12

	Analysis, Function Points Mark II, COSMIC Full Function Points, COCOMO II: A Parametric Productivity Model, Cost Estimation, Staffing Pattern, Effect of Schedule Compression, Capers Jones Estimating Rules of Thumb.	
III	<p>Activity Planning: Introduction, Objectives of Activity Planning, When to Plan, Project Schedules, Projects and Activities, Sequencing and Scheduling Activities, Network Planning Models, Formulating a Network Model, Adding the Time Dimension, The Forward Pass, Backward Pass, Identifying the Critical Path, Activity Float, Shortening the Project Duration, Identifying Critical Activities, Activity-on-Arrow Networks.</p> <p>Risk Management: Introduction, Risk, Categories of Risk, Risk Management Approaches, A Framework for Dealing with Risk, Risk Identification, Risk Assessment, Risk Planning, Risk Management, Evaluating Risks to the Schedule, Boehm's Top 10 Risks and Counter Measures, Applying the PERT Technique, Monte Carlo Simulation, Critical Chain Concepts.</p> <p>Resource Allocation: Introduction, Nature of Resources, Identifying Resource Requirements, Scheduling Resources, Creating Critical Paths, Counting the Cost, Being Specific, Publishing the Resource Schedule, Cost Schedules, Scheduling Sequence.</p>	12
IV	<p>Monitoring and Control: Introduction, Creating the Framework, Collecting the Data, Review, Visualizing Progress, Cost Monitoring, Earned Value Analysis, Prioritizing Monitoring, Getting the Project Back to Target, Change Control, Software Configuration Management (SCM).</p> <p>Managing Contracts: Introduction, Types of Contract, Stages in Contract Placement, Typical Terms of a Contract, Contract Management, Acceptance.</p> <p>Managing People in Software Environments: Introduction, Understanding Behaviour, Organizational Behaviour: A Background, Selecting the Right Person for the Job, Instruction in the Best Methods, Motivation, The Oldham-Hackman Job Characteristics Model, Stress, Stress Management, Health and Safety, Some Ethical and Professional Concerns.</p>	12
V	<p>Working in Teams: Introduction, becoming a Team, Decision Making, Organization and Team Structures, Coordination Dependencies, Dispersed and Virtual Teams, Communication Genres, Communication Plans, Leadership.</p> <p>Software Quality: Introduction, The Place of Software Quality in Project Planning, Importance of Software Quality, Defining Software Quality, Software Quality Models, ISO 9126, Product and Process Metrics, Product versus Process Quality Management, Quality Management Systems, Process Capability Models, Techniques to Help Enhance Software Quality, Testing, Software Reliability, Quality Plans.</p> <p>Project Closeout: Introduction, Reasons for Project Closure, Project Closure Process, Performing a Financial Closure, Project Closeout Report.</p>	12

Books and References:					
Sr. No.	Title	Author/s	Publisher	Edition	Year
1.	Software Project Management	Bob Hughes, Mike Cotterell, Rajib Mall	TMH	6 th	2018
2.	Project Management and Tools & Technologies – An overview	Shailesh Mehta	SPD	1st	2017
3.	Software Project Management	Walker Royce	Pearson		2005

INTRODUCTION TO SOFTWARE PROJECT MANAGEMENT

Unit Structure

- 1.0 Objectives
- 1.1 Introduction
- 1.2 What is a project?
 - 1.2.1 Characteristics of a project
- 1.3 Why is Software Project Management important
 - 1.3.1 Software Project Management
- 1.4 Software Project vs Other Project
- 1.5 Contract Management And Technical Project Management
 - 1.5.1 Common features of contract management and technical project management
 - 1.5.2 Difference between Technical Project Management and Contract Management
- 1.6 Activities Covered by Software Project Management
 - 1.6.1 Project Plans, Methods and Methodologies
- 1.7 Categories of Software Projects
 - 1.7.1 Custom Software Projects
 - 1.7.2 Distributed Computing Projects
 - 1.7.3 Free Software Projects
 - 1.7.4 Software Hosted on Code Plex
- 1.8 Project Charter
 - 1.8.1 Elements of the project charter
 - 1.8.2 Benefits of project charter
- 1.9 Stakeholders
 - 1.9.1 Some Stakeholders in the projects
- 1.10 Setting objectives
 - 1.10.1 Reasons for setting objectives
- 1.11 Business case
 - 1.11.1 Objectives of Business case
- 1.12 Project success and failure
 - 1.12.1 Triple Constraints of Project Management
- 1.13 What is management? Management control
 - 1.13.1 Features of Management
 - 1.13.2 Management Control

- 1.14 Project management life cycles
- 1.15 Traditional V/S Modern Project Management Practices
 - 1.15.1 Traditional Project Management
 - 1.15.2 Advantages of Traditional Project Management
- 1.16 Modern Project Management
 - 1.16.1 Advantages To Modern Project Management
- 1.17 Summary
- 1.18 Questions
- 1.19 Reference

1.0 OBJECTIVES

After completing this unit, you will be able to

- Define software Project Management and its importance.
- Describes the characteristics of software projects.
- Describes projects and its attributes.
- Compares software Project with other projects.
- Differences between project management and contract management.
- Outlines the roles and responsibilities of Project managers.
- Explains in detail Project Management life.
- Categories software Project.
- Explains in details Project Charter.
- Distinguishes between traditional and modern projects.

1.1 INTRODUCTION

Project Management is the discipline of defining and achieving targets while optimizing the use of resources (time, money, people, materials, energy, space, etc) over the course of a project (a set of activities of finite duration). Software Project Management (SPM) is an effort made to develop a non- routine and unique software which will involve a specific timeframe and budget. It will also involve definite specifications and workings of an organization through proper planning. Project management is generally complex as it has multi- disciplinary areas included in various phases of its life span. Software Project Management is a comprehensive management which is planned, implemented, monitored, and controlled throughout all phases of its life to achieve the desired objectives of the developing organizations as well as the organizations that is procuring it. It is an activity which documents, imparts knowledge, takes care of the proper utilization of all the resources and modifies the methods applied by the organization for its development. It is dynamic in nature as the stages, activities and the deliverables keep changing as per the requirements of the stakeholders.

1.2 WHAT IS A PROJECT?

A project is a series of task that is carefully planned to achieve a particular outcome. Project can range from simple to complex and can be managed by either individual or a team. A project is defined as a planned undertaking of series of activities in the beginning to achieve the desired goals. Generally, it is undertaken to explore new and unique methods which will be far better than the old products or services. Its main aim is to accomplish the desired objectives by incorporating interrelated tasks in a time bound manner along with funds and proper utilization of resources. A general definition of Project is: “It is a temporary endeavor with set of well-defined activities that leads achievement of a specific goal(s)”.

1.2.1 Characteristics of a project

A project has the following characteristics.

- Project has specific goal(s)
- It has a definite start date and end date.
- It is not group of routine task or daily task
- Unlike routine activities, project comes to end when its goal(s) is achieved
- Every project requires enough resources in terms of time, skilled workforce, budget, material, and other support

When the project is incorporated with Computer both software and hardware, it creates a network of services and products. It helps to maintain a positive balance between time, attributes, and resources. Project requires proper planning, specific objective, timeframe, and a constrained resource.

1.3 WHY IS SOFTWARE PROJECT MANAGEMENT IMPORTANT?

Unlike machines or buildings, software does not have a physical form, or it is not a tangible product. Today organizations are using software to drive business processes. One can imagine the complexity involved in mapping business process to a software. Also, business process for one organization cannot be same for other, it means requirement of a software for one organization will be different from other. Given the rapid changes in the technology platform as well as globalized but integrated economies induce element of risks in the software already developed or under development. Hence to reduce the risk factor and ensure project delivery will meet stakeholder’s expectations, there is a need to follow structured, process-based approach, which is nothing but software project management.

1.3.1 Software Project Management

It is important due to the following reasons:

- i) It helps to develop a proper schedule along with definite timeline.
- ii) It helps to control the project resources and the environmental risks that occur during the Project development process.
- iii) It is needed to manage and control the scope of the business
- iv) It helps to develop Project with definite timeline and costing associated with the project. It could be increased or decreased as per the needs of the project.
- v) It motivates the project team to remain focused and develops team spirit
- vi) It can alter plans as and when needed by the client or needed in the project
- vii) It also helps to communicate to the stakeholders about the progress and state of the project and seek their opinions for further development
- viii) It helps the project team to be prepared for any unforeseen issues that might arise due to some presumptions that was made in the planning stage of Project
- ix) As the project team had collected inputs of the project from various areas hence, they are able to develop a critical path for the successful completion of the project
- x) In the end, the project report ensures that the knowledge and experiences are properly stored for future usage.

1.4 SOFTWARE PROJECT VS OTHER PROJECT

Most of the applications of Project Management are used in Software Project Management but the products of software projects have number of unique characteristics that make it different from others. Some of the features are as follows

- 1) Flexibility- The software used in the software Project Management helps to make necessary changes as and when needed both by the client and the operational team. It is one of its most effective strengths
- 2) invisibility- The software is not able to physically show the progress of the work. As a result, it is very difficult to quantify the progress

of the work. But in other physical projects, the actual progress can be quantified.

- 3) complexity -Software Projects are generally more complex than other types of projects due to various reasons like programming used in it, the cost associated with it keeps varying with time due to upgrading of software, etc.
- 4) Conformity -Software engineers have to design software as per the demands of the clients but in other projects the engineers have to abide to the laws of the land.
- 5) Technology- In Software Project, technological exchanges are high as the software's are easy to copy and can be hacked. On the other hand, technological exchanges in physical projects are low as they cannot be copied.

1.5 CONTRACT MANAGEMENT AND TECHNICAL PROJECT MANAGEMENT

Generally, Projects in various industries are of 2 types i] In House Projects and ii] Out House Projects. When the industries/ companies use their own software or develop their own personalized software within their organization than it is called as "In House Projects". On the other hand, when the industries / companies contact software developers outside the organization for their personal use by making a contract between the two organizations, than they are called "Out House Projects". In "Out House Projects" the client organization will appoint a manger for monitoring and reviewing the contract and is called the Project Manager. His primarily duty is to closely monitor the development of the project as per the contract, take technical decisions as and when required, to keep the project within the budget and lastly should try to adhere to the timeline. Again, in "Out House Projects" the supplier organization will also appoint a manger who is known as the Technical Project Manager and will specifically look into the technical needs of the client organizations.

1.5.1 Common features of contract management and technical project management

Some of the common features of contract management and technical project management are as follows: -

- a) stakeholders are involved in both.
- b) Team from both the clients and suppliers are involved for accomplishing the project.
- c) They generally evolve out of need and requirements from both the clients and suppliers.
- d) They are interdependent on each other.

e) Standard protocols are maintained by both the clients and suppliers.

1.5.2 Difference between Technical Project Management and Contract Management:-

Some of the differences of contract management and technical project management are as follows: -

S. No	Contract Management	Technical Project Management
1.	It is a part of the procurement function where it confirms that terms and conditions mentioned in the contract are properly adhered too.	It is all about managing all the aspects of the project from planning till completion within the constraints of the main project
2.	They are generally responsible for delivering the said project within the budget and in the time	They are generally responsible to check the progress of the project, its validity as per the need of the organization and timeline
3.	They solely focus on the contract which is a bond between the suppliers and the clients	They generally meet the suppliers on a regular basis in order to emphasis their needs and demands
4.	Their primary duty is to conduct research, do risk analysis and negotiate on the terms of contract.	Their primary duty is of proper documentations which includes from proper planning till completion along with timelines and budgetary areas.
5.	They are more involved in working closely with the clients to help them to understand the paper works that the clients had signed	They are involved in meeting the deadlines of their projects with specification, budget, and timeframes in mind

1.6 ACTIVITIES COVERED BY SOFTWARE PROJECT MANAGEMENT

Software Project Management not only includes development process but are associated with several activities which are primarily required to develop the whole project. There are 5 stages involved in Software Project Management. They are as follows: -

1] Stage 1 – Project Initiation

This is the 1st and the most important step in Software Project Management as it lays down the foundation for the development of the project. This step marks the beginning of the project, and it starts

by studying the needs of the business along with resource allocation, time and money involved in the project. In other words, we can say that the Feasibility study of the projects carried out by gathering valuable information from different sources. In this step, the value of the project is also assessed keeping in mind the main purpose or goal of the project. The main objectives of the project should be very clear as it helps in finalization of the methods to be applied in the project in future to achieve the desired goals. These also play a very vital role in evaluating the success of the project towards the end. As the business requirements are the direct consequence to be achieved in the end, hence in this stage the project manager is made thoroughly aware of the business needs which is kept in the mind in all stages of the project. In this stage, the Project Charter is also prepared by the organization. Project Charter is an official document which authorizes the project manager to handle the project on behalf of the company. Anyone who takes part in the project both in present and in future will also be a part of the project initiation stage as they will be easily identified by the managers as stakeholders. As the business needs are directly related with the goals and objectives of the organization, hence the project manager should also properly understand the objectives of the business needs that lead to the development of the project and will ultimately meet the requirements of the stakeholders.

2] **Stage 2 – Project Planning**

After the feasibility study is found viable for the organization in the 1st stage, planning is done. As the project is new to the organization and has been undertaken for the first time, hence careful planning is needed. It involves identifications of number of activities, milestones, and project deliverables. It should be done by keeping in mind the requirements of the business along with its size and complexities. In this stage, Detail Project Report (DPR) is prepared by incorporating the methodologies to be applied in different phases of the project along with the timelines and the finances involved in it. Planning is an iterative and never-ending process as it might require changes in budget, timeframe, requirements of the stakeholders, etc due to various reasons like natural calamities, change in government rules, etc. It is at this point that the project manager has to use his experience and technical knowledge in developing a sound and realistic project plan. He needs to focus on the goals of the project and narrow down on the project descriptions to achieve the desired results. It is here that the Project Charter which is an official document is prepared in accordance with the mission and vision of the company. This document will give the minute details of the project along with the deadlines and milestones to be achieved in the project. The Project Charter acts a guidebook for the project manager and clearly describes the goals to be achieved in each stage of the project as well as the whole project. It clearly states the definition of the project, its characteristics, the end results, and the project authorities. It is the final official document

handed over to the project manager for the commencement of the project.

3] **Stage 3– Project Execution and Control**

After the completion of the planning, it is time for the execution of the plan to produce estimated product in time and in good quality. Generally, there are 2 types of methods, and they are as follows: -

- i) results in the designing and production of goods or services and plays a vital role in the execution of the project. It is the external appearance of the project which is the user interface of the internal architecture.
- ii) results in the formation of supporting processes to be implemented. The core attributes of the supporting process are risk management, quality assurance, team management, etc. and plan an equal role in project execution. It is also considered to be the code which is required for the implementation of the project.

The execution phase is a part of all project management and is the most physically active and prominent phase of the project. The second part of this stage is Controlling, and its main objective is to quantify and manage the project activities in order to guarantee that they are on the right path to achieve their desired goals. There are 5 Variables of Project Control, and they are 1. Time 2. Cost 3. Quality 4. Scope 5. Risk The controlling process also adheres to the scope, budget, schedule, and quality constraints of the project. It identifies any deviations if found, from the plan and proposes the corrective measures that can be taken to rectify the deviations. Though it is present in all the stages of the project, but it has more importance during this stage.

4] **Stage 4 – Project Closing**

The main aim of the project closure is to ensure that the project has reached its rational level where the project can be said to be completed. In project management, project management and project closure are a formal written assessment of a project. It documents all stages of project management into a feasible report. Through introspection, a project manager learns what worked and what didn't. It is marked by the contract made between the administrative department of the organization and the developers of the project. The project teams ensure that the objective of the project is achieved and is as per the requirements of the company. They also must make sure that the product of the project is completed per the terms and conditions stated in the contract. It's also the most helpful learning tool for teams who hope to plan their next project based on past insights. Once the contract closure has been initiated then the full and final payment of the suppliers to the project must be settled. This is often followed by the administrative closure where they document all the project activities and experiences for future use. A project closure report also shows proof that the project team

delivered what they promised they would in the beginning. A project closure report is the number one way to determine whether a project was successful.

5] **Stage 5– Project Evaluation**

The main objectives of this phase are divided into 3 parts. First part is the **generalized** where all the above four phases are being assessed. This is done by the project team along with the project manager as their leader. They try to analyze both the positive and the negative outcomes of the project along with the experiences gained. They also try to incorporate the best practices that had evolved during the project in their day-to-day affairs. The second part of this project is **individual specific** where the individual performances in various stages of the project are being appraised. Third and not the least, the **third part review** must be conducted by a neutral organization to evaluate the outcome of the project along with the performances of the project manager and his team members as an individual and as a team in achieving the objectives of the project that they were handling. They also review on various other aspects like the quality of the project delivered, customer satisfaction, ethical practices used in the project, team work etc. Towards the end of this stage, it determines whether the project delivered was as per the need of the organization and gave the value of money, time and the resources that were involved in the project.

1.6.1 Project Plans, Methods, And Methodologies

Before the execution of the actual production, proper planning has to be done. It is a dynamic activity and starts from the initial stage of the project and continues till the product is delivered. They has constantly been reviewed and revised as per the latest updates. It involves making sets of plans that guides the project manager and his team members in managing resources, time, cost, risks, etc. Generally, it includes the following: -

- i. generation of the requirements
- ii. analyze the necessities
- iii. design the pilot cases
- iv. design the model
- v. develop the codes
- vi. assess the codes
- vii. compare the actual and the expected outcomes of the project
- viii. installation of the project
- ix. maintenance of the project

This part is followed by a set of activities that has been demarcated in the plan. The plan translates the methods into activities to achieve the desired objectives of the organization. Hence the method includes the following: -

- i. starting and ending date of the project
- ii. division of responsibility in the execution of the project

- iii. list of requirements including software and the resources needed in the project

Methodology is a set of methods that are to follow in a sequential manner to meet with success in the project. It is generally seen that all the methods are interlinked with each other. As a result, the output of one method is the input for the next method. So, by grouping all such methods and techniques, we form the methodology.

1.7 CATEGORIES OF SOFTWARE PROJECTS: -

Software projects are categorized into 4 categories. They are as follows: -

1.7.1 Custom Software Projects - In this, the software application is developed as per the specific requirements of the users or the organization. This software meets the specific needs of the customers and is not like the traditional and off the shelf software. This software is customized either by a third party after a contract is signed between the customer and the client or by the in-house research and development team of the organization. Custom Software are tailored as per the needs of the single entity and would only be used by that single entity.

1.7.2 Distributed Computing Projects - A distributed computing system consists of multiple software components that are on multiple computers but runs as a single system. The computers that are in a distributed system can be physically close together and connected by a local network, or they can be geographically distant and connected by a Wide Area Network. A distributed system permits resource sharing, including software by systems linked to the network. Some of the examples of the distributed systems are Intranets, Internets, www, emails, etc.

1.7.3 Free Software Projects – Free software is software that can be freely used, modified, and reallocated with only one constraint. Any redistributed version of the software must be distributed with the original terms of free use, modification, and distribution. In other words, the user has the liberty to copy, run, download, distribute and do anything for his upgradation. Thus, this software gives the liberty without money and the users can regulate the programs as per their needs.

1.7.4 Software Hosted on Code Plex – Code Plex is Microsoft’s open-source project hosting website. Code Plex is a site for managing open-source software projects, but most of those projects are leniently licensed, commonly written in C# and the building blocks for own open-source project using advanced GUI control libraries. The great thing about permissively licensed building blocks is that one doesn’t have to worry about the project being sucked into GPL if one decides to close the source. Because Code Plex is based on Team Foundation Server, it also provides enterprise bug tracking and build management for open-source project, which is far better than the services provided by Source Forge.

1.8 PROJECT CHARTER

A project charter is a formal, characteristically brief document that describes the project in its wholeness — including what the objectives are, how it will be carried out, and who all are the stakeholders in the project. It is a critical component in planning the project because it is used throughout the project life cycle. A project charter explains the project in clear, concise wording for high level management. Project charters summarize the entirety of projects to support teams rapidly comprehend the goals, tasks, timelines, and stakeholders. The document provides key information about a project and provides approval to start the project. Therefore, it serves as a formal announcement that a new approved project is about to commence. Contained also in the project charter is the appointment of the project manager, the person who is overall responsible for the project. The project charter is a final official document that is prepared in accordance with the mission and visions of the company along with the deadlines and the milestones to be achieved in the project. It acts as a road map for the project manager and clearly describes the objectives that has to be achieved in the project. The project charter clearly defines the projects, its attributes, the end results, and the project authorities who will be handling the project. The project charter along with the project plan provide strategic plans for the implementation of the projects. It is also the green signal for the project manager to commence the project.

1.8.1 Elements of the project charter

In a nutshell, the elements of the project charter which serves the following are: -

- i. Reasons for the project
- ii. Objectives and constraints of the project
- iii. The main stakeholders
- iv. Risks identified
- v. Benefits of the project
- vi. General overview of the budget

1.8.2 Benefits of project charter

Some of the benefits of project charter are as follows: -

- i. It improves the customer relationship
- ii. It improves project management methods
- iii. Expands and enhances the regional and headquarters communications
- iv. Supports in gaining project funding
- v. It recognizes senior management roles and authorities
- vi. Allows development, which is meant at achieving industry best practices

1.9 STAKEHOLDERS

Project stakeholder are entities with an interest in certain project. These stakeholders could probably be inside or outside a company which: sponsor a project, or don't mind spending time or a gain upon a very good completion of a project and might have a positive or negative impact inside the project completion. Samples of project stakeholder include in customer, the person group, the challenge manager, the improvement team, the testers, etc. as a result, it very important that the project manager establishes a sound communication among all the stakeholders and achieve the project requirements for them. They might be a single person or an organization in totality. The stakeholders will be able to deliver important inputs to the projects and it will not only improve the quality of the project but will also help the project to meet with success. Moreover, all the stakeholders do not contribute positively to the project while others have major influences on the outcome of the projects.

1.9.1 Some Stakeholders in the projects are: -

- 1) the client or the end user are the persons who will be using the project
- 2) project team or the persons involved in the development of the project
- 3) project authority / project in charge / project sponsor is the person who had the authority to sanctioned funds and resources and signs the charter on behalf of the organization.
- 4) Project manager is the person who is solely responsible for the execution of the project
- 5) Business partners like the suppliers, customers and the vendors
- 6) Functional managers are the persons who are responsible in different departments of the organization and contribute for the development of the projects.

1.10 SETTING OBJECTIVES

Objectives are defined as some goals that is realistic and achievable and should not be imaginary. Project objectives are the goals that one wants to achieve at the end of the project. It will include deliverables and assets along with more intangible objectives like increase in productivity and motivation. Project objectives should be designed in such a way so that they can be attainable, and time bound along with specific goals which can be measurable at the end of the project.

The objectives of the project should be clearly defined and thoroughly known to the project manager as he will be the main person responsible for the success or failure of the project. Any amount of ambiguity in understanding the objectives and purpose of the project would lead to

disastrous consequences. Clarity on the goals of the project will not only help the organization in getting success but will also help to save time, money and resources. As a result, the 1st step in project management is to understand the motive of the organization behind developing the software project. Once the objectives are clear to the project manager than he can plan accordingly to achieve it keeping in mind the time, money and resources allocated to him for the completion of the project. In brief, **project management objectives** are the successful development of the project's events of initiation, planning, execution, regulation and closure as well as the supervision of the project team's operations towards accomplishing all the agreed upon goals within the set scope, time, quality and budget standards.

1.10.1 Reasons for setting objectives

Some of the reasons for setting objectives are as follows:-

- **The successful development and implementation of all project's procedures.** A project, regardless of its size, generally involves five distinctive project life cycle phases of equal importance: Initiation, Planning and Design, Construction and Execution, Monitoring and Control, Completion. The smooth and uninterrupted development and execution of all the above phases guarantees the success of a project.
- **Productive guidance, efficient communication and apt supervision of the project's team.** Always keep in mind that the success or failure of a project is extremely dependent on teamwork, thus, the key to success is always in project association. To this end, the establishment of good communication is of major importance. On one hand, information needs to be articulated in a clear, unambiguous, and complete way, so everything is comprehended fully by everyone and on the other hand, is the ability to be able listen and receive constructive feedback.
- **The achievement of the project's main goal within the given limitations.** The most important project limitations are, **Scope** in that the main goal of the project is completed within the estimated **Time**, while being of the expected **Quality** and within the estimated **Budget**. Staying within the decided limitations always feeds back into the measurement of a project's performance and accomplishment.
- **Optimization of the allocated necessary inputs** and their application to meeting the project's pre-defined objectives, is a matter where is always space for improvement. All processes and procedures can be reformed and upgraded to enhance the sustainability of a project and to lead the team through the strategic change process.
- **Construction of a complete project which follows the client's exclusive requirements and objectives.** This might mean that you need to shape and reform the client's vision or to negotiate with

them as regards the **project's objectives**, to modify them into feasible goals. Once the client's aims are clearly defined, they usually impact on all decisions made by the project's stakeholders. Meeting the client's hopes and keeping them happy not only leads to a successful collaboration which might help to eliminate surprises during project execution, but also ensures the sustainability of your professional status in the future.

1.11 BUSINESS CASE

Business case states the reasons to adopt the project charter. It provides the justification for undertaking a project, programmed or portfolio. It is very similar to an investment proposal. It evaluates the benefit, cost and risk of alternative options and provides a rationale for the preferred solution. A business case is a way to prove to your client, customer, or stakeholder that the product you are developing is worth the investment. The need for a business case is that it collects the proposal, outline, strategy and marketing plan in one document and offers a full look at how the project will benefit the organization. But one can also proceed without business case in project planning as it is very similar to it. It is a document that provides the top management with all the necessary information needed to select the project that is to be funded. It is generally built on the significance of the business goals and objectives. It also considers the cost of the solution, breakeven point, return on investment and the maintenance cost. Business case handles both the qualitative and quantitative issues in the project. Moreover, the developer of the business case must present convincing facts and figures in favor of the project. A decent business case should contain the following: -

- i) Detail project report along with possible impacts, costs and benefits
- ii) Include all the necessary information's related to the project
- iii) Should be clear and logical in comparing the cost benefits impact on alternative project
- iv) Systematically summarizing all the findings

1.11.1 Objectives of Business case

The objective of the business case is to evaluate and advocate the utilization of information technology to improve the efficiency and effectiveness of the organization. Information Technology are generally undertaken for various reasons like improving customer satisfaction, reducing costs, improving communications, integrating customers, etc. with the underlying objectives of achieving organizational goals. There are number of steps involved in developing the business case and they are as follows: -

- i) **Formation of the project team** – a teamwork is required to develop a business case and the team includes the stakeholders, users, project team and IT experts. This team is formed with the intention to

exchange knowledge, experience and the information in order to develop the software project. As the stakeholders, are the primary ones who will be affected by the project, so their views points should be properly represented in the business case. There are several advantages of having a team develop the business case and they are as credibility, alignment with the organizational goals and access to real costs. As the team comprises of people from different areas of the organization, hence it helps the project manager to overcome all the resistance that he might face in the development of the project.

- ii) **Developing Measurable Organizational Value (MOV)** – in IT projects, the success of the project is assessed through Measurable Organizational Value (MOV). For any project to be successful, the MOV should align with the organization’s mission, objectives and goals. A transparent MOV helps the team to know the road map of the project along with the life cycle of the project. There are certain steps that are to be followed in developing MOV and they are-
 - a) Identifying the desired area of impact
 - b) Identifying the desired value of the project
 - c) Developing an appropriate matrix
 - d) Setting a time frame for achieving the MOV
 - e) Verifying with stakeholders
 - f) Summarize the MOV in a clear and concise statement.
- iii) **Identifying Alternatives** – Alternative solutions to problems and opportunities should be properly accounted in the business case. These alternatives should also enable the company to reach the desired MOV. The business case should put out convincing reasons to bring about any changes and the cost that would be associated with the changes.
- iv) **Defining Feasibility and Assess Risk** – Actually feasibility is the probability of effectively applying an alternative while the risk focuses on what can go against or what is right. The feasibility and the risk associated with each alternative solution should be properly analyzed and assessed. These will help the project manager to identify alternatives that are not worth pursuing.
- v) **Defining Total Cost of Ownership** – total cost of ownership is over and above the cost of purchasing or developing the application. The total cost of ownership of the application needs to be accounted for before any decision is taken on implementing it. As a result, the calculation of total cost of ownership is complex and the project manager must validate the calculation with data sources, expectations and methods for arriving at the cost.
- vi) **Defining Total Benefits of Ownership** - Total Benefits of Ownership comprises of direct and indirect profits that are linked

with each alternative. Profits should be in terms of increased efficiency, improved productivity, improved customer service, improved accuracy, etc. Each alternative has certain tangible and intangible benefits associated with it. The tangible benefits are easy to identify and quantify but intangible benefits are not easily quantified.

- vii) **Analyzing Alternatives** – once the cost and the profits have been recognized than starts the comparisons with the alternative to zero in the alternative that best suits the requirements of the project. Different financial models are being applied to get the desired results.
- viii) **Recommended solutions** – after analyzing each alternative the most suited one is recommended to the customer for the approval.

1.12 PROJECT SUCCESS AND FAILURE

There are certain factors that make a project successful. They are timely delivery of the project. the project developed should be reliable, it should meet the expectations of the client, should be within the budget, the product should be high performing, it should be maintainable and enhance able.

On the other hand, there are some factors that are associated with the failure of the project, and they are as unrealistic projects, inadequate planning, insufficient risk management, poor communication, poor understanding of the objectives, complexities of the project, market competition and many such reasons.

1.12.1 Triple Constraints of Project Management

The three major constraints of project management are :-

- I. Delivering on time
- II. Delivering within allotted budget
- III. Ensuring adherence to scope of the project

1.13 WHAT IS MANAGEMENT? MANAGEMENT CONTROL

Management is the coordination and administration of tasks to achieve a goal. Such administration activities include setting the organization's strategy and coordinating the efforts of staff to accomplish these objectives through the application of available resources. Management can also refer to the seniority structure of staff members within an organization. Management defined as all the activities and tasks undertaken for achieving goals by continuous activities like; planning, organizing, leading and controlling. It is one of the latest areas that has been added to modern day business. As time and requirements are constantly changing very fast, so the role of the managers is also changing. Due to the development of management science, the modern-day

managers are more equipped to meet the challenges of dynamic business environment. In other words, management is an art of getting people together to achieve the desired goals of the business organization. Management is important in all works of our life. **Management is a process of planning, decision making, organizing, leading, motivation and controlling the human resources, financial, physical, and information resources of an organization to reach its goals efficiently and effectively.**

1.13.1 Features of Management:–

Management is the process of setting and reaching goals effectively and efficiently. Management process has some qualities or features.

1. Management is Associated with Group Efforts
2. Management is Purposeful
3. Management is Accomplished Through the Efforts of Others
4. Management is Goal-oriented
5. Management is Indispensable
6. Management is Intangible
7. Management can Ensure Better Life

1.13.2 Management Control:-

is a function of management which supports to check errors to take remedial actions. This is done to curtail deviation from values and safeguard that the definite goals of the organization are attained in a chosen manner. According to modern notions, control is a foreseeing action. Earlier concepts of control were only used when errors were detected. Control in management includes setting standards, measuring actual performance, and taking corrective action in decision making. Control procedures provide managers with the type and amount of information they need to quantify and display the performance. The information from various controls must be as per the needs of the management, departments, or units of operations.

1.14 PROJECT MANAGEMENT LIFE CYCLES

A project management life cycle is a framework encompassing of a set of different high-level stages essential to transform an idea of concept into reality in an orderly and efficient manner. Projects are temporary activities of the organization and have definite aims. They do not form the core activity of the organization. Project life cycles is also used in all works of life. Like in aerospace, government offices, hospitals, hotel, etc. The Project Lifecycles defines the different rational stages in the life of a project, and it starts from the incorporation of the project till the end. The project life cycle structure commonly displays the following characteristics:

- i. In the beginning, cost and staffing stages are low and reach a ultimate level when the efforts are in progress. It again starts to drop speedily as the project begins to halt.
- ii. The typical cost and staffing curve does not spread over to all projects. Substantial expenditures protect the essential resources early in its life cycle.
- iii. Risk and uncertainty are at their heights at the start of the project. These features drop down as the life cycle of the project progresses, decisions are reached, and deliverables are acknowledged.
- iv. The capability to touch the final product of the project without affecting the cost considerably is at its upper most level at the start of the project and decreases as the project progresses towards conclusion. It is indistinct that the cost of constituting new changes and mending errors upsurges the cost of the project as one approaches towards completion.

The project is fragmented into different stages in order to ensure that the project can run smooth and efficiently. By breaking down the projects into various stages, the activities get arranged in a sequential manner and the risk factor also gets reduced. The stages are arranged in such a manner that each stage of the project provides one or more deliverables which are tangible in nature and can be verified. The deliverables at the end of each stage helps the project manager to evaluate the outcome of that stage and take necessary actions as and when required. Though it is said that the stages of project life cycle are linear in sequence but sometimes they overlap to save time, but it is risky to undertake such activities. **There are 6 Phases of Project Management Life Cycle, and they are**

1. Project Initiation

Project initiation is the first stage in Project Management life cycle, where the project starts rolling. It offers a summary of the project, along with the tactics which are essential to achieve the desired results. In this stage, the feasibility and business value of the project are determined.

The project manager starts with a meeting in order to understand the client and stakeholders' requirements, goals, and objectives. It is important to study minute specifications and requirements in order to have a better understanding of the project. once a decision is made to proceed, the project can move on to the next step which is creation of a project team. The Project Charter is measured to be the most significant document of any project.

- i. **Undertake a Feasibility Study** - In the initial stage, it is vital to recognize the feasibility of the project. It is also important to understand the viability from the economic, legal, operational, and technical aspects.

- ii. **Identify the Project Scope** – here the project scope is identified, and it comprises of defining the length, breadth, and depth of the project. On the other hand, it's equally important to plan functions, deadlines, tasks, features, and services.
- iii. **Identify the Project Deliverable**—after identifying the project scope, the next phase is to plan the project deliverables. They include defining the product or services required.
- iv. **Identification of Project Stakeholders** - identification of project stakeholders is important. Meetings among team members and experts helps to identify project stakeholders. Documentation of related information on stakeholders is important for successful completion of the project.
- v. **Develop a Business Case** - Before developing a business case, one should check whether the vital pillars of the project such as feasibility, scope, and identification of stakeholders are in place. The next phase is to come up with a complete business case. After the formation of a statement of work (SoW) and the formation of a team, the project initiation phase comes to an end.

2. Project Planning –

A lot of planning is associated with the project in this phase. On identifying the project objectives, it is time to develop a project plan which could be followed by all. The planning phase decides a set of plans which will guide the team in implementing the phase and thereafter closing it. The program assembled here will surely help you to manage cost, quality, risk, changes, and time. The project plan established should comprise all the important facts associated with the project goals and objectives. It is the most composite phase in which project managers take care of operational requirements, design limitations, and functional requirements.

The project planning stage comprises the following mechanisms:

- i. **Generating a Project Plan** - A project plan is a design of the whole project. An elegant project plan controls the activities, the time frame, dependencies, restrictions involved, and probable risks. It helps the project manager to rationalize the operations.
- ii. **Generating a Resource Plan** - The resource plan delivers information about numerous resource stages essential to achieve a project. Resources used should have applicable Project Management expertise.
- iii. **Budget Estimation** - financial plan benefits one to make the budget and bring project deliverables without surpassing it. The final budget plan states the expenses on material, labour, and equipment. Making a budget plan will aid the team and the project managers to monitor and control the finances throughout the Project life cycle.

- iv. **Collecting Resources** - collecting resources is an important part of project planning and assists to monitor the quality stage of the project. Resources like equipment, money, software solutions, and the workplace should be given to complete the work.
- v. **Forestalling Risks and Potential Quality Barriers** - The risk plan will assist in identifying risks and lessen them. It comprises all the probable risks, the level of severity, and preventive actions to curb it.

3. Project Execution

Project execution is the stage where the execution processes are applied, works and resources are allocated. The technique includes constructing deliverables and satisfying customer needs. Project managers or team leaders achieve the job through resource sharing and by keeping the team members focused. The team starts generating project deliverables and pursue to attain project goals and objectives. The final deliverable of the project takes form during the project execution phase.

- i. **Writing Growth of a Project** - During the project execution phase, it is important to get steady project information as it delivers the essential information and even recognizes the problems.
- ii. **Conduct Regular Meetings** - Before starting a project meeting, the agenda should be clear to all the team members. Proper communication should be done on time.
- iii. **Accomplish Problems** - Problems inside the project are certain to occur. Problems like time management, quality management, weakening in team spirit can hamper the success of a project. So, make sure all problematics issues are solved in the beginning itself.

4. Project Monitoring and Control

The **Project Monitoring and Control** is all about measuring the performance of the project and chasing the development. It is applied during the execution phase and the main goal of this phase is to align with the plan especially related to financial constraints and timelines. It is the accountability of the project manager to make essential modifications connected to resource allocation and guarantee that all the things are on track. Monitoring project after the project execution phase will allow the project manager to take remedial measures.

5. Project Closure

The final phase of the Project Management life cycle phases is similarly significant as all other phases. This phase signifies the final phase of the Project Management life cycle, which is also recognized as the “follow-up” phase. During this time, the ultimate product is completely ready for delivery. Here the project manager and his team focus on product release and product delivery. During this phase, all the happenings associated with the project are wrapped up. The closure phase is not necessarily after a successful completion phase alone but also after the project meets with failures. After the project is completed, it is timely delivery to clients and

it highlights the strengths, identifies the ambiguities and recommends how they could be corrected for future projects.

6. Project Evaluation

It is not possible to immediately evaluate the real value of the software project after its completion as the goals might be long time. By simply appraising the success accomplished in implementing the hardware and the software peripherals of the projects does not amount to having succeeded in the project. The documents produced in the project evaluation stage are very useful for use in the future projects that the organizations might undertake later on. Evaluation of the project team along with the project manager is also carried out in this stage.

1.15 TRADITIONAL V/S MODERN PROJECT MANAGEMENT PRACTICES

The pace of technology change in IT projects makes it difficult for Project Managers to ensure that the project they are implementing is relevant by the time they are done. Most of the methods and methodologies such as “waterfall” assume that every requirement of the project can be identified before any designing or coding occurs.

On the other side, would it not be more pragmatic that the stakeholders describe their vision to the development team and the team development functional software. To overcome these limitations of rational methodologies the agile methodology was introduced. The agile methodology helps the team respond to unpredictability through incremental, iterative work known as sprints.

1.15.1 Traditional Project Management: -

With the fast development in Information Technology, it becomes very difficult for the project managers to keep pace with the changes. In traditional Software practices like the "Waterfall" method, it wants one job to be completed before the start of the next one. Detail Plans are made before the beginning of the project date. Moreover, the successive stages are also plotted out initial to deliver clarity on the work that should be accomplished to reach the desired goal. Many businesses organizations still use it for projects having a fixed budget or deadline.

1.15.2 Advantages of Traditional Project Management are: -

1. **No surprises** - This approach allows hardly any space for flexibility or changes once the project starts rolling. In the beginning itself, the plan is placed before the management and decided by them in the subsequent meeting. This leaves very little space for readjustment and the unintended scope creep is decreased. As both the parties approve on the project timeline and jobs, it offers clarity on the development and assigns responsibilities to everyone in the initial stages only so that they are aware of their duties.

2. **Smooth knowledge transfer** – Elaborate documentation is the most important part of the waterfall methodology. When the information is always easily available, it becomes easier for new team members to adopt quickly. Moreover, the information will not be lost when an employee switches over to another company.
3. **Sets potentials both internally and externally** – much of time is consumed in placing together a complete project timeline for the client to assess. A main advantage is that the client is aware in the initial stage on what to expect and can plan consequently. Moreover, very slight participation is needed after the initial phase, and they have sufficient time to gather the assets they need for a specific phase. Internally, team members can design their time in a better manner which comes in handy when occupied on multiple projects simultaneously.

1.16 MODERN PROJECT MANAGEMENT

Modern project management leverages automated tools to help plan, execute, and organize work. It's also viewed as the more flexible method of the two. More professional service businesses are taking on short-term or even one-time projects, so businesses are looking for alternative to the traditional project management method. This is where the modern project management method flourishes - in a fast-paced environment that can handle mid-project changes swiftly and efficiently.

1.16.1 Advantages to Modern Project Management are:

1. **Juggle more projects at once -**

Instead of having all of your tasks fully outlined at the start of each project (as they are in true, With the use of smart technology, one can create a added flexible technique that will permit one to start the project without having a broad idea of the end result. By doing so, one can readily make modification to the project as per the desires and needs of a client change without returning to the start time.

2. **Minimize risk and human error**

As a smart platform offers enlarged visibility of the project team and the project itself, so each step of the project can be monitored, and remedial steps may be added as and when required and report it much before any actual damage is done to the client relationships. Use of accurate smart platform will inevitably save the client data, calculate the accounts, and update the timeline to the client automatically. This will reduce the manual work, will save time and will make less mistakes.

3. **Be more flexible with time**

In traditional approach to project management, a fixed amount of time is allocated to complete the job. But while engaging smart automated software that logs time and tracks utilization, this will not be a problem. It is so as we will be able to track our project in real-

time bases and reassign jobs as and when needed. One can work remotely on the projects with a correct cloud-based platform which will allow one to share files with team members and communicate with clients from one interactive dashboard and integrated the system- so that one does not have to extend a deadline as cannot go to office.

Selecting the correct project management style is vital to the business. Understanding the advantages and disadvantages of traditional and modern project management is crucial before selecting which is good for you and your team.

1.17 SUMMARY

In this chapter we have seen that software in project management is dedicated to the planning, scheduling, resource allocation, execution, tracking and delivery of software and web projects. Some characteristics of successful software projects are a clear and realistic goal, powerful team leadership, sense of ownership, commitment to quality, getting things done, etc. A project can be defined as a temporary effort to accomplish a unique product, services or results. Some key attributes of a project is it should have a start and finish point, a project should have a fixed budget which is capitalized, a project seeks to make instant changes or benefits and many more. A software project is more complex than any other engineered artifacts. The complexity of a software project cannot be measured until we work on it. The role of a software project manager is to work with cross-functional teams to closely manage new initiatives from start to end, while on the other hand the contract managers are responsible for keeping track of every deadline, deliverable, and other obligations laid out in a company's contracts. Some of the responsibilities of a project manager are organizing and motivation a project team, controlling time management, ensuring customer satisfaction, monitoring progress and many more. A project charter is a formal, typically short document that describes the project in its entirety, including the objectives, method of execution and the details of the shareholders. Traditional projects took place within a clearly defined structure, using channels such as internal mail, telephones, memos and formal meetings. Modern projects utilize technology such as email, internet, and social media to share information with a more flexible team.

1.18 QUESTIONS

- 1) What is Software Project Management?
- 2) Why is Software Project Management important?
- 3) What is project? State it's features.
- 4) State the difference between Software project and other projects.
- 5) State the difference between Contract Management and Technical Project management.
- 6) Describe the activities covered by Software Project Management.

- 7) What are the ways of categorizing Software Project?
- 8) What is Project Charter? Explain briefly
- 9) What is business case? Describe in detail
- 10) What are the reasons behind the success and failure of a project?
- 11) What is Management and Management control?
- 12) Describe the project management life cycle.

1.19 REFERENCE

- Software Project Management Edited By Mandeep Kaur
- Software Project Management (SIE) | 6th Edition 2017 by Bob Hughes (Author), Mike Cotterell (Author), Rajib Mall (Author)
- Introduction to Software Project Management *By* Adolfo Villafiorita
- Project Management Absolute Beginner's Guide by **Greg Horine**
- Project Management for Non-Project Managers **By Jack Ferraro**

PROJECT EVALUATION AND PROGRAM MANAGEMENT

Unit Structure

- 2.0 Objectives
- 2.1 Introduction
- 2.2 Business Case
- 2.3 Project Portfolio Management
- 2.4 Evaluation of Individual Projects
- 2.5 Cost Benefit Evaluation Techniques: -
 - 2.5.1 Introduction
 - 2.5.2 The costs include
 - 2.5.3 The benefits include
- 2.6 Cash flow Forecasting
- 2.7 Techniques of Cost benefit evaluation
 - 2.7.1 Net Profit
 - 2.7.2 Pay-out or the payback period
 - 2.7.3 Return on investment (ROI)
 - 2.7.4 Net present Value
 - 2.7.5 Internal Rate of Return
- 2.8 Risk Evaluation
 - 2.8.1 Introduction
- 2.9 Basic Concepts In Risk Management
 - 2.9.1 What is software risk
 - 2.9.2 Risk Evaluation
- 2.10 Risk Identification
- 2.11 Risk components and drivers
 - 2.11.1 Several Tools in for Mangers in Risk Management
- 2.12 Risk assessment
- 2.13 Decision trees
- 2.14 Program management
- 2.15 Program management framework
- 2.16 Stages in program management
- 2.17 Managing the Allocation of Resources within Program
- 2.18 Strategic program management

- 2.19 Creating a program
 - 2.19.1 Program mandate
 - 2.19.2 The vision statements
 - 2.19.3 The design / blueprint
- 2.20 Some reservations about program management
- 2.21 Aid to Program Management
 - 2.21.1 Dependency Diagrams
 - 2.21.2 Delivery Planning
- 2.22 Benefits Management
- 2.23 Summary
- 2.24 Questions
- 2.25 Reference

2.0 OBJECTIVES

- Define business case and its uses in project management
- Define project portfolio management and evaluation of individual projects
- Describe cost benefits and the different techniques to evaluate it
- Define risk and understand the basic concepts of risk management
- Explain Program management and the stages in it
- Define strategic management and the factors involved in it
- creating Program and its aids
- Advantages and disadvantages of Program management

2.1 INTRODUCTION

As we all know that every organization has various ongoing projects at various levels of progress, each project must be equally profitable to get the Managements approval. The selection standards differ from company to company. The company must balance the total risk of the project with each project so that the project must not become extremely risky. Even after approval the project may have various levels of risk, technical complexities, and strategic intent. There are two major types of companies, both have their own advantages and disadvantages, the companies which only take low risk projects have sure profits, but on contrary to that the company fails to ride the technical wave, and their employees may feel that their talent is unutilized, they are having a slow growth due to absence of opportunities and challenges. Then comes the second type of company which only take up risky projects may fall in a dangerous position in case of implementation of its assignments or if the project fails to deliver. So, its required to take up a balanced project, neither too risky nor too easy. Even though some companies want to take

up perfectly balanced projects it faces difficulties of insufficient resources, scarcity of funds to finance the project, so the companies should carefully scrutinize each project and then take it up. The scrutiny process will determine which project will get how much funding and at what time. Even though the scrutiny process varies from company to company but here we can discuss the common process which is adopted by majority companies. Firstly, in the process of scrutiny there is a Decision-Making committee which receives only those projects which have successfully complies with the companies' organizational standards. Secondly the decision-making committee differentiates the project with other projects ongoing projects of the company on the parameters of risk, cost, difficulties, and time with the projects implemented, on the verge of completion, and underway with the company. Based on the differentiative analysis of the committee the project is either rejected or assigned to the project managers for further execution.

2.2 BUSINESS CASE

Business case is also known as a feasibility study or a project justification. It includes:

- 1) Introduction and Background to the proposal – This is a description of the current environment of the proposed project.
- 2) The proposed project - An overview of the proposed provided
- 3) The market - This would contain information like the estimated demand for the product or service.
- 4) Organizational and operative infrastructure - This describes how the implementation of the project affects the structure of the organization.
- 5) Benefits – Wherever possible a financial value should be put on the benefits of the implemented projects.
- 6) Outline implemented projects - Activities such as marketing, promotion, operational and maintenance infrastructures needed to be considered.
- 7) Costs – A schedule of expected costs associated with the planned approach can be presented after the outline implementation plan is done.
- 8) The financial case – A financial case includes information related to costs and income
- 9) Risks – Here project risk has to be distinguished relating to threats to successful project execution and benefits of delivered project.
- 10) The Management plan - This includes a detailed planning of the management required in all the above steps.

2.3 Project Portfolio Management: -

Project Portfolio Management provides an overview of all the projects that an organization is undertaking or is considering. It is the assemblage of all the different projects, Programs and all the other functionalities of the company which lead to the overall growth of the company. Portfolio management gives a sketch of the different projects the company is considering and based on precedence the fund allocation is done by the company. It also helps to provide effective governance to meet the strategic business objectives. Companies do portfolio managements with the aim of including projects which maximize the portfolio value and exclude the projects which are a potential threat to the company's portfolio valuation. Some major applications of portfolio management are risk examination, reduce or completely stop wastage of resources, keep track of ongoing projects, do proper fund allocation. Some major aspects of project portfolio management are

- A) **Project portfolio definition** – The company must have the details of the projects of the company and a resolution must be taken about which types of projects are to be included whether it should include only renewal projects or only new projects.
- B) **Project Portfolio Management** – After the creation of the portfolio the progress of the project can be tracked, and the detailed costings can also be recorded.
- C) **Project Portfolio Optimization** – Some projects may have huge profits while some projects may have modest profits, there are extremely knowledgeable managers which have the knowledge of tracking the performance on a regular basis and a better balance is achieved.

2.4 EVALUATION OF INDIVIDUAL PROJECTS

The project evaluation can have different types of benefits, it helps to identify the different types of problems that can be encountered while undergoing any project, the non-hardware projects have various problems which are different from the hardware projects like falling behind the time schedule, exceeding the allocated budget, increase in the size of project and difficulties. Some of the reasons for the occurrence of these problems is less upfront work, inadequate user end communication and less training. Feasibility of a project can be evaluated by

- Technical assessment
- Costy –benefit analysis
- Cash flow forecasting

A matter of concern is how to associate the risks associated with the software projects can be lowered, the risks of budget excess and lagging schedule can be lowered by taking a proper scrutiny of the project before starting the project. Another solution for lowering the risk of budget

exceeding and missing the time schedule is to begin with a realistic and an achievable target rather than something unrealistic and wait till the targets are missed. The success of the software project is also affected by the perfect comprehensive report of the existing software and hardware infrastructure. As many a times the existing infrastructure may be a limitation to the success of a project. The objective of project evaluation is to whether the product fulfils the ideas presented at the commencement of the project, and it also helps in assessing the quality of the product.

2.5 COST BENEFIT EVALUATION TECHNIQUES

Cost Benefit Evaluation Techniques reflects the timing of the costs and benefits relative to the size of the investment.

2.5.1 Introduction: -

The most widely used method to accomplish an economic assessment of a fresh project is differentiating the expected costs of development and the profits after the system operation. Basically, it compares the anticipated costs and the anticipated benefits. Though it may look like a very easy process, but it is technically difficult. When the company has multiple projects to be evaluated and then selected for further execution then it becomes important to find the best or rather the most profitable amongst them. In some cases, based on the feasibility it may become imperative that more than one projects are approved, at such a time it becomes necessary to classify the projects based on their importance so that the issue of shortage of resources does not become an issue in the successful execution of the project. Cost Benefit Evaluation helps in assessing the economic financial advantage of the project. We will try to recognize some of the costs and the benefits of carrying out the project

2.5.2 The costs include: -

- 1) **Development Costs** – The salary paid to the people working on the project and that to the consultant if any
- 2) **Setup costs** – This includes the cost of putting the system in place and consists of hardware ancillary equipment as well as cost of training and recruiting staff.
- 3) **Operational costs** – The operational costs are assumed to be a percentage of the development and setup cost, operational costs include support costs, hosting costs, licensing costs, maintenance costs and back costs.

2.5.3 The benefits include: -

- 1) **Direct Benefits** – These benefits arise from various cost cutting techniques like, reduction of staff, faster turnaround, optimum utilization of resources, and newer fresh markets. These benefits are understood after the opening of the system.

- 2) **Indirect Benefits** – These are the subordinate benefits of the project include greater accuracy, on account of user-friendly design resulting reducing errors, improved work output and improved flexibility.
- 3) **Intangible benefits** – These benefits are difficult to measure even after the system is operational but are evident to the user. The benefits are the positive effects of the new system, and include entry to the new markets, increased goodwill, enhanced interest in job reduced staff turnover and thereby, lower recruitment costs. These benefits are a part of the strategic decision making:

Communicating the costs and benefits in common ratio: To arrive at a accurate picture the costs and benefits must be spoken in monetary relationships and new benefits must be estimated where the difference between the total benefits and total costs are expressed in monetary form for better understanding of the project. The business establishment should consider any project that shows an additional benefit should not be sufficient especially when the company has numerous projects to choose from and the resources in hand are limited. There may be better projects to allocate the limited resources.

2.6 CASH FLOW FORECASTING

The procedure of choosing the more desirable projects amongst many profitable investments is made possible by the analysis of cash flows. The idea behind the use of cash flows is to maximize the benefit from using scarce resources. In most cases the scarce resources are funds available for capital investments and the benefits are returns on investment. The objective is to select a project or a combination of projects which would give maximization of the total NPV (Net Present Value). Expenses on the project at the development stage and include staff wages, hardware purchase, etc. This expense cannot be deferred until income from the project is generated and hence it is important to know in advance where the funds for the project are going to come from.

In the preliminary stages of the project there are all outgoing payment and the incoming payment that start producing once the project has been implemented along with any additional cost that is going to be incurred at the end of the project life. It is also important for the manager to distinguish where funding of the project is going to come from. The development expenditure could either be company's own resources or by borrowing from the bank. From the management perspective it is essential to have some forecast of when expenditure such as the payment of salaries will take place and when income will be generated. The manager should consider the timing of incoming and outgoing cash flow and determine whether it is in line with accurate financial plan. Accurate cash flow forecasting is not easy as it is done in the early stages of the project and such as it's difficult to determine the expenses and income of some years in future when estimating future cash flows effects if inflation should be ignored.

2.7 TECHNIQUES OF COST BENEFIT EVALUATION

Here are some of the methods for comparing the projects on the basis of the cash flow forecasting

2.7.1. Net Profit

Net profit is the difference between the total cost and total income of the entire life of the project. It is simple technique of calculating the total benefits of the project how will this method different if the profit relative to the size of the investment. In the table below project Y is generating more benefit from Project X and Project Z but original expenditure in project b is higher than both the projects.

Year	Project X	Project Y	Project Z
0 (year the project is implemented)	-8,00,000	-7,00,000	-6,00,000
1	3,00,000	2,00,000	1,50,000
2	3,00,000	3,00,000	2,50,000
3	3,00,000	3,25,000	2,50,000
Net Profit	1,00,000	1,25,000	50,000

Also new profit techniques do not take into account the timing of payment in our example above comparing projects A and B having to wait for returns as the disadvantage that investment must be funded for a longer period. Moreover, payment estimates in the distant future and less reliable than those in the immediate future.

2.7.2 Pay-out or the payback period

It is the time taken to pay back the initial investment. As a result, the organizations choose the project which has the shortest payback duration. This procedure is to determine how long it takes to be a project return the cost of the original investment.

Advantages of pay out/payback period

- (i) This is a simple method is quite simple and easy to understand it has the advantage of making it clear that there is no profit of project unless the payback is over. When funds are limited, it is always better to risk selecting project having shorter payback period this method is suitable to industries weather risk of obsolescence are very high.
- (ii) the payback period can be compared to a breakeven point the point at which the Costs are full recovery covered but profits are yet to commence.

- (iii) The risk associated with the project arises due to uncertainty associated the cash inflows. A shorter payback period means less uncertainty towards risk.

Disadvantages of Pay-out/payback period

- (i) This method does not add color consideration to the time value of money Cash inflows occurring at every point of time are simply added
- (ii) This method becomes a very inadequate measure of evaluating two projects where cash inflows are inadequate.
- (iii) It stresses capital recovery than profitability. It does not consider the returns from a project after its pay-out period. Therefore, it may not be a good method to evaluate where the comparison is between two projects one involving a long gestation period and other yielding a quick result only for a shorter period.

2.7.3 Return on investment (ROI)

- The accounting rate of return or the Return-on-investment method of evaluating is so named because it parallels traditional accounting concepts of income and investment.
- A project is evaluated by computing a rate of return on the investment, using accounting measures of net income. The formula for the accounting rate of return is

$$ARR = \frac{\text{Annual revenue from project} - \text{Annual expenditure of project}}{\text{Project Investment}} * 100$$

- This rate is compared with the rate expected on other projects had the same funds been invested alternatively on those projects. Sometimes the management compares this rate with the minimum rate (called cut of rate).

Advantages of return of investment

This method is quite simple and popular because it helps to understand and incudes income from project throughout its lifetime

Disadvantages of Return on investment

- This method ignores the timing of cash flows, the duration of cash flows and the time value of money.
- It is based upon a crude average of profits in the future years. It overlooks the consequence of variations in profits from year to year.

Conclusion

These traditional techniques of appraising capital investments decision have two major drawbacks

- (i) They do not consider total benefits throughout the life of the project and
- (ii) The timing of cash inflows is not considered.

Therefore, the two main components of a hypothetically sound appraisal method are that :

- (i) It should be based on total cash stream through the project life
- (ii) It should consider the time value of money of cash flows in each period of project life.

2.7.4 Net Present Value

Under this method all cash inflows and outflows are discounted at a minimum acceptable rate of return, usually the firms cost of capital. If the present value of the cash inflows is greater than the present value of cash outflows the project is acceptable. i.e $NPV > 0$ accept and $NPV < 0$, reject. In other words, a positive NPV means the project earns a rate of return higher than the cost of capital

Net present value = Net investment – total discounted cash inflows

If Net present value > 0 Project is feasible and vice versa.

Merits of NPV

- 1) It distinguishes the value of money against time
- 2) The whole stream of cash flows throughout the project life is calculated.
- 3) A fluctuating concession rate can be constructed into the NPV calculations by changing the denominator.
- 4) NPV can be seen as the addition to the wealth of the shareholders. The criteria of NPV is thus in conformity with the basic fundamental objectives.
- 5) This technique is beneficial for choice of reciprocally exclusive projects.
- 6) An NPV practices the promotional cash flows i.e., special delivery cash flows in relationships to present rupees. The NPV's of different projects can therefore be added/compared. This is termed as the value additive principle indicating the NPV of distinct projects which can be supplemented. It suggests that individual project can be assessed independently of others on its own.

Demerits of NPV

- 1) It is problematic to estimate as well as comprehend and use in evaluation with the payback method or even the ARR method.

- 2) The calculation of concession rate grants grave difficulties. In fact, there is difference of opinion even regarding the exact method to calculating it.
- 3) PV method is an absolute measure. Prima facie among the two projects, the project with a Higher Present Value (or NPV) is favored. But it is likely that this project may also value method may not give dependable results.
- 4) This method may not give satisfactory results in case projects having different effective lives.

2.7.5 Internal Rate of Return

Internal rate of return is the interest rate that discounts an investment's future cash flows to the present so that the present value of cash inflows exactly equals to the current worth of the cash outflow i.e., at that interest rate when the net present value will be equal to zero. The discount rate i.e., cost of capital is considered is the determination of the net present value while in the internal rate of return calculation the net present value is set equal to zero and the concession rate which pleases the factors are determined and is called internal rate of return. Any venture that produces a rate of return larger than the cost of capital should be acknowledged because the project will rise the value of the organization. Unlike the NPV method, calculating the IRR is more difficult. The technique used will depend on whether the cash flows are annuity (equal year wise) or non-uniform.

The following steps are taken to determine the IRR for an annuity (Equal cash flows)

1. Determine the payback period of the proposed pushback.
2. In the table of Present Value of Annuity search for year which is equal to or nearer to the life of the project.
3. In the year column, try to look for two present value or concession factor closest to payback period and amongst them one should be larger and other smaller than it.
4. In the top row of the table note, the two interest rates equivalent to the Present Values as stated above in (3).
5. Determine IRR by interpretation.

When cash flow is not uniform an interest rate cannot be found using annuity tables. As an alternative, trial and error approaches of a computer can be used to calculate the IRR. If the IRR is calculated physically, the primary step is to choose an interest rate that seems rational, and it can be done by calculating the average annual cash flows by the annuity methods. They then calculate the present value of the individual cash flows using that rate. When the net present value is positive then the interest rate used is low, i.e., IRR is higher than the interest rate carefully chosen. A higher

interest rate is then chosen, and the present value of the cash flow is computed again. Moreover, when the new interest rates produce a negative Net present value then a lesser interest rate is to be carefully chosen. The process is repeated until the present value is equal to the present value of the cash outflows. Finding the rate of return using trial and error methods can be tedious but a computer can accomplish the task quickly.

Advantages of internal rate of Return

1. It possesses the advantages which are offered by the NPV criterion such as it considers the value of money, it considers the total cash inflows and outflows.
2. IRR is easier to understand. Business administrators and non-technical persons comprehend the perception of IRR much more voluntarily than they comprehend the concepts of NPV.
3. It does not use the concept of required cost of return (or the cost of capital) but offers a rate of return which is suggestive of the success of the proposal. The calculation of the cost of capital is done later in the project.
4. It is consistent with the overall objective of maximizing shareholders wealth since the acceptance or the otherwise of a project based on comparison of the IRR with the required rate of return.

Limitations of Internal Rate of Return

1. It involves tedious calculations.
2. It produces multiple rates, which is confusing.
3. In assessing reciprocally, special offers in the project with the maximum IRR would be chosen up to the eliminating all others. However, in practice it may not turn out to be one which is the most profitable and consistent with the objective of the firm i.e., maximization of the shareholders.
4. Under IRR method, it is assumed that all intermediate cash flows are reinvested at the IRR rate. It is not logical to think that the same firm can reinvest,

The cash flow at different rates. To have accurate and dependable results, it is understandable that they should be grounded on truthful estimations of the interest rate at which the income should be reinvested.

5. The IRR rule comparing the projects IRR with the opportunity cost of capital. But sometimes there is an opportunity cost of capital for 1 year cash flows a different cost of capital for 2 years cash flows and so on. Here, there is no simple index for appraising the IRR of a project.

2.8 RISK EVALUATION

2.8.1 Introduction: -

Experience has taught us that any project can fail and the reasons for failure are aplenty, lack of effective management, lack of proper Engineering or occurrence of some unforeseen events. A risk is any condition or event whose currently not certain but if were to occur it would have a negative impact on the outcome of project. According to the description of risk, one can originate the following. Occurrence is a probabilistic condition. It will have a negative impact on the project where the event to occur. Risk is those event that may or may not occur but if they were to occur, they would have a negative impact on the project. One may confuse with those events that are likely to occur but those who is exact nature is not known in advance.

However, such events are not risk they are normal events that are likely to happen during the project the only problem is that the exact nature is not known in advance example they are bound to be some problems in programming which need to be identified and dealt with in the project itself. Risks are probabilistic event which everyone is hopeful that they would not occur however if risk events were to materialize, they would surely harm the project. Risks are a normal component which is complement in IT projects. On the other hand , IT projects are themselves a risk as the technology keeps changing and can make the project out of work even before it had been completed.

The positive side is that risks are foreseeable events that the project management team can plan for hence risk management plan is an important necessity for managing large IT projects and is included in the project management plan. The project management plan provides details as to how the risks will be identified, analyzed and controlled. Consider that you are planning to host computer programming contest in your campus for which uninterrupted power supply is an absolute necessity. So, the one clearest that you have identified this power outage. Power Failure is probabilistic event that it may happen or may not. However, if there were to be a power failure the project would be affected. Therefore, to overcome the risk each computer must be provided with an uninterrupted power supply. The above example says that the risk management requires additional cost. The cost to provide an UPS to every computer will be an additional cost.

Hence, risk management can be cost effective if the cost incurred during the risk management is less than the cost which would have been the expense if the risk were to materialize. Another factor of consideration of risk management is that it is impossible to correctly estimate the costs during the estimating the value of risk. Continuing with the above example of the power did not fail then the cost of providing an UPS was a complete waste. No one can say so there was not guarantee that the power would not fail. As risks are probabilistic events there is likelihood that the risk management system would not be used at all. In such a scenario the value

provided with the risk management system is defined in terms of the potential value provided which can only be estimated. Primarily risk management has two components, name risk management and risk control. Risk assessment involves risk identification, analyses and prioritization while risk control involved risk management planning, resolution and monitoring.

2.9 BASIC CONCEPTS IN RISK MANAGEMENT

2.9.1 What is software risk?

Risk divided as various when certain event that happen due to change in various factors in the software development. Risks are condition which may or may not arise that have positive or negative effect on the project's ability to reach set objective. Every project has some degree of the risk that there is quotient identified with IT projects is even higher in the given speed by which that IT technology changes. New technology which may come along and outdate the technology used in the project even before the project has been implemented.

However, most risks can be identified and planned for. Thus, risks are foreseeable events which may make it possibly for the project team to make adequate arrangements to neutralize the disc negative effect they may have on the project. The risk management plan is entering integral component of the overall project management plan. There is management plan is an integral component of the overall project management plan. During the development of the risk the major event is identification of risk and its management. The risk analysis process starts with detailing the risk under the various risk analysis factors as mitigate, monitor, and manage. The format to check and analyze the risk is known as CTC (condition transition consequence). Thus, the risk management process comprises of risk identification, assessment and resolution and is key tools in the smooth completion of the project.

2.9.2 Risk Evaluation

Risk management process is an integral part of the execution phase. Though risk management process is initiated in the project planning phase is continuous right through the execution phase till the completion of the project.

1) Concept of proactive and reactive risk strategies

- (a) Reactive risk strategies
 - (i) These risks have a smaller impact on the project
 - (ii) Reactive risk strategies are usually removed at the time of the occurrence either at the compile time or at the runtime or any other time of the SDLC stage of the project.
 - (iii) This strategy does not have any predefined layout rules

- (b) Proactive risk strategies
 - (i) This will have a great impact on the project deployment
 - (ii) Is estimated and probable risk before the start of the project
 - (iii) The steps for precaution are designed as the risk is analyzed at the beginning
- (c) Types of software risk
 - (i) The various types of the associated risk can be categorized in the basis of the delivery schedule which can be related to time
 - (ii) The risk which reaches the financial impact on the project and the management at termed as financial risk
 - (iii) The one which relates to the operation and the working of the project with regards to the specification and requirement under the technical and functional risk

2) Risk Management process

Creating a central database for all the information related to risks, documents of all risk item and the resolution strategies then adopted. Such a database proof useful to future projects which may encounter similar was they may refer to the process was that was adopted to resolve the risk. Summarizing risk information and including it in regular status meeting. Continuously valuing risk and develop strategies to resolve them

1) Risk identification

Risk identified in the planning phase will evolve over time some risk may be eliminated while some of the new ones may be added to the dist. resolve this should be struck at the list and the new ones should be added

2) Risk assessment

The risk management process is created baseline and kept ready in the project planning phase for the execution phase.

3) Evolution of risks

Generally, probable risks have been identified in the project planning phase, but they become more definitive during the execution phase. The project manager is in a better position to define and specify the Risk items.

4) Management An Iterative Process

This management is an iterative process which is initiated at the project planning phase and then continues right through the entire lifecycle of a project in the execution phase a better idea of the likely risk is developed enabling the project manager to devise action plans to deal with them.

5) Risk meeting

This meeting contributes to the process of risk identification and development of face plan to tackle the risk. Collective mind enables better and faster identification of and resolution of list and should be encouraged a risk identification meeting and not one time process they will have to be held time again.

6) Risk resolution

All risks may not have the same impact on the project. some may be grave while others may cause mild bruising and risk are graded according to the ability to impact the project and resolution strategies are developed.

2.10 RISK IDENTIFICATION

Identification of risk is the most difficult work in project management. Risk is an integral part of an IT project and therefore risk identification at the earliest becomes more critical. This identification is an iterative process where the project manager and his team are always on the Lookout for the risk in the project. Risk identification need to be done throughout the project lifecycle because as the project is newly developed there is high chances that anything unwanted could intervene and pose a threat to the project. An idealistic situation would be one in which all the likely threats would be identified before the execution of the project and prepare oneself to counter it once the threats approach. But the reality is that new risks might originate as the project progress and hence the project manager and his team must be always alert and identify this and nub them at the bud stage. Risk can be categorized based on the size of project and the software developed and then the risk is identified and modified accordingly. The risk which creates a business impact due to the constraints imposed by management or the marketplace during the development phase.

2.11 RISK COMPONENTS AND DRIVERS

The risk components can be identified and defined depending on the criteria of performance, support, cost and the schedule of the project. Some of them are

- Performance risk- if the specification of the clients is not met the final project.
- Support risk- the support the application developer provides after deployment of the project
- Cost risk - to maintain the time and the cost of the project
- Schedule risk - the schedule and the parameters of deliverables should be maintained.

2.11.1 Several Tools in for Mangers in Risk Management

There are **several tools** available to help the project manager in identifying risk

- 1) **Project documents** - it is the first place to start looking for the project risk, project plans, resources, etc. are there to identify the potential risks in different stages of the project.
- 2) **SWOT Analysis**- it is a technique to examine and estimate the strengths, weaknesses, opportunities, and threats in a project. It will also clearly identify the areas of improvement as well as the areas of failures in the project.
- 3) **Brainstorming** - a brainstorming session is one in which the project team sits together and analysis the risks. A brainstorming session is one in which the participants can let the imagination go wild there is no restriction on the number of things to be identified. On the other hand, the team is encouraged to identify anything that remotely resembles a risk.
- 4) **Delphi Technique** - although the brainstorming sessions do manage to identify the likely risk factors, but few participants shy away from identify risk on account of personal reasons. To overcome these limitations, the Delphi Technique is adopted. The Delphi Technique comprises of surveys which are anonymously conducted among team members to build consensus on risk events.
- 5) **Assumption analysis** – as all the projects are based on assumptions, so these assumptions need to be verified in order to test their validity. When the assumptions are not verified then they become potential risks in future in the project. Assumption log should be an integral part of the project document.
- 6) **Root Cause Analysis** – this analyses the negative or the positive effects that the project is experiencing and then takes a dig to the root of the effect to identify the cause

2.12 RISK ASSESSMENT

The analysis of the risk factors is the most crucial step in risk management as it determines the probability of their occurrence and the quantum of the effect they might incur on the project. Risk assessment is carried out in a sequential manner where the quality analysis is first done, and it is then followed by the quantity analysis.

- A] **Quality Risk Analysis** – it is a subjective approach and so does not involve detail analysis of the risk. It does not determine the probability of their occurrence and the quantum of risk that would be faced by the project. the main reason for conducting this is to decide whether the risk qualifies for further analysis. It is generally carried out using a risk matrix which is also called as a probability- impact

matrix. In this Risk is “rated” for its probability and the Impact on a scale to understand the Risk Matrix.

- B] Quantitative Risk Analysis** – after successfully completing Quality Risk Analysis, more serious research on the risks which pose as true potential threats are carried out to plug their chances of occurrence and the quantum of the effect, they might possess on the project. The main drawback of this process is that it takes a lot of time to do the detail analysis and as a result budget associated with it also increases. The probability of the risk of occurrence and the quantum of the effect can be mapped in a controlled environment.

2.13 DECISION TREES

A decision tree is a graphical representation of the logic in a decision-making process and the sequence of the decision points that creates the decision. It has many branches as logical alternate, and it simply sketches the logical structure based on the state policy. It is a useful technique for representing analysis when the decision maker must make a sequence of the decisions. It is referred to as the decision tree as different alternatives form branches from an initial decision point known as the decision node and then moves onto the various options in emanating from different points called Chance Nodes.

A system analyst has the following considerations while constructing decision tree

- a. Branches of the tree should represent the various alternatives available
- b. The flow should proceed from left to right
- c. The values associated each alternative must be shown at the end of the branch
- d. All the alternatives available to the decision maker should branch out from the root node which is the starting point of decision tree.

Decision Rules and Decision Tables are used together. Decision Rules enables decisions to be made better and more cost effectively. As a result, Decisions could be faster and more precise. Decision Rules and Tables are used for Programmable or routine operating decision. It is, therefore, authoritative that decision rules are to be noted for references.

2.14 PROGRAM MANAGEMENT

According to program management institute – “a program is a group of related projects managed in a coordinated manner to obtain the benefits and control not available from managing them individually.” Programs may take in basics of related work outside the possibility of the distinct projects in the program. Some projects inside a program can distribute valuable incremental assistances to the organization even before the

program itself has completed. Program management is a coordinated management of related projects which may include the related business as usual activities that together achieves beneficial change which is of strategic nature for an organization. Programs will differ across different industries and business sectors but there are core program management processes which is used in all the industries.

Program management can be selected as a coordinated organization which gives direction and supports in the application of the Portfolio of projects. These activities together will achieve the outcomes and realize the benefits that are of strategic importance to business. Effective and efficient program management is the key factor for the success to any organization to transform the company's vision and various interrelated strategic objectives spanning across different areas such as project delivery processing and so forth. So, a Program consists of several interlinked projects which contribute to attain of the strategic plan. The projects within a program are related with each other and so need to be coordinated and controlled in a manner that delivers benefits on a larger scale which would not have been possible if managed individually. Project management is a process of handling numerous related projects regularly with the purpose of improving the organization's performance.

When the project program is too large for a single project manager to handle then number of project managers are engaged to run the smaller projects. So smaller projects with the multiple project managers are considered to achieve a single long-term goal, objective or benefit for the organization.

The program manager is not concerned with the day-to-day running of the individual project but is the responsibility of the project managers. The program manager needs to ensure that all the projects are running on target and that each will achieve its overall contribution to the program. The activity undertaken during the program management are

- Setting the baseline
- Approving roles and responsibilities
- Program planning
- Project priority
- Stakeholder communication
- Progress reporting
- Managing benefits
- Quality management
- Risk management
- Issue management
- Program closure

A Program usually starts with a vision of a changed organization and the benefits that will be incurred from the change. Delivering the change organization will involve coordinating several projects and ensuring that their outputs are used to deliver benefits. This will need modification in management of business-as-usual activities

A detailed specification of the end state of the program is called a blueprint however the scale of the program the impact of the dynamic business environment mean that intermittent or regular redefinition may be required. The core management processes are

- **Project coordination** – It identifies, initiating, accelerating, decelerating, redefining, and terminating projects within the Program. Managing interdependencies between projects and within projects and business-as-usual activities
- **Transformation** - Taking project outputs and managing change within business-as-usual so that output delivers results
- **Benefits management** – Defining, quantifying, measuring, and monitoring benefits
- **Stakeholder management and Communications** - Ensuring the relationships are developed and maintained. Thus, permitting productive and the two-way communication with all significant stakeholders

Responsibility for these components lies with three key roles- the program sponsor, a program manager and business change managers

The **sponsor** is accountable for the achievement of the business case and providing senior level commitment to the program.

The program manager is responsible for the day-to-day management of the program and Business change manager is responsible for successful transition and benefit realization

2.15 PROGRAM MANAGEMENT FRAMEWORK

A group of related projects not managed as a Program like to run off course and fail to achieve the desired outcome. There are eight important areas in which Program management framework, and they are:-

- i. Vision** - The vision will usually be a brief statement of the intent communicated down from the leadership.
- ii. Aim and objectives** - The aim and objectives of is more detailed statement and that explains exactly what is required
- iii. Scope** - The scope provides boundaries to the Program explaining what exactly to be delivered at the end of the project

- iv. **Design** - Design is the way in which the project that make a program are put together.
- v. **Approach** – It is the way the Program will be Run
- vi. **Resource Management** - Resource management look at the scheduling and allocation of resources both with short-term and long-term views.
- vii. **Responsibilities** - Responsibilities are identified and located related for each area of the program. Every associate of the Program must undoubtedly understand his or her characters and the roles of the other team members
- viii. **Benefits realization** - Benefits realization is the process at the end of the program by which the benefits which were identified at the beginning of the program are being measured towards the end.

2.16 STAGES IN PROGRAM MANAGEMENT

There are four stages that take a program from the initiation to the finalisation of a project which has a defined business objective or benefits. The four stages in the program management

- 1) **Program identification** - This is the high-level process where the strategy and direction of the organization are decided. It is from this that the Programs require and determines the strategies for the realization of the program. The document for each Program is produced demarcating the business case, aligning it with the strategies and scope as well as with the expected business
- 2) **Program planning** - The Program planning is where the design of the Program takes place. The program manager who is creating the program will do the following
 - Define clear objectives
 - Agree and approach
 - Agree to the roles and responsibilities with the team
 - Set up communication channels
 - Agree with the priorities of the project that make up the program
 - Complete project planning

It is important at this to identify adequate levels of resource from earlier projects and identify the requirements for later project.

- 3) **Program delivery** – in this stage, the project manager runs the project. The Program managers duty in this period is to monitor the progress as a service and report the progress to the direction-finding committee or the leadership. The program manager considers all the

projects and must ensure that the programs is aligned with the overall objectives and strategy of the organization

- 4) **Program closure** - Like projects, Programs too have finite life and are closed after they achieve their defined business objectives or benefits. Before the program is locked out, the program manager must reveal to the management that the wanted benefits to have been achieved, often called *benefits realization*. These benefits are those that were identified in the first stage of program identification

2.17 MANAGING THE ALLOCATION OF RESOURCES WITHIN PROGRAM

A program is a group of related projects which means our resources namely people of the organization must be shared within concurrent projects. Organization generally has a pool of people having expertise like developers, network experts, database designers, etc. These people will have to be shared with the number of the projects running within the program. The program manager will have to ensure the optimal use of the specialist staff and plan the allotment of this staff to the individual project within the program. This means that some activities in the project will have to be delayed until the requirement staff has been completed with the previous task allotted to him. The program manager will have to ensure the highly paid technical staff are utilized to optimum and utilization is not intermittent. Thus, allocation of resources is critical from the point of view of success of the program.

2.18 STRATEGIC PROGRAM MANAGEMENT

The strategic program management consists of six interrelated managerial tasks

- A) **shareholder's analysis** – Shareholders are those who effect or can be effect by the Program. Stakeholder analysis is not a lengthy process but is tricky as it requires a management to identify the conflicting expectation the different stakeholders and their power and influence on the organization.
- B) **Vision mission and objectives**
 - i. **Vision** - Sets the purpose of the business organization. It also states the direction of where to go.
 - ii. **Mission** - The mission statement outlines how the vision is to be translated into reality. It also states what is to be done to achieve the vision.
 - iii. **Objectives** - These are quantifiable target which will enable management to measure the success of the strategy. It enables measuring the success of the strategy

- C) Analysis of factors influencing strategy formation** - Business are subject to various factors which influence strategies and over which they have limited or no control at all.
- i. Environment analysis** - Government policies, change in customer attitude, technological changes are the important factors that the organization should be watchful and predict the environment in which it has to operate
 - ii. Firm analysis** - it is important that the firm identifies own resources and analyses them for their ability to deliver. Firms should allocate and utilize the resources in most efficient manner to get maximum return on investment
 - iii. Industry analysis** - To develop good and sound strategy for the business, it is necessary to understand the industry in which the business operating or proposes to operate. The competitive forces within the industry have a lot of bearing on the strategy formulation. It is important that the strength of the competitors is also analyzed. The size and trends of the industry also need to be considered.
 - iv. Product analysis-** The business needs to analyze the competitive position of its products in the context of the development in the market
- D) SWOT analysis** - SWOT analysis combines the analysis of the firm's internal and external environment. The strength and the weakness of the firm are in the context of the opportunities and threats. The aim of this analysis is to achieve an optimum match of the firm's resources with the environment as well as with their objectives of attaining competitive advantage. The firm should build its own strength, adopt a strategy that either hides the weaknesses or reduces them, makes maximum of the opportunities using its strength and anticipates threats and reduces their exposure
- E) Generate strategic options** - Strategic options are generated on the based on the analysis undertaken so far. The strategies generator should be able to provide the firm competitive advantage, discover alternative strategic course and provide alternative methods to engage strategies
- F) Evaluating strategic options** - It is very problematic to find the strategic option to be selected despite of all the analysis. Strategic management is more of an art than science. Moreover, the decision maker should also try to use all the various quantitative and qualitative techniques available before finalizing the strategy.
- G) Implementation monitoring and review** – In the Implementation stage, the strategy must be clubbed with the operational plan, organization chart, clear job description, procedures and manuals, budgets and control systems.

Budgets confirm that the execution is per the plan. Objectives of the budget and constant monitoring ensures that the strategic objectives are accomplished.

2.19 CREATING A PROGRAM

2.19.1 Program mandate –

The process of generation of a program is activated by the creation of a consensual program mandate. The program mandate is a formal document that mainly gives the details

- i. Abilities of the program
- ii. Development that the ability of the new program will bring to the organization
- iii. Contribution of the program towards organizational goals

The formation of the program mandate is followed by the appointment of a person to lead the Program and holds a prominent position in the organization usually coming from the supporting group. The necessity for the prominent position will specify the importance of the program to the organization while the need for the person to come from the supporting group is that these people have identified the need for the program and that they are totally aware of its implications in the organization

Program brief –

The next step in the program construction is the construction of a program brief. The Program briefly undertakes the study of the feasibility of a program the program. The program brief comprises of

- i. Preliminary vision statement outlines the requirements of the organization
- ii. Highlighting the Benefits that the program will create for the organization
- iii. The program shall also identify the risk that the program is likely to encounter
- iv. Will also highlight the resources required and the timelines for the implementation of the program

2.19.2 The vision statements: -

The program brief provides the supporting organization with sufficient information to decide whether the program is worth undertaking. If the program brief is found to be worthy, then only it will be sent to the next stage where detailed planning is done. A small team will take up the planning and a program manager with the similar experience will be appointed to handle the day-to-day responsibilities of the program. The primary job of the team and the manager is to fine tune the preliminary vision statement. The re-defined vision statement of the program should

be able to highlight the capabilities of the program and how it will improve the organization and its performance. The main drawback of the vision statement for the program is that it will not be able to provide the exact financial details and the performance of the program.

2.19.3 The design / blueprint: -

The design provides the changes to be made in the structure and the processes for the organization to achieve the improved ability as described in the vision statement. The Design or the blueprint provides the following

- i. new processes that are essential for the project
- ii. Changes in the organizational structure
- iii. Staff and skills required to change the structure
- iv. Cost and performance associated with the program

The design states the way the objectives of the organization as stated in the vision statement would be accomplished. The design offers the particulars of the working of the program. The design also provides when the predictable benefits from the enhanced abilities will be realized. The Management structure who will undertake the proposed changes is also to be planned. The program manager will have to make a list of projects that will ultimately enable the organization to achieve its objective. These projects will be listed in the program portfolio in the order in which they should be executed along with the estimated timescales for each of the individual project within the portfolio.

It is natural that the program will affect many different groups within the organization. Some groups will be directly affected while others will have to adjust in their mode of operations in order to facilitate the program. The design should be able to identify all the stakeholders who are having an interest in the program or its outcome. Once identified, they should be included in the communication plan of the program and made sure that they are provided with the updates of the program. It is understandable that it is not possible to plan all the projects within the program in the beginning as more particulars will develop as a program progress. However, a preliminary plan can be drawn detailing the portfolio of projects, cost estimates for each project, expected benefits and risk and resources needed. Once the preliminary plan is ready, it is possible to create a financial plan. Based on the financial plan, the Management and Organization can make a budget arrangement in order to meet the expected cost as per the time they would be required.

2.20 SOME RESERVATIONS ABOUT PROGRAM MANAGEMENT

Program management has a much wider context than that of the project management. According to a Project Management Institute (PMI), “A program is a group of related projects managed in a coordinated manner to obtain benefits and control not available from managing them

individually. Programs may comprise of essentials of connected work outside the scope of the distinct projects in the program. Some projects inside a program can distribute useful incremental assistances to organization before the program itself has completed.”

So, a program comprises of many interlinked projects which contribute to the success of the strategic plan. The projects inside the program relate to each other and hence need to be coordinator and controlled in a manner that delivers increased benefits and it would not have been possible by managing them individually. Project Management in the process of managing several related projects with the intentions of improving the organization’s performance.

Programs may also consist of elements of linked work which are outside the scope of individual projects inside the program. However, the program manager should have the foresightedness of the purpose and status of the projects in a program. He can use this foresightedness to support the project level actions to ensure the program goals are achieved. The project manager should be provided with the program perspective or ideas and approaches to solving project issues that would have an impact on the program.

Within a program, there is a need to identify and manage the cross projects which have dependencies on each other. Generally, the project management office does not have sufficient knowledge of the risk, issues, requirements, and design solutions to be applied to manage the program successfully. In such situations, the program manager is in a better position to provide the insights by proactively informing others on it.

2.21 AID TO PROGRAM MANAGEMENT: -

2.21.1. Dependency Diagrams: -

The program manager must be able to track the dependencies between the projects that make up the program and this is generally done with the help of Dependency Diagrams. A Dependency Diagrams helps to track the critical dependencies of cross projects throughout the Programs. This provides two advantages to the program manager-

- i. It guarantees that the complex network of project and their interdependencies are coordinated and synchronized.
- ii. It helps in tracking of the flow of work completed by different projects teams associated in the program. Moreover, it also ensures that all the works are properly integrated with each other.

However, the dependency diagrams should not be confused with the program plans. Program plans only show the milestones of the different projects and highlights the benefits that would be achieved in the end while the Dependency Diagrams helps to track and coordinate project interdependencies and eases the work pressure of the program manager.

2.21.2 Delivery Planning: -

The Dependency Diagram is just a forerunner to a more detailed Program planning diagram. A detailed program includes tranches of projects. A tranche is a group of projects that deliver their products at the same stage of the program, and it is not possible to move ahead without the completion of this stage. The primary criteria for grouping these projects are that the collective deliverables of these projects act as new benefits to the clients. Moreover, the tranches of projects are formed to avoid clashes for scarce resources as it may permit optional sharing of scarce resources. They can be identified through the project briefs defining the scope and objectives of each individual project in the whole Program. Each project tranche delivers some tangible benefits to the customers.

2.22 BENEFITS MANAGEMENT

The most important aim of benefit management is to ensure that the expected benefits from the desired project and program has been materialized. Benefits management has gained importance as the organizations are becoming more aware of the lack of tangible evidence of the returns on investment in information Technologies in terms of improve productivity. Though many organizations have reorganized their business processes in order to improve the effectiveness and efficiencies, but the expected benefits have not been realized. As a result, benefit management exercises are undertaken to mend the defects of the management. Hence, the benefit management exercises start with –

- i. Defining the expected benefit from the program
- ii. Analyzing the difference in cost and benefits
- iii. Planning the Achievement and measurement of benefits
- iv. Allocating responsibilities for the successful delivery of the benefits
- v. monitoring the realization of benefits

Other benefits from enhanced activities can be of different types.

Some of these benefits are

- Improve quality of services
- Increase productivity
- easier compliances of mandatory changes
- Motivated workforce
- Better internal management
- Fraction in risk
- Revenue management
- Reduction of cost
- Strategic alignment

Moreover, it is necessary that the program should be undertaken with only one benefit in mind. On the other hand, the program should be taken with more than one benefits in mind, and they should be interlinked. They will help to Quantify the benefits and valued with increased productivity. However, some of the merits may be accompanied by demerits for others. For the better functioning of the Benefits Management Business, it is advisable that the developers of the program work in a coordination with the users of the program.

1.23 SUMMARY: -

A business case provides justification for undertaking a project, programme or portfolio. It provides a comprehensive analysis of all the pros and cons of any particular project. It provides the decision makers, stakeholders and the public with a management tool for evidence based and transparent decision making. It is a framework for delivery and performance evaluation of the subsequent policy, strategy or project to follow thereafter. Project portfolio management is a strategy that evaluates potential projects by their prospective success and risks, then delegates staff, resources and timeline in a way that maximizes organizational performance. A cost benefit analysis is the process of comparing the projected or estimated costs and benefits associated with a project decision to determine whether it makes sense from a business perspective. The different cost benefit evaluation techniques are software project management, goal setting motivational software, etc. Risk is the possibility of something bad happening. Risk involves uncertainty. Risk management is the process of identifying, assessing, and controlling threats to an organization's capital and earnings. These risks stem from a variety of sources including financial uncertainty, legal liabilities, technology issues, strategic management errors, accidents and natural disasters. Programme management means managing a programme, it is divided into 4 stages i.e. initiate, plan wisely, execute and time to close. Strategy is an action that managers take to attain one or more of the organizational goal. Some factors involved in it are stakeholders and leaders, project priority, resource allocation, risk assessment and company culture. Everything has its own pros and cons; the advantages of program management involve achieving overall strategic goals of an organization. Improve management of projects interdependencies and impact on the business as usual. Effectively managing the resources, among the projects withing a programme, manage risks etc. The biggest disadvantage of project management is that it sometimes leads to overlapping of authority and responsibility between the top management and project management, where they have different plans in mind which leads to confusion among the team members of the project and further project suffering.

1.24 QUESTION: -

1. What is project portfolio management?
2. How are individual projects evaluated?
3. Explain the cost- benefit evaluation techniques.
4. What is risk management? Describe its details.
5. What is Software risk? State it's types.
6. What is program management? State the core program management processes.
7. Describe the eight important areas of program management framework.
8. Explain in detail the different stages in program management.
9. Briefly describe the various aids to program management.
10. What are the objectives of Benefit Management.

1.25 REFERENCES

- Project and Program Evaluation Consultancy With Terms of Reference, Challenges, Opportunities, and Recommendations by **Moses Jeremiah Barasa Kabeyi** - Durban University of Technology
- The basic project management reference library Cook, Desmond L. | Adams, John R. | Hannah, H.
- <https://www.routledge.com/The-Basics-of-Project-Evaluation-and-Lessons-Learned/Thomas/p/book/9781482204537>
- The Basics of Project Evaluation and Lessons Learned **By** Willis H. Thomas

INTRODUCTION TO STEPWISE PROJECT PLANNING

Unit Structure

- 3.0 Objectives
- 3.1 Introduction
 - 3.1.1 Defining the business need
 - 3.1.2 Business goals and objectives
 - 3.1.3 Undertaking the feasibility study
- 3.2 Creating the Business Case
 - 3.2.1 Drafting a project scope statement
 - 3.2.2 Prioritizing projects
 - 3.2.3 Creating a Financial Plan
 - 3.2.4 Approach to Planning
 - 3.2.5 Creating Milestones
 - 3.2.6 Planning and Contingency Planning
- 3.3 Step 0: Select Project
 - 3.3.1 Step 1: Identify Project Scope and Objectives
 - 3.3.2 Step 2: Identify Project infrastructure
 - 3.3.3 Step 3: Analyze Project Characteristics
 - 3.3.4 Step 4: Identify Project Products and Activities
 - 3.3.5 Step 5: Estimate Effort for Each Activity
 - 3.3.6 Step 6: Identify Activity Risks
 - 3.3.7 Step 7: Allocate Resources
 - 3.3.8 Step 8 -Review/Publicize Plan
 - 3.3.9 Step 9 & 10: Execute Plans/Lower Levels of Planning
- 3.4 Questions
- 3.5 Reference

3.0 OBJECTIVES

- To make aware of the Contents of the project plan.
- To outline the general approach, in project planning.
- Study about feasibility and its importance.
- Define project scope and its description.
- Describes project financial planning and approach.
- Define contingency and its importance.
- Explain in detail the different stages of project planning.

3.1 INTRODUCTION

The project planning phase is the longest and the most significant phase of the project cycle. Without proper and systematic scope planning, a project has a poor chance of success. Team members must decide on the budget, set timelines and identify the resources and any hindrances that one may meet in attaining success in the project. The project team validates the availability of resources, materials and expertise which are critical to the on-time project completion. Project team should spend the quality time in planning a project and should make any plan changes carefully before moving on to the next stage of the project. The team may table their project plans in writing to explain clearly the roles and responsibilities and deadlines of the project. From project manager's point of view, project planning is the first step in the execution of the project. A project plan is iterative process that connects the approach and the intent of the manager. A project plan will provide the details of the processes that will be used in the project and how the project work will be implemented, controlled and commissioned. The first step in project planning is to research the business opportunities or the problems that the project aims to address. Good research will empower the project manager to develop proper understanding of the problem. The research can be done by interviewing the key stakeholders of the project. The project manager needs to know why the project is being started and what it means to complete it. The key success of the project success depends on the clear understanding on the part of the project manager and the stakeholder. The idea of the end result of the project should be mutually developed by the stakeholders and project manager.

3.1.1 Defining the business need

Every project has a driving business need and which ultimately identifies and defines the business need and is the first activity of the project manager or the business analyst to undertake. The business needs help the project manager in defining the project scope and developing the project plan. This also helps the project manager to understand the stakeholder's requirements and end result that the project must accomplish.

3.1.2 Business goals and objectives

Whenever an organisation chooses to start a project, it must have some business goals or objectives that it wishes to accomplish. The goals or objectives could be anything like increasing the efficiency, enhance customer satisfaction, generate revenue, etc. Business goals reflect the business scenario that the organisation predicts once the project will be implemented. So, the first job of the project manager after the goals and objectives are determined is to make a current assessment of the business environment that had prompted the requirement for the change. The difference between the current state and the future predicted state generally indicates the scope of the project. By defining the business goals and objectives of project, the project manager is simply describing the end results of the project. The business goals and objectives help in defining

the time and the cost associated with the project. For achieving the business goals and objectives, the project manager can take help of various tools such as brainstorming, discussions with its focus groups, benchmarking, etc.

3.1.3 Undertaking the feasibility study

A feasibility study is a report of the research that the project manager has undertaken. It helps in defining the validity or scope of the entire project or a part of the project. The feasibility study provides detail information to the management about a problem or the business opportunities that are realizable. Some feasibility studies may also cover the financial aspects of the project that is undertaken. Financial part of the feasibility study will tell the management whether it makes the financial sense in undertaking the project and will also indicate the return on investment. The project manager while making the feasibility report should refrain from expressing his view about the feasibility of the project. The project report should be highly realistic and should cover all the aspects of the project, its ability to address the problem or realise the business opportunities, financial implications and the value that it will add to the organisation. The project manager should be reasonable in assessment and should not be attracted to impose new technology merely for the sake of Technology. The feasibility learning contains of the following: -

1) Executive summary

The purpose of the executive summary is to provide the reader with the brief summary of the findings of the feasibility study

2) Define the business problem or opportunity

Here, the business problems that the organisation is facing and its impact on the functioning of the organisation are debated. The business goals or objectives can be utilized to link the project to the problems. Moreover, the benefits of the proposed technology is also stated in the report. The report should be able to identify the areas or people who are likely to be affected by the introduction of the new technology.

3) Purpose of business study

The main purpose of any feasibility study is to determine the practicality of an opportunity or the solubility of a problem. There are number of reasons for which a feasibility study is done.

- To determine whether a product should be purchased from the market or built in the organisation.
- Compare the various Software and Hardware solutions
- Determine capability resource cap with the new technology

4) Assessment of alternative

A feasibility study includes numerous alternative solutions for the business problem or opportunities. The project manager has to state in the report the alternatives evaluated, basis for selection of the alternative and the manner in which the alternative differs from another.

5) Impacted Areas

Here, the feasibility study identifies the impacted people, addresses the issues concerning the users and determines the capability resource gap. Generally, it cover the process of implementation of the new technology. Some of the issues addressed are;

- a) Possibly downtime the user will experience due to implementation of the project
- b) Phased introduction of the new technology within the organisation.
- c) Assessment of requirement for training, the number of users requiring training, the training period and the resources that will be needed to impart training
- d) Learning curve of the new software
- e) Methods by which the new software will integrate with the organisation's existing software
- f) Specific hardware requirements of new software
- g) Compatibility of new software with existing operating system

6) Financial issues

The feasibility report will address the financial issues related to the project. Some of the problems are mentioned hereby

- i. Pricing of the new technology
- ii. Licensing fee and renewal period
- iii. Cost of training
- iv. Additional financial burden arising due to the requirement of the trained personnel
- v. Cost of labour required in the technology
- vi. Technical support from vendor
- vii. Loss incurred for not adopting the new technology
- viii. Return on investment analysis

Making of a business case document is very much similar to the feasibility report but in most of the cases it is a separate document. Like the feasibility study, the business case can help the management in justifying the cost that will be experienced in the project and its return on investment. The business case is built and the relevance of the business goals and objectives and the cost of the proposed technology that can get the organisation there. The business case takes into account the cost of the solution, breakeven point, return on investment and maintenance cost. Along with the quantitative issues the business case, it may also address qualitative issues such as working comfort, increased efficiency, etc.

3.2.1 Drafting a project scope statement

The project scope statement is the most important document in the project planning process. The project scope statement is based on the project requirement, feasibility study, business goals and objectives and the business case. The project scope statement defines the project boundaries, project deliverable and the work needed to be done by the project team to accomplish those deliverables. The project scope statement is the output of the joint effort of the project manager, project sponsor and the stakeholder. As many people are involved in making and approving the statement, it undergoes considerable modification before it gets the final approval from the people who were in the project. It serves as the main document as the remainder of the stages in the planning process. The deliverables that the project will create can be used as a reference point for future project decisions.

Another significant factor of the project scope statement is that the definition of the project boundary. The project boundary is not only defining the advantages of project will do but also defines the disadvantages of the project. However, it is important to have the utmost clarity on the project boundaries. Another aim of the project scope statement is that it defines key performance indicators against which the performance of the project can be measured. The performance is usually measured at the key project phases otherwise known as milestones

The component of the project scope statement is

- i. Project scope description - It is the description of the deliverables of the project that the client will get up on successful completion of the project.
- ii. Project acceptance criteria - The project acceptance criteria will define what the project should create to be accepted by the organisation
- iii. Miscellaneous deliverables - Besides, the key deliverables, the project scope statement should also define those deliverables that are part and parcel of the project

- iv. Project exclusions - Project exclusions are those deliverables that are not included in the project scope. To avoid the ambiguity the project boundary should be clearly defined
- v. Planning Constraints – The constraints that limit the actions of the project manager are to be clearly defined.
- vi. Project Assumptions – There are certain assumptions that are taken into account in a project. Some of the assumptions continue to support the management in the project.

3.2.2 Prioritizing projects

Every organization has multiple projects that are running at the same time and we may also find the same project manager's managing more than one project at the same time. In such a scenario, it is normal, that there are some projects which are similar and even some that may be conflicting. Every project may not have the same priority and the priorities also keep on fluctuating with time and environment. The management of the organization has to prioritize the projects created on the worth of the organization, the success rate of the project manager, and the resolution of the project. The role of the project sponsor is very vital in confirming that the project gets the priority from the management of the organization. For this to happen, the project sponsor should truly believe in the project, the technology, and the abilities of the project manager. Moreover, the project sponsor should have thump within the management to get priority treatment for the project that are important. Once the project sponsor has entirely believed the idea of the project than he should be ready to defend the projects whenever the need arises. However, for this to materialize, the project sponsor should be frequently updated about the existing status of the project. The role of the project manager is that of a middle man and acts as an intermediary between the project team and the project sponsor. Each and very communication regarding the project has to pass through the project manager.

3.2.3 Creating a Financial Plan

Many a times it has been detected that project managers recommend the use of the latest and most advanced form of technology without matching the utility of the technology with the requirements of the project. However, purchasing technology merely for the sake of technology is not a sensible decision and the project manager should refrain from doing so. While planning for technology the project manager should always keep a keen eye on the financial budget and aspects of the project. Depending on the requirements of the key stakeholders and the budget, the technology should be finalized. Henceforth, while finalizing the technology the project manager should ensure the following;

- Technology selected should enhance the productivity of the company.
- Return on Investment on the project should be within the acceptable boundaries of the project.
- The proposed technology should perfectly integrate with the hardware and software infrastructure of the company.
- Time for unfashionableness of the technology should be checked. Also, the time for next up gradation should be checked properly.
- The breakeven point for the investment in the technology should be checked.
- Vendor credibility should also be checked.

While selecting technology, it is essential for the project manager to recognize which technology will produce the desired results. It is not always that the best technology accessible in the market is the prerequisite of the company. It may also be so that the requirements of the key stakeholders could be met with a lower version technology which is also cost effective. Hence, the project manager should not go for the best technology but the right technology. The right technology is the one that will bring the desired results to the company. Another contributing factor to the selection of technology is the predictable level of quality of the project deliverables. As, quality is a very relative factor, so while selecting the technology it should be ensured that the quality of the project deliverables meets the requirements of the key stakeholders of the project. The project manager should be careful in evaluating the expected level of quality and then make the selection of the technology that will meet the requirements and the objectives of the of the project. The time factor is another important factor of importance as a budgeting concern. Remember, time is money, hence the project manager should also consider the time commitment required from each member of the team as well as his own commitment to the project.

3.2.4 Approach to Planning

The project manager should approach the project planning stage cautiously. He should be aware of the resources in hand, the people who will be assisting him and the time he should give to the planning stage. Although planning is an iterative procedure and the project manager will be revisiting it time and again during the entire course of the project implementation, the project manager should decide in advance the time that he would be spending on the planning phase. The usual practice of determining the time one should spend on planning is that it should be directly proportional to the size of the project and the relevance of the project to the organisation.

3.2.5 Creating Milestones

After having selected the technology, it is time to determine how the project will be completed, resources that will be required and the tasks that

will be involved in the project. One of the most critical activities of the project planning is breaking the project into major tasks or milestones. A task is a simple breakdown of the project in the natural order or sequence of activities indicating what needs to be accomplished before moving on to the next stage of the project. The work breakdown structure indicates the major deliverables of the project and helps increasing the task list. The purpose of creating a task list is to ascertain the time frame for the project, resources needed, and the cost of the project.

3.2.6 Planning and Contingency Planning

Planning is time consuming process, so the team manager should make extra efforts to closely monitor about the time spend by the project team members on planning. Though it had been agreed all the stakeholders to undertake research for planning but not easy to achieve as it is time consuming. At the same time, too much of research will be having detrimental effects on the project. So, it is always good to design certain specific goals and deadlines for the research.

Generally, the project manger is helped in his work of researching by a team. The project manager should assign individual research topics along with the objectives of their research to each of the team members along with the deadlines for the completion of the research. After all the team members have completed the research, the information's should be collected and decision should be made according to the research. Every main plan of the project should have a contingency plan ready simultaneously. A contingency plan is a back up plan which will be kept in reserve and will be resorted to in case the original main plan fails to deliver at any stage of the project. As a result, it is always desirable, essential and beneficial to keep a continency plan ready. Every project has multiple phases running simultaneously and in case the project manager finds out in any stage of the project that the main plan is not working as the requirement then he has the power to implement the continency plan after stopping the main plan.

3.3 STEP 0: SELECT PROJECT

After the business case is ready, it is sent to the management for approval. As in every organisation there are a number of ongoing projects, number of projects are waiting for approval, funding and resource allocation. So, each and every project has to compete with several other in order to see the light of day. The criteria for selection of the project and its inclusion in the company's product portfolio is similar to the analysis for a proposed project alternative. The management of the company has to strike a balance in the portfolio in such a manner so that the total sum of the risk associated with each project does not make the total portfolio a risky one. Generally, the projects selected in the portfolio has varying degrees of risks, technological complexities, size, resource requirements and strategic inputs. If a company only selects projects which have very low risk and technological complexities than it will fail to ride the technological wave and its employees will be feeling that their talent is not been being utilized

properly and will be feeling stagnated due to the lack of the growth, challenges and opportunities. On the other hand, if the company whose project portfolio comprises only of risk and technically Complexities than the projects can end up in a dangerous. Therefore, it is necessary for a company to strike a balance in the Portfolio of projects having varying degrees of risk and technological complexity. Moreover, the organisations may be interested to undertake number of projects but is handicapped by the lack of resources and funds to finance each project and hence it has to prioritise and select the project. The selection process determines which project will be funded in a particular period. Though different organisations have their own selection process but a standard process is adopted by most of the companies. The first step in the screening process is in which the projects receive for approval as screen for the compatibility with the organisational standards. The projects which qualify the standards are sent to the decision-making committee for project approval. The committee that compares the submitted project on various factors such as risk, cost, complexity and time with the other projects in the company which are either on the verge of completion or have been recently implemented. Depending on the comparative analysis of the projects, they are either approved and assigned to the project manager for action or dismissed.

3.3.1 Step 1: Identify Project Scope and Objectives: -

Project scope management includes the process required to ensure the project includes all the work required to carry out the project successfully. It is mainly concerned with defining and controlling the factors in the project. As per the Project Management Body Of Knowledge (PMBOK), the knowledge area of Project Scope Management includes five process namely scope initiation process, scope planning, scope definition, scope verification and scope change control

- a. Project Scope Initiation Process** – In this process, the project sponsor gives the project manager the authority and resources to define the project. But it is only done when the project plan and charters are being developed. It is the first step in the project scope management and develops when the project sponsors authorize and provide resources to the project manager to develop the project scope management plan. This is similar to Business case but gives more details compared to business case. After authorization, the project is planned as per the IT project methodology.
- b. Project Scope Planning Process** - The project scope planning process identifies the area of work inside the project including project's work, activities and deliverables that will help to accomplish the projects MOV. Its main function is to set the boundaries of the project work. It is also important to identify which is not a part of project work in order to avoid future problems. It is not possible to prepare the project plan without project's scope as it contains all the necessary information about of the work, activities and the deliverables that are to be achieved and without its

knowledge, the team will not be in a position to estimate the schedule, budget and the resources that will be requirement in different phases of the projects.

- c. **Project scope definition process** - The project scope definition process identifies the project deliverables. Project deliverables is the work that wants to accomplish in order to distribute a product with exact features and functions as committed to the client. After the setting of the project scope boundaries and developing a project scope statement, the foundation for defining the project scope is prepared. Project scope defines the deliverables that the project teams aim to achieve and the development of detailed project scope is critical to the success of the project. A detailed project scope defines the project boundaries, major deliverables, assumptions and the drawbacks that are documented in the project scope statement. Project deliverables are the features and functions that characterize the final product. The boundaries and deliverables defined by the scope planning and definition facilitate the development of the project charter and plan. The requirements of the project define its boundaries. Moreover, stakeholders need, wants, and expectations, are Analyzed and converted into project deliverables while ensuring that they do not cross the boundaries of the identified project scope. The assumptions and constraints are Analyzed for completeness with additional assumptions and constraints added if felt necessary. The project team members and other stakeholders who have additional insight into the preliminary project scope statement should Analyze and develop the project scope definition. There are two types of deliverables or scope, project-oriented deliverables/scope and product -oriented deliverables/scope. Bifurcation of the deliverables enables the project team to clearly define each deliverable, assign time, resources, and responsibility for its accomplishment. Such a bifurcation and allotment of work leads to better coordination amongst the team member and enables them to identify the place of each deliverable in the larger picture of the project. This facilitates the assigning of resources, estimation of time, and the calculation of cost of completing the work. Bifurcation of deliverables also lays to rest any ambiguity on the project's deliverables and expectations on the part of the stakeholders.
- d. **Project scope verification process** – The project scope defined needs to be verified. Scope verification is the method of procuring the stakeholders official acceptance of the completed project scope and the deliverables associated with it. The Scope verification process checks the scope for accuracy and wholeness. Scope verification guarantees that the project deliverables are accomplished as per the ethics placed in the delivery definition table (DDT). Scope verification includes the review of each deliverable and comparison with the standards specified in the DDT. Scope boundaries and deliverables should be agreed upon by the project sponsor and project manager. In case, the part of the project is being terminated early, than the project scope verification should properly

text the stage and extent of its completion. One may tend to relate project scope verification with quality control; however, they differ. While project scope verification is concerned with the acceptance of the deliverables, quality control is concerned with meeting the specified quality requirements of the deliverables. Also, quality control is performed before scope verification although both can be performed simultaneously. Also, quality control is performed before scope verification although both can be performed simultaneously.

- e. **Project Scope Change Control** - Although the project scope has been set with the great planning and thought, changes are bound to crop up as the project advances and new information or needs that develops. Project scope change is like an unwelcome but unavoidable guest who barges in at any time and in any place. Change is an unavoidable part of IT project management. No matter how much planning goes into the project, the future is unpredictable. This permits the need for Project Scope Change Control to manage the changes. The need for change in project scope may arise due to various reasons like the project manager may discover a better method or a new feature or a request from management or customers to change the deliverables, or in some cases the cause for change is the manager himself. Whatever, be the reason for the change, if not managed properly can derail the entire project. To make changes in IT project management is a difficult process to incorporate, but it is almost a regular feature in every project due to the very nature of the industry. The change control process has to approve the change to initiate modifications in project schedule and budget. The project scope change control process also guards the scope boundaries from expanding pointless due to demands of additional features and functions to the project scope. Moreover, each change request must be documented, the change tracked, request has reviewed, been accepted implemented or rejected or rejected, should and be conveyed to the stakeholder. as Whether it helps in the final scope verification with stakeholders who would recorded in the change log also not be puzzled as to why certain change requests have not been implemented. There is a direct relationship between the project scope, budget and schedule. There are 5 processes that make up the project scope management plan. The size of the project determines whether the scope management plan is to be a separate document or a part of the project charter. If it is large in size and more complex a separate scope management plan document should be created and the summary of it can be included in the project charter. The Cardinal rule for defining the project scope is that it should be aligned to the project's objectives.

3.3.2 Step 2: Identify Project infrastructure: -

As stated earlier, it is authoritative that the project manager develops a clear understanding of the purpose behind the project. Uncertainty should be avoided as it is likely to cost the organisation heavily. Clarity of purpose will help the organization save in terms of cost, time and effort

and the project will be a success. Therefore, the first step in project management is to understand the purpose of the organization behind the project. Once the project manager is aware of what the project should produce, he can move ahead with its planning. Analysis of the Requirements is the first stage in software development process and encompasses those tasks that go into determining the needs or taking account of the possibly conflicting requirements of the various stakeholders. Analysis of the requirement in a project is critical for the accomplishment of the project. Requirements must be actionable, measurable, testable and related to identified business needs or opportunities. Requirements ought to be distinct to a level of detail which is sufficient for a system design. Requirements Analysis consists of three types of activities and they are

- i. **Eliciting requirements:** It is the work of communicating with customers and users to determine about their requirements and it is sometimes called requirements gathering.
- ii. **Analysing requirements:** It regulates whether the specified requirements are clear or not, checks the completeness and its ambiguous nature, or if any contradictory statement is present and then tries to resolve these issues in order to attain the desired objectives of the project.
- iii. **Recording requirements:** The Requirements of the project must be documented either as language documents or as a process specification.

The easiest and most convenient approach for collecting the project requirement is to discuss with the key stakeholders. Discussing about the requirements of the project will help the project manager to understand the requirements better. In some cases, the project manager is assisted by a business analyst who will complete the requirement collection process for the project manager. The project manager can then discuss the requirement with the business analyst and stakeholder to get a clear idea of all the requirements. In the absence of a business analyst, the project manager has to gather all the requirements on his own. Although the work is too large but it will help the project manager in developing a clear understanding of project scope, quality expectations and then identify and address any threats to the project and its results. Many, at times it has been witnessed that the number of stakeholders is too large and it is not feasible for the project manager to interact with each and every stakeholder on a one-to-one basis. In such a situation the project manager can resort to web technology and conduct a survey or ask the stakeholders to nominate representatives amongst them to interact of a daily or project stage bases. The project manager can then meet the representatives and discuss the requirements that will affect the large number of stakeholders.

In addition to the above method, a very effective alternative way of collecting requirement is to passively or maybe actively observing the stakeholders at work. This gives the project manager an idea of the requirements from the project by the stakeholders. Whatever, be the

approach adopted, the project manager should ensure that all the key stakeholders are in agreement with the requirements from the project. Once the requirements and purpose behind the project have been defined, the project manager can determine the time frame that will be required for the completion of the project. Although, the management of the organization has set some time frame for the completion of the project, the project manager should make his own estimation based on the resources and the requirements that are available with him. For estimating the time frame, the project manager should be aware of the end result of the project. The end result of the project can be discussed with the project in charge. After the end result has been decided upon, the project manager can chalk out the path that the project should take. The project manager is solely responsible for setting the goals and deciding the path the project should in order to reach the desired goals. However, the project manager should develop the path after thorough discussion with the all the key stake holders.

IT Project Management is a complex balancing act of technology between the internal and the external factors that are involved in the projects. Some of the external factors are demand, market conditions and the technological changes. Hence, the project manager should ensure the following in the first place;

- Project has clearly defined objectives
- Project has well defined end results
- Project should spell out the exact requirements
- Project should take into account any industry standard and regulation
- Project should also take into account any government regulation that it should abide by
- Project should have reasonable time frame for completion
- Project in charge has the authority to take decisions
- Project should have committed resources

On his part the project manager should be of a very inquisitive mind. He should question each concept, technology and the time that will be associated with its implementation. The project manager should not judge everyone with the same yardstick. Every person in the organisation is bound to vary in their IT knowledge. So, while deciding on the technology that will be utilized in the project needs to seek answers to the following questions;

- i. Effect of the proposed technology on the users
- ii. Effect of the proposed technology on other solutions
- iii. Compatibility of the proposed technology with other operating systems
- iv. Experience of other companies using the proposed technology
- v. Track record of the vendor of the proposed technology

3.3.3 Step 3 : Analyze Project Characteristics: -

The first step in the project initiation stage was the determination of the business need and identification of key stakeholders. The project manager needs to Analyze the project characteristics and this can be done through the project charter. It is the time to draft the Project Charter. A project charter is a detailed official document prepared in line with the company's vision and goal. It describes in detail the finer shades of the project and helps in chalking out deadlines for the milestones to be achieved within the project. The Project Charter serves as a road map for the project manager and states the goals that are to be achieved from the project. A Project Charter provides a clear explanation of the project, its characteristics, the end results and the project authorities. Project authorities are the people who are responsible for the implementation and success of the project. A project charter is the final official authorization to the project manager for the commencement of the project. It is a green signal to the project manager to start the work on the project. The project charter and the project plan provide a tactical plan for the execution of the project. It is an agreement between the project sponsor and the project manager along with his team which documents the project's MOV, defines the infrastructure, summarizes the project plan, defines the roles and responsibilities and states project control mechanism. The project charter and the project plan should be developed simultaneously as the summary of the plan has to be included in the project charter. Moreover, the infrastructure identified in the project charter will influence the project schedule and the estimates in the project plan. In the development of the project charter and project plan, the project manager, the project team, and the project sponsor should be included. This will ensure that they all agree and subscribe to the assumptions and constraints in the project plan and that the goals and objectives of the project are achievable.

1. **Purpose of the Project Charter** - A project charter serves the following purpose -
 - Defines the business need
 - Identifies the project sponsor
 - Authorizes the project
 - Identifies the project manager, grants authority and makes him responsible for the management of the project
2. **The Project Charter Covers the following aspects-**
 - i. **Project's MOV** - the project's MOV is discussed in detail in the business case. The MOV now needs to be agreed upon by both the project sponsor and the project manager and his team. Generally, the project's MOV drives the project and once agreed upon should not be changed. The MOV drives every decision in the project, the project planning process, and the requirement of resources.

- ii. **Project Infrastructure** - The project charter takes into account all the key resources, people, technology, methods, processes that take part in the making of the project. Though, the available infrastructure needs to be taken into account even while developing the project plan. The available infrastructure will help the project manager to plan the schedule and the budget for the project.
- iii. **Summary of the project plan** - The project charter should include a brief summary of the project scope, budget, schedule, quality objectives, key deliverables, and major milestones. Thus, the project charter should serve as a source of information to all those seeking details of the project. It is more like a ready reckoner that can be referred to by anyone seeking information on the project.
- iv. **Roles and Responsibilities** - Not only should the project charter identify all those involved in the project but all define their individual roles and responsibilities and their line of command.
- v. **Project Control Mechanism** - As the project progresses, new data and information arises a need to make changes to the project scope, schedule and budget will certainly be required. But, incorporating these changes is not an easy job and may make the project team lose focus. Therefore, to facilitate change the project charter should outline a process for change management.

3.3.4 Step 4: Identify Project Products and Activities: -

i. **Project Products/ Deliverables**

The next step after the development and approval of the project scope statement is project planning. The primary requirement for project planning is the Work Breakdown Structure (WBS) for which the approved project scope statement is required. The WBS is a deliverable or a product oriented classification that regulates the decomposition of the job to be executed by the project team, to accomplish the project objectives and create the required deliverables or project products. Decomposition indicates the breaking down of the project scope statement into smaller, more manageable components i.e., the project deliverables. The project scope statement defines all the project deliverables. These defined deliverables are then clubbed together to form the Delivery Definition Table (DDT). The Delivery Definition Table contains the sequence of the delivery of the deliverables. The Delivery Definition Table is then used to create the Delivery Structure Table (DSC) which contains the work packages which is then further used to create the Work Breakdown Structure. So, WBS is similar to Bill of Materials (BOM),

wherein a product is broken into its smallest component for which estimation of time and cost is possible. The idea behind WBS is to split each component into smaller components till it reaches its smallest component which is called the work package. The work package is the smallest component in the WBS that facilitates estimation of cost, schedule, resources, and monitoring and control. Each downward levelling in the WBS characterizes an increasingly thorough definition of the project work. Thus, WBS delivers the essential framework for comprehensive cost estimating, guidance for schedule development and control. At the top the WBS offers high-level deliverables each of which are then broken down into more detailed and well-defined deliverables known as the work packages. Further on these work packages are decomposed into activities in the project schedule. The sum of all the work packages comprising of activities should be equal to the project scope. The completion of the project scope will also result in the completion of the product scope. Completion of both scopes would mean the completion of the project.

ii. Benefits of WBS

Dividing complex projects to simpler and manageable tasks is the main purpose of developing WBS. This method is used by project managers to simplify project execution as smaller work packages and activities so that they are easier to manage. Following are the benefits for developing WBS in a project:

- a. WBS serves as an input to key project management activities, mainly, cost budgeting, resource planning, risk management, activity definition, and schedule planning.
- b. WBS illustrates the project scope, so that every team member as well as stakeholder can have a better understanding of it.
- c. WBS provides an accurate and readable project organization.
- d. WBS facilitates accurate assignment of responsibilities to the project team.
- e. WBS indicates the project milestones and control points.

iii. Developing the WBS

Project managers, project team and Subject Matter Experts (SME) characteristically develop a WBS as a predecessor to a detailed project schedule and budget estimate. Though, there are different methods of decomposing project work and creating WBS, the starting point for deriving a WBS would be to identify the main deliverables of the project. The next step would be to convene a meeting of all team members and

subject matter experts, who would then brainstorm all the work required to complete project deliverables successfully. The participation of all team members and topic wise experts increases the probability of the resulting WBS being all-inclusive. Team members and subject matter specialists recognizes all project deliverables and milestones and then break them one at a time into a detailed and chronological list of activities essential to complete them. It is logical to create a WBS based on the nature of the project work. However, this is possible only in projects where it is possible to identify the phases of project. In projects where the focus of the project execution is not clear, the project manager would be the best judge in categorizing the project components and then decomposing. There are no hard and fast rules to be used in the breakdown of the WBS. Some project managers break down the WBS till it is not possible to break any further. But generally, this method is not recommended as the tasks created would be too small to manage, measure, and assign. Therefore, for a small project the work could be broken down into a few days of work while for a large project, the work could be broken down into weeks of work.

The universal rule is the "two weeks" rule where the task is broken into anything not less than two weeks of work. Another rule in creating the WBS is the "8/80" rule. This rule suggests that the smallest work package in the project should take no more than 80 hours to complete or no less than 8 hours to hours complete. of work. In short, no task should be smaller than 8 hours of work and should not be larger than 80. Hours of work. The WBS should be escorted with a WBS Dictionary, which lists and defines WBS elements. The WBS dictionary explains each work package in the WBS. The dictionary identifies each component of the WBS and how they are related to the project scope. The description of each element of the WBS is stated in simple language so that everyone can understand. The WBS dictionary contains the following :

- (i) Code of accounts
- (ii) Description of the task
- (iii) Person responsible for the task
- (iv) Task dependencies
- (v) Schedule of the task
- (vi) Resources required for the task
- (vii) Cost to create the task
- (viii) Quality requirement of the task
- (ix) Technical details of the task
- (x) Acceptance criteria of the task]
- (xi) Contract information

WBS and WBS dictionary are not the Schedule but the building blocks for the schedule, they are not static documents and are subjected to changes as one gathers more information related to project.

iv) **Project Activities**

Generally, it is seen that the top level of the WBS offers high level of deliverables where each of them are decomposed into detailed and well-defined deliverables known as work packages. These work packages are further decomposed to form activities in the project schedule. Activities or tasks are the key input of the project schedule. A list of all the activities to be included in the project schedule is tabulated and it includes the activity name, the activity code/identifier/number, and a brief description of the activity. On the other hand, the activity characteristics encompass additional schedule linked information approximately about each activity such as predecessor and successor activities, logical relationship between the activities, resource requirements of the project, constraints, dates, and assumptions related to each activity. The data for the preparation of the activity list and activity attribute is the WBS and WBS dictionary. As information on the requirements of various activities appears, it is updated in the activity attributes. So, Activity information is a necessary input for other processes and in activity sequencing, resource allocation, estimating duration, and developing the schedule. The project team should get the activity list and attributes reviewed by the project stakeholders before undertaking project scheduling. A milestone is a substantial occurrence in the life of a project. A milestone in a project is accomplished after the completion of various activities. Although, the milestone has no duration, it marks the completion of various activities and is considered a key event. The primary purpose of setting milestones is that they prove useful in setting schedule goals and monitoring the progress of the project activities. Milestones might include completion of certain software modules, installation of hardware, etc. Thus, milestone marks the completion of an important stage in the life of the project and the culmination of activities undertaken to reach that milestone. However, it should be noted that every project deliverable is not a milestone. A milestone is a special event, something which is important for the project and takes it closer to its final destination. The activity definition further defines the project scope and the activity sequencing. It also defines the time of the activity along with resource estimating and the activity duration. So, the activity will clearly define time and the cost of the project. These four project time management processes form the basis of the project schedule.

3.3.5 Step 5: Estimate Effort for Each Activity: -

In most of the tangible projects or products, the size is the most important parameter used to measure the efforts required. But when it comes to software projects it is very difficult to measure the size accurately. Software professionals have been measuring the size of software applications by using different methods such as Source Lines of Code (SLOC), Function Point, Object Point, and Feature Point. By evaluating and estimating the size of the project, it will help in determining the efforts, schedule and cost of executing the project. After having sized the project by using any of the above methods, it is time to transform the size of the deliverable efforts within a comfortable schedule. The total project effort needs to be assigned to the schedule thus it enables the project manager to do a proper resource loading. Once the phase wise resource loading details are available, the project manager can apply the resource rate to each category as per the duration of the assignment. This will provide the project manager with the base cost of the project. To this base cost, the project manager should add cost of project management, configuration management, and other overheads to arrive at the gross cost.

1. Effort Estimation Process

The overall project effort is dependent on the application size and the productivity of the team. The application size can be calculated from the application specification using any one of the estimation methods. As for productivity, the project manager will have to ensure that the team members have the productivity for the technology platform on which the project is being developed. Every programming language has its own average productivity figures that need to be adjusted for the organizations own team. This can be done by using the historic project productivity data. The productivity of the team is based on the competence of the team members as programmers, their expertise with the technology to be used and the software development environment within the organization.

Based on the above; **Effort = Application Size x Productivity**

The effort thus obtained project is the total effort that would be expended for the lifecycle's stages of the projects. But project management and that configuration management efforts need to be added to the final efforts.

3.3.6 Step 6: Identify Activity Risks: -

Risk is an integral part of an IT projects and therefore risk identification at the earliest becomes more critical. Risk identification is an iterative process, wherein the project manager and his team are always on the lookout for risk that may sneak into the project or may have already sneaked in. Risk identification needs to be done throughout the project lifecycle because as the project develops new risk is likely to originate and pose a threat to the project. An idealistic situation would be one in which all the likely risks could be identified before the execution of the project

but the reality is that new risk originates as the project advances and hence the project manager and his team have to be always alert and identify the risk and nub them in their bud stage. They have to do the following

1. To identify the risks are always the biggest and the toughest task in project management.
2. The risk can be categorized on the basis of the size of the project and the software development, the risk identified is then modified.
3. The risks which create a business impact due to the constraints imposed by management or the marketplace during the development phase till the deployment phase.
4. As the customer played the critical role in the project development being the end user, the various characteristics related to the customer for communication are very important.
5. The exact requirement gathering and the analysis to be done for the development of the project at the Software Development Life Cycle has to be determined to avoid the risks.
6. Good quality tools should be used and made available for the developers.
7. New advances in the area of technology and the project development has to be studied for a perfect time lined software development.
8. The major risks are associated if the staff and the team are not as per the required project.

A) **Risk Components and Drivers**

The risk components are defined and identified on the criteria of the performance, support, cost and the schedule of the project :

- i. **Performance Risk** - If the specifications don't meet the final requirements of the project.
- ii. **Support Risk** - The support the application developer provides after deployment of the project
- iii. **Cost Risk** - To maintain the time and the cost of the project.
- iv. **Schedule Risk** - The schedule and the parameters of deliverable should be maintained.

There are several tools available that can help the project manager in identifying risks.

- 1) **Project Documents** - Project documents are the first place to start looking for project risks. The project scope statement, WBS, project plan, resource requirement, etc will help to identify many risks associated with the project.

- 2) **SWOT Analysis** - it is similar to an organizational SWOT where the project is evaluated in terms of its strength, weakness, opportunities and threats. Moreover, the areas are identified where the project could fail to deliver its goals and also Analyze the areas of improvement in the project.
- 3) **Brainstorming** – in brainstorming process is a session in which the project team meets together and discusses about all the possible areas of risk. In this session, the participants let loose their imagination. There are no restrictions on the number of risks that could be identified. Infact, the team members are encouraged to identify probable risk.
- 4) **Delphi Technique** - Although brainstorming sessions do manage to identify the probable risks, a few participants may shy from identifying risks on account of personal reasons. So in order to overcome these limitations the Delphi Technique is adopted. In Delphi technique, numerous anonymous surveys are been conducted amongst team members to build consensus on risk events.
- 5) **Assumption analysis** - Every project is based on certain assumptions. However, these assumptions need to be tested to ensure their validity. Assumptions not tested have the potential to become risk in the future. Therefore, these assumptions need to be tested, researched and the results confirmed as to how they would be affecting the project. An assumption log should form a part of the project documents.
- 6) **Root cause analysis** - This analysis examines an effect that the project is experiencing and then goes to the root of the effect to identify its cause. All the identified risks need to be entered into a risk register. The risk register will contain a log of all the risks and its impact on the project.

3.3.7 Step 7 : Allocate Resources: -

For a project to be fruitful it is advantageous to distinguish in advance the resources that would be essential and the time when they would be required. Resources are required to carry out project tasks and include people, the capital, material, facilities, tools, and equipments. Lack of resources will be a constraint on the completion of the project. However, the biggest resource in IT Projects is People. Projects require people with the right qualification, experience and skill set to accomplish the project objectives. Depending on the nature of the project, deadline for its completion, and the scope of project especially WBS the project manager is able to identify the resource requirement for its success. The resource can then be mapped to the WBS to create a Resource Breakdown Structure. The Resource Breakdown Structure indicates the moment the resource would be required and the project deliverable would be creating. As we have seen that there are various projects dying for scarcity of resources in an organization. The resource breakdown structure will be

able to utilize these resources optimally by assigning resources to project activities. The project schedule will have to consider the availability of project resources to create deliverables. In case the project sponsor has set a deadline even before the project is assigned to the project manager, the project manager should verify the resource requirement based on the project deadline. Projects that fail or are delayed can usually be traced to poor planning, insufficient resources or unrealistic deadlines. The time taken to complete a project depends on the number of hours a resource works in a week, the number of days they will be able to work, and the time they can allocate to this project.

The project manager's first concern during planning is to uncover which skills are needed for the project, which tasks will require the most hours and which skills are needed to complete those tasks. It will help the project manager to propose assigning a resource with the right skills and the highest availability. In order to determine availability, we should first consider the resources that are present in the project in terms of percentage and then adjust and planned the out-of-office resource's project time. Availability Find the right percentage, balance to then solve adjust for the concurrent project demands and plan out-of-office time. To find the right balance to solve the resource equation, the project manager will be able to shorten the project's planned duration and accelerate the finish date. While selecting his team, the project manager may rely on multiple methods to assess the skills of the proposed team member. Some of the basic methods for selection are as follows: -

- I. Prior Experience** - The project manager has worked with the team member on prior projects and is very well versed with the capabilities and shortcomings of the team member.
- II. Prior Track record** - Although the team member to be recruited may not have directly worked with this team member but his track record in the organization makes him a good prospect for being a part of the team.
- III. Management Recommendation** - Not all members are selected by the project manager. Some may be in the team on recommendation of the management. Either management may feel that they are suitable for the job or wants them to get the experience of working on the project.
- IV. Team Member Recommendation** – Existing team members whom you fully trust may recommend others on to the team. The project manager might be able to hit the sponsor's initial deadline by using creative resource allocation, such as a trainer with development skills, or a desktop technician who can set up a server, when traditional resources aren't available and delaying isn't a practical option. The key to making these kinds of choices and settling on the most aggressive yet realistic deadline, is to communicate with the resources' managers. Make sure they agree to commit their people to

the effort and hold them accountable for their active participation on the project team and timely completion of their assigned tasks.

3.3.8 Step 8 -Review/Publicize Plan: -

1. Review of Project Plan

The planning processes and activities are used by the project team to successfully plan and manage the planning project. The planning process is the utmost serious portion of the project management process. The planning process group gathers information from various sources and develops the project management plan

- The efforts consumed on planning should be proportional to the size and complexity of the project. In other words, larger and complex projects demand more planning efforts as compared to smaller and simpler projects. Although, the efforts may vary from project to project, planning is required at each phase of the project with more emphasis given to the development of the project charter and the project plan.
- The planning processes also identify, define and mature the project scope, project cost and project schedule. The planning process is iterative in nature and is subject to constant change and revision. Project planning changes as more information emerges, additional dependencies are discovered, new requirements evolve, and new risks, opportunities, assumptions are identified. Hence, significant changes occur throughout the project lifecycle making it mandatory to revisit the planning process and, in some cases, the initiating process again and again.
- However, experience and good judgment enable the project team to overcome most of the difficulties during the planning process. Supporting processes include a basic description of the project scope, the deliverables, project duration, resource planning, activity planning, cost estimating, schedule estimating, organizational planning and procurement planning.

2. Publicize Project Plan

Perhaps one of the most critical parts of the project plan is to publicize the project plan i.e. to keep the people concerned with the project informed of the project plans and its progress. For this, the project manager needs to identify all the stakeholders and keep them informed of the plan. The requirement of information of all the stakeholders is different and hence distribution of it has to be also planned. The method of distributing, the frequency of distribution and the responsibilities of each person in the project team for distributing the information need to be chalked out.

3.3.9 Step 9 & 10: Execute Plans/Lower Levels of Planning: -

The project plans need to be executed in order to bring the project to reality. The project execution group is entrusted with the task of undertake the work defined in the project plan. The execution job includes the proper coordination among people and resources as well as mixing with the performing activities which will be in accordance with the project plan. This group also addresses the scope defined in the project scope statement and implements approved by the project manager. This group is also entrusted with quality

3.4 QUESTIONS

1. Describe the contents of Project plan
2. Why is it essential to undertake the feasibility study?
3. Why is project scope the most important document in project planning?
4. Why is it necessary to have a contingency plan?
5. Discuss the project scope management process
6. Discuss project scope verification process
7. Write short notes on
 - a. Benefits of WBS
 - b. Project Charter
 - c. Risk components and drivers
8. What is WBS? State its benefits
9. State the scope and objectives of a project
10. Outline the general approach that is considered in project planning

3.5 REFERENCE

- The 'step wise' planning approach to software projects [IEEE Xplore](#) Conference: Project Management for Software Engineers, IEE Colloquium by **Robert Tamblyn Hughes** - University of Brighton.
- **Software Project Management** by [Mike Cotterell](#), [Bob Hughes](#) International Thomson Computer Press, 1995.
- **Project Management for IT-Related Projects:** Textbook for the ISEB Foundation Certificate in IS Project Management by Bob Hughes, Roger Ireland, Brian West, Norman Smith, David I. Shepherd British Computer Society, 2004
- An Introduction to Project Planning Paperback – Import, 1 February 2004 by [Jack Gido](#)
- Software Project Management Edited By Mandeep Kaur



SELECTION OF AN APPROPRIATE PROJECT APPROACH

Unit Structure

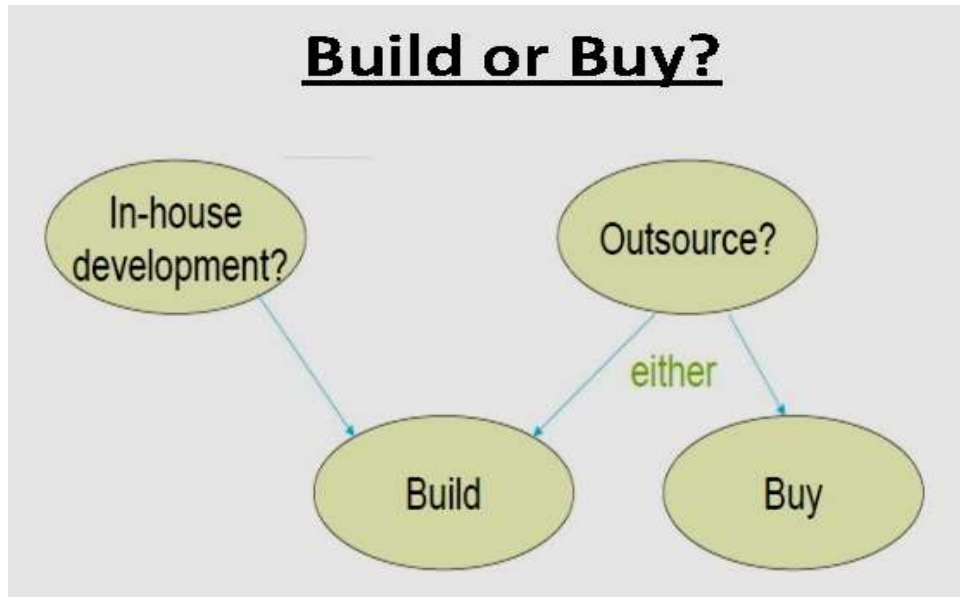
- 4.1 Introduction
- 4.2 Build or Buy?
 - 4.2.1 In- house projects
 - 4.2.2 Outsourced projects
 - 4.2.3 Off-the-shelf projects
- 4.3 Choosing Methodologies and Technologies
 - 4.3.1 Project Methodologies
- 4.4 Software Processes and Process Models
- 4.5 Choice of Process Models
- 4.6 The Waterfall Model
- 4.7 The Spiral Model
- 4.8 Summary
- 4.9 Reference for further reading
- 4.10 Model Questions

4.1 INTRODUCTION

This chapter is concerned with how the characteristics of a project environment and the application to be delivered influence the shape of the plan of a project.

4.2 BUILD OR BUY?

There are three types of projects namely In house projects, outsourced projects and off-the- shelf projects. Based on user requirements one among the three methods can be chosen.



4.2.1 In-house projects

In inhouse projects, developers, and the users of the software in the same organization and often the methods to be used are dictated by organizational standards

4.2.2 Outsourced projects

In outsourced projects the developers and the users of the software in the different organization and the need for tailoring arises as different customers have different needs

Why Outsourcing?

Time is needed to develop the software.

Would often require the recruitment of new technical staff to do the job.

Usually, the new staff won't be needed after the project is completed.

Sometimes due to the novelty of the project there may be lack of executives to lead the effort.

4.2.3 Off-the-shelf projects

In off-the –shelf projects a ready-made software product is purchased.

Why Buying?

- Whether in house or outsourced, software development is still involved.
- The contracting company will not have completely developed software readily available to the client.
- Considerable management effort is needed by the client to establish and manage the contract.

Advantages of off-the-shelf (OTS) software

- Cheaper as supplier can spread development costs over many customers.
- Software already exists
- Can be trialed by potential customer.
- No delay while software being developed.
- Where there have been existing users, bugs are likely to have been found and eradicated.

Disadvantages of off-the-shelf software

- Customer will have same application as everyone else so there is no competitive advantage.
- Customer may need to change the way they work to fit in with OTS application.
- Customer does not own the code and cannot change it.
- Danger of over-reliance on a single supplier.

4.3 CHOOSING METHODOLOGIES AND TECHNOLOGIES

To achieve goals and planned results within a defined schedule and a budget, a manager uses a project. Regardless of which field or which trade, there are assortments of methodologies to help managers at every stage of a project from the initiation to implementation to the closure.

A methodology is a model, which project managers employ for the design, planning, implementation, and achievement of their project objectives. There are different project management methodologies to benefit different projects.

For example, there is a specific methodology, which NASA uses to build a space station while the Navy employs a different methodology to build submarines. Hence, there are different project management methodologies that cater to the needs of different projects spanned across different business domains.

- Identify project as either objective driven or product driven
- Analyze other project characteristics
- Identify high level project risks
- Consider user requirement concerning implementation

Objective driven or product driven

Objective-driven project

- A project is to meet an objective.
- The Client may have a problem and asks a specialist to recommend solutions.

Product-driven project

- A project will be to create a product.
- The details of the product are provided by the client.

Analyze other project characteristics

- Will the software to be implemented a data-oriented or a process-oriented system?
- Will the software to be produced be a general tool or application specific?
- Are there specific tools available for implementing the application?
- Is the system knowledge-based?
- Will the system to be produced makes heavy use of computer graphics?
- Is the system to be created safety critical?
- Is the system designed to carry out predefined services or to be engaging and entertaining?
- What is the nature of the hardware/software environment in which the system will operate?

Identify high-level project risks

- The more uncertainty in the project the more the risk that the project will be unsuccessful.
- Recognizing the area of uncertainty allows taking steps towards reducing its uncertainty
- Uncertainty can be associated with the products, processes, or resources of a project.

Product uncertainty

- How well are the requirements understood?
- The users themselves could be uncertain about what the system is to do.

Process uncertainty

- For the project under consideration, the organization will use an approach or an application building-tool that it never used before.

Resource uncertainty

- The main area of resource uncertainty is the availability of the staff with the right ability and experience.
- Consider User requirements Concerning Implementation
- Imposing uniform applications and technologies throughout whole organization saves time and money at the end of organization.
- A client organization often lays down standards that must be adopted by any contractor providing software for them.

4.3.1 Project Methodologies

Following are the most frequently used project management methodologies in the project management practice:

Adaptive Project Framework

In this methodology, the project scope is a variable. Additionally, the time and the cost are constants for the project. Therefore, during the project execution, the project scope is adjusted to get the maximum business value from the project.

Agile Software Development

Agile software development methodology is for a project that needs extreme agility in requirements. The key features of agile are its short-termed delivery cycles (sprints), agile requirements, dynamic team culture, less restrictive project control and emphasis on real-time communication.

Crystal Methods

In crystal method, the project processes are given a low priority. Instead of the processes, this method focuses more on team communication, team member skills, people, and interaction. Crystal methods come under agile category.

Dynamic Systems Development Model (DSDM)

This is the successor of Rapid Application Development (RAD) methodology. This is also a subset of agile software development methodology and boasts about the training and documents support this methodology has. This method emphasizes more on the active user involvement during the project life cycle.

Extreme Programming (XP)

Lowering the cost of requirement changes is the main objective of extreme programming. XP emphasizes on fine scale feedback, continuous process, shared understanding, and programmer welfare. In XP, there is no detailed requirements specification or software architecture built.

Feature Driven Development (FDD)

This methodology is more focused on simple and well-defined processes, short iterative and feature driven delivery cycles. All the planning and execution in this project type take place based on the features.

Information Technology Infrastructure Library (ITIL)

This methodology is a collection of best practices in project management. ITIL covers a broad aspect of project management which starts from the organizational management level.

Joint Application Development (JAD)

Involving the client from the early stages with the project tasks is emphasized by this methodology. The project team and the client hold JAD sessions collaboratively to get the contribution from the client. These JAD sessions take place during the entire project life cycle.

Lean Development (LD)

Lean development focuses on developing change-tolerance software. In this method, satisfying the customer comes as the highest priority. The team is motivated to provide the highest value for the money paid by the customer.

PRINCE2

PRINCE2 takes a process-based approach to project management. This methodology is based on eight high-level processes.

Rapid Application Development (RAD)

This methodology focuses on developing products faster with higher quality. When it comes to gathering requirements, it uses the workshop method. Prototyping is used for getting clear requirements and re-use the software components to accelerate the development timelines.

In this method, all types of internal communications are considered informal.

Rational Unified Process (RUP)

RUP tries to capture all the positive aspects of modern software development methodologies and offer them in one package. This is one of the first project management methodologies that suggested an iterative approach to software development.

Scrum

This is an agile methodology. The main goal of this methodology is to improve team productivity dramatically by removing every possible burden. Scrum projects are managed by a Scrum master.

Spiral

Spiral methodology is the extended waterfall model with prototyping. This method is used instead of using the waterfall model for large projects.

Systems Development Life Cycle (SDLC)

This is a conceptual model used in software development projects. In this method, there is a possibility of combining two or more project management methodologies for the best outcome. SDLC also heavily emphasizes on the use of documentation and has strict guidelines on it.

Waterfall (Traditional)

This is the legacy model for software development projects. This methodology has been in practice for decades before the new methodologies were introduced. In this model, development lifecycle has fixed phases and linear timelines. This model is not capable of addressing the challenges in the modern software development domain.

Selecting the most suitable project management methodology could be a tricky task. When it comes to selecting an appropriate one, there are a few dozens of factors to be considered. Each project management methodology carries its own strengths and weaknesses. Therefore, there is no good or bad methodology and what you should follow is the most suitable one for your project management requirements.

4.4 SOFTWARE PROCESS AND PROCESS MODELS

Software Processes is a coherent set of activities for specifying, designing, implementing, and testing software systems. There are many different software processes, but all involve:

- Specification – defining what the system should do.
- Design and implementation – defining the organization of the system and implementing the system.
- Validation – checking that it does what the customer wants.
- Evolution – changing the system in response to changing customer needs.

What is a software process model?

A software process model is an abstraction of the software development process. A software process model is an abstract representation of a process that presents a description of a process from some perspective. The models specify the stages and order of a process. So, think of this as a representation of the **order of activities** of the process and the **sequence** in which they are performed.

A model will define the following:

- The tasks to be performed
- The input and output of each task
- The pre and post conditions for each task
- The flow and sequence of each task

The goal of a software process model is to provide guidance for controlling and coordinating the tasks to achieve the product and objectives as effectively as possible.

Factors in choosing a software process

Choosing the right software process model for your project can be difficult. If you know your requirements well, it will be easier to select a model that best matches your needs. You need to keep the following factors in mind when selecting your software process model:

Project requirements

Before choosing a model, take some time to go through the project requirements and clarify them alongside your organizations or team's expectations. Will the user need to specify requirements in detail after each iterative session? Will the requirements *change* during the development process?

Project size

Consider the size of the project you will be working on. Larger projects mean bigger teams, so you'll need more extensive and elaborate project management plans.

Project complexity

Complex projects may not have clear requirements. The requirements may change often, and the cost of delay is high. Ask yourself if the project requires constant monitoring or feedback from the client.

Cost of delay

Is the project highly time-bound with a huge cost of delay, or are the timelines flexible?

Customer involvement

Do you need to consult the customers during the process? Does the user need to participate in all phases?

Familiarity with technology

This involves the developers' knowledge and experience with the project domain, software tools, language, and methods needed for development.

Project resources

This involves the amount and availability of funds, staff, and other resources.

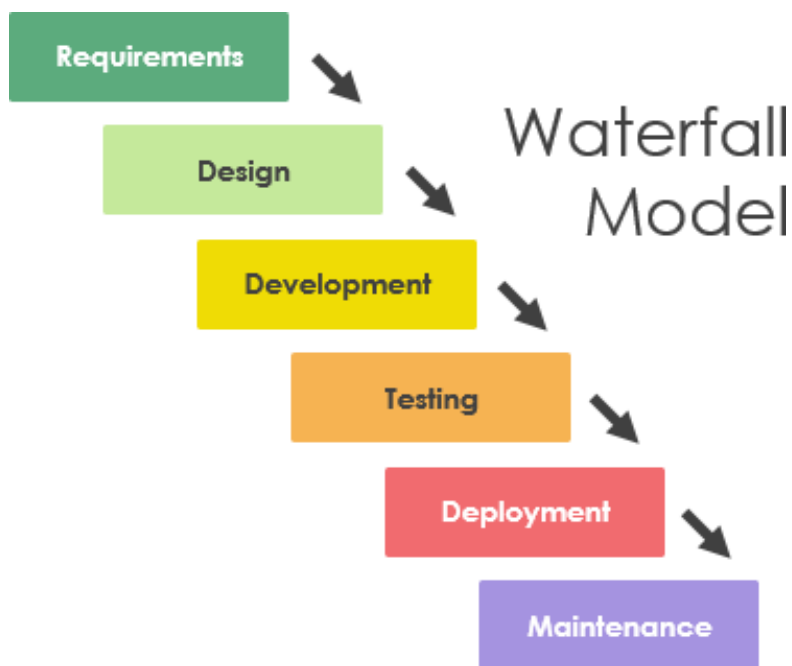
Software Process and Software Development Lifecycle(SDLC) Model

One of the basic notions of the software development process is SDLC models which stand for Software Development Life Cycle models. There are many development life cycle models that have been developed to achieve different required objectives. The models specify the various stages of the process and the order in which they are carried out. The most used, popular, and important SDLC models are given below:

- Waterfall model
- V model
- Incremental model
- RAD model
- Agile model
- Iterative model
- Spiral model
- Prototype model

Waterfall Model

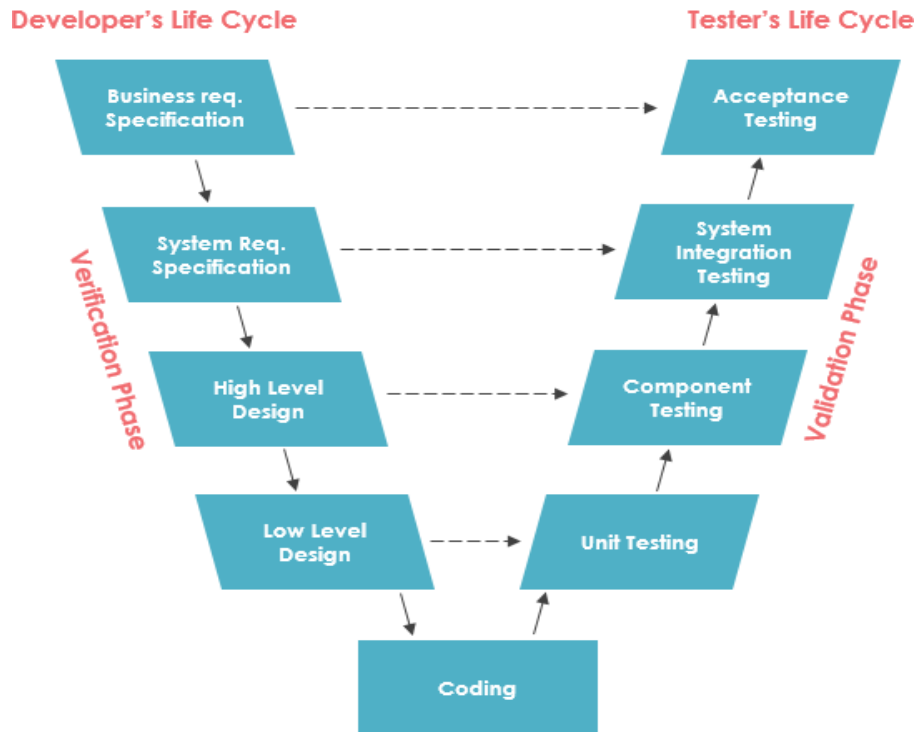
The waterfall model is a breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialisation of tasks. The approach is typical for certain areas of engineering design.



V Model

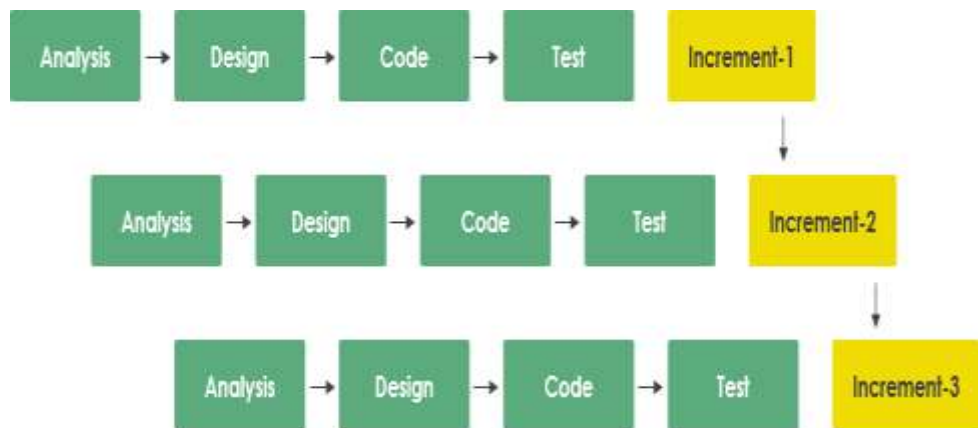
The V-model represents a development process that may be considered an extension of the waterfall model and is an example of the more general V-model. Instead of moving down in a linear way, the process steps are bent upwards after the coding phase, to form the typical V shape. The V-Model demonstrates the relationships between each phase of the development life cycle and its associated phase of testing. The horizontal and vertical axes represent time or project completeness (left-to-right) and level of abstraction (coarsest-grain abstraction uppermost), respectively.

V-Model



Incremental model

The incremental build model is a method of software development where the model is designed, implemented, and tested incrementally (a little more is added each time) until the product is finished. It involves both development and maintenance. The product is defined as finished when it satisfies all its requirements. Each iteration passes through the requirements, design, coding, and testing phases. And each subsequent release of the system adds function to the previous release until all designed functionality has been implemented. This model combines the elements of the waterfall model with the iterative philosophy of prototyping.

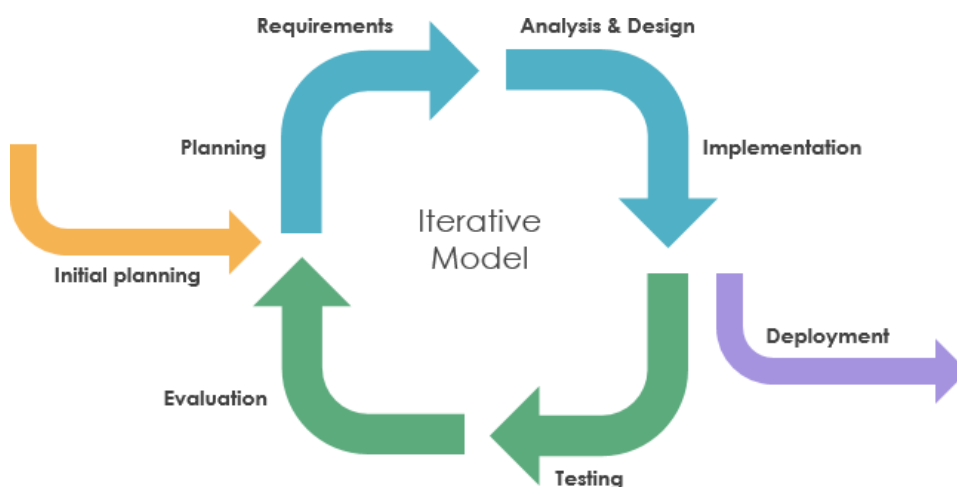


Incremental Model

Iterative Model

An iterative life cycle model does not attempt to start with a full specification of requirements by first focusing on an initial, simplified set user feature, which then progressively gains more complexity and a broader set of features until the targeted system is complete. When adopting the iterative approach, the philosophy of incremental development will also often be used liberally and interchangeably.

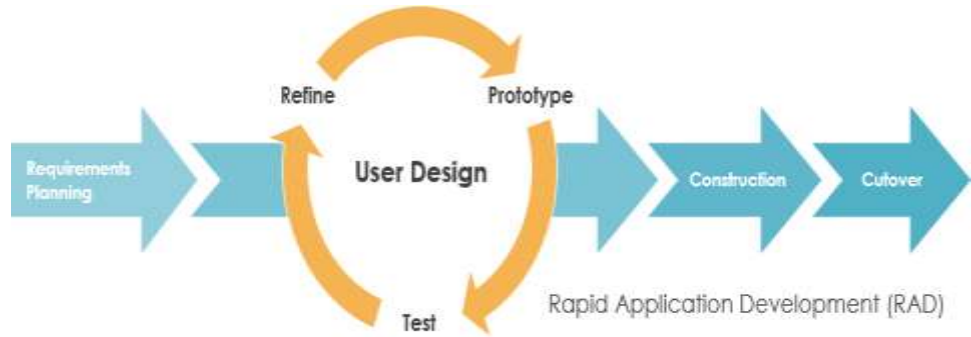
In other words, the iterative approach begins by specifying and implementing just part of the software, which can then be reviewed and prioritized to identify further requirements. This iterative process is then repeated by delivering a new version of the software for each iteration. In a light-weight iterative project the code may represent the major source of documentation of the system; however, in a critical iterative project a formal software specification may also be required.



RAD model

Rapid application development was a response to plan-driven waterfall processes, developed in the 1970s and 1980s, such as the Structured Systems Analysis and Design Method (SSADM). Rapid application development (RAD) is often referred as the adaptive software development. RAD is an incremental prototyping approach to software development that end users can produce better feedback when examining a live system, as opposed to working strictly with documentation. It puts less emphasis on planning and more emphasis on an adaptive process.

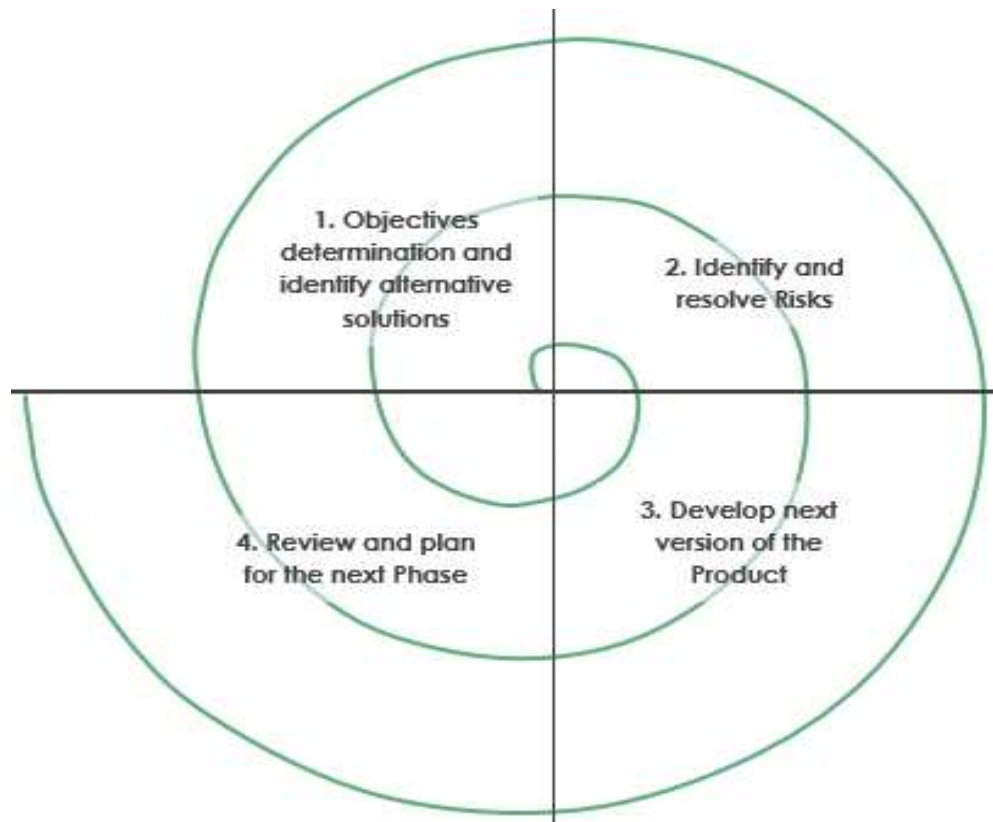
RAD may result in a lower level of rejection when the application is placed into production, but this success most often comes at the expense of a dramatic overrun in project costs and schedule. RAD approach is especially well suited for developing software that is driven by user interface requirements. Thus, some GUI builders are often called rapid application development tools.



Spiral model

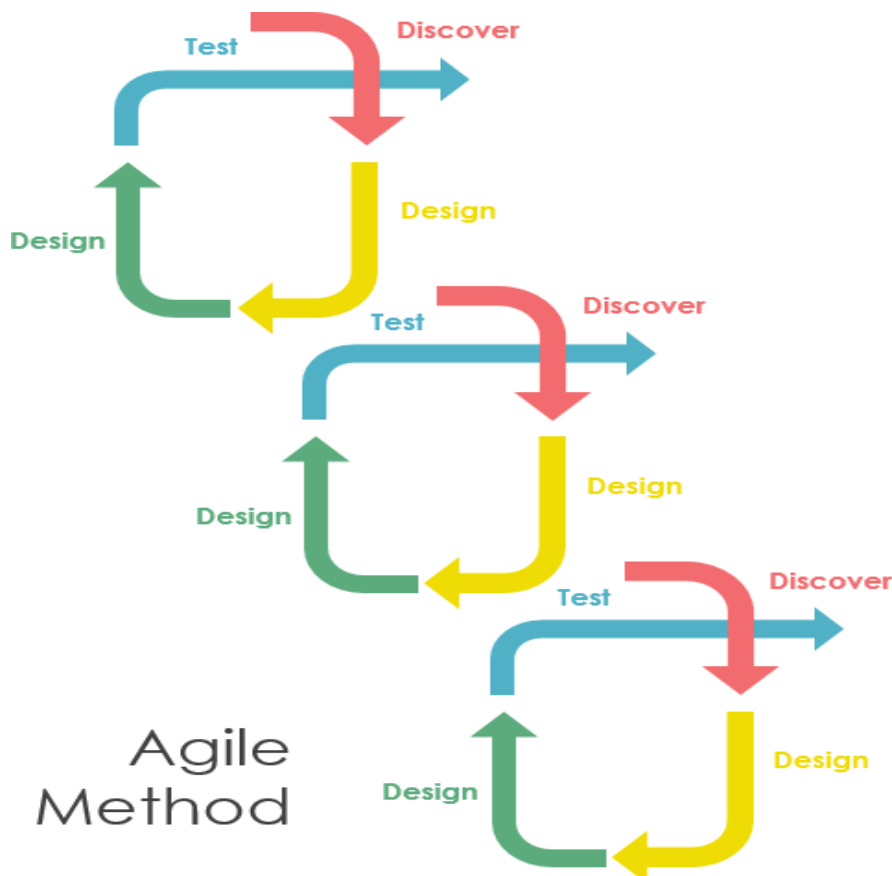
The spiral model, first described by Barry Boehm in 1986, is a risk-driven software development process model which was introduced for dealing with the shortcomings in the traditional waterfall model. A spiral model looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. This model supports risk handling, and the project is delivered in loops. Each loop of the spiral is called a Phase of the software development process.

The initial phase of the spiral model in the early stages of Waterfall Life Cycle that is needed to develop a software product. The exact number of phases needed to develop the product can be varied by the project manager depending upon the project risks. As the project manager dynamically determines the number of phases, so the project manager has an important role to develop a product using a spiral model.



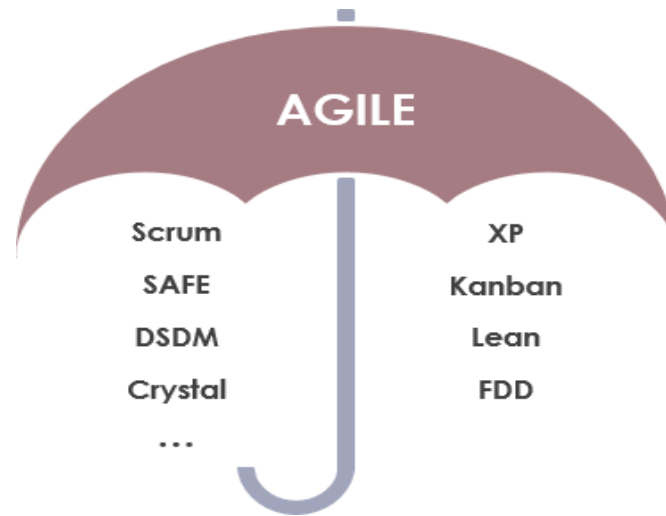
Agile model

Agile is an umbrella term for a set of methods and practices based on the values and principles expressed in the Agile Manifesto that is a way of thinking that enables teams and businesses to innovate, quickly respond to changing demand, while mitigating risk. Organizations can be agile using many of the available frameworks available such as Scrum, Kanban, Lean, Extreme Programming (XP) and etc.



The Agile movement proposes alternatives to traditional project management. Agile approaches are typically used in software development to help businesses respond to unpredictability which refer to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams.

The primary goal of being Agile is empowered the development team the ability to create and respond to change to succeed in an uncertain and turbulent environment. Agile software development approach is typically operated in rapid and small cycles. This results in more frequent incremental releases with each release building on previous functionality. Thorough testing is done to ensure that software quality is maintained.



4.5 CHOICE OF PROCESS MODELS

The word “process” is sometimes used to emphasize the idea of a system in action. To achieve an outcome, the system will have to execute one or more activities: this is its process. This idea can be applied to the development of computer-based systems where several interrelated activities must be undertaken to create a final product. These activities can be organized in different ways, and we can call these process models. A major part of the planning will be the choosing of the development methods to be used and the slotting of these into an overall process model.

The planner needs not only to select methods but also to specify how the method is to be applied. With methods such as SSADM, there is a considerable degree of choice about how it is to be applied: not all parts of SSADM are compulsory. Many student projects have the rather basic failing that at the planning stage they claim that SSADM is to be used: in the event, all that is produced are a few SSADM fragments such as a top-level data flow diagram and a preliminary logical data structure diagram. If this is all the project requires, it should be stated at the outset.

The software process model framework is specific to the project. Thus, it is essential to select the software process model according to the software which is to be developed. The software project is considered efficient if the process model is selected according to the requirements. It is also essential to consider time and cost while choosing a process model as cost and/ or time constraints play an important role in software development. The basic characteristics required to select the process model are project type and associated risks, requirements of the project, and the users.

One of the key features of selecting a process model is to understand the project in terms of size, complexity, funds available, and so on. In addition, the risks which are associated with the project should also be considered. Note that only a few process models emphasize risk assessment. Various other issues related to the project and the risks are listed in Table.

Table Selections on the Basis of the Project Type and Associated Risks

Project Type and Associated Risks	Waterfall	Prototype	Spiral	RAD	Formal Methods
Reliability requirements	No	No	Yes	No	Yes
Stable funds	Yes	Yes	No	Yes	Yes
Reuse components	No	Yes	Yes	Yes	Yes
Tight project schedule	No	Yes	Yes	Yes	No
Scarcity of resources	No	Yes	Yes	No	No

The most essential feature of any process model is to understand the requirements of the project. In case the requirements are not clearly defined by the user or poorly understood by the developer, the developed software leads to ineffective systems. Thus, the requirements of the software should be clearly understood before selecting any process model. Various other issues related to the requirements are listed in Table.

Table Selection on the Basis of the Requirements of the Project

Requirements of the Project	Waterfall	Prototype	Spiral	RAD	Formal Methods
Requirements are defined early in SDLC	Yes	No	No	Yes	No
Requirements are easily defined and understandable	Yes	No	No	Yes	Yes
Requirements are changed frequently	No	Yes	Yes	No	Yes
Requirements indicate a complex System	No	Yes	Yes	No	No

Software is developed for the users. Hence, the users should be consulted while selecting the process model. The comprehensibility of the project increases if users are involved in selecting the process model. It is possible that a user is aware of the requirements or has a rough idea of the requirements. It is also possible that the user wants the project to be developed in a sequential manner or an incremental manner (where a part is delivered to the user for use). Various other issues related to the user's satisfaction are listed in Table.

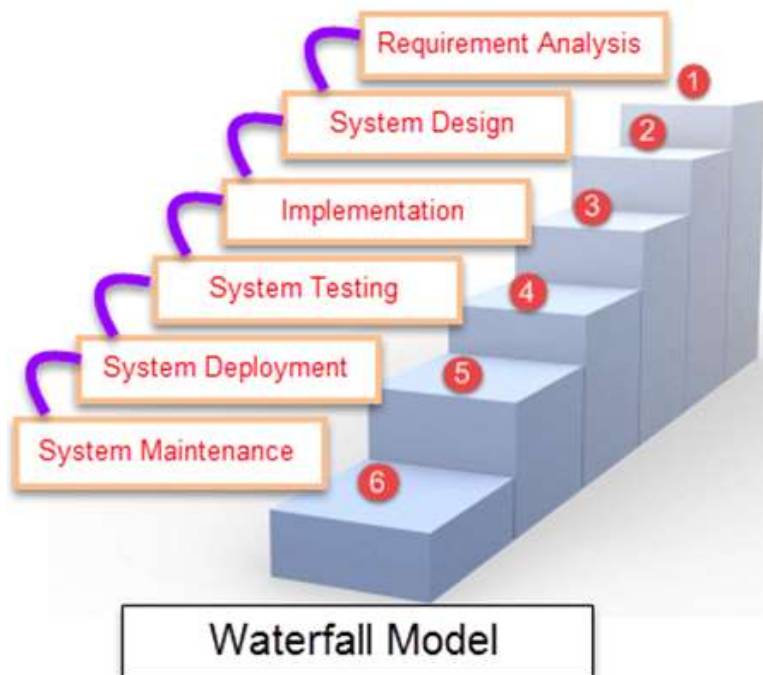
Table Selection on the Basis of the Users

User Involvement	Waterfall	Prototype	Spiral	RAD	Formal Methods
Requires Limited User Involvement	Yes	No	Yes	No	Yes
User participation in all phases	No	Yes	No	Yes	No
No experience of participating in similar projects	No	Yes	Yes	No	Yes

4.6 WATERFALL MODEL

Waterfall Model is a sequential model that divides software development into pre-defined phases. Each phase must be completed before the next phase can begin with no overlap between the phases. Each phase is designed for performing specific activity during the SDLC phase. It was introduced in 1970 by Winston Royce. The Waterfall Model was the first Process Model to be introduced. It is also referred to as a **linear-sequential life cycle model**. It is very simple to understand and use.

The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.



Different Phases of Waterfall Model in Software Engineering

Following are the different Waterfall Model phases:

Different phases	Activities performed in each stage
Requirement Gathering stage	<ul style="list-style-type: none"> • During this phase, detailed requirements of the software system to be developed are gathered from client
Design Stage	<ul style="list-style-type: none"> • Plan the programming language, for Example Java, PHP, .net • or database like Oracle, MySQL, etc. • Or other high-level technical details of the project
Built Stage	<ul style="list-style-type: none"> • After design stage, it is built stage, that is nothing but coding the software
Test Stage	<ul style="list-style-type: none"> • In this phase, you test the software to verify that it is built as per the specifications given by the client.
Deployment stage	<ul style="list-style-type: none"> • Deploy the application in the respective environment
Maintenance stage	<ul style="list-style-type: none"> • Once your system is ready to use, you may later require change the code as per customer request

Advantages and Disadvantages of Waterfall Model

Here are the popular advantages of Waterfall model in Software Engineering with some disadvantages:

Advantages	Dis-Advantages
Before the next phase of development, each phase must be completed	Error can be fixed only during the phase
Suited for smaller projects where requirements are well defined	It is not desirable for complex project where requirement changes frequently
They should perform quality assurance test (Verification and Validation) before completing each stage	Testing period comes quite late in the developmental process
Elaborate documentation is done at every phase of the software's development cycle	Documentation occupies a lot of time of developers and testers
Project is completely dependent on project team with minimum client intervention	Clients valuable feedback cannot be included with ongoing development phase
Any changes in software is made during the process of the development	Small changes or errors that arise in the completed software may cause a lot of problems

Examples of Waterfall Model

In the olden days, Waterfall model was used to develop enterprise applications like Customer Relationship Management (CRM) systems, Human Resource Management Systems (HRMS), Supply Chain Management Systems, Inventory Management Systems, Point of Sales (POS) systems for Retail chains etc.

Waterfall model was used significantly in the development of software till the year 2000. Even after the **Agile manifesto** was published in 2001, Waterfall model continued to be used by many organization till the last decade. These days most project follow **Agile Methodology**, some form of **Iterative model** or one of the other models depending on their project specific requirement.

In the olden days, applications developed in Waterfall Model like CRM Systems, Supply Chain Management Systems etc would usually take a year or longer to develop. With the evolution of technology, there were cases where large scale enterprise systems were developed over a period of 2 to 3 years but were redundant by the time they were completed. There were several reasons for this.

- By the time the applications were developed in C, C++ etc, new languages (relatively speaking) like Java, .Net etc would replace them with web-based functionality.
- Even if the application was developed using a new technology, factors like more competitors entering the market, cheaper alternatives becoming available, better functionality using newer technologies, change in customers requirement etc. increase the risk of developing an application over several years.
- However, there are some areas where Waterfall model was continued to be preferred.
- Consider a system where human life is on the line, where a system failure could result in one or more deaths.
- In some countries, such mishaps could lead to imprisonment for those who are accountable.
- Consider a system where time and money were secondary considerations and human safety was first.
- In such situations, Waterfall model was the preferred approach.
- Development of Department of Defense (DOD), military and aircraft programs followed Waterfall model in many organizations.
- This is because of the strict standards and requirements that have to be followed.
- In such industries, the requirements are known well in advance and contracts are very specific about the deliverable of the project.

- DOD Agencies typically considered Waterfall model to be compatible with their acquisition process and rigorous oversight process required by the government.

Having said that, even these industries are being disrupted using Iterative model and Agile methodology by organizations like Space X and others. Waterfall model was also used in banking, healthcare, control system for nuclear facilities, space shuttles etc

When to use the waterfall model

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- This model is used only when the requirements are very well known, clear and fixed.
- Application is not complicated and big
- Environment is stable
- Technology and tools used are not dynamic and is stable
- Product definition is stable.
- Technology is understood.
- There are no ambiguous requirements
- Ample resources with required expertise are available freely
- The project is short.

In Waterfall model, very less customer interaction is involved during the development of the product. Once the product is ready then only it can be demonstrated to the end users. Once the product is developed and if any failure occurs then the cost of fixing such issues is very high, because we need to update everything from document till the logic. In today's world, Waterfall model has been replaced by other models like iterative, agile etc.

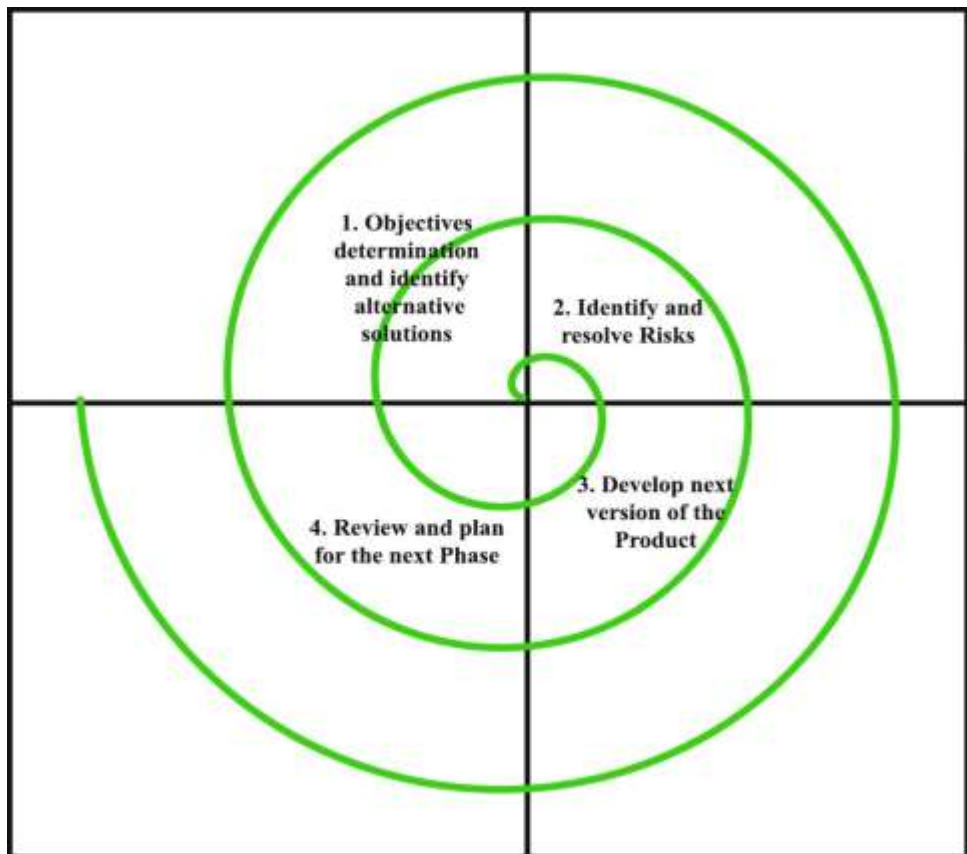
4.7 SPIRAL MODEL

Spiral model is one of the most important Software Development Life Cycle models, which provides support for **Risk Handling**. In its diagrammatic representation, it looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called a **Phase of the software development process**. The exact number of phases needed to develop the product can be varied by the project manager depending upon the project risks. As the project manager dynamically determines the number of phases, so the project manager has an important role to develop a product using the spiral model. The Radius of the spiral at any point represents the expenses(cost) of the project so far, and the angular dimension represents the progress made so far in the current phase.

This Spiral model is a combination of iterative development process model and sequential linear development model i.e., the waterfall model with a very high emphasis on risk analysis. It allows incremental releases of the product or incremental refinement through each iteration around the spiral.

The **risk-driven** feature of the spiral model allows it to accommodate any mixture of a specification-oriented, prototype-oriented, simulation-oriented, or another type of approach. An essential element of the model is that each period of the spiral is completed by a review that includes all the products developed during that cycle, including plans for the next cycle. The spiral model works for development as well as enhancement projects.

The below diagram shows the different phases of the Spiral Model: –



Each phase of the Spiral Model is divided into four quadrants as shown in the above figure. The functions of these four quadrants are discussed below-

- **Objectives determination and identify alternative solutions:** Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.
- **Identify and resolve Risks:** During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the

risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.

- **Develop next version of the Product:** During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.
- **Review and plan for the next Phase:** In the fourth quadrant, the Customers evaluate the so far developed version of the software. In the end, planning for the next phase is started.

Steps of the spiral model

While the phases are broken down into quadrants, each quadrant can be further broken down into the steps that occur within each one. The steps in the spiral model can be generalized as follows:

- The new system requirements are defined in as much detail as possible. This usually involves interviewing several users representing all the external or internal users and other aspects of the existing system.
- A preliminary design is created for the new system.
- A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system and represents an approximation of the characteristics of the final product.
- A second prototype is evolved by a fourfold procedure: (1) evaluating the first prototype in terms of its strengths, weaknesses, and risks; (2) defining the requirements of the second prototype; (3) planning and designing the second prototype; (4) constructing and testing the second prototype.
- The entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation and other factors that could result in a less-than-satisfactory final product.
- The existing prototype is evaluated in the same manner as was the previous prototype, and, if necessary, another prototype is developed from it according to the fourfold procedure outlined above.
- The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.
- The final system is constructed, based on the refined prototype.
- The final system is thoroughly evaluated and tested. Routine maintenance is carried out on a continuing basis to prevent large-scale failures and to minimize downtime.

Risk Handling in Spiral Model

A risk is any adverse situation that might affect the successful completion of a software project. The most important feature of the spiral model is handling these unknown risks after the project has started. Such risk resolutions are easier done by developing a prototype. The spiral model supports coping up with risks by providing the scope to build a prototype at every phase of the software development.

The Prototyping Model also supports risk handling, but the risks must be identified completely before the start of the development work of the project. But in real life project risk may occur after the development work starts, in that case, we cannot use the Prototyping Model. In each phase of the Spiral Model, the features of the product are defined and analyzed, and the risks at that point in time are identified and are resolved through prototyping. Thus, this model is much more flexible compared to other SDLC models.

Why is Spiral Model called Meta Model?

The Spiral model is called a Meta-Model because it subsumes all the other SDLC models. For example, a single loop spiral represents the Iterative Waterfall Model. The spiral model incorporates the stepwise approach of the Classical Waterfall Model. The spiral model uses the approach of the **Prototyping Model** by building a prototype at the start of each phase as a risk-handling technique. Also, the spiral model can be considered as supporting the evolutionary model – the iterations along the spiral can be considered as evolutionary levels through which the complete system is built.

When to use Spiral Model?

As mentioned before, the spiral model is best used in large, expensive, and complicated projects. Other uses include:

- projects in which frequent releases are necessary.
- projects in which changes may be required at any time.
- long term projects that are not feasible due to altered economic priorities.
- medium to high-risk projects.
- projects in which cost, and risk analysis is important.
- projects that would benefit from the creation of a prototype; and
- projects with unclear or complex requirements

Spiral Model Application

The Spiral Model is widely used in the software industry as it is in sync with the natural development process of any product, i.e., learning with maturity which involves minimum risk for the customer as well as the

development firms. The following pointers explain the typical uses of a Spiral Model –

- When there is a budget constraint and risk evaluation is important.
- For medium to high-risk projects.
- Long-term project commitment because of potential changes to economic priorities as the requirements change with time.
- Customer is not sure of their requirements which is usually the case.
- Requirements are complex and need evaluation to get clarity.
- New product line which should be released in phases to get enough customer feedback.
- Significant changes are expected in the product during the development cycle.

Advantages of Spiral Model:

Below are some advantages of the Spiral Model.

- **Risk Handling:** The projects with many unknown risks that occur as the development proceeds, in that case, Spiral Model is the best development model to follow due to the risk analysis and risk handling at every phase.
- **Good for large projects:** It is recommended to use the Spiral Model in large and complex projects.
- **Flexibility in Requirements:** Change requests in the Requirements at later phase can be incorporated accurately by using this model.
- **Customer Satisfaction:** Customer can see the development of the product at the early phase of the software development and thus, they habituated with the system by using it before completion of the total product.

Disadvantages of Spiral Model:

Below are some main disadvantages of the spiral model.

- **Complex:** The Spiral Model is much more complex than other SDLC models.
- **Expensive:** Spiral Model is not suitable for small projects as it is expensive.
- **Too much dependability on Risk Analysis:** The successful completion of the project is very much dependent on Risk Analysis. Without very highly experienced experts, it is going to be a failure to develop a project using this model.
- **Difficulty in time management:** As the number of phases is unknown at the start of the project, so time estimation is very difficult.

4.8 SUMMARY

This chapter is concerned with how the characteristics of a project environment and the application to be delivered influence the shape of the plan of a project. Introduction to most common process models and selection of the most appropriate of them for a project is also a part of this chapter.

4.9 REFERENCE FOR FURTHER READING

1. <https://www.slideshare.net/tumetr1/selection-of-an-appropriate-project-approach>
2. <https://slidetodoc.com/selection-of-an-appropriate-project-approach-contents-introduction/>
3. <https://slideplayer.com/slide/14117777/>
4. http://net481.yolasite.com/resources/Ch04_project_approach.pdf

4.10 MODEL QUESTION

1. Explain in detail about build or buy project approach
2. Discuss about choosing Methodologies and Technologies in project approach
3. Explain software process models
4. Discuss about Waterfall Model
5. Discuss on The Spiral Model

SOFTWARE PROTOTYPING

Unit Structure

- 5.0 Software Prototyping: Introduction
 - 5.0.1 What is software prototyping?
 - 5.0.2 The Advantages and Disadvantages of Software Prototyping
- 5.1 Incremental Delivery
 - 5.1.1 What is Incremental Model?
 - 5.1.2 Characteristics of Incremental model
 - 5.1.3 Advantages and Disadvantages of Incremental model
- 5.2 Dynamic Systems Development Method (DSDM)
 - 5.2.1 Definition of dynamic systems development method (DSDM)
 - 5.2.2 Key principles of the dynamic systems development method
 - 5.2.3 DSDM life cycle
- 5.3 Atern Project
 - 5.3.1 The Structure of an Atern Project
 - 5.3.2 Atern Principles
 - 5.3.3 The Roles and Responsibilities of an Atern Project
- 5.4 Rapid Application Development
 - 5.4.1 RAD Model - Pros and Cons
- 5.5 Agile Model :Introduction
 - 5.5.1 Phases of Agile Model
 - 5.5.2 Agile Testing Methods
 - Extreme Programming (XP)
 - Scrum
 - Lean Software Development
- 5.6 Managing Iterative Model
 - 5.6.1 Phases of Iterative Model
 - 5.6.2 *When to use the Iterative Model?*
 - 5.6.3 *Advantages (Pros) of Iterative Model*
 - 5.6.4 *Disadvantages (Cons) of Iterative Model*
- 5.7 Selecting the most appropriate process model
 - 5.7.1 Selection Process Parameters for a Software Life Cycle Model
 - 5.7.2 How to select the right SDLC
- 5.8 Summary
- 5.9 References for further reading
- 5.10 Model question

5.0 SOFTWARE PROTOTYPING: INTRODUCTION

The Software Prototyping refers to building software application prototypes which displays the functionality of the product under development but may not actually hold the exact logic of the original software.

Software prototyping is becoming very popular as a software development model, as it enables to understand customer requirements at an early stage of development. It helps get valuable feedback from the customer and helps software designers and developers understand about what exactly is expected from the product under development.

5.0.1 What Is Software Prototyping?

Software prototyping refers to the process of visualising a software product before it has been created. Creating software from scratch requires a great investment in the form of time, money, and effort. That is why most clients prefer to have a visual prototype developed before work is put into the development of the actual product. The prototype acts as a 'model' closely replicating the appearance, and sometimes the functionality, of the product that the client has in mind. Have you ever beta tested a software application? Have you played a game or used a program whose publishers said it wasn't quite up to par and they needed your opinions before developing the final product? If so, you have participated in one form of software prototyping.

5.0.2 The Advantages and Disadvantages of Software Prototyping

Are you wondering whether you should go for a prototype or tell your development team to get directly to the coding part? Here are a few upsides and downsides of prototyping that might help you make the smarter decision.

Get Started Right Away

Creating a new software application can sometimes feel like a very daunting task. It is impossible to think of everything you want your software to do. If you are not entirely sure, you don't have to unnecessarily delay production. As a start, you can bounce a few ideas off of your software provider. Their designers can get started with a rough model which can then be refined gradually.

Clearly Define Your Vision

Do you remember the first step of prototyping in software development? Gathering requirements! Before they start working on the prototype, the developers will want to know what you want the software to do. Once they start with the design process, the product will start to take a more certain shape and form. This will solve one problem for both you and the developer:

- When you see your product on-screen, you will be able to interact with it and see if there is anything you would want to change. Your idea might turn out way better than you had imagined. If not, you can easily let the developers know of anything that you feel is missing from the prototype.
- In case the developers miss something in the initial requirement gathering phase, they can now be clear on everything the product is supposed to be.

Communicate And Collaborate – Stay Involved!

Prototyping software requires a lot of back-and-forth between the client and the developer. With the right development team, you can both always stay in sync. Clients are encouraged to provide their feedback to the design team after each iteration. The consistent communication between both parties also helps to manage expectations better.

Get A Visual Guide

This one is especially for those who like to see quick results and is arguably the best (read: most fun!) part about prototyping in software development. While a basic prototype will probably help you make sure that the developers have got your ideas right, a high-fidelity prototype will allow you to see what your product will look like once it is complete. You can even interact with a high-fidelity prototype, go through the various screens, click on buttons, and make sure the app flows smoothly.

Achieve Greater Creativity

Considering how communication-intensive a process software prototyping is, it creates the chance for ideas to flow freely. Greater levels of collaboration, both between the client and the developers and amongst the development team members, allow all participants to pitch in and collectively solve problems. This proves particularly useful when trying to work out the kinks in the prototype, leading it to perfection.

Early User Acceptance Testing

Acquainting users with a new software application is quite a task. Imagine heading an organisation with hundreds of employees and feeling the need to switch to a new type of accounting software. When the new software is implemented organisation-wide, many of your employees might have a hard time trying to catch up with this drastic change. Despite extensive training sessions, some users might still face problems. This problem could very well be solved with the help of software prototyping in software engineering because it will help you catch any problems way before development is started on the new software.

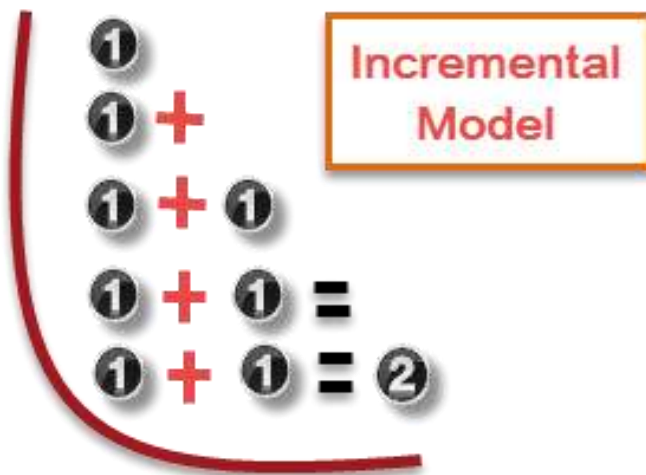
After thinking long and hard, we could come up with only two downsides to the software prototyping activity.

- It will add an extra step to the software development process.
- Over the course of this activity, you might want to add a lot more functionality to your app than you had originally intended.

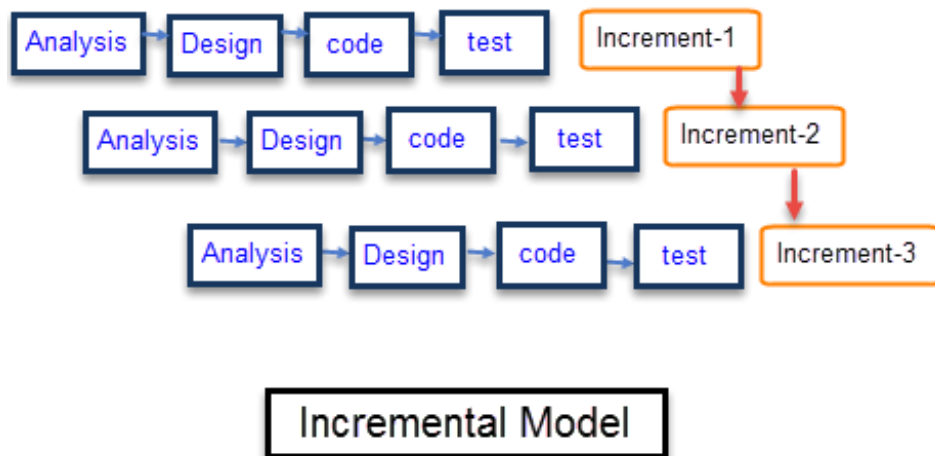
5.1 INCREMENTAL DELIVERY

5.1.1 What is Incremental Model?

Incremental Model is a process of software development where requirements are broken down into multiple standalone modules of software development cycle. Incremental development is done in steps from analysis design, implementation, testing/verification, maintenance.



Each iteration passes through the **requirements, design, coding and testing phases**. And each subsequent release of the system adds function to the previous release until all designed functionality has been implemented.



The system is put into production when the first increment is delivered. The first increment is often a core product where the basic requirements are addressed, and supplementary features are added in the next

increments. Once the core product is analyzed by the client, there is plan development for the next increment.

5.1.2 Characteristics of an Incremental model

- System development is broken down into many mini development projects
- Partial systems are successively built to produce a final total system
- Highest priority requirement is tackled first
- Once the requirement is developed, requirement for that increment are frozen

Incremental Phases	Activities performed in incremental phases
Requirement Analysis	Requirement and specification of the software are collected
Design	Some high-end functions are designed during this stage
Code	Coding of software is done during this stage
Test	Once the system is deployed, it goes through the testing phase

1.1.3 Advantages and Disadvantages of Incremental Model

Advantages	Disadvantages
The software will be generated quickly during the software life cycle	It requires a good planning designing
It is flexible and less expensive to change requirements and scope	Problems might cause due to system architecture as such not all requirements collected up front for the entire software lifecycle
Throughout the development stages changes can be done	Each iteration phase is rigid and does not overlap each other
This model is less costly compared to others	Rectifying a problem in one unit requires correction in all the units and consumes a lot of time
A customer can respond to each building	
Errors are easy to be identified	

5.2 DYNAMIC SYSTEMS DEVELOPMENT METHOD (DSDM)

DSDM (Dynamic Systems Development Method), the longest-established Agile method, launched in 1995, is the only Agile method to focus on the management of Agile projects. Arie van Bennekum represented DSDM at the launch of the Agile Alliance and their Agile Manifesto in 2001. DSDM has mainly operated in the corporate environment where it consistently demonstrates its ability to successfully work within and complement existing corporate processes. Practicing evolutionary development itself DSDM's latest version (Atern) incorporates those improvements.

5.2.1 Definition of dynamic systems development method (DSDM)

Like the wider agile family of methodologies, dynamic systems development method is an iterative approach to software development but adds additional discipline and structure to the process. Central to DSDM is the principle that “any project must be aligned to clearly defined strategic goals and focus upon early delivery of real benefits to the business’s advocates refer to it as a 'grown-up' version of agile for the corporate world.

5.2.2 Key principles of the dynamic systems development method

DSDM is structured around eight key principles:

1. **Focus on the business need:** DSDM teams must establish a valid business case and ensure organizational support throughout the project
2. **Deliver on time:** Work should be time-boxed and predictable, to build confidence in the development team.
3. **Collaborate:** DSDM teams must involve stakeholders throughout the project and empower all members of the team to make decisions.
4. **Quality:** To ensure high quality, the level of quality should be agreed with the business at the start of the project. This is enforced through continuous testing, review, and documentation.
5. **Build incrementally from firm foundations:** Teams must do enough design work up front (EDUF) to ensure they know exactly what to build, but not too much to slow development.
6. **Developer Iteratively:** Take feedback from the business and use this to continually improve with each development iteration. Teams must also recognize that details emerge as the project or product develops and they must respond to this.
7. **Communicate continuously and clearly:** Holding daily stand-up sessions, encouraging informal communication, running workshops and building prototypes are all key DSDM tools. Communicating

through documents is discouraged - instead, documentation must be lean and timely.

8. **Demonstrate control:** The project manager and team leader should make their plans and progress visible to all and focus on successful delivery.

5.2.3 DSDM life cycle

DSDM life cycle that defines 3 different unvarying cycles, preceded by 2 further life cycle activities:

1. **Feasibility Study:**

It establishes the essential business necessities and constraints related to the applying to be designed then assesses whether the application could be a viable candidate for the DSDM method.

2. **Business Study:**

It establishes the use and knowledge necessities that may permit the applying to supply business value; additionally, it is the essential application design and identifies the maintainability necessities for the applying.

3. **Functional Model Iteration:**

It produces a collection of progressive prototypes that demonstrate practicality for the client. (Note: All DSDM prototypes are supposed to evolve into the deliverable application.) The intent throughout this unvarying cycle is to collect further necessities by eliciting feedback from users as they exercise the paradigm.

4. **Design and Build Iteration:**

It revisits prototypes designed throughout useful model iteration to make sure that everyone has been designed during a manner that may alter it to supply operational business price for finish users. In some cases, useful model iteration and style and build iteration occur at the same time.

5. **Implementation**

It places the newest code increment (an “operationalized” prototype) into the operational surroundings. It ought to be noted that:

- the increment might not 100% complete or,
- changes are also requested because the increment is placed into place. In either case, DSDM development work continues by returning to the useful model iteration activity.

5.3 ATERN PROJECT

5.3.1 The Structure of an Atern Project

Atern differs from more common agile approaches as it encompasses the entire project lifecycle and not just software development (where Scrum prevails). It incorporates project management disciplines and provides mechanisms to ensure that the project benefits are clear, the proposed solution is feasible and there are solid foundations in place before detailed work is started.

There are seven phases to an Atern project:

Phase	Key Responsibilities
Pre-project	Initiation of the project, agreeing the Terms of Reference for the work
Feasibility	Typically, a short phase to assess the viability and the outline business case (justification).
Foundations	Key phase for ensuring the project is understood and defined well enough so that the scope can be baselined at a high level and the technology components and standards agreed before the development activity begins.
Exploration	Iterative development phase during which teams expand on the high-level requirements to demonstrate the functionality
Engineering	Iterative development phase where the solution is engineered to be deployable for release
Deployment	For each Increment (set of timeboxes) of the project the solution is made available.
Post project	Assesses the accrued benefits.

The Exploration and Engineering phases are often merged, as the method is flexible, allowing them to be organized to best suit the situation.

5.3.2 Atern Principles

Many organisations guide general behaviour with high-level values and culture. Well-understood principles are better guides than detailed process procedures. In Atern principles are used to provide guidance throughout the project.

Atern has eight underlying principles, and the complete framework can be directly derived from these. The principles are based on best practice in its truest sense. They define "the way things are done". Breaking one of these principles can lead to failure, as these are the basic building blocks for Atern, and bind together all the other elements of Atern.

Principal	Description
Focus on the Business Need	Deliver what the business needs when it needs it. The true business priorities must be understood with a sound business case.
Deliver on Time	Timeboxes are planned and the timeframe set. The dates never change; features are varied depending on business priorities, to achieve the deadline.
Collaborate	Teams work in a spirit of active co-operation and commitment. Collaboration encourages understanding, speed and shared ownership. The teams must be empowered and include the business representatives.
Never Compromise on Quality	A solution must be "good enough". The level of quality is set at the outset. Projects must test early and continuously and review constantly.
Build Incrementally from Firm Foundations	Increments allow the business to take advantage of work before the final product is complete, encouraging stakeholder confidence and feedback. This is based on doing just enough upfront analysis to proceed and accepting that detail emerges later.
Develop Iteratively	Accept that work is not always right first time. Use Timeboxes to allow for change yet continuously confirm that the solution is the right one.
Communicate Continuously and Clearly	Use facilitated workshops, daily standups, modeling, prototyping, presentations and encourage informal face-to-face communication.
Demonstrate Control	The team needs to be proactive when monitoring and controlling progress in line with Foundations Phase. They need to constantly evaluate the project viability based on the business objectives.

5.3.3 The Roles and Responsibilities of an Atern Project

Atern defines the roles and responsibilities in such a way that it easy to imagine how existing roles and positions would fit into an Atern project.

Project Roles

Role	Key Responsibilities
Business Sponsor	Owns the business case. Ensures funding and resourcing. Guarantees effective decision-making and deals with escalations rapidly.
Project Manager	Entry point for project governance. High-level planning. Monitors progress, resource availability, project configuration, manages risk and escalated issues.
Business Visionary	Owns the business vision and impact on wider business changes. Monitors progress against the vision. Contributes to key requirements, design and review sessions.
Technical Coordinator	Agrees and controls technical architecture. Advises and coordinates teams. Identifies and manages technical risk. Ensures non-functional requirements are met.

Solution Development Roles

Role	Key Responsibilities
Team Leader	Focuses team to deliver on time. Encourages full team participation. Manages detailed time box activities and day-to-day activities. Ensures testing and review activities are scheduled and completed.
Business Ambassador	Contributes to all requirements, design, and review sessions. Provides the business view for all day-to-day decision making. Describes business scenarios to help design and test the solution. Provides assurance that the solution is correct. Coordinates business acceptance.
Solution Developer	Creates the solution and participates fully in all appropriate QA activities.
Solution Tester	Works with business roles to define test scenarios for the solution. Carries out full technical testing reporting results to the Team Leader and Technical Coordinator.
Business Analyst	Supports communication between business and technical members of the team. Manages all required products related to business requirements. Ensures business implications of day-to-day decisions are properly thought through.
Business Advisor	Provides specialist input, for example an accountant or a tax advisor. Usually an intended user of the solution.

Other Roles

Role	Key Responsibilities
Atern Coach	Helps teams new to Atern teams get the most out of Atern. Tailors Atern for the needs of the project. Not all aspects are needed all the time!
Workshop Facilitator	Manages and organizes workshops. Responsible for the context not the content. Independent.
Other Specialists	Experts required on a short-term basis, possibly technical e.g., Load-Test specialists etc.

5.4 Rapid Application Development

The **RAD (Rapid Application Development)** model is based on prototyping and iterative development with no specific planning involved. The process of writing the software itself involves the planning required for developing the product.

Rapid Application Development focuses on gathering customer requirements through workshops or focus groups, early testing of the prototypes by the customer using iterative concept, reuse of the existing prototypes (components), continuous integration and rapid delivery.

5.4.1 RAD Model - Pros and Cons

RAD model enables rapid delivery as it reduces the overall development time due to the reusability of the components and parallel development. RAD works well only if high skilled engineers are available and the customer is also committed to achieve the targeted prototype in the given time frame. If there is commitment lacking on either side the model may fail.

The advantages of the RAD Model are as follows –

- Changing requirements can be accommodated.
- Progress can be measured.
- Iteration time can be short with use of powerful RAD tools.
- Productivity with fewer people in a short time.
- Reduced development time.
- Increases reusability of components.
- Quick initial reviews occur.
- Encourages customer feedback.
- Integration from very beginning solves a lot of integration issues.

The disadvantages of the RAD Model are as follows –

- Dependency on technically strong team members for identifying business requirements.

- Only system that can be modularized can be built using RAD.
- Requires highly skilled developers/designers.
- High dependency on Modelling skills.
- Inapplicable to cheaper projects as cost of Modelling and automated code generation is very high.
- Management complexity is more.
- Suitable for systems that are component based and scalable.
- Requires user involvement throughout the life cycle.
- Suitable for project requiring shorter development times.

5.5 AGILE MODEL: INTRODUCTION

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like –

- Planning
- Requirements Analysis
- Design
- Coding
- Unit Testing and
- Acceptance Testing.

At the end of the iteration, a working product is displayed to the customer and important stakeholders.

5.5.1 Phases of Agile Model

Following are the phases in the Agile model are as follows:

1. Requirements gathering
2. Design the requirements
3. Construction/ iteration
4. Testing/ Quality assurance
5. Deployment
6. Feedback

1. **Requirements gathering:** In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.

2. **Design the requirements:** When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.
3. **Construction/ iteration:** When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.
4. **Testing:** In this phase, the Quality Assurance team examines the product's performance and looks for the bug.
5. **Deployment:** In this phase, the team issues a product for the user's work environment.
6. **Feedback:** After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

5.5.2 Agile Testing Methods

- Extreme Programming(XP)
- Scrum
- Lean Software Development
- Crystal
- Dynamic Software Development Method(DSDM)
- Feature Driven Development(FDD)

Extreme Programming(XP)

Extreme Programming (XP) is one of the numerous Agile frameworks applied by IT companies. But its key feature — emphasis on technical aspects of software development — distinguishes XP from the other approaches.

Software engineer Ken Beck introduced XP in the 90s with the goal of finding ways to write high-qualitative software quickly and being able to adapt to customers' changing requirements. XP is a set of engineering practices. Developers have to go beyond their capabilities while performing these practices. That's where the "extreme" in the framework's title comes from. To get a better understanding of these practices, we'll start with describing XP's lifecycle and the roles engaged in the process.

The process and roles of extreme programming

The XP framework normally involves 5 phases or stages of the development process that iterate continuously:

1. **Planning**, the first stage, is when the customer meets the development team and presents the requirements in the form of user stories to describe the desired result. The team then estimates the stories and creates a release plan broken down into iterations needed to cover the required functionality part after part. If one or more of the stories can't be estimated, so-called *spikes* can be introduced which means that further research is needed.
2. **Designing** is a part of the planning process but can be set apart to emphasize its importance. It's related to one of the main XP values that we'll discuss below — simplicity. A good design brings logic and structure to the system and allows to avoid unnecessary complexities and redundancies.
3. **Coding** is the phase during which the actual code is created by implementing specific XP practices such as coding standards, pair programming, continuous integration, and collective code ownership (the entire list is described below).
4. **Testing** is the core of extreme programming. It is the regular activity that involves both unit tests (automated testing to determine if the developed feature works properly) and acceptance tests (customer testing to verify that the overall system is created according to the initial requirements).
5. **Listening** is all about constant communication and feedback. The customers and project managers are involved to describe the business logic and value that is expected.

Development process entails the cooperation between several participants, each having his or her own tasks and responsibilities. Extreme programming puts people in the center of the system, emphasizing the value and importance of such social skills as communication, cooperation, responsiveness, and feedback. So, these roles are commonly associated with XP:

1. **Customers** are expected to be heavily engaged in the development process by creating user stories, providing continuous feedback, and making all the necessary business decisions related to the project.
2. **Programmers or developers** are the team members that create the product. They are responsible for implementing user stories and conducting user tests (sometimes a separate **Tester** role is set apart). Since XP is usually associated with cross-functional teams, the skill set of such members can be different.
3. **Trackers or managers** link customers and developers. It's not a required role and can be performed by one of the developers. These people organize the meetups, regulate discussions, and keep track of important progress KPIs.
4. **Coaches** can be included in the teams as mentors to help with understanding the XP practices. It's usually an outside assistant or external consultant who is not involved in the development process but has used XP before and so can help avoid mistakes.

Extreme programming practices

The practices of XP are a set of specific rules and methods that distinguishes it from other methodologies. When used in conjunction, they reinforce each other, help mitigate the risks of the development process, and lead to the expected high-quality result. XP suggests using 12 practices while developing software which can be clustered into four groups.

EXTREME PROGRAMMING PRACTICES	
Group	Practices
Feedback	<ul style="list-style-type: none"> ✓ Test-Driven Development ✓ The Planning Game ✓ On-site Customer ✓ Pair Programming
Continual Process	<ul style="list-style-type: none"> ✓ Continuous Integration ✓ Code Refactoring ✓ Small Releases
Code understanding	<ul style="list-style-type: none"> ✓ Simple Design ✓ Collective Code Ownership ✓ System Metaphor ✓ Coding Standards
Work conditions	<ul style="list-style-type: none"> ✓ 40-Hour Week

XP main practices

Test-Driven Development

Is it possible to write a clear code quickly? The answer is yes, according to XP practitioners. The quality of software derives from short development cycles that, in turn, allow for receiving frequent feedback. And valuable feedback comes from good testing. XP teams practice test-driven development technique (TDD) that entails writing an automated unit test before the code itself. According to this approach, every piece of code must pass the test to be released. So, software engineers thereby focus on writing code that can accomplish the needed function. That's the way TDD allows programmers to use immediate feedback to produce reliable software. You can learn more about improving software testing in our dedicated article.

The Planning Game

This is a meeting that occurs at the beginning of an iteration cycle. The development team and the customer get together to discuss and approve a product's features. At the end of the planning game, developers plan for the upcoming iteration and release, assigning tasks for each of them.

On-site Customer

As we already mentioned, according to XP, the end customer should fully participate in development. The customer should be present all the time to answer team questions, set priorities, and resolve disputes if necessary.

Pair Programming

This practice requires two programmers to work jointly on the same code. While the first developer focuses on writing, the other one reviews code, suggests improvements, and fixes mistakes along the way. Such teamwork results in high-quality software and faster knowledge sharing but takes about 15 percent more time. In this regard, it's more reasonable trying pair programming for long-term projects.

Code Refactoring

To deliver business value with well-designed software in every short iteration XP teams also use refactoring. The goal of this technique is to continuously improve code. Refactoring is about removing redundancy, eliminating unnecessary functions, increasing code coherency, and at the same time decoupling elements. *Keep your code clean and simple, so you can easily understand and modify it when required* would be the advice of any XP team member.

Continuous Integration

Developers always keep the system fully integrated. XP teams take iterative development to another level because they commit code multiple times a day, which is also called continuous delivery. XP practitioners understand the importance of communication. Programmers discuss which parts of the code can be re-used or shared. This way, they know exactly what functionality they need to develop. The policy of shared code helps eliminate integration problems. In addition, automated testing allows developers to detect and fix errors before deployment.

Small Releases

This practice suggests releasing the MVP quickly and further developing the product by making small and incremental updates. Small releases allow developers to frequently receive feedback, detect bugs early, and monitor how the product works in production. One of the methods of doing so is the continuous integration practice (CI) we mentioned before.

Simple Design

The best design for software is the simplest one that works. If any complexity is found, it should be removed. The right design should pass all tests, have no duplicate code, and contain the fewest possible methods and classes. It should also clearly reflect the programmer's intent.

XP practitioners highlight those chances to simplify design are higher after the product has been in production for some time. Don Wells advises writing code for those features you plan to implement right away rather than writing it in advance for other future features: "The best approach is to create code only for the features you are implementing while you search for enough knowledge to reveal the simplest design. Then refactor incrementally to implement your new understanding and design."

Coding Standards

A team must have common sets of coding practices, using the same formats and styles for code writing. Application of standards allows all team members to read, share, and refactor code with ease, track who worked on certain pieces of code, as well as make the learning faster for other programmers. Code written according to the same rules encourages collective ownership.

Collective Code Ownership

This practice declares a whole team's responsibility for the design of a system. Each team member can review and update code. Developers that have access to code won't get into a situation in which they don't know the right place to add a new feature. The practice helps avoid code duplication. The implementation of collective code ownership encourages the team to cooperate more and feel free to bring new ideas.

System Metaphor

System metaphor stands for a simple design that has a set of certain qualities. First, a design and its structure must be understandable to new people. They should be able to start working on it without spending too much time examining specifications. Second, the naming of classes and methods should be coherent. Developers should aim at naming an object as if it already existed, which makes the overall system design understandable.

40-Hour Week

XP projects require developers to work fast, be efficient, and sustain the product's quality. To adhere to these requirements, they should feel well and rested. Keeping the work-life balance prevents professionals from burnout. In XP, the optimal number of work hours must not exceed 45 hours a week. One overtime a week is possible only if there will be none the week after.

Advantages and disadvantages of XP

XP practices have been debated upon for decades, as its approach and methods are rather controversial in a number of aspects and can't be applied in just any project. Here, we'll try to define the pros and cons of XP methodology.



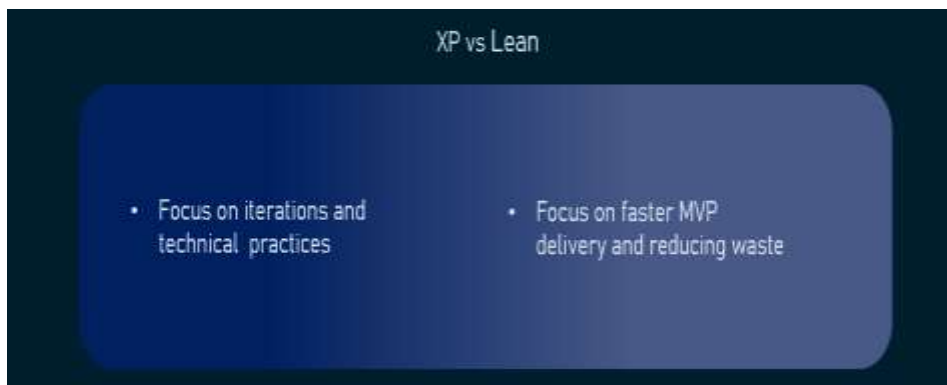
XP pros and cons in a nutshell

Comparison of XP to other frameworks

As we mentioned above, XP is part of the agile methodology. It shares the main agile principles, i.e., frequent releases, short development cycles, constant communication with the customer, cross-functional teams, and so on. For this reason, XP is often confused with other popular Agile frameworks such as Scrum, Kanban, and Lean. Check our detailed whitepaper to get more in-depth information or the infographics for a quick summary of the main agile methods. Here, we'll briefly compare them and see what the main differences are.

But before we dive in, it's important to note that XP is not really a project management framework, even though a lot of its practices overlap with those from the project management domain. So, its primary focus is on the technical aspects of development and the implementation of specific practices rather than the management and organizational sides.





XP vs Scrum and Lean in a nutshell

Extreme programming vs Scrum

Scrum is commonly associated with self-organizing teams. It also usually has sprints that are 2 to 4 weeks long, while XP iterations are shorter, taking 1 to 2 weeks. Besides, XP is much more flexible with possible changes within iterations, while Scrum doesn't allow any modifications after the sprint backlog is set. Another difference is that in XP the customer prioritizes features and decides on the order of their development, but in Scrum the team itself determines what to work on first.

Scrum's main roles are Product Owner, Scrum Master, and Scrum Team, which are different from those in XP.

However, there is no need to choose between XP and Scrum. Incorporating XP practices and Scrum techniques is considered quite effective with XP focusing on engineering aspects and Scrum organizing the process.

Extreme programming vs Lean

It's hard to compare XP and Lean because the latter is more of a philosophy or approach to the development process and bringing value to the customer. Its core principles include eliminating waste, deciding as late as possible, delivering as early as possible, and so on. So, Lean's focus is not on time-boxed iterations or specific engineering practices as in XP, but largely on a fast MVP delivery and reducing time waste.

Scrum

SCRUM is an agile development process focused primarily on ways to manage tasks in team-based development conditions.

There are three roles in it, and their responsibilities are:

- **Scrum Master:** The scrum can set up the master team, arrange the meeting and remove obstacles for the process
- **Product owner:** The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.

- **Scrum Team:** The team manages its work and organizes the work to complete the sprint or cycle.

What is Lean Software Development (LSD)?

Lean Software Development (LSD) is an agile framework based on optimizing development time and resources, eliminating waste, and ultimately delivering only what the product needs. The Lean approach is also often referred to as the Minimum Viable Product (MVP) strategy, in which a team releases a bare-minimum version of its product to the market, learns from users what they like, don't like and want to be added, and then iterates based on this feedback.

Advantages of LSD

LSD has proved to improve software development in following ways :

1. LSD removes the unnecessary process stages when designing a software so that it acts as a time saver as it simplifies the development process.
2. With focus on MVP, Lean Software Development prioritizes essential functions so this removes the risk of spending time on valueless builds.
3. It increases involvement power of your team as more and more members participate due to which the overall workflow becomes optimized, and losses get reduced.

Key Principles of Lean Software Development

There are 7 established lean principles which come with a set of tactics, practices and processes that build more efficient software products :

Eliminating Waste

Waste reduction, being the first rule in Lean engineering, defines its entire purpose. For the most part, the methodology tries to fight these 9 types of waste.

- Unnecessary features and code
- More tasks in log than can be completed
- Delays in the engineering process
- Vague requirements
- Inefficient communication
- Issues with quality
- Unneeded and crippling bureaucracy
- Data duplications

1. Costs of aforementioned

To identify and eliminate waste, regular meetings are held by Project Managers after each short iteration. They allow team members to report progress, point out bottlenecks and suggest which changes to implement during next iterations, which facilitates learning and allows improvements to the code to be implemented in small, manageable increments.

2. Building Quality In

Efficient quality management is, too, a guiding principle in lean development methodology, as issues in this area lead to different types of waste. Repetitive testing of the code, mistakes in logging and their resolution take time and therefore drive costs of development higher; lean strives to address such nuances before they even happen.

Various tactics are used in lean, and all related agile development types, to ensure that quality is maintained all along the process. Engineering is kept flexible. Every small iteration, each loop is followed by an immediate assessment. The time between software development stages is always reduced as much as possible and trade-offs (occasional sacrifices of quality for other project dimensions – time, costs and scope) are regularly discussed and considered.

Additionally, to fix bugs before the fact either Pair Programming or Test-Driven Development can be applied.

Pair Programming is the application of “two heads better than one” principle to software engineering. Each task is being completed by two developers, combined experience of which allows to figure out more effective solutions, foresee possible issues better and deliver higher quality than one of them would, singly.

Test-driven programming turns conventional “build, then scrutinize” approach upside down. Tests are written before the code is, so that an engineer, knowing precisely how features’ conditions are going to be checked, works out all probable scenarios whilst developing.

3. Amplifying knowledge

Lean software development originated from lean manufacturing, where the ultimate goal had always been a simplified, standardized, pipeline production which requires no knowledge and rare modifications. Therefore, when lean’s concept “amplify learning” was introduced to physical business, it was a game changer.

In software engineering, however, the importance of learning was never in doubt and lean development methodology, perhaps, only proven it once more. Each time the code is written, engineers reflect on it immediately and then incorporate, during following iterations,

the lessons they have learned. That's true of lean and all agile methodologies. To ensure that knowledge isn't accumulated exclusively by one engineer, who's writing a particular piece of code, lean methodology often uses paired programming.

Besides that, learning is amplified through ample code reviewing, meetings and establishment of metrics from data that are cross-team applicable. You, as a client, get to voice your feedback to the development team upon each iteration; collecting it and adjusting future efforts to your requirements is paramount to all lean developers.

4. Delaying commitment

It happens or rather used to happen often in traditional project management, that your application, though meeting the spec precisely, turned out completely unfit for the market by the date of release. Too many changes had emerged since your requests – in the business environment, in technologies your competitors use and in market's course overall; changes that had not been addressed over the course of development.

Lean software development methodology recognizes this threat. It always leaves room for improvement by postponing irreversible decisions until all the needed experimentation is done and as much info as possible is gathered; until you've checked and examined your requirements comprehensively and there are no doubts as to their relevance.

The methodology strives always to construct software to be flexible, so that when new knowledge is made available, engineers can act upon it without wrecking completely what's been done earlier. As all new projects, nowadays, are bound to face uncertainty, so the importance of this is hard to overestimate.

5. Delivering fast

Historically, meticulous, and long-term planning used to be the key to success in business. Only when each aspect of your strategy had been worked out thoroughly, agreed upon, strict milestones and pace of development had been established, you were considered ready to enter the software market.

As practice showed, however, such approach often led to a catastrophe. It made engineers spend too much time on building complex, monolithic systems packed with unneeded features. It restrained them from adapting the software to the ever-changing environment and client requirements.

As a result, lean engineers came up with the concept of MVP (minimum viable product) and overall opposite philosophy: build

quickly, include little functionality, and launch a product to the market as fast as possible. Then, study the reaction.

Such approach allows to enhance a piece of software incrementally, based on the feedback collected from real customers, and ditch everything that is of no value.

6. Respecting the team

Lean software development is a system aimed at empowering team members, rather than controlling them. It goes beyond establishing basic human courtesy; it instills trust within each project. Engineers are granted freedom to make important development decisions, based on knowledge they receive whilst writing code and their own judgment. Providing, of course, that they're experienced enough to do so.

Such approach contributes a lot to a faster application of changes to software that are needed to reflect the changes in the environment, and it keeps your developers motivated.

And what's more important than team's motivation?

Setting up a collaborative atmosphere, however, and keeping the perfect balance of control within the project is hard. Developers should be let to do their thing, implement changes that they feel are necessary, but they're also ought to report on their decisions; to explain the approach, they intend to take to managers and, more importantly, to you – the client. In the end, it's you who are in charge of the overall course.

7. Optimizing the entire value stream

According to Mary and Tom Poppedniecks, sub-optimizing is one of those unfortunate tendencies that, though being unproductive, still occurs often in traditional IT departments. Managers choose to break each issue into multiple constituent parts, which they then have their teams fix separately, without optimizing entire systems. Lean software development opposes that and stands for focusing on value stream.

At Perfection, for instance, we find people that are best suitable for each specific project and organize them into complete, standalone teams. That way, there aren't delays, disruptions, and miscommunications that would happen, surely, if project members were scattered across various departments.

Lean principles allow to optimize team's workflow, create unity among everyone involved in the project, inspire a sense of shared responsibility and shared objectives, which translates into higher performance.

Weakness in LSD :

- Make it scalable as other frameworks since it strongly depends on team involved.
- It is hard to keep with pace so it is not easy for developers to work with team members as conflict may occur in between them.
- It leads to difficult decision-making process as it is mandatory for customers to clearly set their requirements for the development not to be interrupted.

Lean Software Development is one of the proactive approaches that drives your body through productivity and cleanliness. It closely connects to Agile methodology, knowledge-sharing experience, fast product delivery. All processes and stags of development are accurately built to deliver the product at minimum cost in a timely manner.

5.6 MANAGING ITERATIVE MODEL

In this Model, you can start with some of the software specifications and develop the first version of the software. After the first version if there is a need to change the software, then a new version of the software is created with a new iteration. Every release of the Iterative Model finishes in an exact and fixed period that is called iteration.

The Iterative Model allows the accessing earlier phases, in which the variations made respectively. The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process.

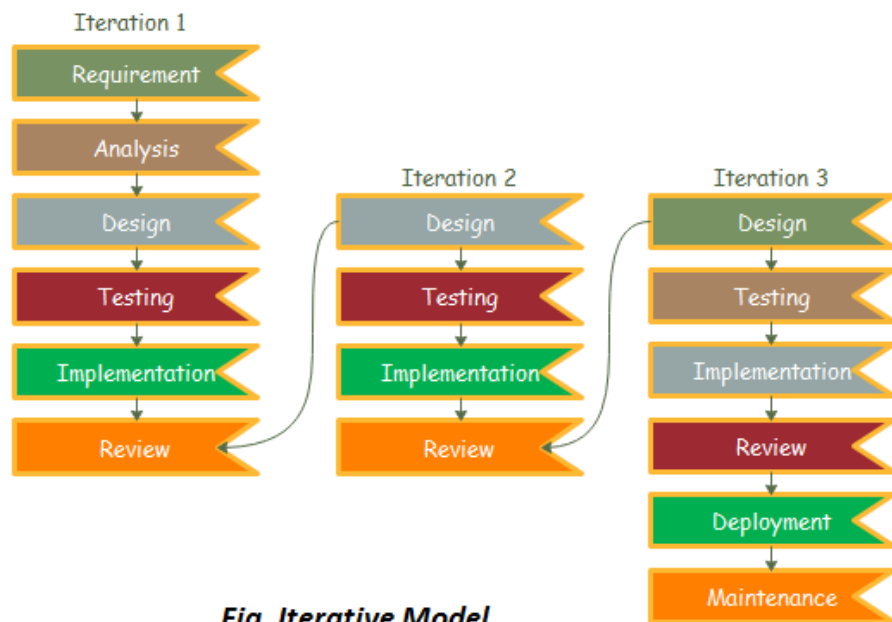


Fig. Iterative Model

5.6.1 Phases of Iterative Model

Requirement gathering & analysis: In this phase, requirements are gathered from customers and check by an analyst whether requirements will fulfil or not. Analyst checks that need will achieve within budget or not. After all of this, the software team skips to the next phase.

Design: In the design phase, team design the software by the different diagrams like Data Flow diagram, activity diagram, class diagram, state transition diagram, etc.

Implementation: In the implementation, requirements are written in the coding language and transformed into computer programmes which are called Software.

Testing: After completing the coding phase, software testing starts using different test methods. There are many test methods, but the most common are white box, black box, and grey box test methods.

Deployment: After completing all the phases, software is deployed to its work environment.

Review: In this phase, after the product deployment, review phase is performed to check the behaviour and validity of the developed product. And if there are any error found then the process starts again from the requirement gathering.

Maintenance: In the maintenance phase, after deployment of the software in the working environment there may be some bugs, some errors or new updates are required. Maintenance involves debugging and new addition options.

5.6.2 When to use the Iterative Model?

1. When requirements are defined clearly and easy to understand.
2. When the software application is large.
3. When there is a requirement of changes in future.

5.6.3 Advantages (Pros) of Iterative Model

1. Testing and debugging during smaller iteration is easy.
2. A Parallel development can plan.
3. It is easily acceptable to ever-changing needs of the project.
4. Risks are identified and resolved during iteration.
5. Limited time spent on documentation and extra time on designing.

5.6.4 Disadvantages (Cons) of Iterative Model

1. It is not suitable for smaller projects.
2. More Resources may be required.

3. Design can be changed again and again because of imperfect requirements.
4. Requirement changes can cause over budget.
5. Project completion date not confirmed because of changing requirements.

5.7 SELECTING THE MOST APPROPRIATE PROCESS MODEL

Selecting a Software Development Life Cycle (SDLC) methodology is a challenging task for many organizations and software engineers. What tends to make it challenging is the fact that few organizations know what the criteria are to use in selecting a methodology to add value to the organization. Fewer still understand that a methodology might apply to more than one Life Cycle Model. Before considering a framework for selecting a given SDLC methodology, we need to define the different types and illustrate the advantages and disadvantages of those models (please see the Software Development Life Cycle Models and Methodologies).

5.7.1 Selection Process Parameters for a Software Life Cycle Model

Selection Process parameters plays an important role in software development as it helps to choose the best suitable software life cycle model. Following are the parameters which should be used to select a SDLC.

Requirements characteristics :

- Reliability of Requirements
- How often the requirements can change
- Types of requirements
- Number of requirements
- Can the requirements be defined at an early stage
- Requirements indicate the complexity of the system

Development team :

- Team size
- Experience of developers on similar type of projects
- Level of understanding of user requirements by the developers
- Environment
- Domain knowledge of developers
- Experience on technologies to be used
- Availability of training

User involvement in the project :

- Expertise of user in project
- Involvement of user in all phases of the project
- Experience of user in similar project in the past

Project type and associated risk :

- Stability of funds
- Tightness of project schedule
- Availability of resources
- Type of project
- Size of the project
- Expected duration for the completion of project
- Complexity of the project
- Level and the type of associated risk

5.7.2 How to select the right SDLC

Selecting the right SDLC is a process that the organization can implement internally or consult for. There are some steps to get the right selection.

STEP 1: Learn the about SDLC Models

SDLCs are the same in their usage. To select the right SDLC, you should have enough experience and be familiar with the SDLCs that will be chosen and understand them correctly. As described in the software development life cycle models article, models are like the tools that important to know each tool usage to know which context it can fit into. Imagine the image below by Jacob Lawrence, if the carpenter did not know the tools he will use, what will be the results? Did you visualize the disaster?

STEP 2: Assess the needs of Stakeholders

We must study the business domain, stakeholders concerns and requirements, business priorities, our technical capability and ability, and technology constraints to be able to choose the right SDLC against their selection criteria.

STEP 3: Define the criteria

Some of the selection criteria or arguments that you may use to select an SDLC are:

- Is the SDLC suitable for the size of our team and their skills?
- Is the SDLC suitable for the selected technology we use for implementing the solution?

- Is the SDLC suitable for client and stakeholders concerns and priorities?
- Is the SDLC suitable for the geographical situation (distributed team)?
- Is the SDLC suitable for the size and complexity of our software?
- Is the SDLC suitable for the type of projects we do?
- Is the SDLC suitable for our software engineering capability?
- Is the SDLC suitable for the project risk and quality insurance?

What are the criteria?

Factors	Waterfall	V-Shaped	Evolutionary Prototyping	Spiral	Iterative and Incremental	Agile
Unclear User Requirement	Poor	Poor	Good	Excellent	Good	Excellent
Unfamiliar Technology	Poor	Poor	Excellent	Excellent	Good	Poor
Complex System	Good	Good	Excellent	Excellent	Good	Poor
Reliable system	Good	Good	Poor	Excellent	Good	Good
Short Time Schedule	Poor	Poor	Good	Poor	Excellent	Excellent
Strong Project Management	Excellent	Excellent	Excellent	Excellent	Excellent	Excellent
Cost limitation	Poor	Poor	Poor	Poor	Excellent	Excellent
Visibility of Stakeholders	Good	Good	Excellent	Excellent	Good	Excellent
Skills limitation	Good	Good	Poor	Poor	Good	Poor
Documentation	Excellent	Excellent	Good	Good	Excellent	Poor
Component reusability	Excellent	Excellent	Poor	Poor	Excellent	Poor

STEP 4: Decide

When you define the criteria and the arguments you need to discuss with the team, you will need to have a decision matrix and give each criterion a defined weight and score for each option. After analyzing the results, you should document this decision in the project artifacts and share it with the related stakeholders.

STEP 5: Optimize

You can always optimize the sdlc during the project execution, you may notice upcoming changes do not fit with the selected sdlc, it is okay to align and cope with the changes. You can even make your own sdlc model which optimum for your organization or the type of projects you are involved in.

5.8 SUMMARY

In Software Engineering, Prototype methodology is a software development model in which a prototype is built, test and then reworked when needed until an acceptable prototype is achieved. 1) Requirements gathering and analysis, 2) Quick design, 3) Build a Prototype, 4) Initial user evaluation, 5) Refining prototype, 6) Implement Product and Maintain; are 6 steps of the prototyping process. Type of prototyping models are 1) Rapid Throwaway prototypes 2) Evolutionary prototype 3) Incremental prototype 4) Extreme prototype Regular meetings are essential to keep the project on time and avoid costly delays in prototyping approach. Missing functionality can be identified, which helps to reduce the risk of failure as Prototyping is also considered as a risk reduction activity in SDLC. Prototyping may encourage excessive change requests.

5.9 REFERENCE FOR FURTHER READING

1. <https://www.geeksforgeeks.org/software-engineering-prototyping-model/>
2. https://www.tutorialspoint.com/sdlc/sdlc_software_prototyping.htm
3. <https://study.com/academy/lesson/what-is-software-prototyping-definition-models-tools.html>
4. <https://www.guru99.com/software-engineering-prototyping-model.html>
5. <https://t4tutorials.com/software-prototypes-software-engineering/>

5.10 MODEL QUESTION

1. What is software prototyping explain its Advantages and Disadvantages.
2. Discuss Incremental Model and its Characteristics?
3. Explain Advantages and Disadvantages of Incremental model
4. Explain in detail about Key principles Dynamic Systems Development Method (DSDM)
5. Discuss about DSDM life cycle
6. Explain Atern Project concepts
7. Discuss on RAD model?
8. what are the phases of agile model?
9. Explain Agile testing methods?
10. Discuss on phases of iterative model along with its pros and cons?
11. Discuss on the Selection Process Parameters for a Software Life Cycle Model?

SOFTWARE EFFORT ESTIMATION

Unit Structure

- 6.0 Software Effort Estimation: Introduction
- 6.1 Where are the Estimates Done?
- 6.2 Problems with Over- and Under-Estimates
- 6.3 Basis for Software Estimating
- 6.4 Software Effort Estimation Techniques
 - 6.4.1 Bottomup Estimating
 - 6.4.2 The Top-down Approach and parametric models
 - 6.4.3 Expert Judgement
 - 6.4.4 Estimating by Analogy
 - 6.4.5 Albrecht Function Point Analysis
 - 6.4.6 Function Points Mark II
 - 6.4.7 COSMIC Full Function Points
- 6.5 COCOMO II: A Parametric Productivity Model
 - 6.5.1 Cost Estimation
 - 6.5.2 Staffing Pattern
- 6.6 Effect of Schedule Compression
- 6.7 Capers Jones Estimating Rules of Thumb
- 6.8 Summary
- 6.9 Reference for further reading
- 6.10 Model questions

6.0 SOFTWARE EFFORT ESTIMATION: INTRODUCTION

Effective software project estimation is one of the most challenging and important activities in software development. Proper project planning and control is not possible without a sound and reliable estimate. The software industry doesn't estimate projects well and doesn't use estimates appropriately. We suffer far more than we should as a result and we need to focus some effort on improving the situation. Under-estimating a project leads to under-staffing it (resulting in staff burnout), under-scoping the quality assurance effort (running the risk of low-quality deliverables) and setting too short a schedule (resulting in loss of credibility as deadlines are missed). For those who figure on avoiding this situation by generously padding the estimate, over-estimating a project can be just about as bad for the organization! If you give a project more resources than it really needs without sufficient scope controls, it will use them. The

project is then likely to cost more than it should (a negative impact on the bottom line), take longer to deliver than necessary (resulting in lost opportunities), and delay the use of your resources on the next project.

6.1 WHERE ARE ESTIMATES DONE?

Estimates are carried out at various stages of a software project. At each stage, the reasons for the estimate and the methods used will vary.

Strategic planning the costs of computerizing potential applications might need to be estimated to help decide what priority to strategic planning given to each project. Such estimates might also influence the numbers of various detail, types of development staff to be recruited by the organization. **Feasibility study** This ascertains that the benefits of the potential system which justifies the costs. **System specification** Most system development methodologies distinguish between the definition of the users' requirements and the design of the documents as how those requirements are to be fulfilled. The effort needed to implement different design proposals will need to be estimated. Estimates at the design stage will also confirm that the feasibility study is still valid, considering all that has been learnt during detailed requirements analysis. The estimate at this stage cannot be based only on the user requirement but technical plan is also needed

Evaluation of suppliers' proposals In the case of the IOE maintenance group accounts subsystem, for example, IOE might consider putting the actual system-building out to tender. Staff in the software houses that are considering a bid would need to scrutinize the system specification and produce estimates on which to base proposals. Amanda might still be required to carry out her own estimate to help judge the bids received. IOE might wish to question a proposal that seems too low: they might wonder, for example, whether the proposer had properly understood the requirements. If, on the other hand, the bids seem too high, they might reconsider in-house development. **Project planning** as the planning and implementation of the project progresses to greater levels of detail, more detailed estimates of smaller work components will be made. As well as confirming the earlier and more broad-brush estimates, these will help answer questions about, for example, when staff will have completed tasks and be available for new activities.

6.2 PROBLEMS WITH OVER- AND UNDER-ESTIMATES

A project leader such as Amanda will need to be aware that the estimate itself, if known to the development team, will influence the time required to implement the system. An over-estimate might cause the project to take longer than it would otherwise. This can be explained by the application of two 'laws'.

Parkinson's Law ' Work expands to fill the time available', which implies that given an easy target staff will work less hard. **Brooks' Law** The effort required to implement a project will go up disproportionately with

the number of staff assigned to the project. As the project team grows so will the effort that has to go into management, co-ordination, and communication. This has given rise, in extreme cases, to the notion of Brooks' Law: 'putting more people on a late job makes it later'. If there is an over-estimate of the effort required, then this might lead to more staff being allocated than are needed and managerial overheads will be increased. This is more likely to be of significance with large projects.

Some have suggested that while the under-estimated project might not be completed on time or to cost, it might still be implemented in a shorter time than a project with a more generous estimate. There must, however, be limits to this phenomenon where all the slack in the project is taken up.

The danger with the under-estimate is the effect on quality. Staff, particularly those with less experience, might respond to pressing deadlines by producing work which is sub-standard. Since we are into laws, this might be seen as a manifestation of Weinberg's zeroth law of reliability: 'if a system does not have to be reliable, it can meet any other objective'. In other words, if there is no need for the program actually to work, you can meet any programming deadline that might be set! Sub-standard work might only become visible at the later, testing, phases of a project, which are particularly difficult to control and where extensive rework can have catastrophic consequences for the project completion date.

Because of the possible effects on the behaviour of development staff caused by the size of estimates, they might be artificially reduced by their managers to increase pressure on staff. This will work only where staff are unaware that this has been done. Research has found that motivation and morale are enhanced where targets are achievable. If, over a period, staff become aware that the targets set are unattainable and that projects are routinely not meeting their published targets, then this will help to destroy motivation. Furthermore, people like to think of themselves as winners and there is a general tendency to put success down to our own efforts, while failure is blamed on the organization

6.3 BASIS FOR SOFTWARE ESTIMATION

Estimation is the process of finding an estimate, or approximation, which is a value that can be used for some purpose even if input data may be incomplete, uncertain, or unstable.

Estimation determines how much money, effort, resources, and time it will take to build a specific system or product. Estimation is based on –

- Past Data/Past Experience
- Available Documents/Knowledge
- Assumptions
- Identified Risks

Observations on Estimation

- Estimation need not be a one-time task in a project. It can take place during –
 - Acquiring a Project.
 - Planning the Project.
 - Execution of the Project as the need arises.
- Project scope must be understood before the estimation process begins. It will be helpful to have historical Project Data.
- Project metrics can provide a historical perspective and valuable input for generation of quantitative estimates.
- Planning requires technical managers and the software team to make an initial commitment as it leads to responsibility and accountability.
- Experience can aid greatly.
- Use at least two estimation techniques to arrive at the estimates and reconcile the resulting values. Refer Decomposition Techniques in the next section to learn about reconciling estimates.
- Plans should be iterative and allow adjustments as time passes and more details are known.

General Project Estimation Approach

The Project Estimation Approach that is widely used is **Decomposition Technique**. Decomposition techniques take a divide and conquer approach. Size, Effort and Cost estimation are performed in a stepwise manner by breaking down a Project into major Functions or related Software Engineering Activities.

Step 1 – Understand the scope of the software to be built.

Step 2 – Generate an estimate of the software size.

Step 3 – Generate an estimate of the effort and cost. You can arrive at the effort and cost estimates by breaking down a project into related software engineering activities.

Step 4 – Reconcile estimates: Compare the resulting values from Step 3 to those obtained from Step 2. If both sets of estimates agree, then your numbers are highly reliable.

Step 5 – Determine the cause of divergence and then reconcile the estimates.

Estimation Accuracy

Accuracy is an indication of how close something is to reality. Whenever you generate an estimate, everyone wants to know how close the numbers

are to reality. You will want every estimate to be as accurate as possible, given the data you have at the time you generate it. And of course, you don't want to present an estimate in a way that inspires a false sense of confidence in the numbers. Important factors that affect the accuracy of estimates are -

- The accuracy of all the estimate's input data.
- The accuracy of any estimate calculation.
- How closely the historical data or industry data used to calibrate the model matches the project you are estimating.
- The predictability of your organization's software development process.
- The stability of both the product requirements and the environment that supports the software engineering effort.
- Whether or not the actual project was carefully planned, monitored, and controlled, and no major surprises occurred that caused unexpected delays.

Estimation Issues

Often, project managers resort to estimating schedules skipping to estimate size. This may be because of the timelines set by the top management or the marketing team. However, whatever the reason, if this is done, then at a later stage it would be difficult to estimate the schedules to accommodate the scope changes.

While estimating, certain assumptions may be made. It is important to note all these assumptions in the estimation sheet, as some still do not document assumptions in estimation sheets. Even good estimates have inherent assumptions, risks, and uncertainty, and yet they are often treated as though they are accurate.

The best way of expressing estimates is as a range of possible outcomes by saying, for example, that the project will take 5 to 7 months instead of stating it will be complete on a particular date or it will be complete in a fixed no. of months. Beware of committing to a range that is too narrow as that is equivalent to committing to a definite date.

Estimation Guidelines

One should keep the following guidelines in mind while estimating a project –

- During estimation, ask other people's experiences. Also, put your own experiences at task.
- Assume resources will be productive for only 80 percent of their time. Hence, during estimation take the resource utilization as less than 80%.

- Resources working on multiple projects take longer to complete tasks because of the time lost switching between them.
- Include management time in any estimate.
- Always build in contingency for problem solving, meetings and other unexpected events.
- Allow enough time to do a proper project estimate. Rushed estimates are inaccurate, high-risk estimates. For large development projects, the estimation step should really be regarded as a mini project.
- Where possible, use documented data from your organization's similar past projects. It will result in the most accurate estimate. If your organization has not kept historical data, now is a good time to start collecting it.
- Use developer-based estimates, as the estimates prepared by people other than those who will do the work will be less accurate.
- Use several different people to estimate and use several different estimation techniques.
- Reconcile the estimates. Observe the convergence or spread among the estimates. Convergence means that you have got a good estimate. Wideband-Delphi technique can be used to gather and discuss estimates using a group of people, the intention being to produce an accurate, unbiased estimate.
- Re-estimate the project several times throughout its life cycle.

6.4 SOFTWARE EFFORT ESTIMATION TECHNIQUES

Barry Boehm, in his classic work on software effort models, identified the main ways of deriving estimates of software development effort as:

algorithmic models which use 'effort drivers' representing characteristics of the target system and the implementation environment to predict effort.

expert judgment where the advice of knowledgeable staff is solicited.

analogy where a similar, completed, project is identified, and its actual effort is used as a basis for the new project.

Parkinson which identifies the staff effort available to do a project and uses that as the 'estimate';

price to win where the 'estimate' is a figure that appears to be sufficiently low to win a contract.

top-down where an overall estimate is formulated for the whole project and is then broken down into the effort required for component tasks.

bottom-up where component tasks are identified and sized and these individual estimates are aggregated.

6.4.1 Bottom-up estimating

Estimating methods can be generally divided into bottom-up and top-down approaches. With the bottom-up approach, the estimator breaks the project into its component tasks and then estimates how much effort will be required to carry out each task. With a large project, the process of breaking down into tasks would be a repetitive one: each task would be analysed into its component sub-tasks and these in turn would be further analysed. This is repeated until you get to components that can be executed by a single person in about a week or two. The reader might wonder why this is not called a top-down approach: after all you are starting from the top and working down! Although this top-down analysis is an essential precursor to bottom-up estimating, it is really a separate one - that of producing a Work Breakdown Structure (WBS). The bottom-up part comes in adding up the calculated effort for each activity to get an overall estimate.

6.4.2 The top-down approach and parametric models

The top-down approach is normally associated with parametric (or algorithmic) models. These may be explained using the analogy of estimating the cost of rebuilding a house. This would be of practical concern to a house-owner who needs sufficient insurance cover to allow for rebuilding the property if it were destroyed. Unless the house-owner happens to be in the building trade it is unlikely that he or she would be able to work out how many bricklayer-hours, how many carpenter-hours, electrician-hours and so on would be required. Insurance companies, however, produce convenient tables where the house-owner can find an estimate of rebuilding costs based on such parameters as the number of storeys and the floor space that a house has. This is a simple parametric model.

The effort needed to implement a project will be related mainly to variables associated with characteristics of the final system. The form of the parametric model will normally be one or more formulae in the form: $\text{effort} = (\text{system size}) \times (\text{productivity rate})$. For example, system size might be in the form 'thousands of lines of code' (KLOC) and the productivity rate 40 days per KLOC. The values to be used will often be matters of subjective judgment.

A model to forecast software development effort therefore has two key components. The first is a method of assessing the size of the software development task to be undertaken. The second assesses the rate of work at which the task can be done. For example, Amanda at IOE might estimate that the first software module to be constructed is 2 KLOC. She might then judge that if she undertook the development of the code, with her expertise she could work at a rate of 40 days per KLOC and complete the work in 2×40 days, that is, 80 days, while Ken, who is less experienced, would need 55 days per KLOC and take 2×55 that is, 110 days to complete the task? Some parametric models, such as that implied by function points, are focused on system or task size, while others, such

are COCOMO are more concerned with productivity factors. Having calculated the overall effort required, the problem is then to allocate proportions of that effort to the various activities within that project.

The top-down and bottom-up approaches are not mutually exclusive. Project managers will probably try to get several different estimates from different people using different methods. Some parts of an overall estimate could be derived using a top-down approach while other parts could be calculated using a bottom-up method.

At the earlier stages of a project, the top-down approach would tend to be used, while at later stages the bottom-up approach might be preferred.

Comparison between bottom-up approach and top-down approach

Bottom-up

- Use when no past project data
- Identify all tasks that must 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

6.4.3 Expert judgment

This is asking someone who is knowledgeable about either the application area or the development environment to give an estimate of the effort needed to carry out a task. This method will most likely be used when estimating the effort needed Expert judgment as an estimating method'.

The estimator would have to carry out impact analysis in order to judge the proportion of code that would be affected and from that derive an estimate. Someone already familiar with the software would be in the best position to do this.

Some have suggested that expert judgment is simply a matter of guessing, but experts tend to use a combination of an informal analogy approach where similar projects from the past are identified and bottom-up estimating.

6.4.4 Estimating by analogy

The use of analogy is also called case-based reasoning. The estimator seeks out projects that have been completed (source cases) and that have similar characteristics to the new project (the target case). The effort that has been recorded for the matching source case can then be used as a base estimate for the target. The estimator should then try to identify any

differences between the target and the source and adjust the base estimate for the new project.

This might be a good approach where you have information about some previous projects but not enough to draw generalized conclusions about what variables might make good size parameters.

A problem here is how you identify the similarities and differences between the different systems. Attempts have been made to automate this process. One software application that has been developed to do this is ANGEL. This identifies the source case that is nearest the target by measuring the Euclidean distance between cases. The source case that is at the shortest Euclidean distance from the target is deemed to be the closest match. The Euclidean distance is calculated:

$$\text{distance} = \text{square-root of } ((\text{target_parameter}_1 - \text{source_parameter}_1)^2 + \dots + (\text{target_parameter}_n - \text{source_parameter}_n)^2)$$

6.4.5 Albrecht function point analysis

This is a top-down method that was devised by Allan Albrecht when he worked for IBM. Albrecht was investigating programming productivity and needed some way to quantify the functional size of programs independently of the programming languages in which they had been coded. He developed the idea of function points (FPs).

The basis of function point analysis is that computer-based information systems comprise five major components, or external user types in Albrecht's terminology, that are of benefit to the users:

- External input types are input transactions that update internal computer files.
- External output types are transactions where data is output to the user. Typically, these would be printed reports, since screen displays would come under external inquiry- types.
- Logical internal file types are the standing files used by the system. The term 'file' does not sit easily with modern information systems. It refers to a group of data that is usually accessed together. It might be made up of one or more record types. For example, a purchase order file might be made up of a record type PuRCHASEORDER plus a second that is repeated for each item ordered on the purchase order - PurchaseOrderItem.
- External interface file types allow for output and input that might pass to and from other computer applications. Examples of this would be the transmission of accounting data from an order processing system to the main ledger system or the production of a file of direct debit details on a magnetic or electronic medium to be passed to the Bankers Automated Clearing System (BACS). Files shared among applications would also be counted here.

- External inquiry types - note the US spelling of inquiry - are transactions initiated by the user that provide information but do not update the internal files. The user inputs some information that directs the system to the details required.

6.4.6 Function points Mark II

The Mark II method has been recommended by the CCTA (Central Computer and Telecommunications Agency), which lays down standards for UK government projects. At one time this Mark II approach seemed to be a good method to use with SSADM, but some difficulties are now apparent. The 'Mark II' label implies an improvement and replacement of the Albrecht method. The Albrecht (now IFPUG) method, however, has had many refinements made to it and FPA Mark II remains a minority method used mainly in the UK.

6.4.7 COSMIC Full Function Points

COSMIC function points are a unit of measure of software functional size. The size is a consistent measurement (or estimate) which is very useful for planning and managing software and related activities. The process of measuring software size is called functional size measurement (FSM). COSMIC functional size measurement is applicable to business, real-time and infrastructure software at any level of decomposition. It is independent of the technology or processes used to develop the system. It is an ISO standard. It is a refined improvement over its predecessors. The unit of size is the COSMIC Function Point or CFP.

Uses

Once you have measured (or estimated) the size in COSMIC Function Points you can then use this as the base metric to :

- Estimate development effort
- Estimate project duration
- Estimate project quality achievement
- Estimate test effort
- Control scope creep
- Assess the value a software asset
- Estimate maintenance and replacement costs
- Assess the achievement of quality (defect removal rates)
- As the basis for fixed price contracts
- and more.

Based on Principles

The COSMIC Function Point sizing method of measuring software requirements is based on two main principles:

1. The 'Software Context Model'

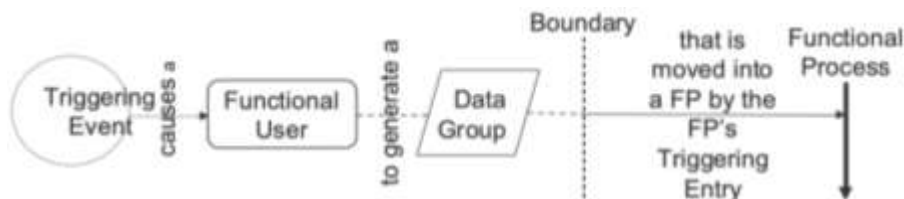
Defines the software to be measured

- Software is bounded by hardware and typically structured into **layers**.
- The **scope** of any piece of software to be measured shall depend on the **purpose** of the measurement and shall be confined wholly within a single layer.
- The **functional users** of a piece of software to be measured shall be identified from its Functional User Requirements (FUR) as the senders and/or intended recipients of data to/from the software respectively.
- A precise COSMIC size measurement of a piece of software requires that its FUR is known at a **level of granularity** at which its **functional processes** and sub-processes may be identified.
- An approximate COSMIC size measurement is possible if its FUR are measured at a high level of granularity by an approximation approach and scaled to the level of granularity of the functional processes.

2. The 'Generic Software Model'

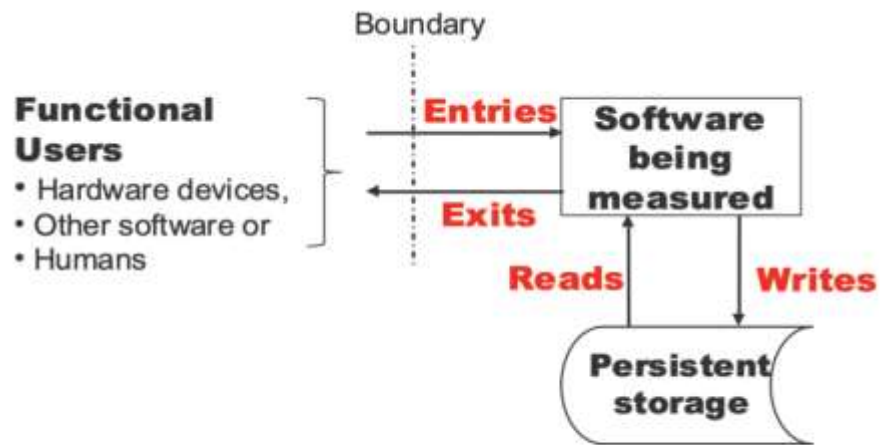
Generic concepts applicable to all software

- A piece of software interacts with its functional users across a **boundary**, and with **persistent storage** within the boundary.
- The FUR of a piece of software can be mapped into unique **functional processes**.
- Each functional process is started by its **triggering Entry** data movement. The data group moved by the triggering Entry is generated by a functional user in response to a **triggering event**.
- A functional process shall include at least one Entry data movement and either a Write or an Exit data movement. There is no upper limit to the number of data movements in a functional process



N.B. There can be many-to-many relationships at each stage along this chain, except that any one functional process may have only one triggering Entry

- Each functional process consists of sub-processes, **data movements (DMs)** and **data manipulations**.
- As an approximation for measurement purposes, the COSMIC method assumes that the functionality of any data manipulation is accounted for by the data movement with which it is associated.
- There are four data movement types, **Entry, Exit, Write and Read**.
- A data movement moves a single **data group**, which consists of a unique set of **data attributes** that describe a single **object of interest**.



The 'Data Movement' is the unit of measure: 1 CFP

6.5 COCOMO II: A PARAMETRIC PRODUCTIVITY MODEL

In modern software development practice, it is crucial to **know the cost and time required for the software development** before establishing new software projects. One of the efficient cost estimation models which are extensively applied to many software projects is called "**Constructive Cost Model (COCOMO)**".

COCOMO is a procedural software cost estimation model proposed by Barry W. Boehm in 1981. This cost estimation model is extensively used in predicting **the effort, development time, average team size and effort required to develop a software project**. The distinctiveness of this strategy is that COCOMO estimates the cost based on the **number of lines of source code** and **sets of subjective assessment (cost drivers)** of product, hardware, personnel, and project attributes. This provides transparency to the model which allows software managers to understand why the model gives the estimates it does. Moreover, the baseline **COCOMO originally underlies a waterfall model lifecycle**. The table below indicates the criteria for selecting the type of software project to be applied for further calculation in the model.

	Organic	Semi-detached	Embedded
Project size (lines of source code)	2,000 to 50,000	50,000 to 300,000	300,000 and above
Team Size	Small	Medium	Large
Developer Experience	Experienced developers needed	Mix of Newbie and experienced developers	Good experience developers
Environment	- Familiar Environment	- Less familiar environment	- Unfamiliar environment (new) - Coupled with complex hardware
Innovation	Minor	Medium	Major
Deadline	Not tight	Medium	Very tight
Example(s)	Simple Inventory Management system	New Operating system	Air traffic control system

Table 1. Comparison between three classes of software project in terms of size, team size, developer experience, environment, innovation, and deadline.

Types of COCOMO model

COCOMO model allows software manager to decide how detailed they would like to conduct the cost estimation for their own project. COCOMO can unveil the efforts and schedule of a software product based on the size of the software in different levels. There are **basic COCOMO**, **Intermediate COCOMO**, and **Detailed/Completed COCOMO**.

Basic COCOMO model the basic COCOMO is used for **rough calculation** which **limits accuracy in calculating software estimation**. This is because the model solely considers based on lines of source code together with constant values obtained from software project types rather than other factors which have major influences on Software development process. **Intermediate COCOMO model** Intermediate COCOMO model is an extension of the Basic COCOMO model which includes a set of cost drivers into account to enhance more accuracy to the cost estimation model as a result. **Complete/detailed COCOMO model** the model incorporates all qualities of both Basic COCOMO and Intermediate COCOMO strategies on each software engineering process. The model accounts for the influence of the individual development phase (analysis, design, etc.) of the project.

Pros

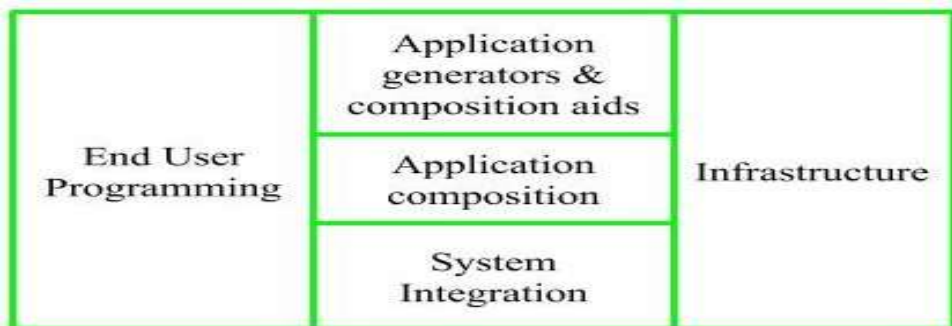
- COCOMO is transparent, one can see how it works unlike other models such as SLIM.
- COCOMO works on historical data or the experience, therefore it is predictable and more accurate.
- Easy to implement factors, as the drivers help to estimate the impact of different factors that affect the projects.
- Easy to estimate the total cost of the projects. This is because COCOMO uses a regression formula from historical projects.

Cons

- Cocomo ignores the requirements of the project and all the related documentation related to the project.
- Cocomo ignores customer skills, cooperation, knowledge, and other parameters.
- When the Cocomo cannot establish a good understanding of the project between the customers and the developers.
- Cocomo is dependent, If there are changes to the actual amount of time spent on these phases, it will affect the accuracy.
- There are certain factors that are beyond the control of the developers or customers such as hardware malfunctions and failures.

COCOMO-II is the revised version of the original Cocomo (Constructive Cost Model) and is developed at University of Southern California. It is the model that allows one to estimate the cost, effort and schedule when planning a new software development activity.

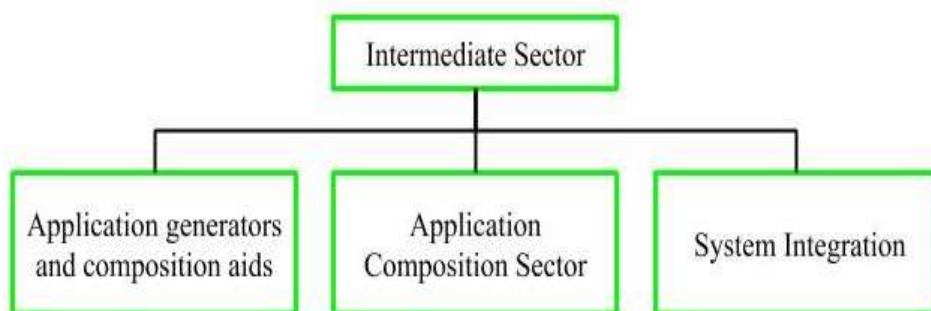
It consists of three sub-models:



1. End User Programming:

Application generators are used in this sub-model. End users write the code by using these application generators. **Example** – Spreadsheets, report generator, etc.

2. Intermediate Sector:



(a) Application Generators and Composition Aids –

This category will create largely prepackaged capabilities for user programming. Their product will have many reusable components. Typical firms operating in this sector are Microsoft, Lotus, Oracle, IBM, Borland, Novell.

(b) Application Composition Sector –

This category is too diversified and to be handled by prepackaged solutions. It includes GUI, Databases, domain specific components such as financial, medical, or industrial process control packages.

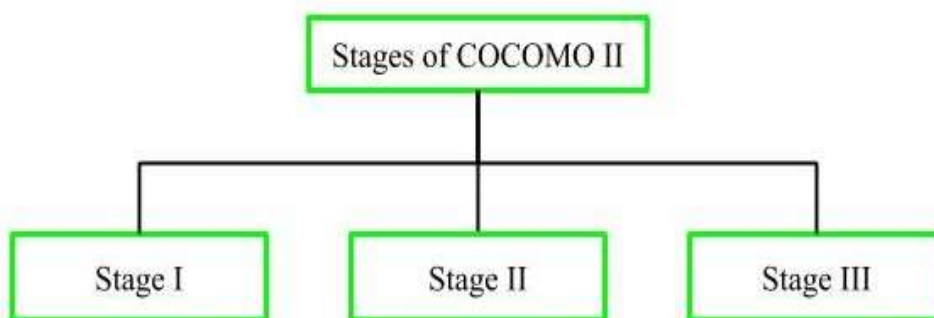
(c) System Integration –

This category deals with large scale and highly embedded systems.

3. Infrastructure Sector:

This category provides infrastructure for the software development like Operating System, Database Management System, User Interface Management System, Networking System, etc.

Stages of COCOMO II:



1. Stage-I:

It supports estimation of prototyping. For this it uses *Application Composition Estimation Model*. This model is used for the prototyping stage of application generator and system integration.

2. **Stage-II:**

It supports estimation in the early design stage of the project when we less know about it. For this it uses *Early Design Estimation Model*. This model is used in early design stage of application generators, infrastructure, system integration.

3. **Stage-III:**

It supports estimation in the post architecture stage of a project. For this it uses *Post Architecture Estimation Model*. This model is used after the completion of the detailed architecture of application generator, infrastructure, system integration.

The Constructive Cost Model (COCOMO) is an algorithmic software cost estimation model developed by Barry Boehm. The model uses a basic regression formula, with parameters that are derived from historical project data and current project characteristics. COCOMO was first published in 1981 Barry W.

References to this model typically call it COCOMO 81. In 1997 COCOMO II was developed and finally published in 2000 in the book *Software Cost Estimation with COCOMO II*. COCOMO II is the successor of COCOMO 81 and is better suited for estimating modern software development projects. It provides more support for modern software development processes and an updated project database.

COCOMO II is tuned to modern software life cycles. The original COCOMO model has been very successful, but it doesn't apply to newer software development practices as well as it does to traditional practices. COCOMO II targets modern software projects and will continue to evolve over the next few years.

COCOMO II has three different models:

The Application Composition Model

Suitable for projects built with modern GUI-builder tools. Based on new Object Points.

The Early Design Model

You can use this model to get rough estimates of a project's cost and duration before you've determined its entire architecture. It uses a small set of new Cost Drivers, and new estimating equations. Based on Unadjusted Function Points or KSLOC.

The Post-Architecture Model

This is the most detailed COCOMO II model. You'll use it after you've developed your project's overall architecture. It has new cost drivers, new line counting rules, and new equations.

6.5.1 Cost estimation

For instance, detailed COCOMO will perform cost estimation in each development phase of Waterfall Model.

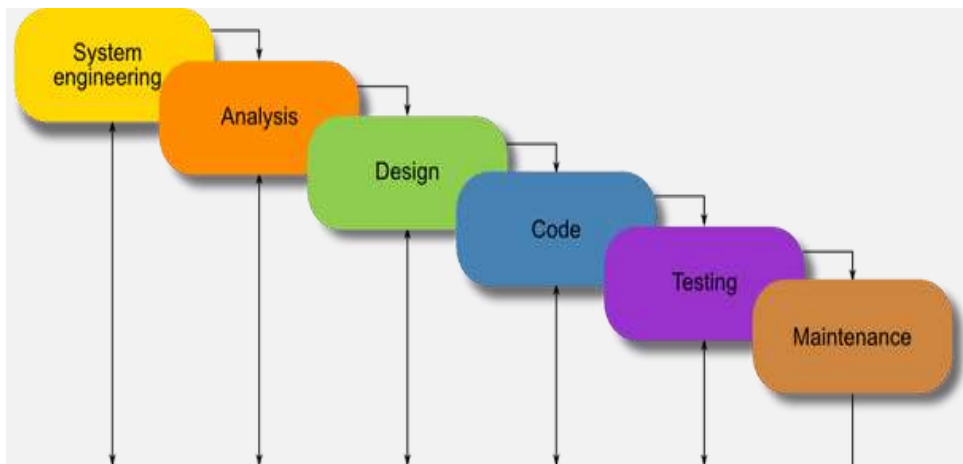


Figure 1. An illustration of classical Waterfall Model.

Calculation

In COCOMO, the general calculation steps of COCOMO-based cost estimation are the following:

1. Get an initial estimate of the development effort from evaluation of **thousands of delivered lines of source code (KDLOC)**.
2. Determine a set of **15 cost factors** from various attributes of the project.
3. Calculate the effort estimate by **multiplying the initial estimate with all the multiplying factors** i.e., multiply the values in previous steps.

Now, let's apply these steps to the real example in **Basic COCOMO** first.

Question statement: Suppose the project was estimated to be 400 KDLOC calculate the effort, development time, and time of each of the 3 modes

Note: the basic COCOMO is used in Organic mode by default.

The arithmetic formula of Basic COCOMO is

Effort applied to the project: $E = a_b(KLOC)^{b_b}$ (in Person-month)

Development time: $D = c_b(E)^{d_b}$ (in month)

Manpower required: $P = \frac{E}{D}$ (in Person)

Where a_b, b_b, c_b, d_b are constants for each category of software product

Figure 2. Formula for Basic COCOMO

Each of the constant a, b, c, d can be defined as show in the Table 2.

SW project	a_b	b_b	c_b	d_b
Organic	2.4	1.05	2.5	0.38
Semi detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Table 2. Constant values corresponding to software project type as stated in previous section.

The calculations of Basic COCOMO corresponding to each of software project types are shown in the Table 3.

Basic mode →	$E = a_b(KLOC)^{b_b} = 2.4(400)^{1.05} = 1295.31 \text{ PM}$ $D = c_b(E)^{d_b} = 2.5(1295.31)^{0.38} = 38.07 \text{ months}$ $P = \frac{E}{D} = \frac{1295.31}{38.07} \approx 1295 \text{ people}$
Semi Detached →	$E = a_b(KLOC)^{b_b} = 3.0(400)^{1.12} = 2462.79 \text{ PM}$ $D = c_b(E)^{d_b} = 2.5(2462.79)^{0.35} = 38.45 \text{ months}$ $P = \frac{E}{D} = \frac{2462.79}{38.45} \approx 64 \text{ people}$
Embedded →	$E = 4772.81 \text{ PM}$ $D = 38 \text{ PM}$

Table 3. The calculation of Basic COCOMO on each software project type.

Rather than the sole consideration on the number of lines of the source code as shown in Basic COCOMO, **Intermediate COCOMO** introduces sets of 15 subjective assessment called “Cost Drivers” to ensure that other aspects of Software Development are considered in the cost estimation. Cost drivers are divided into 4 groups including, product attributes, hardware attributes, personal attributes, and project attributes.

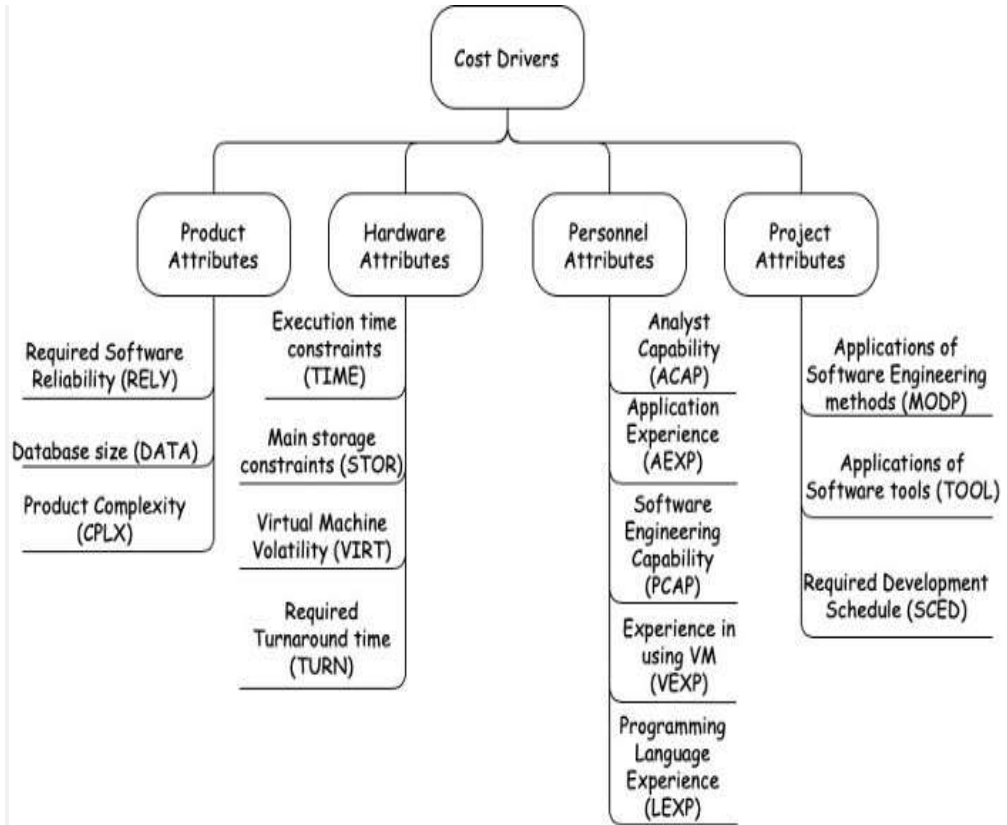


Figure 3. Cost drivers

Each of cost driver is rated on the scale of are very low to extremely high to calculate the specific effort multiplier and each of them returns an adjustment factor which multiplied yields in the **total EAF (Effort Adjustment Factor)**.

The scale includes very low, low, normal, high very high, extra high, accordingly. The adjustment factor is 1 for a cost driver that is judged as normal. In practice, typical values for EAF range from 0.9 to 1.4.

Cost Drivers	RATINGS					
	Very low	Low	Nominal	High	Very High	Extra High
Product Attributes						
RELY	0.75	0.88	1.00	1.15	1.40	..
DATA	..	0.94	1.00	1.08	1.16	..
CPLX	0.70	0.85	1.00	1.15	1.30	1.65
Computer Attributes						
TIME	1.00	1.11	1.30	1.66
STOR	1.00	1.06	1.21	1.56
VIRT	..	0.87	1.00	1.15	1.30	..
TURN	..	0.87	1.00	1.07	1.15	..

Figure 5. Ratings for cost drivers under product attributes and computer attributes

Cost Drivers	RATINGS					
	Very low	Low	Nominal	High	Very high	Extra high
Personnel Attributes						
ACAP	1.46	1.19	1.00	0.86	0.71	..
AEXP	1.29	1.13	1.00	0.91	0.82	..
PCAP	1.42	1.17	1.00	0.86	0.70	..
VEXP	1.21	1.10	1.00	0.90
LEXP	1.14	1.07	1.00	0.95
Project Attributes						
MODP	1.24	1.10	1.00	0.91	0.82	..
TOOL	1.24	1.10	1.00	0.91	0.83	..
SCED	1.23	1.08	1.00	1.04	1.10	..

Figure 6. Ratings for cost drivers under personnel attributes and project attributes

By taking all cost drivers into account represented by EAF, the arithmetic cost estimation formula for Intermediate COCOMO can be derived as follow:

Total effort required for the project: $E = a_i(KLOC)^{bi} * (EAF)$
 Total time required for project development: $D = C_i(E_i)^d$
 EAF can be calculated by multiplying all values that have been obtained after categorizing each cost drive.
 KLOC is the size of the code for the project in Kilo lines of code.
 a, b, c, d are the constant parameters for the software project

Figure 7. Formula for Intermediate COCOMO with an inclusion of Effort Adjustment Factor (EAF)

6.5.2 Staffing Pattern

Putnam was the first to study the problem of what should be a proper staffing pattern for software projects. He extended the classical work of Norden who had earlier investigated the staffing pattern of general research and development type of projects. To appreciate the staffing pattern desirable for software projects, we must understand both Norden’s and Putnam’s results.

Norden’s Work

Norden studied the staffing patterns of several R&D projects. He found the staffing patterns of R&D projects to be very different from the manufacturing or sales type of work. Staffing pattern of R&D types of projects changes dynamically over time for efficient manpower utilization. He concluded that staffing pattern for any R&D project can be approximated by the Rayleigh distribution curve.

Putnam's Work

Putnam studied the problem of staffing of software projects.

He found that staffing pattern for software development projects has characteristics very similar to R&D projects. He adapted the Rayleigh-Norden curve to relate the no of delivered lines of code to the effort and the time required to develop the product. Initially less no of developers are needed. As the project progresses and more detailed work is performed, the number of developers increases and reaches a peak during product delivery. After delivery, the no. of project staff falls consistently during product maintenance.

6.6 EFFECT OF SCHEDULE COMPRESSION

There are 2 Techniques for Schedule Compression: Fast Tracking and Crashing. No matter how well you plan a project, actual results will differ from what you have planned. **In terms of schedule, the actual durations of tasks can take longer than planned.** To meet the project deadline, you need to take corrective actions to get back on track and these are called schedule compression methods. Schedule compression helps you to **get your project back on track.** There are two methods for schedule compression.

Schedule Compression Definition

Schedule compression techniques are applied during develop schedule process if a project is behind the schedule. The objective of schedule compression is to **try to compress the** schedule without changing project scope. Because, if a project scope has not changed, and if the project is behind schedule, you can meet the planned deadline only by compressing the remaining schedule of the project. There are 2 approaches for schedule compression

The 1st schedule compression technique: What is fast tracking in project management?


First approach is Fast Tracking. Let's give fast tracking project management definition. In fast tracking schedule compression technique, critical path activities are performed in parallel instead of series. This is possible only the activities are not in mandatory dependency. Because, if two activities are depending on each other by nature, you cannot do these two activities in parallel. For instance, you cannot start testing of a screen before completing development. If critical path activities are depending on each other because of resource dependency or if there is a discretionary dependency, you can fast track those activities to complete remaining activities faster.

Time Management Master of Project Academy

Develop Schedule
Schedule Compression:

- Objective of schedule compression is to try to compress the schedule without changing project scope

Fast Tracking: doing critical path activities in parallel instead of series. Increases more risk, rework and requires more attention.



Let's visualize fast tracking over a sample. As a fast-tracking example, let's consider that there are 3 activities that need to be completed each other: Activity #1, Activity #2 and Activity #3. If you are aiming to finish the project faster, you can perform Activity #2 and Activity #3 in parallel after completing activity #1. This is called fast tracking schedule compression approach.

The 2nd schedule compression technique: What is crashing in project management?

Second technique is crashing. **In crashing schedule compression technique, there is a trade-off between cost and schedule.** If the scope is the same and project is behind schedule, another option for compressing the schedule is putting extra resources on remaining activities of the project. Because if it is possible to assign more than one resource on an activity, activity duration will decrease respectively. This will help to complete the project faster. However, since these extra resources were not in the initial plan, there will be an additional cost if crashing is used for schedule compression.


Time Management Master of Project Academy

Develop Schedule
Schedule Compression:

Crashing: Cost vs schedule tradeoffs to determine how to compress schedule.
E.g. putting extra resource for one activity (shortens time, increases cost)

If you are behind schedule, follow the steps below:

1. Check risks and re-estimate
2. Fast-track the project
3. Crash the project
4. Reduce scope
5. Cut quality



5 major steps to follow if the project is behind the schedule

If a project is behind the schedule, there are 5 major steps that must be followed in sequence.

- **First, check risks and re-estimate.** Because, for the remaining activities, if the risks that were considered during planning are no

longer valid, re-estimation of the remaining activities can result in shorter activity durations. Re-estimation will show how long will it take to complete remaining activities of the project.

- **If re-estimation results in a later deadline for project completion, Fast-tracking the project must be considered.** Remaining critical path activities are evaluated and possible activities that can be performed in parallel can be done to shorten the duration of the project. This is one of the advantages of fast tracking in project management over crashing because fast-tracking does not bring an extra cost to project.
- **The third step is crashing the project.** Extra resources are planned, and extra budget is allocated to accommodate the increasing costs. Since more resources will work on remaining activities of the project, it is expected to finish the project on time.
- **The fourth step is scope reduction.** Reducing the scope can help to reduce the remaining activities in the project and if the customer agrees, reducing the scope can help to complete the project on time.
- **The fifth step is cutting the quality.** Achieving a certain level of quality means cost and time. If the customer agrees to decrease its quality expectations, cutting quality can help you to complete the project faster. Note that, 4th and 5th steps are not recommended the course of actions in a project

CAPERS JONES ESTIMATING RULES OF THUMB

Accurate software estimating is too difficult for simple rules of thumb. Yet despite the availability of more than 50 commercial software estimating tools, simple rules of thumb remain the most common approach. Rules based on the function point metric are now replacing the older LOC rules.

Introduction

For many years manual estimating methods were based on the “lines of code” (LOC) metric and several workable rules of thumb were developed for common procedural programming languages such as Assembly, COBOL, FORT RAN, PASCAL, PL/I and the like.

Tables 1 and 2 illustrate samples of the LOC based rules of thumb for procedural languages in two forms: Table 1 uses “months” as the unit for work, while table 2 uses “hours” as the unit for work. Both hourly and monthly work metrics are common in the software literature, with the hourly form being common for small programs and the monthly form being common for large systems.

Table 1: Rules of Thumb Based on LOC Metrics for Procedural Languages

(Assumes 1 work month = 132 work hours)

Size of Program in LOC	Coding LOC per Month	Coding Effort (Months)	Testing Effort Percent	Noncode Effort Percent	Total Effort (Months)	Net LOC Per Month
1	2500	0.0004	10.00%	10.00%	0.0005	2083
10	2250	0.0044	20.00%	20.00%	0.0062	1607
100	2000	0.0500	40.00%	40.00%	0.0900	1111
1,000	1750	0.5714	50.00%	60.00%	1.2000	833
10,000	1500	6.6667	75.00%	80.00%	17.0000	588
100,000	1200	83.3333	100.00%	100.00%	250.0000	400
1,000,000	1000	1000.0000	125.00%	150.00%	3750.0000	267

As can be seen, the “monthly” form of this table is not very convenient for the smaller end of the spectrum, but the “hourly” form is inconvenient at the large end.

Table 2: Rules of Thumb Based on LOC Metrics for Procedural Languages

(Assumes 1 work month = 132 work hours)

Size of Program in LOC	Coding LOC per Month	Coding Effort (Months)	Testing Effort Percent	Noncode Effort Percent	Total Effort (Months)	Net LOC Per Month
1	18.94	0.05	10.00%	10.00%	0.06	15.78
10	17.05	0.59	20.00%	20.00%	0.82	12.18
100	15.15	6.60	40.00%	40.00%	11.88	8.42
1,000	13.26	75.43	50.00%	80.00%	173.49	5.76
10,000	11.36	880.00	75.00%	100.00%	2,420.00	4.13
100,000	9.09	11,000.00	100.00%	150.00%	38,500.00	2.60
1,000,000	7.58	132,000.00	125.00%	150.00%	495,000.00	2.02

Also, the assumption that a work month comprises 132 hours is a tricky one, since the observed number of hours worked in a month can run from less than 120 to more than 170. Because the actual number of hours varies from project to project, it is best to replace the generic rate of “132” with an actual or specific rate derived from local conditions and work patterns.

The development of Visual Basic and its many competitors such as Realizer have changed the way many modern programs are developed. Although these visual languages do have a procedural source code portion, quite a bit of the more complex kinds of “programming” are done using button controls, pull-down menus, visual worksheets, and reusable components. In other words, programming is being done without anything that can be identified as a “line of code” for measurement or estimation purposes. By the end of the century perhaps 30% of the new software applications will be developed using either object-oriented languages or visual languages (or both).

For large systems, programming itself is only the fourth most expensive activity. The three higher-cost activities can not really be measured or estimated effectively using the lines of code metric. Also, the fifth major cost element, project management, cannot easily be estimated or measured using the LOC metric. The usefulness of a metric such as lines of code which can only measure and estimate one out of the five major software cost elements is a significant barrier to economic understanding.

The Development of Function Point Metrics

By the middle 1970's IBM's software community was topping 25,000 and the costs of building and maintaining software were becoming a significant portion of the costs and budgets for new products.

Programming languages were exploding in numbers, and within IBM applications were being developed in assembly language, APL, COBOL, FORTRAN, RPG, PL/I, PL/S (a derivative of PL/I) and perhaps a dozen others. Indeed, many software projects in IBM and elsewhere used several languages concurrently, such as COBOL, RPG, and SQL as part of the same system.

Allan J. Albrecht and his colleagues at IBM White Plains were tasked with attempting to develop an improved methodology for sizing, estimating, and measuring software projects. The method they developed is now known as “function point analysis” and the basic metric they developed is termed a “function point.”

In October of 1979 Allan Albrecht presented the function point metric at a conference in Monterey, California sponsored jointly by IBM and two IBM user groups, SHARE and GUIDE. Concurrently, IBM placed the basic function point metric into the public domain. Now that the function point metric has been in use for almost 20 years on many thousands of software projects, a new family of simple rules of thumb has been derived.

These new rules are based on function points, and encompass software sizing algorithms, schedule algorithms, quality algorithms, and other interesting topics. This article contains a set of ten simple rules of thumb that cover various aspects of software development and

maintenance. The rules assume the version 4.1 function point counting rules published by the International Function Point Users Group (IFPUG).

The International Function Point Users Group (IFPUG) is a non-profit organization which has become the largest software metrics association in history. Between IFPUG in the United States and other function point affiliates in about 20 countries, more than 2000 corporations and 15,000 individuals now comprise the function point community. Membership in function point user groups has been growing at more than 50% per year, while usage of lines of code has been declining for more than 10 years.

Users of other kinds of function points such as Mark II, COSMIC, web-object points, story points, engineering function points, etc. should seek out similar rules from the appropriate sponsoring organization. However, most of the function point variants have the interesting property of creating function point totals about 15% larger than IFPUG function points.

The following set of rules of thumb are known to have a high margin of error. They are being published in response to many requests for simple methods that can be used manually or with pocket calculators or spreadsheets. The best that can be said is that the rules of thumb are easy to use and can provide a “sanity check” for estimates produced by other and hopefully more rigorous methods.

SIZING RULES OF THUMB

The function point metric has transformed sizing from a very difficult task into one that is now both easy to perform and comparatively accurate.

Sizing Source Code Volumes

Now that thousands of software projects have been measured using both function points and lines of code (LOC), empirical ratios have been developed for converting LOC data into function points, and vice versa. The following rules of thumb are based on “logical statements” rather than “physical lines.” For similar information on almost 500 programming languages refer to my book Applied Software Measurement

Rule 1: Sizing Source Code Volumes

One function point = 320 statements for basic assembly language

One function point = 125 statements for the C programming language

One function point = 107 statements for the COBOL language

One function point = 71 statements for the ADA83 language

One function point = 15 statements for the SMALLTALK language

The overall range of non-commentary logical source code statements to function points ranges from more than 300 statements per function point for basic assembly language to less than 15 statements per

function point for object-oriented languages with full class libraries and many program generators. However, since many procedural languages such as C, Cobol, Fortran, and Pascal are close to the 100 to 1 mark, that value can serve as a rough conversion factor for the general family of procedural source code languages.

Sizing Paper Deliverables

Software is a very paper intensive industry. More than 50 kinds of planning, requirements, specification, and user-related document types can be created for large software projects. For many large systems and especially for large military projects, the costs of producing paper documents costs far more than source code. The following rule of thumb encompasses the sum of the pages that will be created in requirements, specifications, plans, user manuals, and other business-related software documents.

Rule 2: Sizing Software Plans, Specifications, and Manuals Function points raised to the 1.15 power predicts a pproximate page counts for paper documents associated with software projects.

Paperwork is such a major element of software costs and schedules that it cannot safely be ignored. Indeed, one of the major problems with the “lines of code” (LOC) metric was that it tended to conceal both the volumes of paper deliverables and the high costs of software paperwork.

Sizing Creeping User Requirements

The function point metric is extremely useful in me asuring the rate at which requirements creep.

Rule 3: Sizing Creeping User Requirements Creeping user requirements will grow at an average rate of 2% per month from the design through coding phases.

Assume that you and your clients agree during the requirements to develop an application of exactly 100 function points. This rule of thumb implies that every month thereafter, the original requirements will grow by a rate of 2 function points. Since the design and coding phases of a 100-function point project are usually about 6 months, this rule would imply that about 12% new features would be added and the final total for the application would be 112 function points rather than the initial value of 100 function points.

Sizing Test Case Volumes

The function point metric is extremely useful for test case sizing, since the structure of function point analysis closely parallels the items that need to be validated by testing. Commercial software estimating tools can predict the number of test cases for more than a dozen discrete forms of testing. This simple rule of thumb encompasses the sum of all test cases:

Rule 4: Sizing Test Case Volumes Function points raised to the 1.2 power predicts the approximate number of test cases created.

A simple corollary rule can predict the number of times each test case will be run or executed during development: assume that each test case would be executed approximately four times during software development.

Sizing Software Defect Potentials

The “defect potential” of an application is the sum of bugs or errors that will occur in five major deliverables: 1) requirements errors; 2) design errors; 3) coding errors; 4) user documentation errors; 5) bad fixes, or secondary errors introduced in the act of fixing a prior error.

One of the many problems with “lines of code” metrics is the fact that more than half of software defects are found in requirements and design, and hence the LOC metric is not capable of either predicting or measuring their volumes with acceptable accuracy. Because the costs and effort for finding and fixing bugs is usually the largest identifiable software cost element, ignoring defects can throw off estimates, schedules, and costs by massive amounts.

Rule 5: Sizing Software Defect Potentials Function points raised to the 1.25 power predicts the approximate defect potential for new software projects.

A similar corollary rule can predict the defect potentials for enhancements. In this case, the rule applies to the size of the enhancement rather than the base that is being updated: Function points raised to the 1.27 power predicts the approximate defect potential for enhancement software projects. The higher power used in the enhancement rule is because of the latent defects lurking in the base product that will be encountered during the enhancement process. Incidentally, if you are doing complex client-server applications the 1.27 power actually matches the defect potentials somewhat better than the 1.25 power. Client-server applications are often very buggy, and the higher power indicates that fact.

Sizing Software Defect Removal Efficiency

The defect potential is the life-cycle total of errors that must be eliminated. The defect potential will be reduced by somewhere between 85% (approximate industry norms) and 99% (best in class results) prior to actual delivery of the software to clients. Thus the number of delivered defects is only a small fraction of the overall defect potential.

Rule 6 :Sizing Defect Removal Efficiency

Each software review, inspection, or test step will find and remove 30% of the bugs that are present. The implication of this rule means that a series of between six and 12 consecutive defect removal operations

must be utilized to achieve very high-quality levels. This is why major software producers normally use a multi-stage series of design reviews, code inspections, and various levels of testing from unit test through system test.

RULES OF THUMB FOR SCHEDULES, RESOURCES, AND COSTS

After the sizes of various deliverable items and potential defects have been quantified, the next stage in an estimate is to predict schedules, resources, costs, and other useful results.

Estimating Software Schedules

Rule 7 calculates the approximate interval from the start of requirements until the first delivery to a client:

Rule 7: Estimating Software Schedules

Function points raised to the 0.4 power predicts the approximate development schedule in calendar months.

Among our clients, the range of observed schedules in calendar months varies from a low of about 0.32 to a high or more than 0.45. Table 4 illustrates the kinds of projects whose schedules are typically found at various power levels, assuming a project of 1000 function points in size:

Table 4: Software Schedules in Calendar Months (Assumes 1000 function points from requirements to delivery)

Power	Schedule in Calendar Months	Projects Within Range
0.32	9.12	
0.33	9.77	Agile
0.34	10.47	Extreme
0.35	11.22	Web
0.36	12.02	OO software
0.37	12.88	Client-server software
0.38	13.80	Outsourced software
0.39	14.79	MIS software
0.40	15.85	Commercial software
0.41	16.98	Systems software
0.42	18.20	
0.43	19.50	Military software
0.44	20.89	
0.45	22.39	

The use of function points for schedule estimation is one of the more useful byproducts of function points that has been developed in recent years.

Estimating Software Staffing Levels

Rule 8 is based on the concept of “assignment scope” or the amount of work for which one person will normally be responsible. Rule 8 includes software developers, quality assurance, testers, technical writers, data base administrators, and project managers.

Rule 8: Estimating Software Development Staffing Levels Function points divided by 150 predicts the approximate number of personnel required for the application.

The rule of one technical staff member per 150 function points obviously varies widely based on the skill and experience of the team and the size and complexity of the application. A corollary rule can estimate the number of personnel required to maintain the project during the maintenance period:

Rule 9: Estimating Software Maintenance Staffing Levels Function points divided by 750 predicts the approximate number of maintenance personnel required to keep the application updated.

The implication of rule 9 is that one person can perform minor updates and keep about 750 function points of software operational. (Another interesting maintenance rule of thumb is: Raising the function point total to the 0.25 power will yield the approximate number of years that the application will stay in use.)

Among our clients, the “best in class” organizations are achieving ratios of up to 3,500 function points per staff member during maintenance. These larger values usually indicate a well-formed geriatric program including the use of complexity analysis tools, code restructuring tools, reengineering and reverse engineering tools, and full configuration control and defect tracking of aging legacy applications.

Estimating Software Development Effort

The last rule of thumb in this article is a hybrid rule that is based on the combination of rule 7 and rule 8:

Rule 10: Estimating Software Development Effort Multiply software development schedules by number of personnel to predict the approximate number of staff months of effort.

Since this is a hybrid rule, an example can clarify how it operates. Assume you are concerned with a project of 1000 function points in size:

- Using rule 7, or raising 1000 function points to the 0.4 power, indicates a schedule of about 16 calendar months.
- Using rule 8, or dividing 1000 function points by 150 indicates a staff of about 6.6 full time personnel.
- Multiplying 16 calendar months by 6.6 personnel indicates a total of about 106 staff months to build this project.

6.8 SUMMARY

Effort estimation is a key factor for software project success, defined as delivering software of agreed quality and functionality within schedule and budget. Traditionally, effort estimation has been used for planning and tracking project resources. Effort estimates are over-optimistic and there is a strong over-confidence in their accuracy. The mean effort overrun seems to be about 30% and not decreasing over time. However, the measurement of estimation error is problematic, assessing the accuracy of estimates. The strong overconfidence in the accuracy of the effort estimates is illustrated by the finding that, on average, if a software professional is 90% confident or “almost sure” to include the actual effort in a minimum-maximum interval, the observed frequency of including the actual effort is only 60-70%.

Currently the term “effort estimate” is used to denote as different concepts such as most likely use of effort (modal value), the effort that corresponds to a probability of 50% of not exceeding (median), the planned effort, the budgeted effort or the effort used to propose a bid or price to the client. This is believed to be unfortunate, because communication problems may occur and because the concepts serve different goals.

6.9 REFERENCE FOR FURTHER READING

<https://www.Javatpoint.com>

<https://www.Geeksforgeeks.org>

<https://tutorialspoint.com>

<https://datafloq.com/read/7-innovative-uses-of-clustering-algorithms/6224>

https://en.wikipedia.org/wiki/Cluster_analysis

6.10 MODEL QUESTIONS

1. Where are the estimates done?
2. Brief on Software effort estimation techniques?
3. Explain COCOMO II Model?
4. Explain in detail about capers jones Estimating rules of thumb?

ACTIVITY PLANNING

Unit Structure

- 7.0 Objectives
- 7.1 Introduction
- 7.2 Objective of Activity Planning
- 7.3 When to Plan?
- 7.4 Project Schedules
- 7.5 Projects and Activities
 - 7.5.1 Defining Activities
 - 7.5.2 Identifying Activities
- 7.6 Sequencing and Scheduling Activities
- 7.7 Network Planning Models
- 7.8 Formulating to a Network Model
- 7.9 Adding the Time Dimension
- 7.10 The Forward Pass
- 7.11 The Backward Pass
- 7.12 Identifying the Critical Path
- 7.13 Activity Float
- 7.14 Shortening the Project Duration
- 7.15 Identifying Critical Activities
- 7.16 Activity-on-Arrow Networks
- 7.17 Summary
- 7.18 Exercises
- 7.19 References

7.0 OBJECTIVES

After going through this unit, you will be able to:

- Produce an activity plan for Project
- Estimate the overall duration of a Project
- Create a critical path and a precedence network for a Project

7.1 INTRODUCTION

A detailed plan for the project includes a schedule indicating the start and completion time for each activity which will make us to understand the following:

- Ensure that appropriate resources will be available precisely when required.
- Avoid different activities competing for the same resources at the same time
- Produce a detailed schedule showing which staffs carry out each activity
- Produce a detailed plan against which actual achievement may be measured
- Replan the project during its life to correct drift from the target

To be effective, a plan must be stated as a set of targets, the achievement or non-achievement of which can be unambiguously measured. The activity plan does this by providing a target start and completion date for each activity (or a window within which each activity may be carried out). The starts and completions of activities must be clearly visible and this is one of the reasons why it is advisable to ensure that each and every project activity produces some tangible product or 'deliverable'. Monitoring the project's progress is then, at least in part, a case of ensuring that the products of each activity are delivered on time.

As a project progresses it is unlikely that everything will go according to plan. Much of the job of project management concerns recognizing when something has gone wrong identifying its causes and revisiting the plan to mitigate its effects. The activity plan should provide a means of evaluating the consequences of the meeting any of the activity target dates and guidance as to how the plan might most effectively be modified to bring the project back to target. We shall see that the activity plan may well also offer guidance as to which components of a project should be most closely monitored.

7.2 OBJECTIVE OF ACTIVITY PLANNING

In addition to providing project and resource schedules, activity planning aims to achieve a number of other objectives which may be summarized as follows.

- **Feasibility assessment:** - Is the project possible within required timescale and resource constraints? However, it is not until we have constructed a detailed plan that we can forecast a completion date with any reasonable knowledge of its achievability
- **Resource allocation:** - What are the most effective ways of allocating resources to the project. When should the resources be available? The project plan allows us to investigate relation between timescales and resource availability (in general, allocating additional resources to the project shortens its duration) and the efficacy of additional spending on resource procurement
- **Detailed costing:** - How much will the project cost and when is that expenditure likely to take place? After producing an activity plan

and allocating specific resources we can obtain more detailed estimates of costs and their timing

- **Motivation:** - Providing targets and being seen to monitor achievements against the targets is an effective way of motivating staff. Particularly where they have been involved in setting those targets in the first place
- **Coordination:** - When do the staff in different department needs to be available to work on a particular project and when do staff need to be transformed between projects? The project plan, particularly with large project involving more than a single project team. provides an effective vehicle for communication and coordination among.

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 a minimum cost. These are not, in themselves concerned with meeting quality targets. which, generally impose constraints on the scheduling process. One effective way of shortening project durations is to carry out activities in parallel .Clearly we cannot undertake all the activities at the same time — some require the completion of others before they can start and there are likely to be resource constraints limiting how much may be done simultaneously .Activity scheduling will, however, give us an indication of the cost of these constraints in terms of lengthening timescales and provide us with an indication of how timescales may be shortened by relaxing those constraint .

7.3 WHEN TO PLAN?

Planning is an ongoing process of refinement, each iteration becoming more detailed and more accurate than the last. Over successive iterations, the emphasis and purpose of planning will shift. During the feasibility study and project start-up, the main purpose of planning will be to estimate timescales and the risks of not achieving target completion dates or keeping within budget. As the project proceeds beyond the feasibility study, the emphasis will be placed upon the production of activity plans for ensuring resource availability and cash flow control. Throughout the project, until the final deliverable has reached the customer, monitoring and replanning must continue to correct any drift that might prevent meeting time or cost targets.

7.4 PROJECT SCHEDULES

Before work commences on a project or, possibly, a stage of a larger project, the project plan must be developed to the level of showing dates when each activity should start and finish and when and how much of each resource will be required. Once the plan has been refined to this level of detail, we call it a project schedule. Creating a project schedule comprises four main stages.

The first step in producing the plan is to decide what activities need to be carried out and in what order they are to be done. From this we can construct an ideal activity plan that is, a plan of when each activity would ideally be undertaken were resources not a constraint.

The ideal activity plan will then be the subject of an activity risk analysis, aimed at identifying potential problems. This might suggest alterations to the ideal activity plan and will almost certainly have implications for resource allocation.

The third step is resource allocation. The expected availability of resources might place constraints on when certain activities can be carried out, and our ideal plan might need to be adapted to take account of this.

The final step is schedule production. Once resources have been allocated to each activity, we will be in a position to draw up and publish a project schedule, which indicates planned start and completion dates and a resource requirements statement for each activity.

7.5 PROJECTS AND ACTIVITIES

7.5.1 Defining Activities

Before we try to identify the activities that make up a project it is worth reviewing what we mean by a project and its activities and adding some assumptions that will be relevant when we start to produce an activity plan.

- A project is composed of a number of interrelated activities.
- A project may start when at least one of its activities is ready to start.
- A project will be completed when all of the activities it encompasses have been completed.
- An activity must have a clearly defined start and a clearly defined endpoint normally marked by the production of a tangible deliverable.
- If an activity requires a resource (as most do) then that resource requirement must be forecastable and is assumed to be required at a constant level throughout the duration of the activity.
- The duration of an activity must be forecastable – assuming normal circumstances and the reasonable availability of resources. • Some activities might require that others are completed before they begin (these are known as precedence requirements).

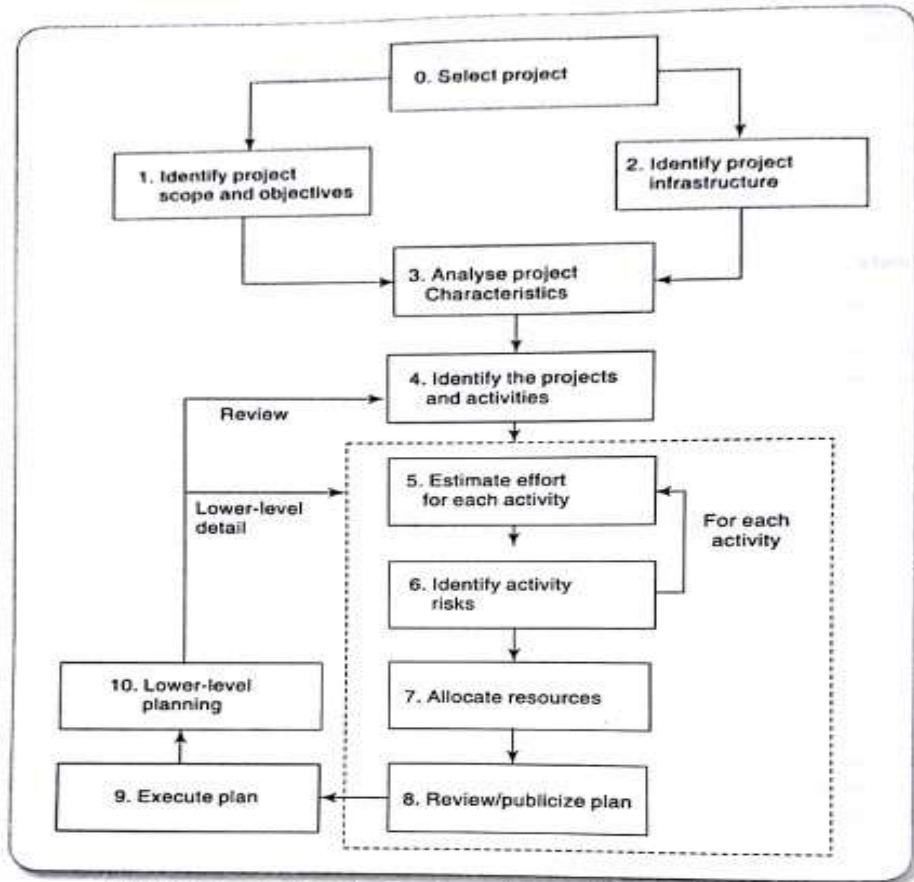


FIGURE 6.1 Activity planning is carried out in Steps 4 and 5

7.5.2 Identifying Activities

Essentially there are three approaches to identifying the activities or tasks that make up a project — we shall call them the activity-based approach, the product-based approach and the hybrid approach.

The activity-based approach

The activity-based approach consists of creating a list of all the activities that the project is thought to involve. This might require a brainstorming session involving the whole project team or it might stem from an analysis of similar past projects. When listing activities, particularly for a large project, it might be helpful to subdivide the project into the main life-cycle stages and consider each of these separately.

Rather than doing this in an ad hoc manner, with the obvious risks of omitting or double-counting tasks, a much-favoured way of generating a task list is to create a Work Breakdown Structure (WBS). This involves identifying the main (or high-level) tasks required to complete a project and then breaking each of these down into a set of lower-level tasks. Figure 6.2 shows a fragment of a WBS where the design task has been broken down into three tasks and one of these has been further decomposed into two tasks.

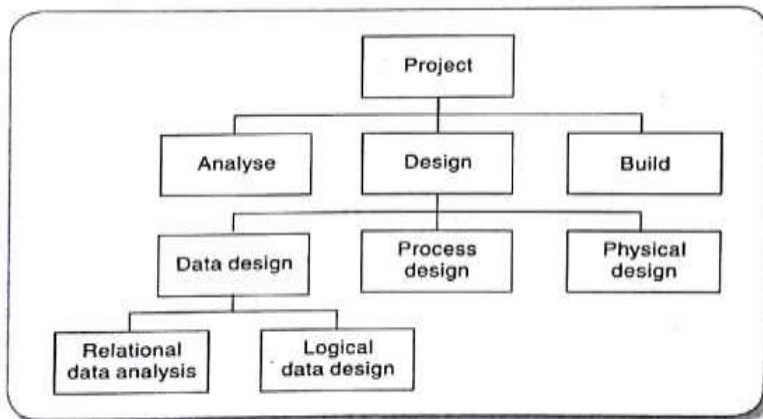


FIGURE 6.2 A fragment of an activity-based Work Breakdown Structure

Activities are added to a branch in the structure if they contribute directly to the task immediately above — if they do not contribute to the parent task, then they should not be added to that branch. The tasks at each level in any branch should include everything that is required to complete the task at the higher level.

When preparing a WBS, consideration must be given to the final level of detail or depth of the structure. Too great a depth will result in a large number of small tasks that will be difficult to manage, shallow structure will provide insufficient detail for project control. Each branch should, however, be broken down at least to a level where each leaf may be assigned to an individual or responsible section within the organization.

Advantages claimed for the WBS approach include the belief that it is much more likely to result in a task catalogue that is complete and is composed of non-overlapping include task definitions activities. Note that it is only the leaves of the structure that comprise the list of along with task Input activities in the project — higher-level nodes merely represent collections of activities

The WBS also represents a structure that may be refined as the project proceeds. In the early part of a project, we might use a relatively high-level or shallow WBS, which can be developed as information becomes available, typically during the project's analysis and specification phases.

Once the project's activities have been identified (whether or not by using a WBS), they need to be sequenced in the sense of deciding which activities need to be completed before others can start.

Product-Based Approach

It consists of producing a Product Breakdown Structure and a Product Flow Diagram. The PFD indicates, for each product, which other products are required as inputs. The PFD can therefore be easily transformed into an ordered list of activities by identifying the transformations that turn some products into others. Proponents of this approach claim that it is less likely that a product will be left out of a PBS than that an activity might be omitted from an unstructured activity list.

This approach is particularly appropriate if using a methodology such as SSADM or USDP (Unified Software Development Process), which clearly specifies, for each step or task, each of the products required and the activities required to produce it.

In the USDP, products are referred to as artifacts - see Figure 3.3 — and the sequence of activities needed to create them is called a workflow— see Figure 3.4 for an example. Some caution is needed in drawing up an activity network from these workflows. USDP emphasizes that processes are iterative. This means that it may not be possible to map a USDP process directly onto a single activity in a network. In Section 4.18 we saw how one or more iterated processes could be hidden in the single execution of a larger activity. All projects, whether they contain iterations or not, will need to have some fixed milestones or time-boxes if progress towards a planned delivery date is to be maintained. These larger activities with the fixed completion dates would be the basis of the activity network.

Hybrid Approach

The WBS illustrated in Figure 3.2 is based entirely on a structuring of activities• Alternatively, and perhaps more commonly, a WBS may be based upon the project products as illustrated in Figure 3.5, which is in turn based on a simple list of final deliverables and, for each deliverable, a set of activities required to produce that product.

The degree to which the structuring is product-based or activity-based might be influenced by the nature of the project and the particular development method adopted. As with a purely activity-based WBS, having identified the activities we are then left with the task of sequencing them.

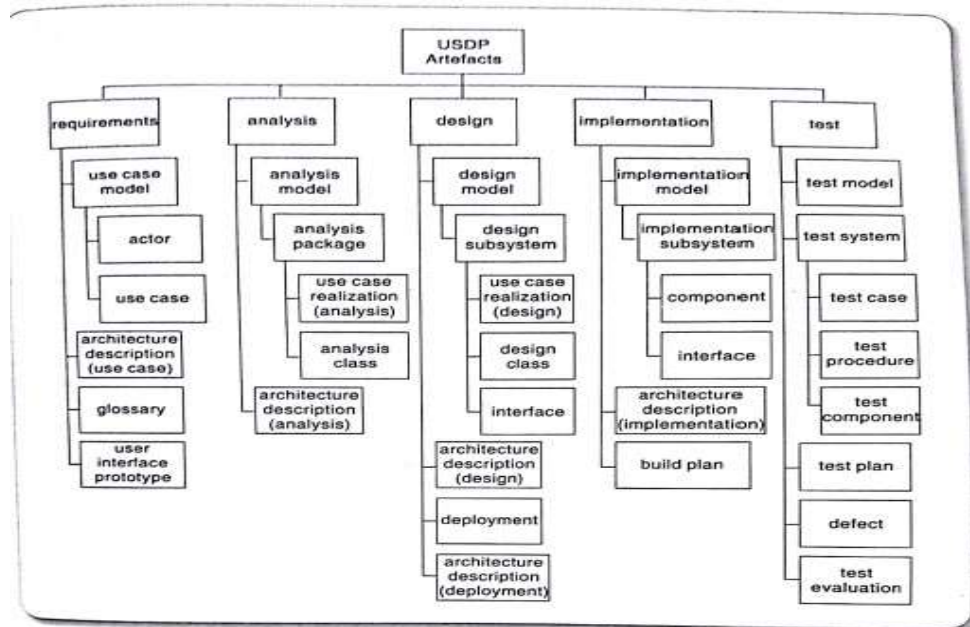


FIGURE 6.3 USDP product breakdown structure based on artefacts identified in Jacobson, Booch and Rumbaugh (1999)

A framework dictating the number of levels and the nature of each level in the structure may be imposed on a WBS. For example, in their MITP

methodology, IBM recommends that the following five levels should be used in a WBS:

- Level 1: Project
- Level 2: Deliverables such as software, manuals and training courses

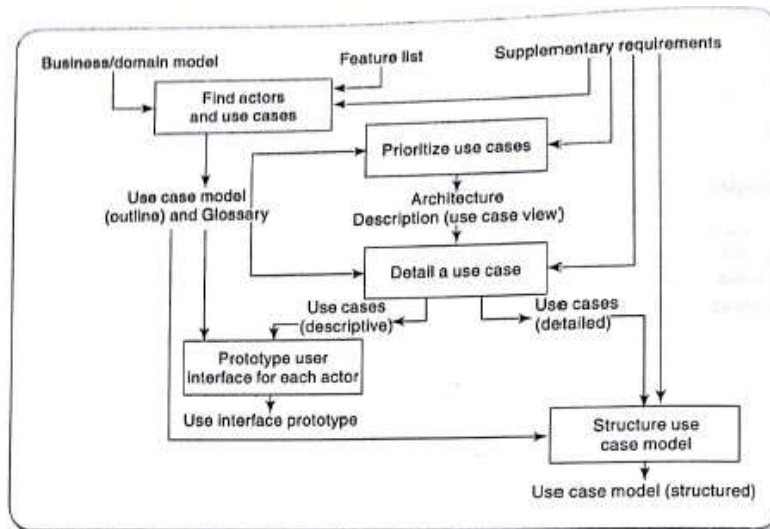


FIGURE 6.4 A structuring of activities for the USDP requirements capture workflow based on Jacobson, Booch and Rumbaugh (1999)

- Level 3: Components, which are the key work items needed to produce deliverables, such as the modules and tests required to produce the system software
- Level 4: Work-packages, which are major work items, or collections of related tasks, required to produce a component
- Level 5: Tasks, which are tasks that will normally be the responsibility of a single person

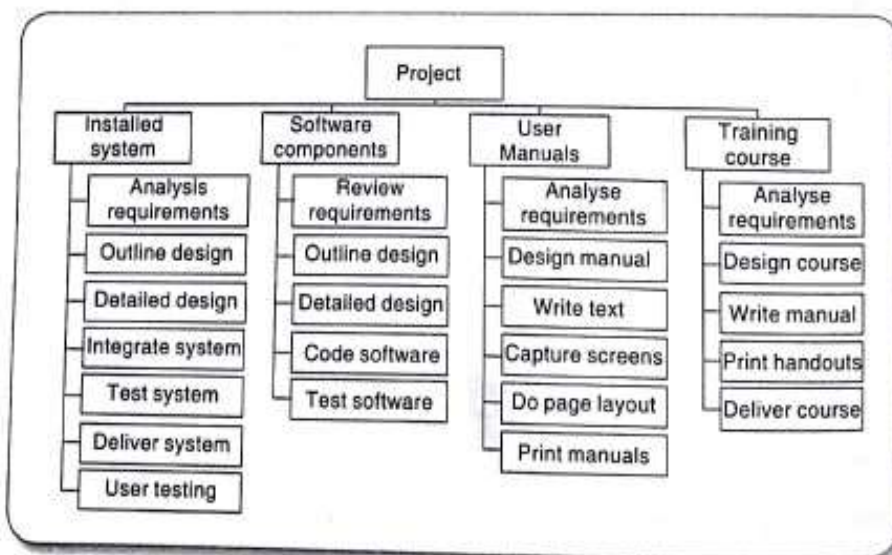


FIGURE 6.5 A hybrid Work Breakdown Structure based on deliverables and activities

7.6 SEQUENCING AND SCHEDULING ACTIVITIES

Throughout a project, we will require a schedule that clearly indicates when each of the project's activities is planned to occur and what resources it will need. We shall be considering scheduling in more detail in Chapter 8 but let us consider in outline how we might present a schedule for a small project.

The chart shown has been drawn up taking account of the nature of the development Process (that is, certain tasks must be completed before others may start) and the resources that are available (for example, activity C follows activity B because Andy cannot work on both tasks at the same time). In drawing up the chart, we have therefore done two things — we have sequenced the tasks (that is, identified the dependencies among activities dictated by the development process) and scheduled them (that is, specified when they should take place). The scheduling has had to take account of the availability of staff and the ways in which the activities have been allocated to them. The schedule might look quite different were there a different number of staff or were we to allocate the activities differently

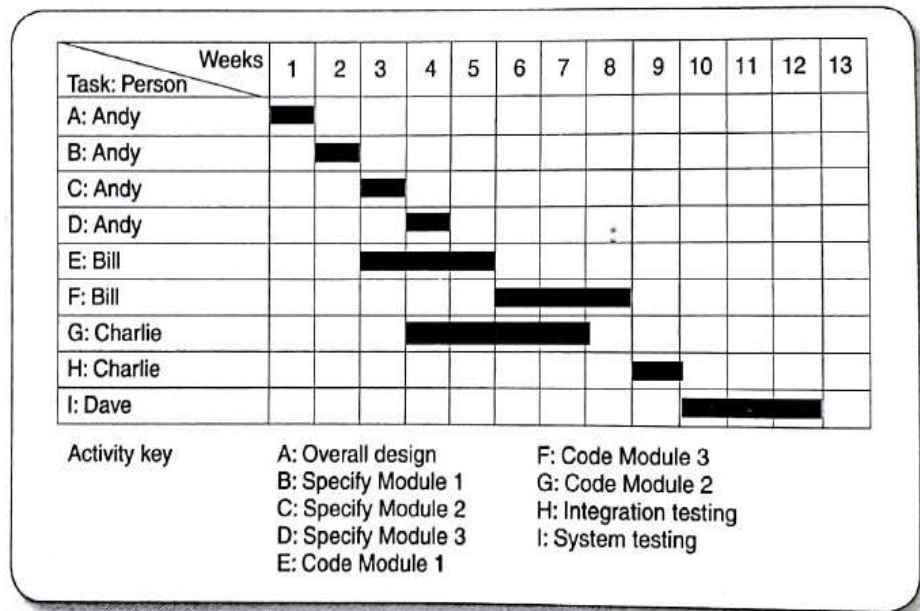


FIGURE 6.6 A project plan as a bar chart

In the case of small projects, this combined sequencing—scheduling approach may be quite suitable, particularly where we wish to allocate individuals to particular tasks at an early planning stage. However, on larger projects it is better to separate out the two activities: to sequence the tasks according to their logical relationships and then to schedule them taking into account resources and other factors. Approaches to scheduling that achieve this separation between the logical and the physical use networks to model the project and it is these approaches that we will consider in subsequent sections of this chapter.

These project scheduling techniques model the project's activities and their relation as a network. In the network, time flows from left to right. These techniques were originally developed in the 1950s — the two best known being CPM (Critical Path Method) and PERT (Program Evaluation Review Technique).

Both of these techniques used an activity-on-arrow approach to visualizing the project as a network where activities are drawn as arrows joining circles, or nodes, which represent the possible start and/or completion of an activity or set of activities. More recently a variation on these techniques, called precedence networks, has become popular. This method uses activity-on-node networks where activities are represented as nodes and the links between nodes represent precedence (or sequencing) requirements. This latter approach avoids some of the problems inherent in the activity-on-arrow representation and provides more scope for easily representing certain situations. It is this method that is adopted in the majority of computer applications currently available. These three methods are very similar and it must be admitted that many people use the same name (particularly CPM) indiscriminately to refer to any or all of the methods.

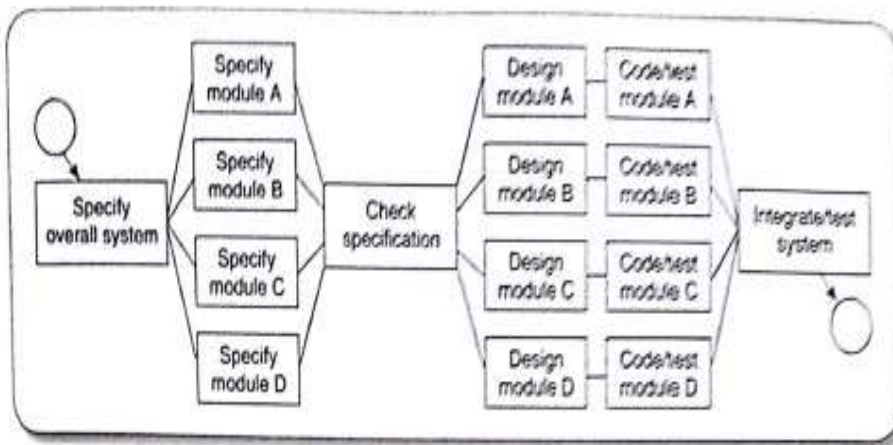


FIGURE 6.7 The IOE annual maintenance contracts project activity network fragment with a check-point activity added

7.8 FORMULATING TO A NETWORK MODEL

The first stage in creating a network model is to represent the activities and their interrelationships as a graph. In activity-on-node we do this by representing activities as nodes (boxes) in the graph — the lines between nodes represent dependencies.

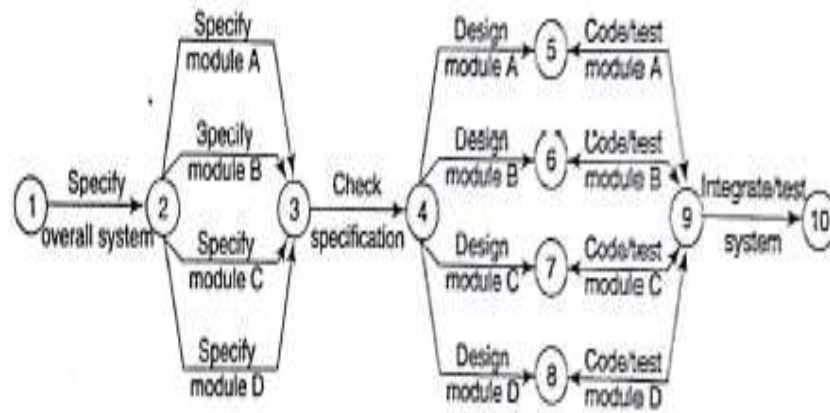


FIGURE 6.8 The IOE annual maintenance contracts project activity network fragment represented as a CPM network

Constructing precedence networks:

Before we look at how networks are used, it is worth spending a few moments considering some rules for their construction. A project network should have only one start node Although it is logically possible to draw a network with more than one starting node, it is undesirable to do so as it is a potential source of confusion. In such cases (for example, where more than one activity can start immediately the project starts) it is normal to invent a 'start' activity which has zero duration but may have an actual start date. A project network should have only one end node The end node designates the completion of the project, and a project may finish only once! Although it is possible to draw a network with more than one end node, it will almost certainly lead to confusion if this is done. Where the completion of a project depends upon more than one 'final' activity it is normal to invent a 'finish' activity. A node has duration A node represents an activity and, in general, activities take time to execute. Notice, however, that the network in Figure 3.7 does not contain any reference to durations. This network drawing merely represents the logic of the project - the rules governing the order in which activities are carried out

Links normally have no duration Links represent the relationships between activities In Figure 3.9 installation cannot start until program testing is complete, program testing cannot start until both coding and data take-on have been completed.

Precedents are the immediately preceding activities in figure6,9, the activity 'program test' cannot start until both 'Code' and 'Data take-on' have been completed and activity 'Install' cannot start until Program test' has, finished. 'Code' and 'Data take-on' can therefore be said to be precedents of 'Program test'. And 'Program test' is a precedent of 'Install'. Note that we do not speak of 'Code' and 'Data take-on' as precedents of 'brush' - that relationship is implicit in the previous statement.

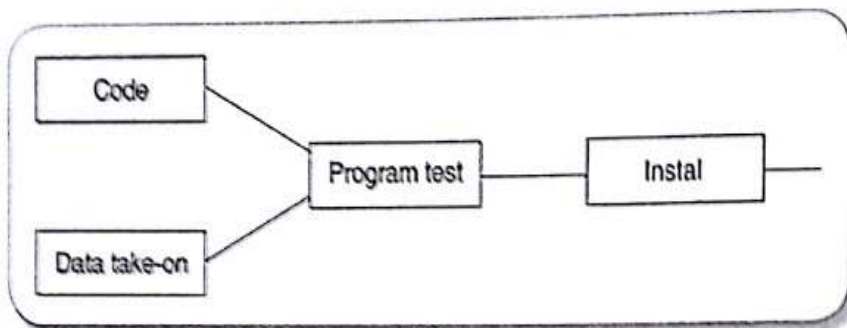


FIGURE 6.9 Fragment of a precedence network

Time moves from left to right. If at all possible, networks are drawn so that time moves from left to right. It is rare that this convention needs to be flouted, but some people add arrows to the lines to give a stronger visual indication of the time flow of the project.

fig.3.10 demonstrates a loop in a network. A loop is an error in that it represents a situation that cannot be occurred in practice. While loops, in the sense of iteration, may occur in practice, they cannot be directly represented in a project network.

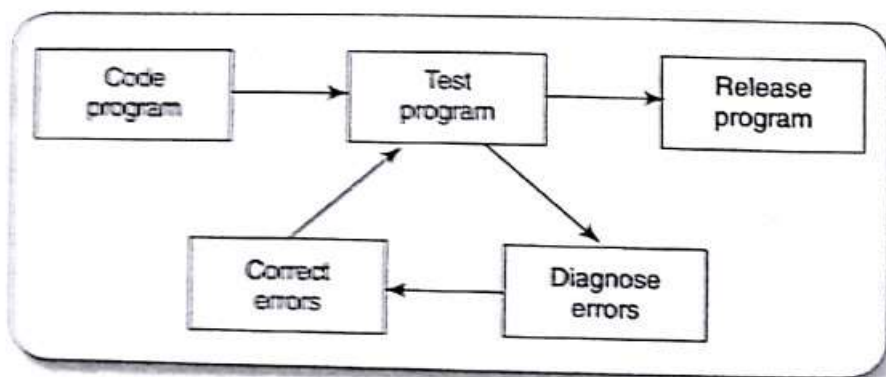


FIGURE 6.10 A loop represents an impossible sequence

Although it is easy to see the loop in this simple network fragment, very large networks can easily contain complex loops which are difficult to spot when they are initially constructed. Fortunately, all network planning applications will detect loops and generate error messages when they are found. A network should not 'twain dangles'. A dangling activity such as 'Write user manual' in Figure 3.1 I should not exist as it is likely to lead to errors in subsequent analysis. Indeed, in many cases, dangling activities indicate errors in logic when activities are added as an afterthought. If, in Figure 3.1 I, we mean to indicate that the project is complete once the software has been installed and the user manual written then we should redraw the network with a final completion activity — which, at least in this case, is probably a more accurate representation of what should happen. The redrawn network is shown in Figure 3.12.

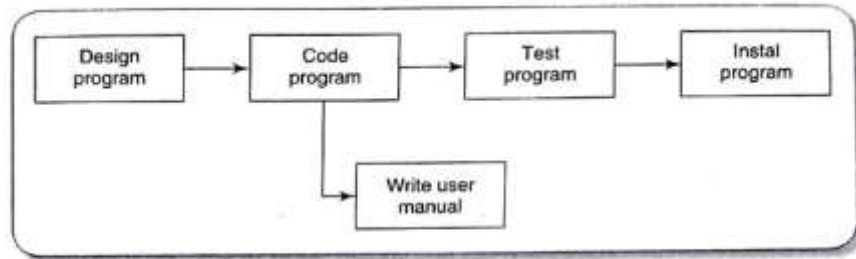


FIGURE 6.11 A dangle

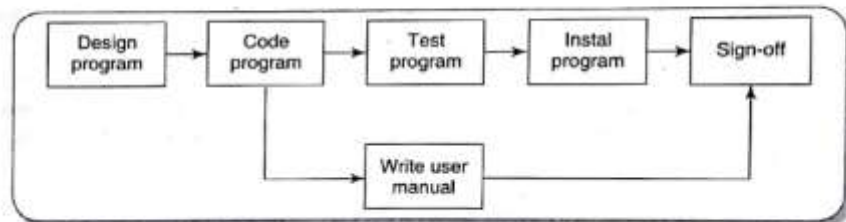


FIGURE 6.12 Resolving the dangle

Representing Lagged Activities

We might come across situations where we wish to undertake two activities in parallel, so long as there is a lag between the two. We might wish to document amendments to a program as it is being tested — particularly if evaluating a prototype. In such a case, we could designate an activity 'test and document amendments'. This would, however, make it impossible to show that amendment recording could start, say, one day after testing had begun and finish a little after the completion of testing. Where activities can occur in parallel with a time lag between them, we represent the lag with a duration on the g, e linking arrow as shown in Figure 3.13. This indicates that documenting amendments can start one day after the start of prototype testing and will be completed two days after prototype testing is completed.

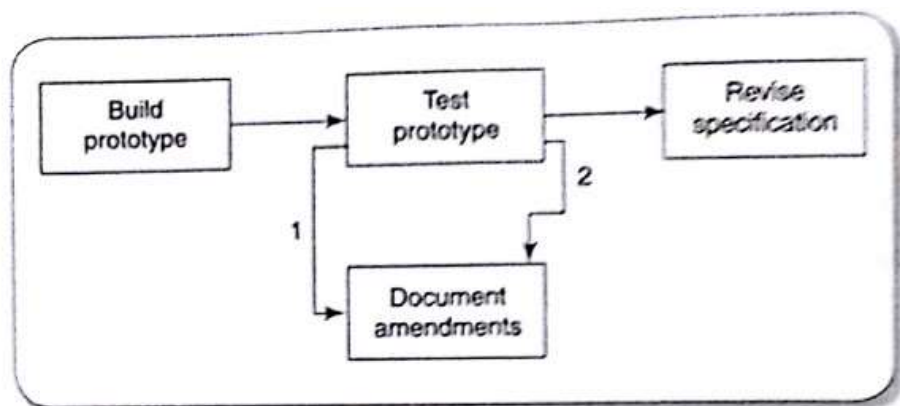


FIGURE 6.13 Indicating lags

Hammock Activities

Hammock activities are activities which_ in themselves, have zero duration but are assumed to start at the same time as the first 'hammocked' activity and to end at the same as the last one. They are normally

used for representing overhead costs or other resources that will be incurred or used at a constant rate over the duration of a set of activities.

Labelling Conventions

There are a number of differing conventions that have been adopted for entering information on an activity -on-node network. The one adopted here is shown on the left and is based on the British standard BS4335

7. ADDING THE TIME DIMENSION

Having created the logical network model indicating what needs to be and the relationships between those activities, we are now ready to start thinking about when each activity should be undertaken.

The critical path approach is concerned with two primary objectives: planning the project in such a way that it is completed as quickly as possible; and identifying those activities where a delay is likely to affect the overall end date of the project or later activities*start dates the method requires that for each activity we have estimate of its duration. The network is then analysed by carrying out a forward pass to calculate the earliest dates at which activities may commence, and the project be completed, and a backward pass, to calculate the latest start dates for activities and the critical path.

In practice, we would use a software application to carry out these calculations; for anything but the smallest of projects. It is important, though, that we understand how the calculation; are carried out in order to interpret the results correctly and understand the limitations of the method.

7.10 THE FORWARD PASS

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

Where an actual start date is known, the calculations may be carried out using actual dates. Alternatively, we can use day or week numbers and that is the approach we shall adopt here, by convention, dates the end of a period and the project is therefore shown as starting at the end of week zero (or the beginning of week 1).

7.11 THE BACKWARD PASS

The second stage in the analysis of a critical path network is to carry out a backward pass to calculate the latest date at which each activity may be started and finished without delaying the end date of the project. In calculating the latest dates, we assume that the latest finish date for the project is the same as the earliest finish date - that is, we wish to complete the project as early as possible.

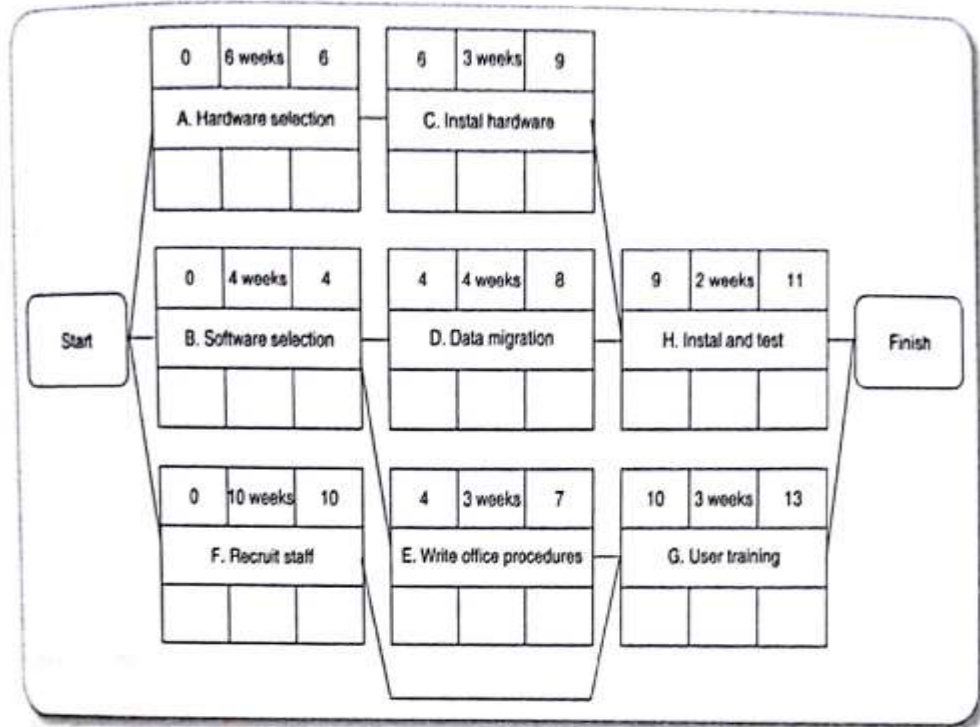


FIGURE 6.15 The network after the forward pass

Figure 3.16 illustrates our network after carrying out the backward pass. The latest activity dates are calculated as follows.

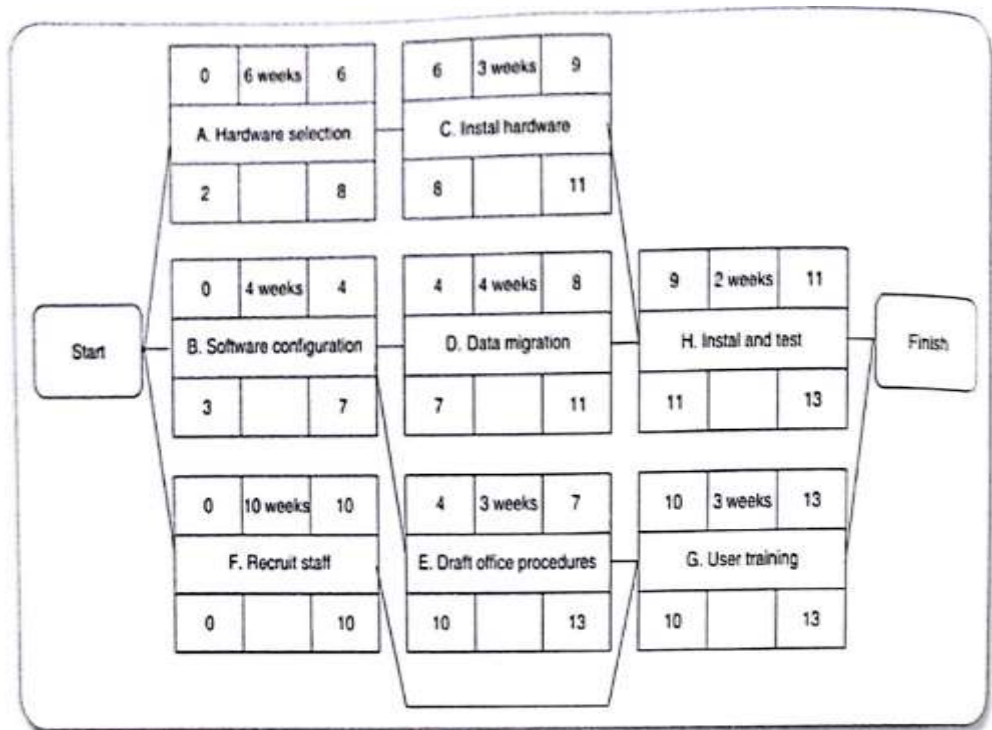


FIGURE 6.16 The network after the backward pass

- The latest completion date for activities G and H is assumed to be week 13.

- Activity H must therefore start at week 11 at the latest ($13 - 2$) and the latest start date for activity G is week 10 ($13 - 3$).
- The latest completion date for activities C and D is the latest date at which activity H must start that is week 11. They therefore have latest start dates of week 8 ($11 - 3$) and week 7 (11 respectively).
- Activities E and F must be completed week 10 so their earliest start dates are weeks 7 ($10 - 3$) and 0 ($10 - 10$) respectively.
- Activity B must be completed by week 7 (the latest start date for both activities D and E) and so its latest start is week 3 ($7 - 4$).
- Activity A must be completed by week 8 (the latest start date for activity C) so its latest start is week 2 ($8 - 6$).
- The latest start date for the project start is the earliest of the latest start dates for activities A, B and F.

This is week zero. This is, of course, not very surprising since it tells us that if the project does not start on time it won't finish on time.

7.12 IDENTIFYING THE CRITICAL PATH

There will be at least one path through the network (that is, one set of successive activities) that defines the duration of the project. This is known as the critical path. Any delay to any activity on this critical path will delay the completion of the project. The difference between an activity's earliest start date and its latest start date (or, equally, the difference between its earliest and latest finish dates) is known as the activity's float — it is a measure of how much the start or completion of an activity may be delayed without affecting the end date of the project. Any activity with a float of zero is critical in the sense that any delay in carrying out the activity will delay the completion date of the project as a whole. There will always be at least one path through the network joining those critical activities — this path is known as the critical path and is shown bold in Figure 3.17.

The significance of the critical path is two-fold.

- In managing the project, we must pay particular attention to monitoring activities on the critical path so that the effects of any delay or resource unavailability are detected and corrected at the earliest opportunity.

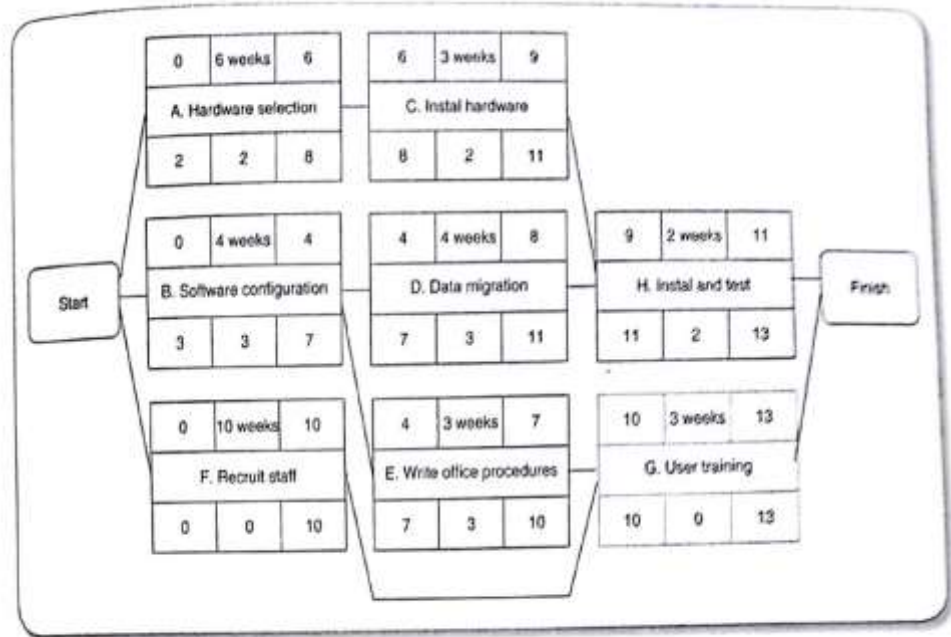


FIGURE 6.17 The critical path

- In planning the project, it is the critical path that we must shorten if we are to reduce the overall duration of the project.

Figure 3.17 also shows the activity span. This is the difference between the earliest start date and the latest finish date and is a measure of the maximum time allowable for the activity. However, it is subject to the same conditions of interpretation as activity float, which is discussed in the next section.

7.13 ACTIVITY FLOAT

Although the total float is shown for each activity, it really 'belongs' to a path through the network. Activities A and C in Figure 3.17 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!

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

- **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 the earliest start date of the succeeding activity. This might be considered a more satisfactory measure of float for publicizing to the staff involved in undertaking the activities
- **Interfering float:** The difference between total float and free float. This is quite commonly used, particularly in association with the free float. Once the free float has been used (or if it is zero), the

interfering float tells us by how much the activity may be delayed without delaying the project end date — even though it will delay the start of subsequent activities.

7.14 SHORTENING THE PROJECT DURATION

If we wish to shorten the overall duration of a project, we will normally consider attempting to reduce durations. In many cases, this can be done by applying more resources to the task - working overtime or procuring additional staff, for example. The critical path indicates where we must look to save time - if we are trying to bring forward the end date of the project, there is clearly no point in attempting to shorten non-critical activities. Referring to Figure 3.17, it can be seen that we could complete the project in week 12 by reducing the duration of activity F by one week (to 9 weeks).

As we reduce activity times along the critical path, we must continually check for any new critical path emerging and redirect our attention where necessary.

There will come a point when we can no longer safely, or cost-effectively, reduce critical activity durations in an attempt to bring forward the project end date. Further savings, if needed, must be sought in a consideration of our work methods and by questioning the logical sequencing of activities. Generally, time savings are to be found by increasing the amount of parallelism in the network and the removal of bottlenecks (subject always, of course, to resource and quality constraints).

7.15 IDENTIFYING CRITICAL ACTIVITIES

The critical path identifies those activities which are critical to the end date of the project; however, activities that are not on the critical path may become critical. As the project proceeds, activities will invariably use up some of their float and this will require a periodic recalculation of the network. As soon as the activities along a particular path use up their total float then that path will become a critical path and a number of hitherto non-critical activities will suddenly become critical.

It is therefore common practice to identify near-critical paths — those whose lengths are within, say, 10-20% of the duration of the critical path or those with a total float of less than, say, 10% of the project's uncompleted duration.

The importance of identifying critical and near-critical activities is that it is they that are most likely to be the cause of delays in completing the project. We shall see, in the next three chapters, that identifying these activities is an important step in risk analysis, resource allocation and project monitoring.

7.16 ACTIVITY-ON-ARROW NETWORKS

The developers of the CPM and PERT methods both originally used activity-on-arrow network. Although now less common than activity-on-node networks, they are still used and introduce an additional useful Concept — that of events. We will therefore take a brief look at how they are drawn and analysed using the same project example shown in Table 3.1.

In activity-on-arrow networks activities are represented by links (or arrows) and the nodes represents events of activities (or groups of activities) starting or finishing. Figure 3.18 illustrates our previous example (see Figure 3.14) drawn as an activity-on-arrow network.

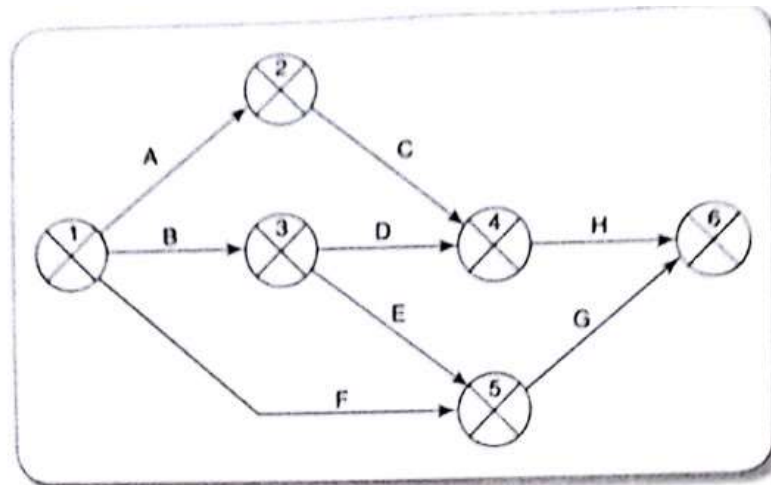


FIGURE 6.18 An activity-on-arrow network

Activity-On-Arrow Network Rules and Conventions

A project network may have only one start node. This is a requirement of activity-on-arrow network rather than merely desirable as is the case with activity-on-node networks.

A project network may have only one end node. Again this is a requirement for activity-on-arrow network.

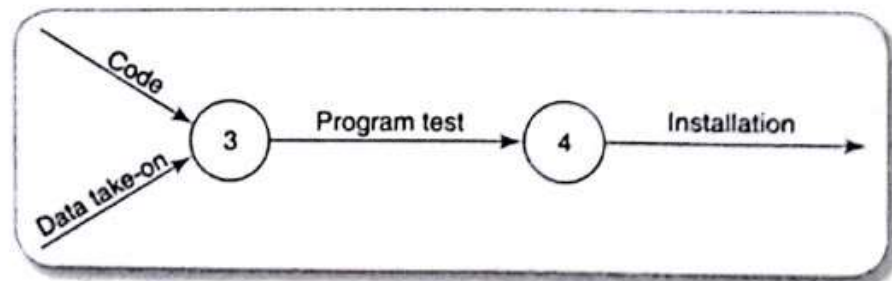


FIGURE 6.19 Fragment of a CPM network

A link has duration: A link represents an activity and, in general, activities take time to execute. Notice, however, that the network in Figure 3.18

does not contain any reference to durations. The links are not drawn in any way to represent the activity durations. The network drawing merely represents the logic of the project, the rules governing the order in which activities are to be carried out.

Nodes have no duration Nodes are events and, as such, are instantaneous points in time. The source node is the event of the project becoming ready to start and the sink node is the event of the project becoming completed. Intermediate nodes represent two simultaneous events -- the event of all activities leading into a node having been completed and the event of all activities leading out of that node being in a position to be started.

In Figure 3.19, node 3 is the event that both 'coding' and 'data take-on' have been completed and activity 'program test' is free to start. Installation may be started only when event 4 has been achieved, that is as soon as 'program test' has been completed.

Time moves from left to right as with activity-on-node networks, activity-on-arrow networks are drawn, if at all possible, so that time moves from left to right.

Nodes are numbered sequentially There are no precise rules about node numbering, but nodes should be numbered so that head nodes (those at the 'arrow' end of an activity) always have a higher number than tail events (those at the 'non-arrow' end of an activity). This convention makes it easy to spot loops. A network may not contain loops Figure 6,2() demonstrates a loop in an activity-on-arrow network. As discussed in the context of precedence networks, loops are either an error of logic or a situation that must be resolved by itemizing iterations of activity groups.

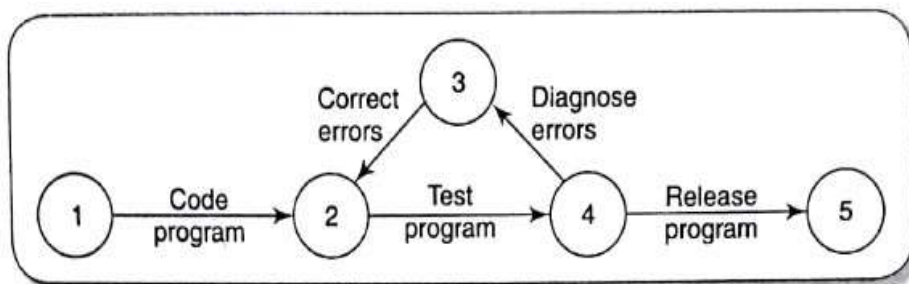


FIGURE 6.20 A loop represents an impossible sequence

A network may not contain dangles A dangling activity, such as 'Write user manual' in Figure 3.21, cannot exist, as it would suggest there are two completion points for the project. If, in Figure 3.21, node 5 represents the true project completion point and there are no activities dependent on activity 'Write user manual', then the network should be redrawn so that activity 'Write user manual' starts at node 2 and terminates at node 5 — in practice, we would need to insert a dummy activity between nodes 3 and 5. In other words, all events, except the first and the last, must have at least one activity entering them and at least one activity leaving them and all activities must start and end with an event.

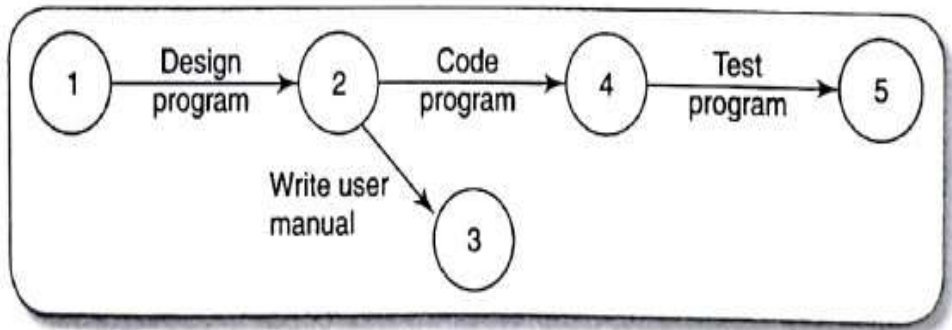


FIGURE 6.21 A dangle

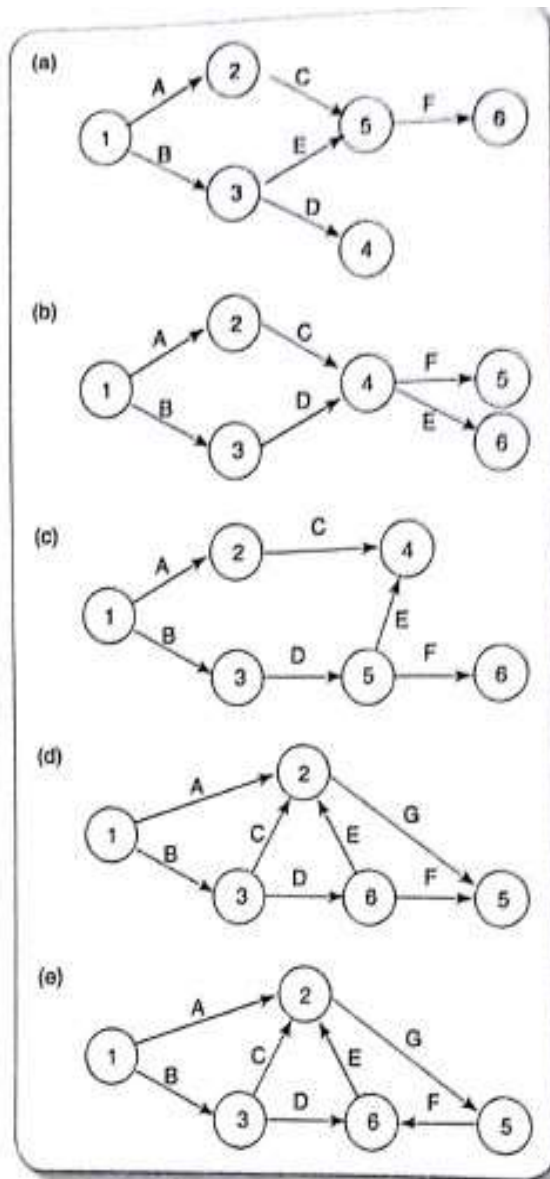


FIGURE 6.22 Some activity networks

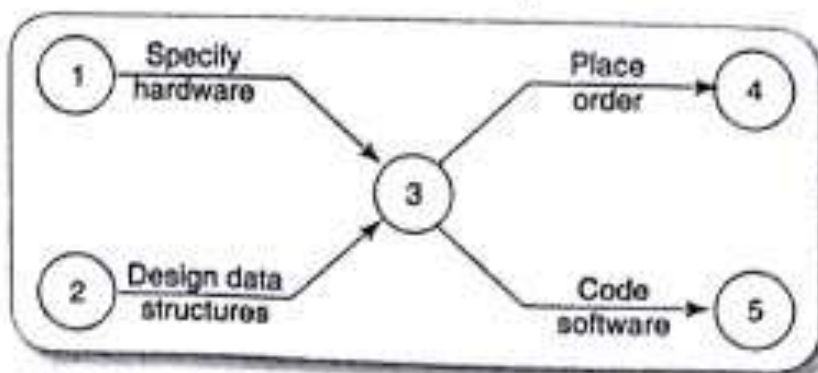


FIGURE 6.23 Two paths with a common node

Using Dummy Activities

When two paths within a network have a common event although they are, in other respects, independent, a logical error such as that illustrated in Figure 3.23 might occur. Suppose that, in a particular project, it is necessary to specify a certain piece of hardware before placing an order for it and before coding the software. Before coding the software, it is also necessary to specify the appropriate data structures, although clearly, we do not need to wait for this to be done before the hardware is ordered. Figure 3.23 is an attempt to model the situation described above, although it is incorrect in that it requires both hardware specification and data structure design to be completed before either an order may be placed, or software coding may commence. We can resolve this problem by separating the two (more or less) independent paths and introducing a dummy activity to link the completion of 'specify hardware' to the start of the activity 'code software'. This effectively breaks the link between data structure design and placing the order and is shown in Figure 3.24.

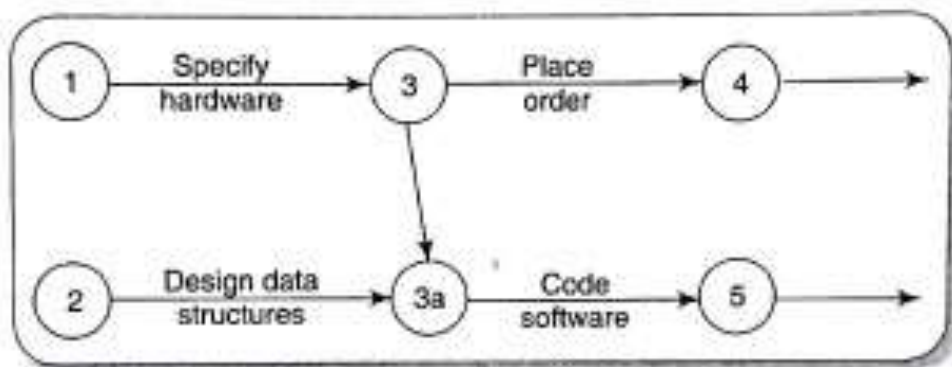


FIGURE 6.24 Two paths linked by a dummy activity

Dummy activities, shown as dotted lines on the network diagram, have a zero duration and use no resources. They are often used to aid in the layout of network drawings as in Figure 3.25. The use of a dummy activity where two activities share the same start and end nodes makes it easier to distinguish the activity endpoints.

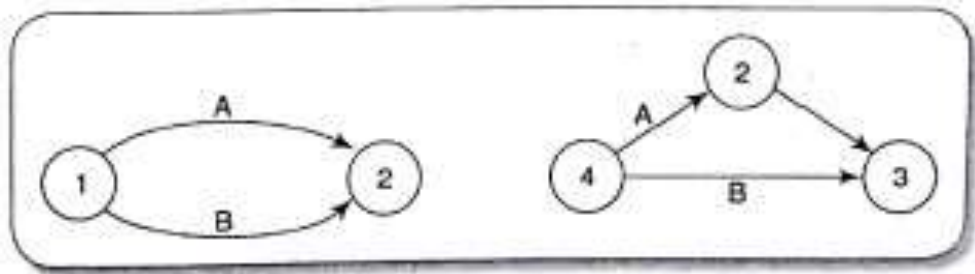


FIGURE 6.25 Another use of a dummy activity

These are problems that do not occur with activity-on-node networks.

Representing Lagged Activities

Activity-on-arrow networks are less elegant when it comes to representing lagged parallel activities. We need to represent these with pairs of dummy activities as shown in Figure 3.2. Where the activities are lagged because a stage in one activity must be completed before the other may proceed, it is likely to be better to show each stage as a separate activity.

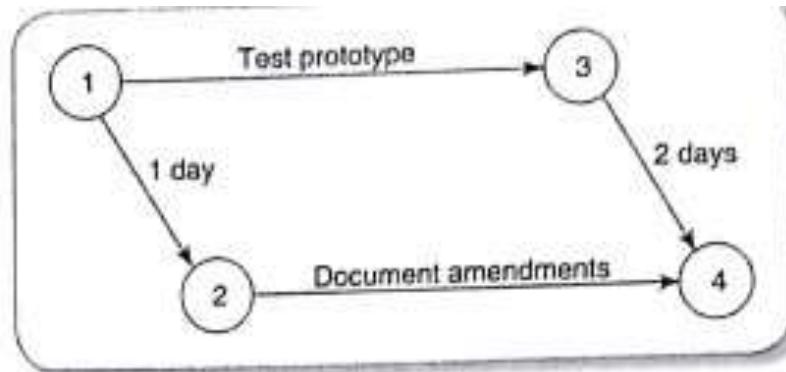


FIGURE 6.26 Using the ladder technique to indicate lags

Activity Labelling

There are several differing conventions that have been adopted for entering information on an activity-on-arrow network. Typically, the diagram is used to record information about the events rather than the activities - activity-based information (other than labels or descriptions) is generally held on a separate activity table.

One of the more common conventions for labelling nodes, and the one adopted here, is to divide the node circle into quadrants and use those quadrants to show the event number, the latest and earliest dates by which the event should occur, and the event slack (which will be explained later).

Network Analysis

Analysis proceeds in the same way as with activity-on-node networks, although the discussion places emphasis on the events rather than activity start and completion times.

The forward pass is carried out to calculate the earliest date on which each event may be achieved and the earliest dates on which each activity may be started and completed. The earliest date for an event is the earliest date by all activities upon which it depends can be completed. Using Figure 6.18 and Table 6.1, the calculation proceeds according to the following reasoning.

- Activities A, B and F may start immediately, so the earliest date for event 1 is zero and the earliest start date for these three activities is also zero.
- Activity A will take 6 weeks, so the earliest it can finish is week 6 (recorded in the activity table). Therefore, the earliest we can achieve event 2 is week 6.
- Activity B will take 4 weeks, so the earliest it can finish and the earliest we can achieve event 3 is week 4.

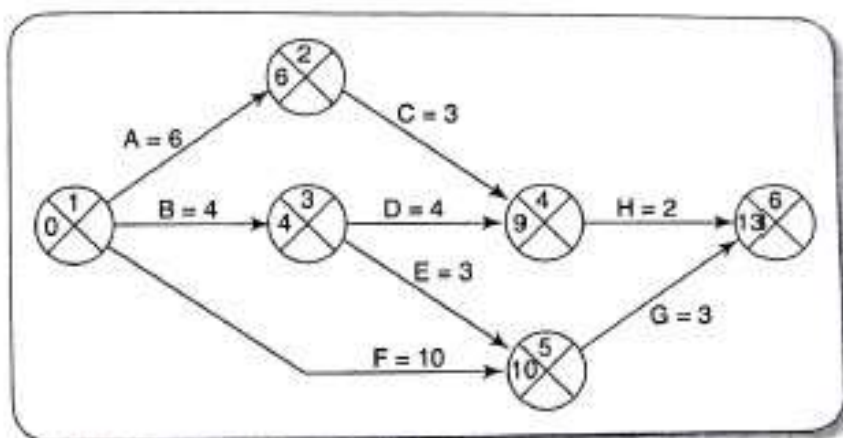


FIGURE 6.27 A CPM network after the forward pass

- Activity F will take 10 weeks, so the earliest it can finish is week 10 — we cannot, however, tell whether or not this is also the earliest date that we can achieve event 5 since we have not, as yet, calculated when activity E will finish.
- Activity E can start as early as week 4 (the earliest date for event 3) and, since it is forecasted to take 3 weeks, will be completed, at the earliest, at the end of week 7.
- Event 5 may be achieved when both E and F have been completed, that is, week 10 (the later of 7 and 10).
- Similarly, we can reason that event 4 will have the earliest date of week 9. This is the later of the earliest finish for activity D (week 8) and the earliest finish for activity C (week 9).
- The earliest date for the completion of the project, event 6, is therefore the end of week 13 — the later of 11 (the earliest finish for H) and 13 (the earliest finish for G).

The results of the forward pass are shown in Figure 3.27 and Table 3.3.

The backward pass the second stage is to carry out a backward pass to calculate the latest date at which each event may be achieved, and each activity started and finished, without delaying the end date of the project. The latest date for an event is the latest date by which all immediately following activities must be started for the project to be completed on time. As with activity-on-node networks, we assume that the latest finish date for the project is the same as the earliest finish date — that is, we wish to complete the project as early as possible. Figure 3.28 illustrates our network and Table 3.4 the activity table after carrying out the backward pass — as with the forward pass, event dates are recorded on the diagram and activity dates on the activity table.

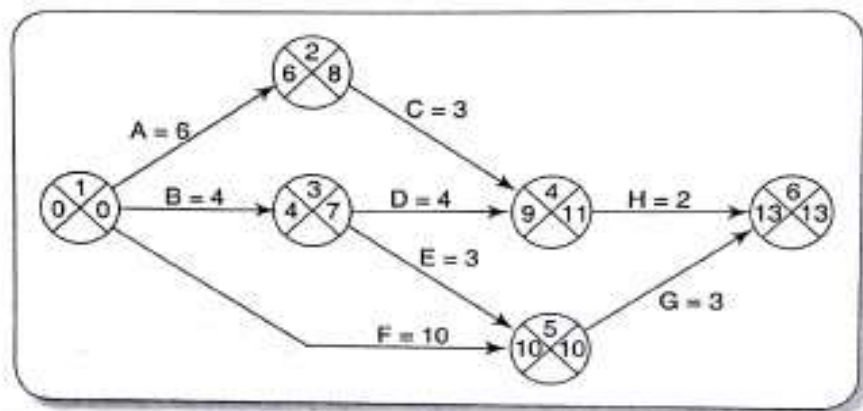


FIGURE 6.28 CPM network after the backward pass

TABLE 6.4 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	

Identifying the critical path, The critical path is identified in a way similar to that used in activity-on-node networks. We do, however, use a different concept, that of slack, in identifying the path. Slack is the difference between the earliest date and the latest date for an event — it is a measure of how late an event may be without affecting the end date of the project. The critical path is the path joining all nodes with a zero slack (Figure 3.29).

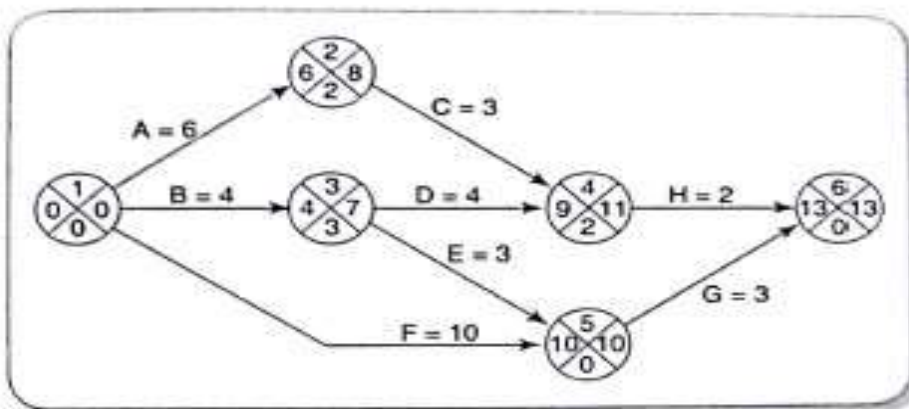


FIGURE 6.29 Critical path

7.17 SUMMARY

In this chapter, we have discussed the use of the critical path method and precedence networks to obtain an ideal activity plan. This plan tells us about the order in which we should execute activities and the earliest and latest we can start and finish them.

These techniques help us to identify which activities are critical to meeting a target completion date.

In order to manage the project, we need to turn the activity plan into a schedule that will specify precisely when each activity is scheduled to start and finish. Before we can do this, we must consider what resources will be required and whether or not they will be available at appropriate times. As we shall see, the allocation of resources to an activity may be affected by how we view the importance of the task and the risks associated with it. In the next two chapters we look at these aspects of project planning before we consider how we might publish a schedule for the project.

7.18 EXERCISES

- Draw an activity network using either activity-on-node or activity-on-arrow network conventions for each of the following projects:
 - Redecorating a room
 - Choosing and purchasing a desktop computer
 - Organizing and carrying out a survey of users' opinions of an information system
- If you have access to a project planning application, use it to produce a project plan for the IOE annual maintenance contracts project. Base your plan on that used for Exercise 3.2 and verify that your application reports the same information as you calculated manually when you did the exercise.
- Based on your answer to Exercise 3.2, discuss what options Amanda might consider if she found it necessary to complete the project earlier than day 104.

4. Create a precedence activity network using the following details:
5. Calculate the earliest and latest start and end dates and the float associated with each activity in the network you have created for further exercise 4 above. From this identify the critical path.
6. Draw up a precedence activity network for the following scenario: The specification of an ICT application is estimated as likely to take two weeks to complete. When this activity has been completed, work can start on three software modules, A, B and C. Design/coding of the modules will need 5, 10 and 10 days respectively. Modules A and B can only be unit-tested together as their functionality is closely associated. This joint testing should take about two weeks. Module C will need eight days of unit testing. When all unit testing has been completed, integrated system testing will be needed, taking a further three weeks. This testing will be based on the functionality described in the specification and will need 10 days of planning.
7. For the activity network in further exercise 6 above, derive the earliest and latest start dates for each activity and the earliest and latest finish dates. Work out the shortest project duration. If only two software developers were available for the design and coding of modules, what effect would this have on the project duration?
8. What are the limitations of the precedence and CPM activity network notations?
9. Consider a software project with five tasks T1-T5. Duration of the five tasks in weeks is 3, 2, 3, 5, and 2 respectively. T2 and T4 can start when T1 is complete. T3 can start when T2 is complete. T5 can start when both T3 and T4 are complete. Draw the CPM network representation of the project. When is the latest start date of the task T3? What is the float time of the task T4? Which tasks are on the critical path?
10. Consider a software project with five tasks that are denoted by T1, T2, T3, T4 and T5, Duration of these five tasks (in days) are 15, 10, 12, 25 and 10, respectively. T2 and T4 can start when T1 is complete. T3 can start when T2 is complete. T5 can start when both T3 and T4 are complete. What will be the latest start date of the task T3? What is the slack time of the task T4?
11. Why is it necessary for a project manager to decompose the tasks of a project using Work breakdown structure (WBS) into finer level tasks before constructing the task schedule? To what granularity level should the tasks be decomposed? Explain your answer.
12. For each of the following questions, exactly one option is correct. Select the appropriate option.
 - (i) Which one of the following charts would be the most useful to decompose the project activities into smaller tasks that are more meaningfully managed?
 - (a) PERT chart

- (b) GANTT chart
 - (c) Task network representation
 - (d) Work breakdown structure
- (ii) Which one of the following is the critical path for the activity network given below?
- (a) A, B, F,H, G, land L (b) A, B,KH,I and L
 - (c) A,C,D,J, K and L (d) A, C,D,G,K and L
- (iii) Consider a portion of the network diagram given below. What is the LF of activity F?
- (a) 10 (b) 11
 - (c) 16 (d) 17
- (iv) In a PERT chart, in which one of the following situations is a dummy activity required?
- (a) In the PERT chart, two or more activities have the same ending events.
 - (b) The PERT chart contains two or more activities that have identical starting and ending events.
 - (c) In the PERT chart, two or more activities have different ending events.
 - (d) In the PERT chart, two or more activities have the same starting events
- (v) Using the data in the following table, what is the total project duration?
- (a) 20 (b) 27
 - (c) 37 (d) 44
- (vi) PERT method differs from CPM in which one of the following aspects.
- (a) PERT uses statistical time durations whereas CPM uses deterministic time durations,
 - (b) PERT uses dummy activities whereas CPM does not.
 - (c) PERT uses free float, whereas CPM uses total float in critical path calculations.
 - (d) PERT uses activity on arc whereas CPM uses activity on node networks.
- (vii) Which one of the following is true of a critical path in a PERT chart?
- (a) It is the path having maximum number of tasks.
 - (b) It is the shortest path in terms of time.
 - (c) It is the longest path in terms of time.
 - (d) It is the path with the largest amount of slack time.

- (viii) Which one of the following statements regarding critical paths in a PERT chart is true?
 - (a) A critical path through a PERT chart is any path through the chart that contains the least number of edges.
 - (b) Some activities on the critical path can have slack.
 - (c) Every PERT chart has exactly one critical path.
 - (d) It is possible that in the PERT chart for a project, there can be multiple critical paths, all having the same duration.
- (ix) In a PERT chart, an activity has an early start (ES) of 3 days, a late start (LS) of 13 days, an early finish (EF) of 16 days and a late finish (LF) of 26 days. Which one of the following can be inferred regarding this activity?
 - (a) It is on the critical path. (b) It is not progressing well.
 - (c) It is progressing well. (d) It is not on the critical path.
- (x) Suppose you have estimated the nominal duration of your project to be 4 months and you have planned to complete the work by deploying three developers. However, the customer request you to complete the work in 3 months. In this case, what will be the manpower requirement as per Putnam's results?
 - (a) 6 (b) 8
 - (c) 10 (d) 20

Answer questions (xi) and (xii) for a project whose activities, their precedence ordering, estimated time for completion are given in the following table.

- (xi) Which one of the following sequence of activities is on the critical path?
 - (a) A-E-F (b) A-B-C-G
 - (c) A-B-C-D-F (d) A-B-F
- (xii) Which one of the following paths has the greatest slack time?
 - (e) A-E-F (f) A-B-C-G
 - (g) A-B-C-D-F (h) A-B-F

7.19 REFERENCES

- a) Software Project Management by Bob Hughes, Mike Cotterell, Rajib Mall, Tata McGraw Hill, 6th Edition, 2018.
- b) Project Management and Tools & Technologies – An overview by Shailesh Mehta, Shroff Publishers, 1st Edition, 2017.
- c) Software Project Management by Walker Royce, Pearson Publication, 2005



RISK MANAGEMENT

Unit Structure

- 8.0 Objectives
- 8.1 Introduction
- 8.2 Risk
- 8.3 categories of Risk
- 8.4 Risk Management Approaches
 - 8.4.1 Reactive approaches
 - 8.4.2 Proactive approach
- 8.5 A framework for dealing with Risk
- 8.6 Risk Identification
- 8.7 Risk Assessment
- 8.8 Risk Planning
- 8.9 Risk Management
 - 8.9.1 Contingency
 - 8.9.2 Deciding on the risk action
 - 8.9.3 Creating and maintaining the risk register
- 8.10 Evaluating Risk to the schedule
- 8.11 Boehm's Top 10 Risks and counter Measures
 - 8.11.1 Risk Mitigation Monitoring, and Management
- 8.12 Applying the PERT Technique
 - 8.12.1 Using expected duration
 - 8.12.2 Activity standard deviation
 - 8.12.3 The likelihood of meeting targets
 - 8.12.4 Calculating the standard deviation of each project event
 - 8.12.5 Calculating the z values
 - 8.12.6 Calculating z values to probabilities
 - 8.12.7 Advantages of PERT
- 8.13 Monte Carlo simulation
- 8.14 Critical chain concepts
 - 8.14.1 Deriving most likely activity durations
 - 8.14.2 using latest start dates for activities
 - 8.14.3 Inserting project and feeder buffers
 - 8.14.4 Project execution
- 8.15 Summary
- 8.16 Exercises
- 8.17 References.

8.0 OBJECTIVES

After going through this unit, you will be able to:

- Identify the factor putting a project at a risk
- Categorize and prioritize actions for risk elimination or containment
- Quantify the likely effect of risk on project timescales.

8.1 INTRODUCTION

In an earlier chapter we had seen that how the software for the new annual maintenance contracts application was to be produced which included estimation of how long each task would take. But suppose if one of the developer leaves for better paid job then it might further delay the process and getting the replacement immediately for the vacant post is bit difficult which will impact the overall growth. Such type of activities is termed as “Risks” which are uncertain about its occurrences and one has to be ready to take it and face the consequences of it which can be sometimes fruitful.

8.2 RISK

- PM-BOK defines risk as 'an uncertain event or condition that, if it occurs, has a positive or negative effect on a project's objectives'.
- The UK government sponsored project management standard, defines risk as 'the chance of exposure to the adverse consequences of future events.

The key elements of risk as follow

- It relates to the future The future is inherently uncertain. Some things which seem obvious project is over, for example that the costs relate to the future The future is inherently uncertain Some things which seem obvious when project is over, for example that the costs were underestimated or that a new technology was overly difficult to use, might not have been so obvious during planning.
- It involves cause and effect For example, a cost over-run might be identified as a risk, but cost over-run describes some damage, but does not say what causes it Is it, for example, an inaccurate estimate of effort, the use of untrained staff, or a poor specification Both the cause (or hazanf, such as 'inexperienced staff and a particular type of negative outcome such as lower productivity. should be defined for each risk

8.3 CATEGORIES OF RISK

- An ICT project management is normally given the objective of installing the required application by a specified deadline and within an agreed budget. Other objective might be set, especially with regard to quality requirements
- Project risks are those that could prevent the achievement of the objective given to the project manager and the project team.

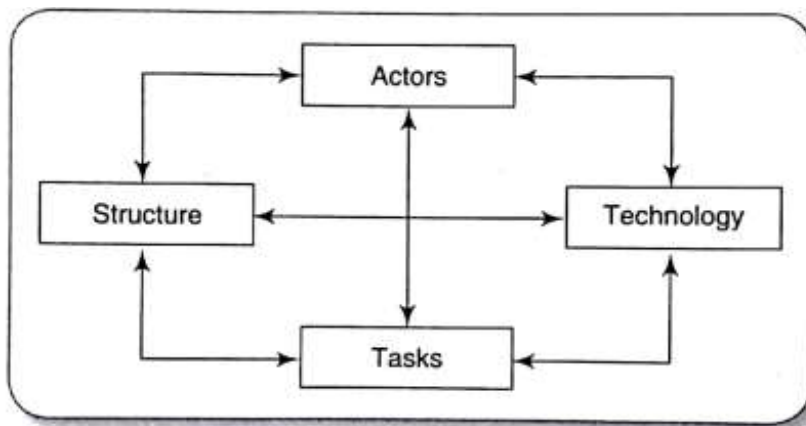


FIGURE 7.2 Lyttinen-Mathiassen-Ropponen risk framework

- In this figure 'Actors' refers to all the people involved in the development of the application in question. A typical risk in this area is that high staff turnover leads to the appropriateness of the technologies and to possible faults with them.
- Structure describes the management structured and system, including those affecting planning and control.
- Task 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
- All boxes are interlinked .Risk often arise from the relationships between factors –for example technology and people.

8.4 RISK MANAGEMENT APPROACHES

Risk management approaches can broadly be classified into reactive and proactive approaches. The latter approaches are much more effective in risk handling. And therefore, used wherever possible.

Reactive approaches

- Reactive approaches take no action until an unfavorable event occurs. Once an unfavorable event occur these approaches try to contain the adverse effects associated with the risk and take steps to prevent future occurrence of the same risk events

- An example of such a risk management strategy can be the following Consider a project in which the server hosting the project data crashes. Once this risk event has occurred, the team members may put best effort to recover the data and also initiate the practice of taking regular backups so that in future such a risk event does not recur.

Proactive approaches

- The proactive approaches try to anticipate the possible risks that the project is susceptible. After identifying the possible risk, actions are taken to eliminate the risks. If a risk cannot be avoided, these approaches suggest making plans to contain the effect of the risk .
- For example, if manpower turnover is anticipated (some personnel may leave the project), then thorough documentation may be planned Also, more than one developer may work on a work item and also some stand by man power may be planned Obviously proactive approaches incur lower cost and time overruns when ttd events occur and, therefore, are much more preferred by scam: However, when some risks cannot be anticipated, a reactive approach is usually followed.

8.5 FRAMEWORK FOR DEALING WITH RISK

- (i) Risk identification
- (ii) Risk analysis and prioritization
- (iii) Risk planning
- (iv) Risk monitoring

Steps (i) to (iii) above will probably be repeated. When risks that could prevent a project success are identified plans can be made to duce or remove their threat. The plans are then reassessed to ensure that the original risks are reduced sufficiently, and no new risks made inadvertently introduced take the risk that staff inexperience with a new technology could lead to delays in software development. To reduce this risk, consultant's expert in the new technology might be recruited. However, the use of consultants might introduce the new risk that knowledge about the new technology is not transferred to the permanent staff, making subsequent software maintenance problematic. Having identified this new risk further risk reduction activities can be planned.

8.6 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
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

- **Checklist**

Checklists are simply lists of the risks that have been found to occur regularly in software development projects. A specialized list of software development risks by Barry Boehm. In a modified version Ideally a group of representative project stakeholders examines a checklist identifying risks applicable to their project. Often the checklist suggests potential countermeasures for each risk

- **Brainstorming**

- Ideally, representatives of the main stakeholders should be brought together once some kind of preliminary plan has been drafted, they then identify, using their individual knowledge of different parts of the project the problems that might occur. This collaborative approach may generate a sense of ownership in the project
- Brainstorming might be used with Brigitte's Brightmouth payroll implementation project as she realizes that there are aspects of college administration of which she is unaware.

8.7 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 using the formula

Risk exposure = (potential damage) X (probability of occurrence)

Using the most rigorous but not necessarily the most practical approach, the potential damage would be assessed as a money value. Say 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 be estimated that where the computer is located there is a 1 in 1000 chance of a fire actually happening, a probability of 0.001. The risk exposure in this case would be

500,000 X 0.001

8.8 RISK PLANNING

Having identified the major risks and allocated priorities, the task is to decide how to deal with them. The choices discussed will be

- Risk acceptance
- Risk avoidance
- Risk reduction and mitigation
- Risk transfer
- **Risk acceptance**

This is the do-nothing option. We will already in the risk prioritization process, have decided to ignore some risk in order to concentrate on the more likely damaging. We could decide that the damage inflicted by some risk would be less than the cost of action that might reduce the probability of a risk happening.

- **Risk avoidance**

Some activities may be prone to accident that is best to avoid them altogether. If you are worried about shark they don't go into the water. For example given all the problems with developing software solution from scratch manager might decide to retain existing clerical method or to buy an off-the-shelf solution.

- **Risk reduction and mitigation**

Risk mitigation can sometime be distinguished from risk reduction. Risk reduction attempt to reduce likelihood of the risk occurring. Risk mitigation taken to ensure that the impact of the risk is lessened.

when its occur. for example, taking ,taking regular backup of data storage would reduce the impact of data corruption but not its likelihood .Mitigation is closely associated with contingency planning which is discussed presently.

- **Risk Transfer**

In this case the risk is transferred to other person or organization you might the except the supplier to quote a higher figure to cover the risk that the project takes longer than the average excepted time.

8.9 RISK MANAGEMENT

- **Contingency**

Risk reduction activities would appear to have only a small impact on reducing the probability of some risks, for example staff absence through illness. While some employers encourage their employees to adopt a healthy lifestyle, it remains likely that some project team members will at some point be brought down by minor illnesses such as flu. These kinds of risk need a contingency plan. This is a planned action to be carried out if the particular risk materializes. If a team member involved in urgent work were ill then the project manager might draft in another member of staff to cover that work.

The preventative measures that were discussed under the 'Risk reduction' heading above will usually incur some cost regardless of the risk materializing or not. The cost of a contingency measure will only be incurred if the risk materializes. However, there may be some things that have to be done for the contingency action to be feasible. An obvious example is that back-ups of a database must be taken if the contingency action when the database is corrupted is to restore it from back-ups. There would be a cost associated with taking the back-ups.

- **Deciding on the risk actions**

- I. Five generic responses to a risk have been discussed above. For each actual risk, however, specific actions likelihood of particular risks; see, for example, Bochum's 'top ten' software engineering risks in Table 7.1. Have to be planned. In many cases experts have produced lists recommending practical steps to cope with the likelihood of a particular risk.
- II. Whatever the countermeasures that are considered, they must be cost-effective. On those occasions where a risk exposure value can be calculated as a financial value using the (value of damage) X (probability of occurrence) formula-recall the cost-effectiveness of a risk reduction action can be assessed by calculating the risk reduction leverage (RRL).

III. Risk reduction leverage = $(RE_{\text{before}} - RE_{\text{after}}) / (\text{Cost of the risk reduction})$ RE_{before} is the risk exposure, as explained, before risk reduction actions have been taken. RE is the risk exposure after taking the risk reduction action. An RRL above 1.00 indicates that the reduction in risk exposure achieved by a measure is greater than its cost. To take a rather unrealistic example, it might cost £200,000 to replace a hardware configuration used to develop a software application. There is a 1% chance of a fire (because of the location of the installation, say). The risk exposure would be 1% of £200,000 that is £2,000. Installing fire alarms at a cost of £500 would reduce the chance of fire to 0.5% the new risk exposure would be £1,000, a reduction of £1,000 on the previous exposure. The RRL would be $(2000 - 1000) / 500$, that is 2.0, and the action would therefore be deemed worthwhile.

- **creating and maintaining the risk register**

When the project planners have picked out and examined what appear to be the most threatening risks to the project, they need to record their findings in a risk register. The typical content of such a register . After work starts on the project more risks will emerge and be added to the register. At regular intervals, probably as part of the project control life cycle described in Chapter 9, the risk register should be reviewed and amended. Many risks threaten just one or two activities, and when the project staffs have completed this risk can then be 'closed' as no longer relevant. In any case, as noted earlier, the probability and impact of a risk are likely to change during the course of the project.

8.10 EVALUATING RISKS TO THE SCHEDULE

This illustrated the point that a forecast of the time needed to do a job is most realistically presented as a graph of likelihood of a range of figure, with the most

Likely duration as the peak and the chances of the job taking longer or shorter shown as curves sloping down on either side of the peak. Thus we can show that a job might take five days but there is a small chance it might need four or six days, and a smaller chance of three or seven days, and so on. If a task in a project takes longer than planned we might hope that some other task might take less and thus compensate for this delay. In the following sections we will examine PERT, a technique which takes account of the uncertainties in the duration of activities

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 Register Page

within a project. We will also touch upon Monte Carlo simulation, which is a more powerful and flexible tool that tackles the same problem

A drawback to the application of methods like PERT is that in practice there is a tendency for developer to work to the schedule even if a task could be completed more quickly. Even if tasks are completed earlier than planned. project managers are not always quick to exploit the opportunities to start subsequent activities earlier than schedule. Critical chain management will be explored as a way of tackling this problem.

8.11 BOEHM'S TOP 10 RISKS AND COUNTER MEASURES

Boehm has identified the top 10 risks that a typical project suffers from and has recommended a set of countermeasures for each. We briefly review these in the following.

1. **Personnel shortfall:** This risk concerns shortfall of project personnel. The shortfall may show up as either project personnel may lack some specific competence required for the project tasks or personnel leaving the project (called manpower turnover) before

project completion. The countermeasures suggested including staffing with top talent, job matching, team building, and cross-training of personnel.

2. **Unrealistic schedules and budgets:** The suggested countermeasures include the project manager working out the detailed milestones and making cost and schedule estimations based on it. Other countermeasures are incremental development, software reuse, and requirements scrubbing. It may be mentioned that requirements scrubbing involves removing the overly complex and unimportant requirements, in consultation with the customers.
3. **Developing the wrong functions:** The suggested countermeasures include user surveys and user participation, developing prototypes and eliciting user feedback, and early production users' input and getting user feedback on it.
4. **Developing the wrong user interface:** The countermeasures suggested for this risk include prototyping, scenarios and task analysis, and user participation.
5. **Gold-plating:** Gold-plating as discussed in Chapter 1, concerns development of features that the team members consider nice to have and, therefore, decide to develop those even though the customer has not expressed any necessity for those. The countermeasures suggested for this risk include requirements scrubbing, prototyping and cost-benefit analysis.
6. **Continuing stream of requirements changes:** The countermeasures suggested for this risk include incremental development, high change threshold and information hiding.
7. **Shortfalls in externally furnished components:** This concerns the risk that the components developed by third party are not up to the mark. The countermeasures suggested for this risk include benchmarking, inspections, reference checking and compatibility analysis.
8. **Shortfalls in externally performed tasks:** This concerns the risk that the work performed by the contractors may not be up to the mark. The countermeasures suggested for this risk include reference checking, pre-award audits, award-fee contracts, competitive design or prototyping and team building.
9. **Real-time performance shortfalls:** The countermeasures suggested for this risk include simulation. Benchmarking, modeling, prototyping, instrumentation, and tuning.
10. **Straining computer science capabilities:** The countermeasures suggested for this risk include technical analysis, cost-benefit analysis, and prototyping.

It is usually advisable for the project manager to develop a risk mitigation, monitoring, and management (RMMM) plan for a project. An important component of this document is a risk table. Each row of the table contains the name of the risk, its probability, and its impact on the project. For each risk in the risk table, the specific conditions or events that need to be monitored to check whether the risk has occurred is mentioned. The possible ways in which the risk can be avoided (mitigation) is also documented. A contingency plan to contain the effect of the risk is also documented.

8.12 APPLYING THE PERT TECHNIQUE

Using PERT to evaluate the effects of uncertainty

PERT was developed to take account of the uncertainty surrounding estimates of task durations. It was developed in an environment of expensive, high-risk, and state-of-the-art projects - not that dissimilar to many of today's large software projects.

The method is very similar to the CPM technique (indeed many practitioners use the terms PERT and CPM interchangeably) but, instead of using a single estimate for the duration of each task, 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 for this.
- **Pessimistic time:** the worst possible time, allowing for all reasonable eventualities but excluding 'acts of God and warfare' (as they say in most insurance exclusion clauses). We shall call this b.

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

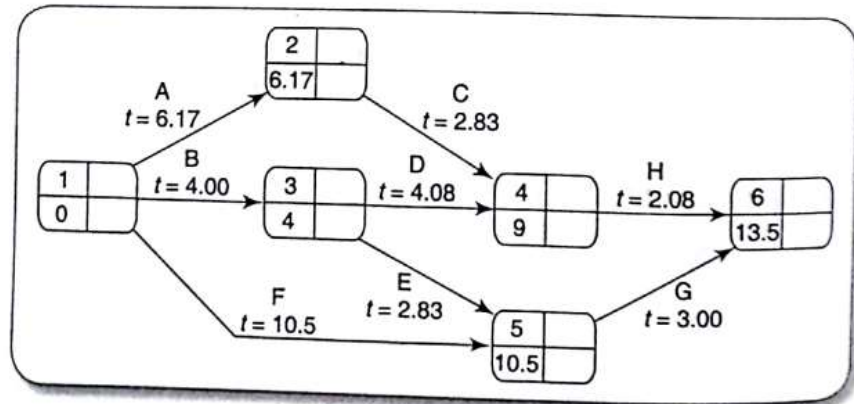
$$t_e = a+4m+b/6$$

Using expected durations

The expected durations are used to carry out a forward pass through a network, using the same method as the CPM technique. In this case, however, the calculated event dates are not the earliest possible dates but the dates by which we expect to achieve those events.

The PERT network illustrated in Figure 7.6 indicates that we expect the project to take 13.5 weeks. In Figure 7.6 we have used an activity-on-arrow network as this form of presentation makes it easier to separate visually the estimated activity data (expected durations and, later, their

standard deviations) from the calculated data (expected completion dates and target completion dates). The method can, of course, be equally well supported by activity-on-node diagrams.



The Pert Network After The forward pass

Activity standard deviation

A quantitative measure of the degree of uncertainty of an activity duration estimate may be obtained by calculating the standard deviation s of an activity time, using the formula

$$S = (b - a) / 6$$

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 of risk for each activity. The activity expected durations and standard deviation for our sample project.

The likelihood of meeting targets

The main advantage of the PERT technique is that it provides a method for estimating the probability of meeting or missing target dates. There might be only single target date – the project completion – but we might wish to set an additional intermediate target.

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

Expected time and standard deviation

Calculating the Z values

The Z value calculated for each node that has a targeted date. It is equivalent to the number of standard deviation between the node expected and target date. It is calculating using this formula.

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

converting z values

Advantages of PERT

We have seen that by requesting multi-valued activity duration estimates and calculating expected dates PERT focuses attention on the uncertainty of forecasting. We can use the technique to calculate the standard deviation for each task and use this to rank them according to their degree of risk. Using this ranking, we can see, for example, that activity F is the one regarding which we have greatest uncertainty, whereas activity C should, in principle, give us relatively little cause for concern.

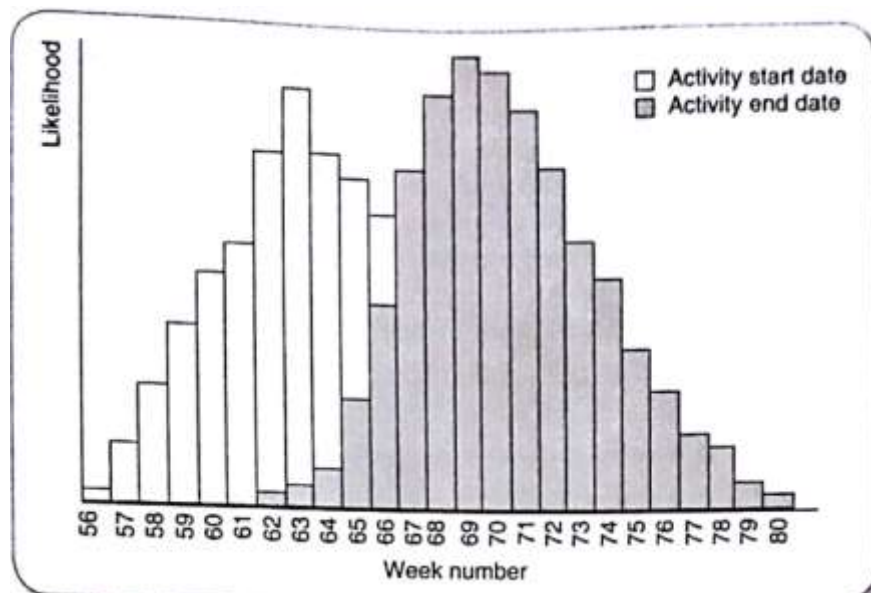
If we use the expected times and standard deviations for forward passes through the network we can, for any event or activity completion, estimate the probability of meeting any set target. By setting target dates along the critical path, we can focus on those activities posing the greatest risk to the project's schedule.

8.13 MONTE CARLO SIMULATION

- As an alternative to the PERT technique, we can use Monte Carlo simulation approach. Monte Carlo simulation is a class of general analysis techniques that are 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. Since this technique is based on repeated computation of random numbers, it becomes easier to use this technique when available as a computer program
- When Monte Carlo simulation is used to analyse the risk of not meeting the project deadline, the project completion time is first modeled as a mathematical expression involving the probability distributions of the completion times of various project activities and their precedence relationships. Activity durations can be specified in a variety of forms, depending upon the information available. If, for example, we have historic data available about the durations of similar activities as shown in the probability chart in Figure 7.4 we might be able to specify durations as pertinent probability distributions. With less information available, we should, at least, be able to provide three time estimates as used by PERT
- Monte Carlo simulation essentially evaluates a range of input values generated from the specified probability distributions of the activity durations. It then calculates the results repeatedly; each time using a

different of random values generated from the given probability functions. Depending upon the number of probalistic

- Parameters and the ranges specified for them, a Monte Carlo simulation could involve thousands or even millions of calculations to complete. After the simulation results are available, these are analyzed summarized and represented graphically, possibly using a histogram as shown in Figure 7.9. The main steps in mind in carrying out Monte Carlo simulation for a project consisting of n activities are as follow
 - **Step 1:** Express the project completion time in terms of the duration of the n activities (x) and their dependences as a precedence graph, $d=f(x_1, x_2, \dots, x_n)$.
 - **Step 2:** Generate a set of random inputs, $X_1, X_2 \dots X_n$ using specified probability distributions.
 - **Step 3:** Evaluate the project completion time expression and store the result in d
 - **Step 4:** Repeat Steps 2 and 3 for the specified number of times.
 - **Step 5:** Analyze the results d in summarize and display using a histogram as the one shown in Figure given below

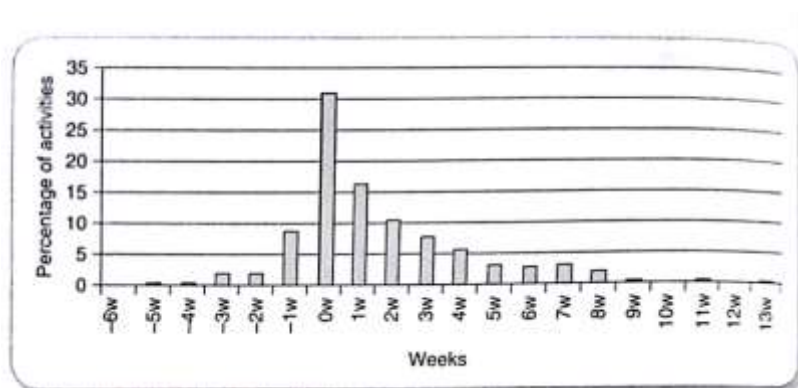


To appreciate the advantage of Monte Carlo simulations over a manual approach, consider the following. In the manual approach, a few combinations of each project duration are chosen (such as best case, worst case, and most likely case), and the results recorded for each selected scenario. In contrast, in Monte Carlo simulation, hundreds or thousands of possible random sampling of probability distribution functions of the activity durations are considered as samples for evaluation of the project completion time expression to produce outcomes. Monte Carlo simulation

is expected to give a more realistic result than manual analysis of a few cases, especially because manual analysis implicitly gives equal weights to all scenarios.

8.14 CRITICAL CHAIN CONCEPTS

- This chapter has stressed the idea that the forecast for the duration of an activity cannot in reality be a single number but must be a range of durations that can be displayed on a graph such as Figure 7.3. However, we would want to pick one value in that range which would be the target.
- The duration chosen as the target might be the one that seems to be the most likely. Imagine someone who cycles to work each day. It may be that on average it takes those about 45 minutes to complete the journey, but on some days, it could be more and on others it could be less. These journey times could be plotted on a graph like the one in Figure 7.3. If the cyclist had a very important meeting at work, it is likely that they would give themselves more time - say an extra 15 minutes than the average 45 minutes to make sure that they arrived in time. In the discussion above on the PERT risk technique the most likely duration was the middle value, and the pessimistic estimate was the equivalent of the $45+15=60$ minute
- Of course, there will be some days when the cyclist will beat the average of 45 minutes. When a project is being executed, the project manager will be forced to focus on the activities where the actual durations exceed the target. Activities which are completed before the target date are likely to be overlooked. These early completions, properly handled, could put some time in hand that might still allow the project to meet its target completion date if the later activities are delayed.



Percentage of activities early or late (after van Genuchten 1991)

Deriving 'most likely activity duration

- The target date generated by critical chain planning is one where it is estimated that there is a 50% chance of success - this approximates to the expected time identified in the PERT risk method. In some

explanations of critical chain project planning, it is suggested that the most likely activity duration can be identified by halving the estimates provided. This assumes that the estimates given to the planner will be 'safe' ones based on a 95% probability of them being achieved. If you look at Figure 7.3, the 95% estimate would be 9 days and half of that (4.5 days) would not be a reasonable target as it would have a probability of only 10% of success. It also assumes that a probability profile has a bell-shaped normal distribution (like the example in Figure 7.3). If you look at the distribution which resulted from van Genuchten's research, see that it is certainly not bell-shaped. Other critical chain experts suggest deducting 33% from the safe estimate to get the target estimate - which seems less unreasonable.

- However, what appear to be arbitrary managerial reductions in the estimates may not be a good way to motivate developers, especially if these staff supplied the estimates in the first place. A better approach would be to ask developers to supply two estimates. One of these would be a 'most likely' estimate and the other would include a safety margin or comfort zone. From now on we are going to assume that this is what has happened. In fact we will use the figures already presented in Table 7.6 in this new role (Table 7.8).

Using latest start dates for activities

Working backwards 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 of the project on to other work. It is also argued - with some justification according to van Genuchten's research above - that most developers would tend to start work on the task at the latest start time anyway. However, it does make every activity "critical". If one is late the whole project is late. That is why the next steps are needed.

Inserting project and feeder buffer

- To cope with activity overruns, a project buffer is inserted at the end of the project before the target completion date. One way of calculating this buffer is as the equivalent of 50% of the sum of lengths of the comfort zones' that have been removed from the critical chain. The critical chain is the longest chain of activities in the project, taking account of both task and resource dependencies. This is different from the critical path a latter only takes account of task dependencies. A resource dependency is where one activity has to wait for a resource (usually a person in software development) which is being used by another activity to become available. If an activity on this critical chain is late, it will push the project completion date further into the project buffer. That the buffer should be 50% of the total comfort zones for critical chain activities is based on the grounds that if the estimate for an activity was calculated as having a 50% chance of being correct, the buffer

would only need to be called upon by the 50% of cases where the estimate was not correct.

- An alternative proposal is to sum the squares of the comfort zones and then take the square root of the total. This is based on the idea that each comfort zone is the equivalent to the standard deviation of the activity. Back and look at the section headed 'Calculating the standard deviation of each project event' in Section 7.12. This method of calculation still produces a figure which is less than simply summing all the comfort zones. This is justified on the grounds that the contingency time needed for a group of activities is less than the sum of the individual contingency allowances as the success of some activities will compensate for the shortfalls in others.
- Buffers are also inserted into the project schedule where a subsidiary chain of activities feeds into the critical chain. These feeding buffers could once again be set at 50% of the length of the 'comfort zone' removed from the subsidiary or feeding chain.

Project execution

When the project is executed, the following principles are followed

No chain of tasks should be started earlier than scheduled, but once it has been started it should be finished as soon as possible - this invokes the relay race principle, where developers should be ready to start their tasks as soon as the previous, dependent, tasks are completed.

- Buffers are divided into three zones: green, amber, and red, each of an even (33%) size:
- Green, where no action is required if the project completion date creeps into this zone.
- Amber, where an action plan is formulated if the project completion dates move into this zone.
- Red, where the action plan above is executed if the project penetrates this zone.
- Critical chain planning concepts have the support of a dedicated group of enthusiasts. However, the full application of the model has attracted controversy on various grounds. Our personal view is that the ideas of
- requiring two estimates: the most likely duration/effort and the safety estimate which includes additional time to deal with problems that could arise with the task, and
- placing the contingency time, based on the 'comfort zone' which is the difference between the most likely and safety estimates, in common buffers rather than associating it with individual activities are sound ones that could usefully be absorbed into software project management practice

8.15 SUMMARY

- In this chapter we have seen how to identify and manage the risks that might affect the success of a project. Risk Management is concerned with assessing and prioritizing risks and drawing up plans for addressing those risks before they become problems.
- This chapter has also described the techniques for estimating the effect of risk on the project's activity network and schedule.
- Many of the risks affecting software project can be reduced by allocating more experienced staff to those activities that are affected.

8.16 EXERCISES

- Suppose you are the project manager of a large software development project. List three common types of risk that your project might suffer. Point out the main steps that you would follow to effectively manage risks in your project.
 - Schedule slippage is a very common form of risk that almost every project manager has to deal with. Suppose you are the project manager of a medium sized project. Explain how you would manage the risk of schedule slippage.
 - Select the appropriate option
- 1) Which one of the following is an important objective of risk exposure analysis?
 - a) Collecting information that can be used for future risk analysis.
 - b) Defining risk avoidance strategies for various risks.
 - c) Estimating the impact of the risk on the project.
 - d) Assessing risk response strategies for the identified risks.
 - 2) Suppose four risks named R1, R2, R3 and R4 have been identified and assigned the probabilities of occurrence of 0.1, 0.2, 0.3 and 0.4 respectively. The likely damages due to the four risks are Rs. 50,000, Rs. 100,000, Rs. 70,000 and Rs. 60,000 respectively. Which has the highest risk exposure?
 - a) R1
 - b) R2
 - c) R3
 - d) R4
 - 3) Which one of the following is not a displacement strategy for mitigating for controlling?
 - a) Mitigation
 - b) simulation
 - c) avoidance
 - d) acceptance

- 4) Which one of the following is the most appropriate sequence of strategies that can be adopted for dealing with positive risks?
- Avoid mitigate transfer and accept
 - Transfer mitigate avoid and exploit
 - Exploit share enhance and accept
 - Mitigate Enhance exploit and accept
- 5) Purchasing insurance cover can be considered to be an example of which one of the following risk handling strategies?
- Mitigation
 - Transfer
 - Acceptance
 - Avoidance
- 6) Which one of the following can be considered to be the most accurate definition of proactive risk management?
- Monitoring risks throughout the project and taking appropriate actions to contain them
 - Identifying, analyzing and prioritizing risks and developing a risk response plan
 - Developing a risk management plan to prevent occurrence of various types of risks
 - Mitigating transferring or accepting risk as and when a risk becomes reality
- 7) Which of the following techniques is not suggested by Boehm to handle the risks of gold plating?
- Requirements scrubbing
 - Cross training of personnel
 - Cost benefit analysis
 - Prototyping Dsadsa
- 8) Assume that you are the project manager software development project. Sunrise Engineering Works the hardware vendor has intimated you that a problem in customs clearance is preventing your network equipment from being delivered on time and may get delayed by several months. For handling this risk, you have arranged for leasing a network equipment from a local company as an interim arrangement. Which one of the following the risk response strategies have you adopted?
- Transference
 - Acceptance
 - Mitigation
 - Avoidance

8.17 REFERENCES

- a) Software Project Management by Bob Hughes, Mike Cotterell, Rajib Mall, Tata McGraw Hill, 6th Edition, 2018.
- b) Project Management and Tools & Technologies – An overview by Shailesh Mehta, Shroff Publishers, 1st Edition, 2017.
- c) Software Project Management by Walker Royce, Pearson Publication, 2005

RESOURCE ALLOCATION

Unit Structure

- 9.0 Objectives
- 9.1 Introduction
- 9.2 Nature of Resources
- 9.3 Identify Resources Requirements
- 9.4 Scheduling Resources
- 9.5 Creating Critical Paths
- 9.6 Counting the Cost
- 9.7 Being Specific
- 9.8 Publishing the Resources Schedule
- 9.9 Cost Schedules
- 9.10 Summary
- 9.11 Exercises

9.0 OBJECTIVES

After going through this unit, you will be able to:

- Identify the resources required for a project
- Make the demand for resources more even throughout the life of a project
- Produce a work plan and resource schedule

9.1 INTRODUCTION

In general, the allocation of resources to activities will lead us to review and modify the ideal activity plan. It may cause us to revise stage or project completion dates. In any event, it is likely to a narrowing of the time spans within which activities may be scheduled.

The result of resource allocation will normally be several schedules, including:

- an activity schedule indicating the planned start and completion dates for each activity.
- a resource schedule showing the dates on which each resource will be required and the level of that requirement.
- a cost schedule showing the planning cumulative expenditure using resources over time.

9.2 NATURE OF RESOURCES

A resource is any item or person required for the execution of the project. This covers many things – from paper clip to key personnel – and it is unlikely that we would wish to itemize every resource required, let alone draw up a schedule for their use. Stationery and other standard office supplies, for example, need not normally be the concern of the project manager – ensuring an adequate is the role of the office manager. The project manager must concentrate on those resources which, without planning, might not be available when required.

Some resources, such as a project manager, will be required for the duration of the duration of the project whereas others, such as a specific software developer, might be required for a single activity. The former, while vital to the success of the project, does not require the same level of scheduling as the latter. The manager may have to request the use of a developer who belongs to a pool of resources controlled at programmer level.

In general, resources will fall into one of seven categories.

- **Labour :** The main items in this category will be members of the development project team such as the project manager, systems analysts and software developers. Equally important will be the quality assurance team and other support staff and any employees of the client organization who might be required to undertake or participate in specific activities.t team such as the project manager, system analysts and software developers.
- **Equipment:** obvious items will include workstations and other computing and office equipment. We must not forget that staff also need basic equipment such as desks and chairs.
- **Materials:** Materials are items that are consumed, rather than equipment that is used. They are of little consequence in most software projects but can be important for some software that is to be widely distributed might, for example, require supplies of disks to be specially obtained.
- **Space:** For projects that are undertaken with existing staff, space is normally readily available. If any additional staff (recruited or contracted) should be needed, then office space will need to be found.
- **Services:** some projects will require procurement of specialist services - development of a wide area distributed system, for example, requires scheduling of telecommunication services.
- **Time:** time is the resource that is being offset against the other primary resources - project timescales can sometimes be reduced by increasing other resources and will almost certainly be extended if they are unexpectedly reduced.
- **Money:** money is a secondary resource – it is used to buy other resources and will be consumed as other resources are used.

9.3 IDENTIFYING RESOURCE REQUIREMENTS

the first step in producing a resources allocation plan is to list the resources that will be required along with the expected level of demand. This will normally be done by considering each activity in turn and identifying the resources required. It is likely, however, that there will also be resources required that are not activity specific but are part of the projects infrastructure or required to support other resources. It is like other resources in that it is available at a cost - in this case interest charges.

9.4 SCHEDULING RESOURCES

Having produced the resource requirements list. The next stage is to map this on to the activity plan to assess the distribution of resources required over the duration of the project. This is best done by representing the activity plan as a bar chart and using this to produce a resource histogram for each resource.

Each activity has been scheduled to start at its earliest start date a sensible initial strategy, since we would, other things being equal, wish to save any float to allow for contingencies. Earliest start scheduling, as is the case with Amanda's project, frequently creates resources histograms that start with a peak and then tail off.

Changing the level of resources on a project over time, particularly personnel, generally adds to the cost of a project. Recruiting staff has costs and, even where staff are transferred internally, time will be needed for familiarization with the new project environment.

The resource histogram poses problems in that it calls for two analyst/designers to be idle for twelve days, one for seven days and one for two days between the specification and design stage. It is unlikely that IOE would have another project requiring their skills for exactly those periods of time. This raises the question whether this idle time should be charged to Amanda's project. This ideal resource histogram will be smooth with, perhaps, an initial build- up and a staged run – down.

Some project planning software tools will carry out resources smoothing automatically, although they are unlikely to consider all the factors that could be used by a project manager. Many project planning software tools will produce resources histograms based on earliest activity start dates.

In practice, resources must be allocated to a project on an activity-by-activity basis and finding the "best" allocation can be time consuming and difficult. As soon as a member of the project team is allocated to an activity, that activity acquires a scheduled start and finish date, and the team member becomes unavailable for other activities for that period. Thus, allocating a resource to one activity limits the flexibility for resource allocation and scheduling of other activities.

It is therefore helpful to prioritize activities so that resources can be allocated to competing activities in some rational order. The priority must almost always be to allocate resources to critical path activities and then to those activities that are most likely to affect others. In that way, lower-priority activities are made to fit around the more critical, already scheduled activities.

Of the various ways of prioritizing activities, two are described below.

- Total float priority Activities are ordered according to their total float, those with the smallest total float having the highest priority. In the simplest application of this method, activities are allocated in ascending order of total float. However, as scheduling proceeds, activities will be delayed (and not available at their earliest start dates) and total floats will be reduced. It is therefore the floats (and hence reorder the list) each time an activity is delayed.
- Ordered list priority with this method, activities that can proceed at the same time are ordered according to a set of simple criteria. An example of this is Burman's priority list, which considers activity duration as well as total float:
 1. Shortest critical activity
 2. Critical activities
 3. Shortest non-critical activity
 4. Non-critical activity with least float
 5. Non-critical activities

Unfortunately, resource smoothing, or even containment of resource demand to available levels, is not always possible within planned timescales - deferring activities to smooth out resource peaks often puts back project completion. Where that is the case, we need to consider ways of increasing the available resource levels or altering working methods

9.5 CREATING CRITICAL PATHS

Scheduling resources can create new critical paths. Delaying the start of an activity because of lack of resources will cause that activity to become critical if this uses up its float. Furthermore, a delay in completing one activity can delay the availability of a resource required for a later activity. If the later one is already critical, then the earlier one might now have been made critical by linking their resources

Amanda's revised schedule, which still calls for four analyst/designers but only for a single day, is illustrated in the solution to Exercise 8.2 (check it in the back of the book if you have not done so already). Notice that in rescheduling some of the activities she has introduced additional critical activities. Delaying the specification of module C has used up all its float - and that of the subsequent activities along that path! Amanda now has two critical paths - the one shown on the precedence network and the new one.

In a large project, resource-linked criticalities can be quite complex - a hint of the potential problems may be appreciated by looking at the next exercise.

9.6 COUNTING THE COST

The discussion so far has concentrated on trying to complete the project by the earliest completion date with the minimum number of staff. We have seen that doing these places constraints on when activities can be carried out and increases the risk of not meeting target dates.

Alternatively, Amanda could have considered using additional staff or lengthening the overall duration of the project. The additional costs of employing extra staff would need to be compared to the costs of delayed delivery and the increased risk of not meeting the scheduled date. The relationship between these factors is discussed later in this chapter.

9.7 BEING SPECIFIC

Allocating resources and smoothing resource histograms is relatively straightforward where all resources of a given type can be considered equivalent. When allocating Labourers to activities in a large building project we need not distinguish among individuals - there are likely to be many Labourers and they may be treated as equals so far as skills and productivity are concerned.

This is seldom the case with software projects. We saw in Chapter 5 that, because of the nature of software development, skill and experience play a significant part in determining the time taken and, potentially, the quality of the final product. Except for extremely large projects, it makes sense to allocate individual members of staff to activities as early as possible, as this can lead us to revise our estimate of their duration.

In allocating individuals to tasks, several factors need to be considered.

- **Availability:** We need to know whether a particular individual will be available when required. Reference to the departmental work plan determines this but the wise project manager will always investigate the risks that might be involved - earlier projects might, for example, overrun and affect the availability of an individual.
- **Criticality** Allocation of more experienced personnel to activities on the critical path often helps in shortening project durations or at least reduces the risk of overrun.
- **Risk** We saw how to undertake activity risk assessment in the previous chapter. Identifying those activities posing the greatest risk, and knowing the factors influencing them, helps to allocate staff. Allocating the most experienced staff to the highest-risk activities is likely to have the greatest effect in reducing overall project uncertainties. More experienced staff are, however, usually more expensive.

- **Training** It will benefit the organization if positive steps are taken to allocate junior staff to appropriate non-critical activities where there will be sufficient slack for them to train and develop skills. There can even be direct benefits to the project since some costs may be allocated to the training budget.
- **Team building:** The selection of individuals must also take account of the final shape of the project team and the way they will work together. This and additional aspects of team management are discussed in Chapter 12.

9.8 PUBLISHING THE RESOURCE SCHEDULE

In allocating and scheduling resources, we have used the activity plan (a precedence network in the case of the examples in this chapter), activity bar charts and resource histograms. Although good as planning tools, they are not the best way of publishing and communicating project schedules. For this we need some form of work plan. Work plans are commonly published as either lists or charts such as that illustrated in Figure 8.7. In this case, Amanda has chosen not to include activity floats (which could be indicated by shaded bars) as she fears that one or two members of the team might work with less urgency if they are aware that their activities are not critical.

Notice that, somewhat unusually, it is assumed that there are no public holidays or other non-productive periods during the 100 days of the project and that none of the team has holidays for the periods they are shown as working.

9.9 COST SCHEDULES

It is now time to produce a detailed cost schedule showing weekly or monthly costs over the life of the project. This will provide a more detailed and accurate estimate of costs and will serve as a plan against which project progress can be monitored.

Calculating cost is straightforward where the organization has standard cost figures for staff and other resources. Where this is not the case, then the project manager will have to calculate the costs.

In general, costs are categorized as follows:

- **Staff costs:** These will include staff salaries as well as the other direct costs of employment such as the employer's contribution to social security funds, pension scheme contributions, holiday pay and sickness benefit. These are commonly charged to projects at hourly rates based on weekly work records completed by staff. Note that contract staff are usually charged by the week or month - even when they are idle.

- **Overheads:** Overheads represent expenditure that an organization incurs, which cannot be directly related to individual projects or jobs, including space rental, interest charges and the costs of service departments (such as human resources). Overhead costs can be recovered by making a fixed charge on development departments (in which case they usually appear as a weekly or monthly charge for a project), or by an additional percentage charge on direct staff employment costs. These additional charges or on-costs can easily equal or exceed the direct employment costs.
- **Usage charges:** In some organizations, projects are charged directly for use of resources such as computer time (rather than their cost being recovered as an overhead). This will normally be on an 'as used' basis.

9.10 SUMMARY

In this chapter we have discussed the problems of allocating resources to project activities and the conversion of an activity plan to a work schedule. We have seen the importance of the following:

- Identifying all the resources needed.
- Arranging activity starts to minimize variations in resource levels over the duration of the project.
- Allocating resources to competing activities in a rational order of priority.
- Taking care in allocating the right staff to critical activities.

9.11 EXERCISES

1. Burman's priority ordering for allocating resources to activities considers the activity duration as well as its total float. Why do you think this is advantageous?
2. If you have access to project planning software, use it to produce an activity plan for Amanda's project and include the staff resource requirements for each activity.

Explore the facilities of your software and answer the following questions.

- Can you set up resource types and ask the application to allocate individuals to tasks?
- Will your software allow you to specify productivity factors for individual members of staff so that the duration of an activity depends upon who is carrying it out?
- Will your software carry out resource smoothing or provide a minimum cost solution?

- Can you replicate Amanda's work schedule (see Figure 8.7) - or produce a better one?
3. On a large project, it is often the responsibility of a team leader to allocate tasks to individuals. Why might it be unsatisfactory to leave such allocations entirely to the discretion of the team leader?
 4. In scheduling her project, Amanda ignored the risks of absence due to staff sickness. What might she have done to estimate the likelihood of this occurring and how might she have taken account of the risk when scheduling the project?
 5. (a) Draw up an activity network and calculate the earliest finish for the following project:

Activity	Duration	Depends on	Resource type
A	3 days		SA
B	1 day	A	SD
C	2 days	A	SD
D	4 days	A	SD
E	3 days	B	SC
F	3 days	C	SC
G	6 days	D	SC
H	3 days	E, F, G	SA

SA = systems analyst; SD = systems designer; SC = software coder

- (b) Produce a table showing the number of specialists of each type needed on each day of the project if every activity was started as soon as possible. How many of each type of resource will need to be recruited for the project as a whole if the earliest finish date is to be preserved?
 - (c) What impact would there be on the project if there were only two systems designers?
 - (d) What impact would there be on the project if there was only one systems designer, but you had three software coders?
 - (e) Assuming that the systems designers were employed for the duration of the project, what would be the percentage utilization of the systems designers in the case of both (c) and (d) above?
6. (a) Draw up an activity network for the activities below, identifying the critical path.

Activity	Duration	Depends on	Resource type
A	2 days		SA
B	10 days	A	SD
C	2 days	A	SD
D	2 days	C	SC
E	3 days	C	SC
F	2 days	C	SC
G	4 days	B, D, E, F	SA

SA = systems analyst; SD = systems designer; SC = software coder

- (b) Draw up a resource table showing the number of each type of resource needed on each day of the project and assuming that there is only one systems designer.
- (c) Identify the best way of revising the plan to remove resource clashes.
7. Consider a software development project with seven tasks T1-17. The estimated duration of these seven tasks in weeks are 3, 2, 3, 5, 2, 4, and 5 respectively. T2 and T4 can start when T1 is complete. T3 can start when T2 is complete. T5, T6, and T7 can start when both T3 and T4 are complete. If developer: A is available from the start of the project and developer B and C become available after three weeks of the start of the project. Schedule the project and show your results in the form of a bar chart and resource histogram.
8. For each of the following questions, exactly one option is correct. Select the appropriate option.
- (ii) Which one of the following is not true about resource histograms?
- A resource histogram is a representation of the distribution of the resources required over the duration of the project.
 - Based on the resource histogram, some activities may be delayed reducing the maximum demand of a resource.
 - A resource histogram is used to estimate activity durations.
 - The initial activity network is refined based on the resource histogram.
- (iii) Which one of the following is false regarding resource scheduling?
- Resource scheduling may lead to changing the duration of some activities on the PERT chart.
 - Resource scheduling may not affect the critical path.
 - Resource scheduling usually shortens the critical path.
 - Resource scheduling can create additional critical paths.

MONITORING AND CONTROL

Unit Structure

- 10.0 Objectives
- 10.1 Introduction
- 10.2 Creating the Framework
 - 10.2.1 Responsibility
 - 10.2.2 Assessing Progress
 - 10.2.3 Setting Checkpoints
 - 10.2.4 Taking Snapshots
- 10.3 Collecting the Data
 - 10.3.1 Partial Completion Reporting
 - 10.3.2 RAG Reporting
- 10.4 Review
 - 10.4.1 Utility of Review
 - 10.4.2 Candidate Work Product for Review
 - 10.4.3 Review Roles
 - 10.4.4 Review Process
 - 10.4.5 Data Collection
- 10.5 Visualizing Progress
 - 10.5.1 Gantt Chart
 - 10.5.2 Slip Chart
 - 10.5.3 Timeline
- 10.6 Cost Monitoring
- 10.7 Earned Value Analysis
 - 10.7.1 Baseline Budget
 - 10.7.2 Monitoring Earned Value
 - 10.7.3 Schedule Variance
 - 10.7.4 Time Variance
 - 10.7.5 Cost Variance
 - 10.7.6 Performance Ratios
- 10.8 Prioritizing Monitoring
- 10.9 Getting the Project Back on Target
 - 10.9.1 Maintaining the Business Case
 - 10.9.2 Exception Planning
- 10.10 Change Control
 - 10.10.1 Change Control Procedures
 - 10.10.2 Changes in Scope of a System
 - 10.10.3 Configuration Librarian's Role
- 10.11 Software Configuration Management (SCM)

- 10.11.1 Context in which Configuration Management is Necessary
- 10.11.2 Few Terminologies
- 10.11.3 Purpose of Software Configuration Management
- 10.11.4 Configuration Management Process
- 10.11.5 Modifications to Work Product Under Configuration Control
- 10.11.6 Release Management
- 10.11.7 Open-Source Configuration Management Tools
- 10.12 Summary
- 10.13 List of References
- 10.14 Unit End Exercise

10.0 OBJECTIVES

After going through this chapter, you will be able to understand

- The need to monitor the progress of the project
- The risk of slippages
- The ways to visualize and assess the planned and actual state of the project
- Countermeasures to revise target back on track in case of drift
- How to control changes to requirements of the project.

10.1 INTRODUCTION

- Once the work schedules have been published and the project has begun, the focus must be on progress.
- This necessitates monitoring what is happening, comparing actual achievement to the schedule, and revising plans and schedules as needed to bring the project as close to completion as possible.

10.2 CREATING THE FRAMEWORK

- Controlling a project and ensuring that goals are met requires regular monitoring.
- There could be a mismatch between the expected and actual outcomes.
- So, replanning may be required to get the project back on track or target date can be revised.
- Projects face four types of shortfalls namely delay in meeting target dates, quality, inadequate functionality and costs going over the target.

- The following flowchart depicts the project control cycle and shows the various aspect of project planning to monitoring and revising the plan in case of drifts.

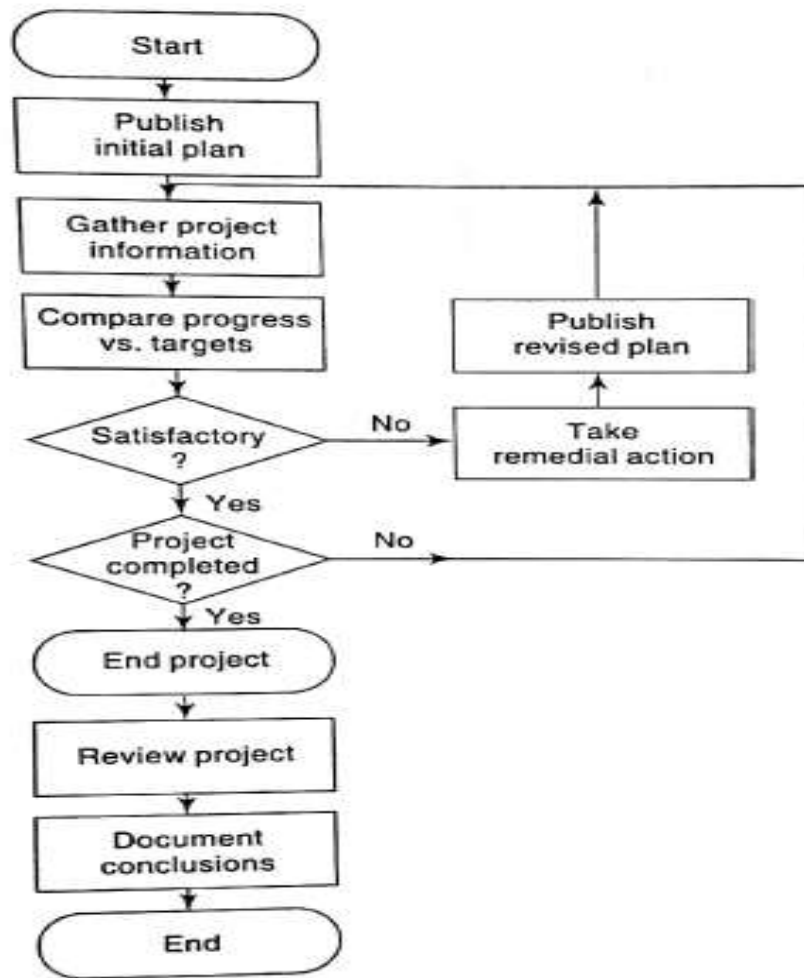


Figure 10.1 – Project Control Cycle

10.2.1 RESPONSIBILITY

- The overall responsibility for ensuring satisfactory progress on a project is often the role of the project steering board.
- Day to day responsibility is on the project manager and he delegates work to team leaders.
- In small projects, employees report directly to project manager but in large projects the team leaders collate reports and send to manager.

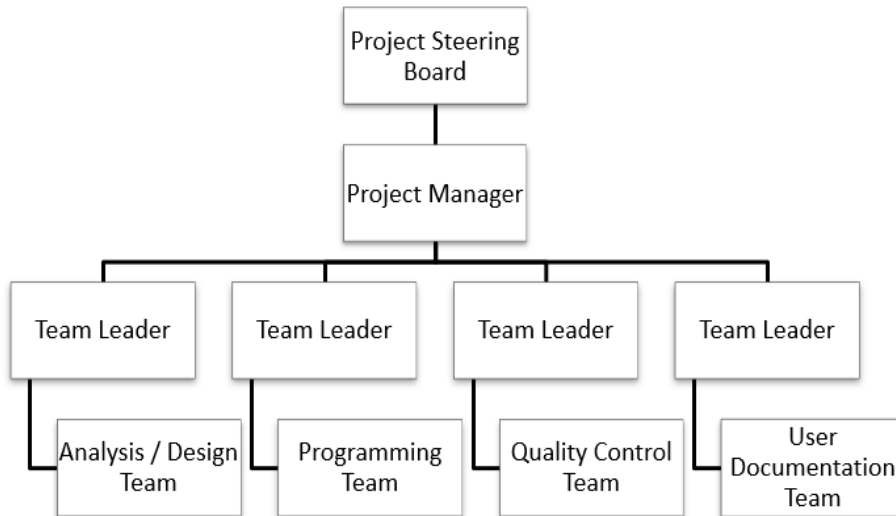


Figure 10.2 – Project Reporting Structure

- Progress reports can be oral or written, formal or informal and regular or adhoc. Sample table shows the various reporting categories

Report Type	Example
Oral Formal Regular	Progress meetings held weekly or monthly
Oral Formal Adhoc	Review meetings held at end of stage
Written Formal Regular	Preparation of Job sheets and progress report
Written Formal Adhoc	Preparation of change or exception reports
Oral Informal Adhoc	Having social interaction

10.2.2 ASSESSING PROGRESS

- Some information used to evaluate mission development may be amassed routinely, even as different information may be precipitated via way of means of particular events.
- This information has to be objective and tangible whether or not or now no longer a specific report has been delivered.

10.2.3 SETTING CHECKPOINTS

- Checkpoints are necessary to demonstrate how well a project is performing.
- They help in identify the important risks, issues, and out of tolerance conditions.

- They perform a global assessment for the whole lifecycle, not just the current situation of an individual perspective or intermediate product.
- So, checkpoints may be regular or tied to specific events.

10.2.4 TAKING SNAPSHOTS

- The frequency of progress reviews will rely on the dimensions and diploma of threat of the challenge.
- Major or project-degree development critiques will normally take region at specific factors all through the existence of a challenge generally called assessment points or control points.
- Weekly collection of information and its assessment is usually preferred.
- End stage assessment is also a common practice adopted by PRINCE2.

10.3 COLLECTING THE DATA

- Managers try to decompose long activities in more controllable tasks than one or two weeks.
- It will still be necessary to collect information on partially completed activities and, in particular, the prevention of the remaining amount of work.
- It can be difficult to make such predictions precisely.
- But in case of series of products, it is easy to estimate the partial completion of activities.

10.3.1 PARTIAL COMPLETION REPORTING

- Numerous associations utilize standard bookkeeping frameworks with week-by-week timesheets to charge staff time to individual positions.
- The staff time booked to a task demonstrates the work did and the charges to the undertaking.
- It does not, in any case, mention to the undertaking manager what has been created or regardless of whether errands are on time.
- So weekly time sheets are adapted by breaking tasks down to activity level and collecting information about work done in addition to time spent.

[Company Name]

[Street Address]
[Address 2]
[City, ST ZIP Code]

Week ending: 4/3/2005

Weekly Time Sheet

Employee: _____
 Manager: _____
 Employee phone: _____
 Employee e-mail: _____

Day		Project Name	Task	Regular Hours	Overtime Hours	Total
Monday	3/28/2005					
Tuesday	3/29/2005					
Wednesday	3/30/2005					
Thursday	3/31/2005					
Friday	4/1/2005					
Saturday	4/2/2005					
Sunday	4/3/2005					
Total hours						

Employee signature Date

Manager signature Date

Figure 10.3 – Weekly Time Sheet

10.3.2 RAG (RED/AMBER/GREEN) REPORTING

- The drawback of the previous scheme is to ask for estimated completion dates.
- This scheme overcomes by asking the likelihood of meeting the planned target date.
- So, in traffic light or RAG method, we recognize the key (first level) components for a piece of work.
- It Breaks these vital components into constituent’s components (second level) and survey every one of the second level components on scale Green – 'on track', Amber – 'not on track but rather recoverable' and Red – 'not on track and recoverable just with trouble'
- It then surveys every one of the second level assessments to show up at first level evaluations.
- Finally review first and second level assessment to produce the final evaluation.

In Progress																	
Project	Prog. Priority	Overall Priority	Cat	Actual Start Date	Planned Delivery Date	Original Delivery Date	Mar-12	Apr-12	May-12	Jun-12	Jul-12	Aug-12	Sep-12	Oct-12	Nov-12	Dec-12	Jan-13
COM001 Office365 for Students	1	1	VI	Jul-12	Jan-13	Jul-12	G	G	A	A	G	A					
COM003 Business Intelligence Needs Analysis - Discovery Phase	2	2	VI	Jul-12	Sep-12	Sep-12					G	G	0				
ITS092 BOXI Upgrade	3	2	E	Oct-11	Jul-13	Jul-12	A	A	A	G	G	G					
COM002 Kofax Upgrade	4	2	E	Aug-12	Jul-12	Jun-12		G	A	A	A	A	0				

Figure 10.4 – RAG Reporting Sheet

- RAG assessment highlights risk of non-achievement.
- It does not attempt to estimate work completed or compute predictable delays.

10.4 REVIEW

- Review of work items is a significant mechanism for checking the advancement of a project and guaranteeing the quality of the work items.
- It is important to kill as many deformities in these work items to understand a result of acceptable quality.
- Testing is a convincing effective defect removal technique, but testing is relevant to just executable code.
- Review is pertinent to all work items even the non-executable work items.
- Review is more cost-effective in removing defects.

10.4.1 UTILITY OF REVIEW

- Apart from review being the most cost-effective defect removal technique, it offers other advantages too.
- A review usually aids in identifying any deviations from standards, as well as issues that may affect the software's maintenance.
- Reviewers make suggestions for how to improve the work items.

- A review meeting, in recognizing defects, frequently provides learning opportunities for not only the creator of a work items, but also the other review meeting participants.
- Participants in the review gain a thorough understanding of the work items under consideration, making it simpler for them to communicate with or use the work items in their work.

10.4.2 CANDIDATE WORK PRODUCT FOR REVIEW

- Apart from final work items and end products, other items are also reviewed such as requirements specification documents, user interface specification and design documents, architectural, high-level and detailed design documents, test plan and the designed test cases and lastly project management plan and configuration management plan

10.4.3 REVIEW ROLES

- Every review meeting requires few members (Moderator, Recorder and Reviewer) to play key roles in the review process.
- The **moderator** plans and organises meetings, distributes review materials, and leads and moderates review sessions.
- The **recorder** logs the defects discovered as well as the time and effort expended.
- The **reviewers** (review team members) go over the work product and make specific suggestions to the author about the existing flaws as well as ways to improve it.

10.4.4 REVIEW PROCESS

- There are four important activities in the review process namely planning, preparation, review meeting and rework and follow-up.
- **Planning** - The input to the planning phase is the work products which is ready for review. The project manager appoints moderator and with his consultation nominates the review team. The moderator is responsible to schedule all review meetings.
- **Preparation** - To initiate the review process, a brief preparation meeting is scheduled and copies of work products are handed over to team. Author presents overview of work. Moderator highlights objectives of review and individual team prepare review logs and record their observation.

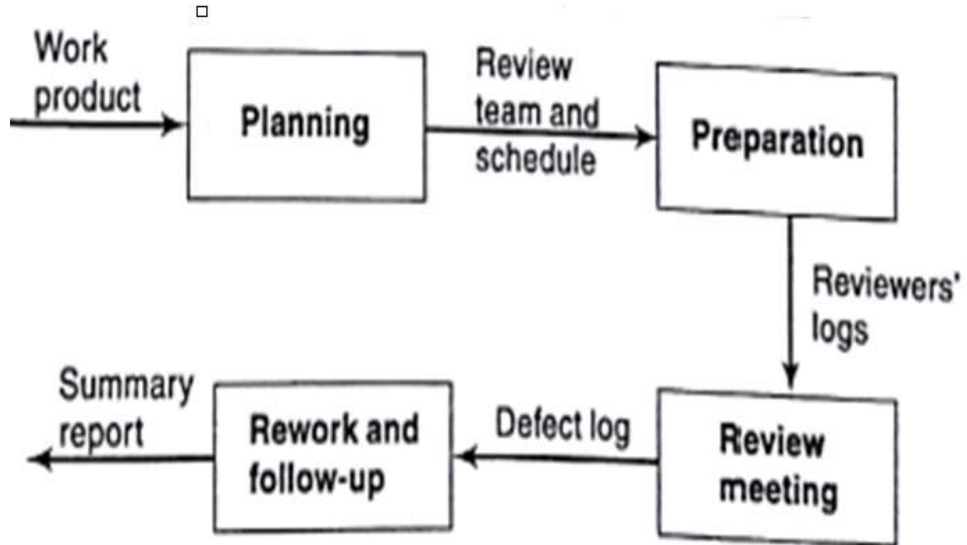


Figure 10.5 – Review Process Model

- **Review Meeting** – The reviewer presents their observations and author along with other reviewers respond to it and moderator ensures discussions are focussed and productive. At the end the recorder scribes the defects and prepares defect log with review statistics.
- **Rework and follow-up** – The author raise all issues by team and brings in relevant modification. A rejoinder is prepared against the defect log. The corrected work along with rejoinder is distributed to team for confirmation. At the end a summary report of review is prepared.

10.4.5 DATA COLLECTION

- Because a review meeting is an entirely human endeavour, the data representing the meeting's results may be lost if not properly recorded.
- Through addition to recording all defects, data on the time spent by reviewers on the review activity must be recorded.
- Review data is captured in three reports: Review Preparation Log, Review Log, and Review Summary Report.
- The **Review Preparation Log** is prepared by reviewer which contains data about defects, location, criticality and time spent in doing review.
- The **Review Log** is prepared by author and contains those defects that are agreed upon.
- The **Review Summary Report** summarizes the review data with information such as total defects and amount of time spent on each review.

- After gathering data on project progress, a manager must find a way to effectively present that data.
- Some of these methods provide a static image, a single snapshot, while others attempt to demonstrate how the project has progressed and changed over time.

10.5.1 GANTT CHART

- The Gantt chart is the most basic and oldest technique for tracking project progress.
- This is primarily an activity bar chart that shows scheduled activity dates and durations, often supplemented with activity floats.
- The chart records reported progress, and today's cursor provides an instant visual indication of which activities are ahead or behind schedule.

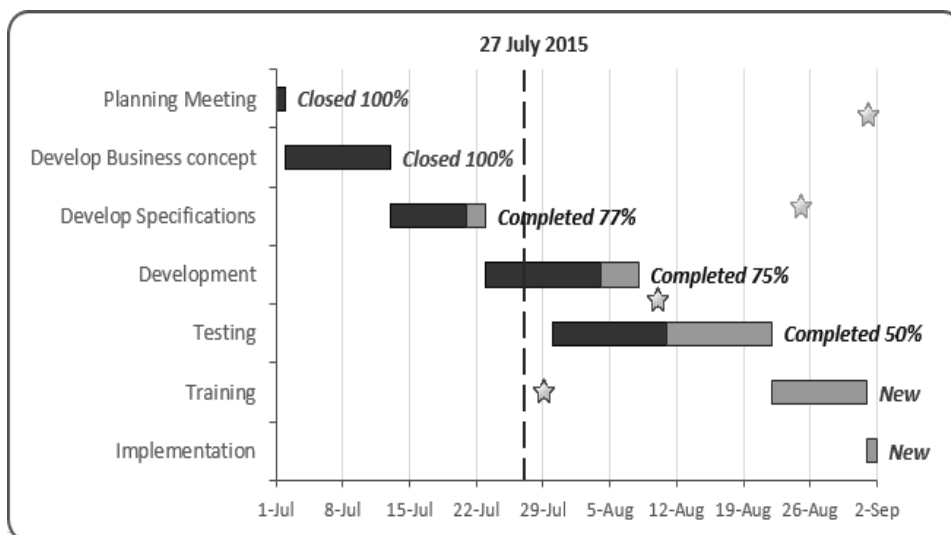


Figure 10.6 - Sample Gantt Chart

10.5.2 SLIP CHART

- It is a very equivalent alternative that some project managers prefer.
- This is because they believe it provides a more visible visual indication of activities that are not progressing according to plan – the further the skip line bends, the significantly larger the deviation from the plan.
- Jagged lines indicate need for rescheduling.

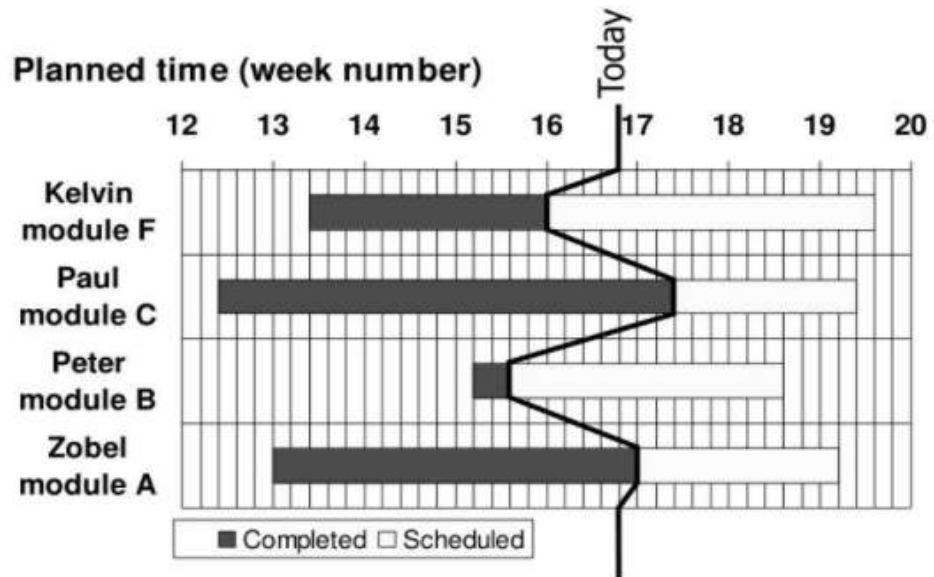


Figure 10.7 - Sample Slip Chart

10.5.3 TIMELINE

- One drawback of the charts discussed thus far is that they do not actually demonstrate the slippage of the project completion date throughout the project's life cycle.
- Analysing and comprehending trends in the project thus far allows us to forecast the project's future progress.
- The timeline chart is a method of recording and displaying how targets have changed over the course of a project.
- It is useful during project execution stage and post implementation stage as reasons for changes and delays can be known which can be avoided in future.

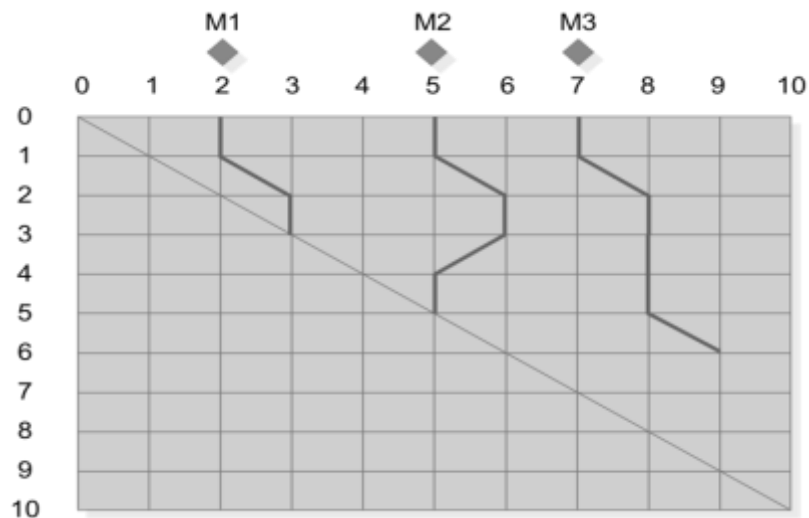


Figure 10.8 – Sample Timeline Chart

- Expenditure control is a vital component of project control not only because it provides an indication of the effort that has taken into a project, but also because it delivers an indication of the effort that has gone into a project.
- A project may be completed on time but over budget due to the addition of additional resources.
- A project may be delayed because the staff that was initially committed has not been deployed and so project will be late but under budget in this case.
- As a result, both accomplishments and costs must be tracked.
- A cumulative expenditure chart is a straightforward way to compare actual and planned expenditures.
- When we add projected future costs calculated by adding the estimated costs of unfinished work to the costs already incurred, the costs chart becomes more useful.
- A computer-based planning tool is used, and cost schedule revisions are generally provided automatically after actual expenditure has been recorded.

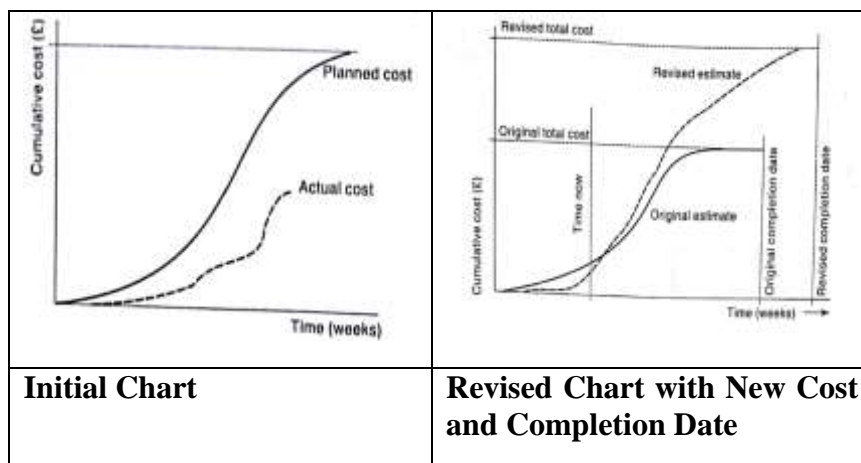


Figure 10.9 – Cumulative Expenditure Chart

10.7 EARNED VALUE ANALYSIS

- Earned value analysis has attracted increasing attention in recent years and can be thought of as a refinement of cost monitoring.
- It is rooted on assigning a 'value' to each task or work package based on the initial expenditure forecasts.

- Planned value (PV) or Budgeted cost of work scheduled (BCWS) is the initial estimate of the effort/cost to complete a task (compare to the concept of a 'price').
- A task that has not yet begun is assigned an earned value of zero, and when completed, it is credited with the task's original planned value.
- Earned value (EV) or Budgeted cost of work performed (BCWP) is the sum of the PVs for the work completed at this time.
- Consistent methods of assigning earned value in software projects include
- **0/100 technique** – It means initial value is zero and on completion given 100% of budgeted value.
- **50/50 technique** – It means initial value of 50% is started and on completion given 100% of budgeted value.
- **75/25 technique** - It means initial value of 75% is started and on completion given remaining 25% of budgeted value.
- **Milestone technique** – It is based on fulfillment of milestones based on initial plan.
- **Percentage Complete technique** – It is based on some objectively measuring criteria.
- The 0/100 is preferred choice

10.7.1 BASELINE BUDGET

- It is the first stage in setting up earned value analysis.
- It is based on the project plan and depicts the projected increase in earned value over time.
- Earned value can be expressed in monetary terms, person-hours, or work days.

10.7.2 MONITORING EARNED VALUE

- The following task is to track earned value as the project progresses after having created the baseline budget.
- This is accomplished by tracking the completion of tasks, the start of activities, and the achievement of milestones.
- Actual cost (AC) or actual cost of work performed (ACWP) can be used to collect the actual cost of each task.
- The performance statistics can be depicted in earned value chart which can also be shown directly.

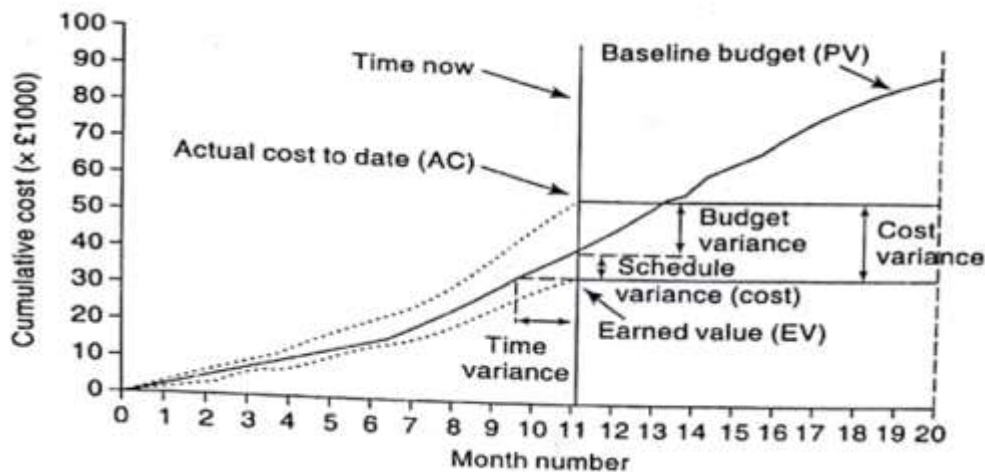


Figure 10.10 – Earned Value Tracking Chart

10.7.3 SCHEDULE VARIANCE

- Schedule variance is expressed in cost terms as $EV - PV$ and indicates the extent to which the value of completed work differs from that anticipated.
- A negative SV indicates that the project is running behind schedule.

10.7.4 TIME VARIANCE

- It denotes the time difference between when the specified EV should have been reached and when it was actually reached.

10.7.5 COST VARIANCE

- The difference between the earned value or budgeted cost and the actual cost of completed work is calculated as $EV - AC$.
- A negative CV indicates that the project is over budget.

10.7.6 PERFORMANCE RATIOS

- Two commonly traced ratios are considered as ‘value-for-money’ namely Cost Performance Index ($CPI = EV / AC$) and Schedule Performance Index ($SPI = EV / PV$).
- A value greater than one implies that work is being accomplished more efficiently than planned, whereas a value less than one implies that work is costing more and moving more slowly than anticipated.
- CPI can be used to generate a revised project cost estimate.
- $BAC / CPI = EAC$ where EAC stands for estimated on completion; BAC stands for budgeted on completion.
- Given the current rate of progress, SPI can be used to project the project's possible duration.

- $SAC/SPI = TEAC$ where TEAC stands for time estimate at completion; SAC stands for schedule at completion.

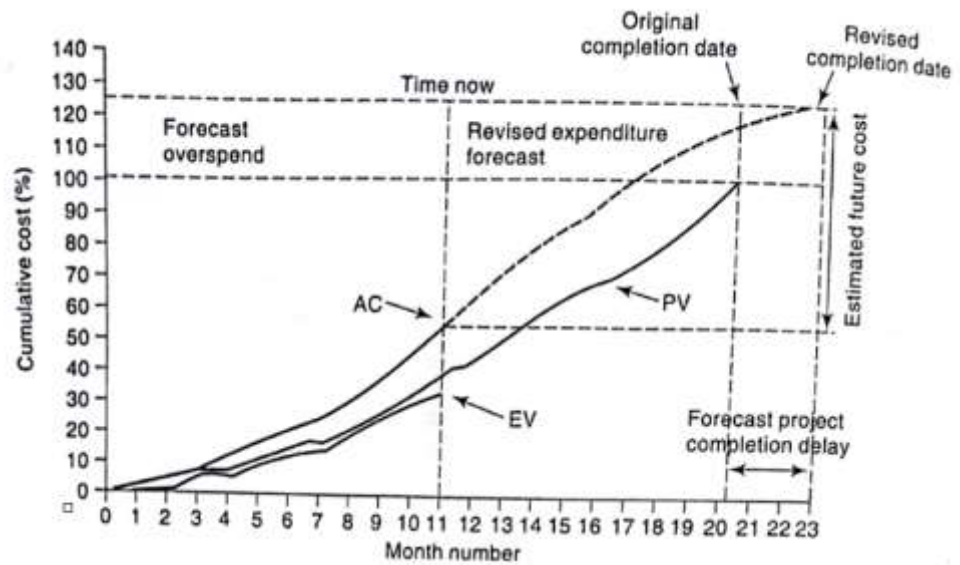


Figure 10.11 - Earned value chart with revised forecast

10.8 PRIORITIZING MONITORING

- In terms of the level of monitoring used, all aspects of the project will be treated equally.
- We must not forget, however, that monitoring takes time and resources that could be better used elsewhere.
- The list that follows prioritizes the deciding level used for monitoring.
- **Critical path activities** – They can cause delay in project completion date and so requires high priority close monitoring.
- **Activities with no free float** – They may cause delay in subsequent activity which might not affect project completion date but availability of resources could be a hitch which could be an area of concern.
- **Activities with less than a specified float** – They may be taken care by regular monitoring.
- **High risk activities** – They must be identified at initial stages as they may cause overrun or overcost as they may have high variance.
- **Activities using critical resources** - They are highly expensive as they consume staff resources if not controlled on time.

- Almost every project will face delays and unexpected events at some point.
- The project manager's job is to recognise when this happens and to try to mitigate the problem's effects as quickly as possible.
- The project manager makes every effort to keep the project's scheduled end date intact.
- When developing plans to get a project back on track, there are two main approaches to consider.
- **Reducing the critical path** – This can be handled by adding resources especially staff, increase use of current resources, reallocate staff to critical activities, reduce scope and finally reduce quality. This way we can control the timescales and cost of critical activities. Shortening the critical path may cause some other paths to become critical which should be taken care off.
- **Changing the requirements for activity precedence** - Consider the constraints that force some activities to be postponed while others are completed. Divide an activity into two parts: one that can begin immediately and one that must wait. It is obviously critical to be aware that quality may be compromised. It is also critical to evaluate the extent to which changes in work practises increase risk.

10.9.1 MAINTAINING THE BUSINESS CASE

- The project sponsor's main concern when making decisions about project management is whether the project's business case has been preserved.
- If costs rise, the value of the benefits at the end of the project falls.
- If completion date is delayed and functionalities shortened, in this case also the expected benefits would be reduced than what was expected.
- The project could then be cancelled in the following cases.

10.9.2 EXCEPTION PLANNING

- The project manager is usually allowed to change the specifics of a plan as long as the agreed-upon project outcomes are delivered on time and within budget.
- A few changes to the plan may have a repercussion on the project's delivery date, scope, or costs.
- These, in turn, may have an impact on the business case.

- The project manager would have to obtain the approval of the project's business sponsors.
- One challenge is to develop an exception report that explains why the deviations from the existing plan occurred.
- After reviewing the report and approving one of the options, the Project Board may charge the project manager with developing a more detailed exception plan.

10.10 CHANGE CONTROL

- When developing a document, such as user requirements, many different forms of the document may be created as it goes through development and review cycles.
- At this juncture, any change management process would be very informal and adaptable.
- It is expected that the final version will be created at some point.
- This is the baseline, and it is effectively frozen.
- Baselined products serve as the foundation for future product development.
- As a result, any changes to the baselined document may have an impact on other parts of the project.
- So, changes to baselined documents needs to be rigorously controlled.

10.10.1 CHANGE CONTROL PROCEDURES

- The steps involved are
- Step 1 - A change may be perceived as necessary by one or more users.
- Step 2 - User organization considers that the change is valid and noteworthy, and it is forwarded to development management.
- Step 3 - A developer is tasked with determining the feasibility and cost of implementing the change.
- Step 4 - Development management reports the cost of the change to user management, and user management decides whether to proceed.
- Step 5 - A smaller group (Change Control Board) will be tasked with finalising changes up to a certain amount of expenses.

- Step 6 - One or maybe more developers are authorised to duplicate components that will be altered.
- Step 7 - The Copies have been altered.
- Step 8 - Following initial testing, a test version may be made available to users for acceptance testing.
- Step 9 - When all users are satisfied, the operational release is authorised, and master copies of configuration items are replaced.

10.10.2 CHANGES IN SCOPE OF A SYSTEM

- This is referred to as software creep.
- The size of the system progressively increasing is a common phenomenon in IS development projects.
- Changes to requirements requested by users are one cause of this.
- As a result, the project's scope must be carefully constantly monitored.
- At key milestones, one method is to re-estimate the system size in terms of SLOC or FP.

10.10.3 CONFIGURATION LIBRARIAN'S ROLE

- The configuration librarian alternatively called project librarian or configuration manager is responsible for control of changes and documentation.
- His duties would be
- Recognizing items subject to change control.
- The creation and upkeep of a centralised repository for master copies of software products and project documentation.
- The establishment and operation of a formal set of procedures for dealing with changes.
- The keeping of records regarding who has access to which library items and the status of every library item.

10.11 SOFTWARE CONFIGURATION MANAGEMENT (SCM)

- When we consider changes occurring on all work products and when there are numerous variants of the product, the manual change management process becomes overburdened.
- In this case, a systematic software configuration management process with an effective methodology must be implemented.

- SCM is involved with tracking and controlling software changes.
- The various work products associated with software change on a regular basis in any development and maintenance environment.
- Several issues can arise if an appropriate configuration management system is not deployed.

10.11.1 CONTEXT IN WHICH CONFIGURATION MANAGEMENT IS NECESSARY

- Work products are altered as development actions are performed out during the development phase.
- The work products change during the maintenance phase due to different types of enhancements and adjustments, including bug fixes.
- As a result, the state of work products is constantly changing and is referred to as a software product's configuration.
- SCM is concerned with proficiently tracking and controlling a software product's configuration throughout its entire life cycle.
- A configuration management tool must be deployed for effective configuration management.

10.11.2 FEW TERMINOLOGIES

- **Configuration** – The state of various work products under configuration control.
- **Version** – The configuration of a software product changes as development and maintenance activities are performed on it. It is frequently necessary to refer to the configuration that existed at a specific point in time. A version is thus a configuration that existed at a specific point in time.
- **Revision** – It is a numbering scheme used to identify the current state of a configuration item.
- **Baseline** – A software configuration that has been formally reviewed and agreed upon and serves as the foundation for future development.
- **Variant** – These are versions that are meant to coexist with one another.

10.11.3 PURPOSE OF SOFTWARE CONFIGURATION MANAGEMENT

- Proper configuration of work items is essential and if not carried out properly then it can lead to several problems such as
- **Problems associated with concurrent access** - The ability to control access to various deliverable objects is possibly the most important reason for configuration management. Several issues can

arise if strict discipline is not enforced regarding the updating and storage of various work products.

- **Undoing Changes** - It is simple to undo a portion of a revision or even roll back development to a previous version. It is extremely difficult to reverse a change unless a proper configuration management system exists.
- **System accounting** - System accounting refers to the process of recording who made a specific change to a configuration item, what change was made, and when the change was made. Acknowledging why changes were made and whether some changes are redundant, as well as comparing the performance of different versions, will benefit from knowing the who, what, and when of changes.
- **Handling variants** - It is frequently necessary to create variants. Keeping track of all variants, their versions, and revisions is a difficult task in this situation without a configuration management system.
- **Accurate determination of project status** - Typically, a project manager will perform configuration management using a configuration management tool. Furthermore, a configuration management tool aids in the tracking of various deliverable objects, allowing the project manager to quickly and unambiguously determine the current state of the project. The configuration management tool allows the developer to make controlled changes to the various components.
- **Preventing unauthorized access to the work products** - Configuration management aids in the implementation of a controlled change process. As a result, it is possible to prevent unauthorized changes to work products.

10.11.4 CONFIGURATION MANAGEMENT PROCESS

- It is accomplished through two primary activities: Configuration Identification and Configuration Control.
- Typically, project managers divide the work products associated with a software development process into three categories: controlled, pre-controlled, and uncontrolled.
- Controlled work products are those that have been configured. To change these, team members must follow some formal procedures.
- Pre-controlled work products are not currently under configuration control but will be in the future.
- Work products that are not controlled will not be subject to configuration control.

- Work products that can be controlled include both controlled and pre-controlled work products.
- Typical controllable work products include requirements specification document, design documents, tools used to build the system, source code for each module, test cases and problem reports
- Configuration control is a component of a configuration management system that has the most significant impact on developers' daily operations.
- Configuration control restricts unauthorized changes to controlled objects and allows only authorized changes.
- Some members may be granted permission by the project manager to change or access specific work products.
- To modify a controlled work product, such as a code module, a developer can obtain a private copy of the module via a reserve operation.
- Configuration management tools limit the number of modules that can be reserved by a team member at any given time.

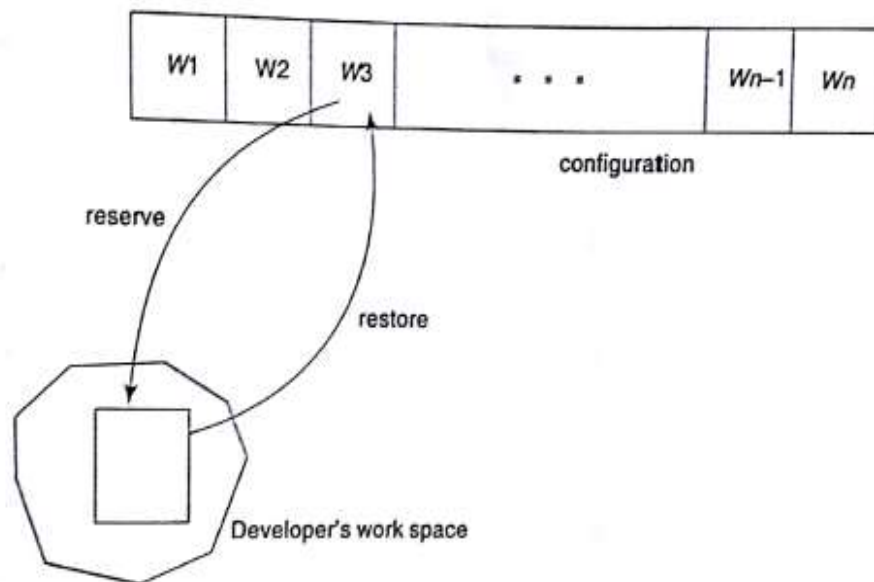


Figure 10.12 – Work Product Modification in SCM

- Once a work product is reserved, no one else will be able to reserve it until the reserved module is restored.
- Thus, by preventing multiple developers from reserving a module at the same time, the problems associated with concurrent access are addressed.

10.11.5 MODIFICATIONS TO WORK PRODUCT UNDER CONFIGURATION CONTROL

- When a developer needs to make a change to a work product, they first submit a reserve request.
- A team member's reserve request is honored only if the project manager has granted that member appropriate authorization for the specific work product.
- Following the successful execution of the reserve command, a private copy of the work product is created in their local directory.
- Then, on their private copy, they can make all necessary changes to the work product.
- Once they have completed all necessary changes, the changes must be restored in the configuration management repository.
- However, restoring the modified work product to the system configuration necessitates the approval of a Change Control Board (CCB).
- The CCB is usually made up of members of the development team.
- The CCB reviews the changes made to the controlled work product and certifies certain aspects of the change that needs to be carried out such as
 - Change is well-intentioned.
 - The effects of the change have been considered and documented by the developer.
 - Changes interact well with other developers' changes.
 - Appropriate individuals have validated the change.

10.11.6 RELEASE MANAGEMENT

- It is preferable for a software development project to implement an appropriate release management process.
- It systematizes the work done by developers to include a new release of software, as well as the work done by users to obtain and use a new release smoothly and effortlessly.
- The release process should require little action on the part of the developer to post a new release of software and little effort on the part of the users to download and install it.

10.11.7 OPEN-SOURCE CONFIGURATION MANAGEMENT TOOLS

- SCCS and RCS are popular configuration management tools found on nearly all UNIX systems.
- It can be used to control and manage multiple versions of text files.
- They do not work with binary files.
- They provide an efficient method of storing versions that use the least amount of disc space.
- Change control features include the ability to limit the number of people who can create new versions and the ability to check components in and out.

10.12 SUMMARY

- Planning is useless unless the plan's execution is monitored.
- Longer activities should be subdivided to make them more manageable.
- The delivery of project products should be used to gauge progress.
- In order to effectively communicate information, progress must be visually appealing.
- Costs, as well as elapsed time, must be tracked.
- Delayed projects are frequently brought back on track by streamlining critical path activity times or comforting some precedence constraints.

10.13 LIST OF REFERENCES

- “Software Project Management”, Sixth Edition, Bob Hughes, Mike Cotterell, Rajib Mall, Mc Graw Hill Education.
- <https://www.guru99.com/software-configuration-management-tutorial.html>
- <https://www.gristprojectmanagement.us/software/cost-monitoring.html>
- <https://www.jigsawacademy.com/blogs/data-science/earned-value-analysis>

10.14 UNIT END EXERCISES

1. The scenario is for removing bugs from the code. Which technique would be cost effective – Review or testing? State proper reasons.
2. A work with a PV of £40,000 should have been completed by now. Some of that work is not done so EV is only £35,000. £55,000 had actually been spent to get that EV. Calculate SV and CV. Also calculate CPI and SPI. The budget at completion was £100,000. The planned total duration for the project was 23 months. Calculate EAC & TEAC.
3. Suppose a project is to be completed in one year at the cost of £100,000. After 3 months, you realize that the project is 30% complete at the cost of £40,000. Assess the performance of the project.
4. Identify other reasons why there is tendency for software creep.
5. What are the pros and cons of putting all the work products in a project under configuration control?

MANAGING CONTRACTS

Unit Structure

- 11.0 Objectives
- 11.1 Introduction
- 11.2 Types of Contracts
 - 11.2.1 Fixed price contracts
 - 11.2.2 Time and Materials contracts
 - 11.2.3 Fixed price per unit delivered contracts
 - 11.2.4 Open tendering process
 - 11.2.5 Restricted tendering process
 - 11.2.6 Negotiated tendering process
- 11.3 Stages in Contract Placement
 - 11.3.1 Requirement Analysis
 - 11.3.2 Evaluation plan
 - 11.3.3 Invitation to tender (ITT)
 - 11.3.4 Evaluation of proposals
- 11.4 Typical terms of a Contract
- 11.5 Contract Management
- 11.6 Acceptance
- 11.7 Summary
- 11.8 List of References
- 11.9 Unit End Exercise

11.0 OBJECTIVES

After going through this chapter, you will be able to understand

- Types of contracts
- Stages in contract placement
- How to evaluate the aspects of a contract?
- Major terminologies in a contract
- How to administer a contract from signing to final acceptance till project end

11.1 INTRODUCTION

- It is sometimes more cost-effective to hire outside software developers for new development while a smaller group of in-house software developers maintain and support existing systems.
- When money is plentiful but other less flexible types of resources are scarce, purchasing goods and services rather than "doing it yourself" is appealing.
- As a result, an acquisition project requires just as much thought and planning as an internal development project.
- It is important that customer community find time to clarify the exact needs in the initial phase itself and then ensure that goods and services delivered be satisfactory.
- The negotiating power of customer would be powerful if business is going to be valuable.
- Probable suppliers will carefully assess the cost and time spent responding to client request as the final contract cannot be guaranteed.

11.2 TYPES OF CONTRACTS

- External resources may be required in the form of services.
- A contract for a completed software package, on the other hand, could be placed as a
 - **Bespoke system** – It is created specifically for one customer
 - **Off-the-shelf** – It is purchased 'as is'. It is referred as shrink-wrapped software.
 - **Customized off-the-shelf (COTS)** – It is a core system customised to meet the needs of a specific customer.
- When purchasing equipment, a contract for the supply of goods is typically used.
- When it comes to software, this could be considered a service or the granting of a licence to use the software, which remains the supplier's property.
- These distinctions have legal consequences.
- Another classification of contracts based on the calculation of payment to suppliers is fixed price contracts, time and materials contracts and fixed price per unit delivered contracts.

- Another approach of classification is based on contractor selection namely open, restricted, and negotiated tendering process.
- We shall discuss the above two classifications in detail.

11.2.1 FIXED PRICE CONTRACTS

- When the contract is signed, the price is set.
- The customer understands that if the contract terms are not changed, this is the price they will pay upon completion.
- For this to work, the customer requirements must be established from the start.
- Once development has begun, the customer cannot modify their prerequisites without renegotiating the contract price.
- The advantages are known customer expenditure and supplier motivation.
- The disadvantages of this techniques are higher prices to allow for contingency, difficulties in modifying requirements, upward pressure on the cost of changes and threat to system quality.

11.2.2 TIME AND MATERIALS CONTRACTS

- The customer is charged a fixed rate per unit of effort under this type of contract.
- The supplier would provide an initial cost estimate depending on the current insight of the customer's requirements, but this does not constitute the grounds for the final payment.
- Typically, the supplier bills the customer on a regular basis for work completed.
- The advantages are ease of changing requirements and lack of price pressure.
- The disadvantages are customer liability and lack of incentive for supplier.
- Customers dislike this approach because it appears to give the supplier a blank check.
- The hiring of contract development staff, on the other hand, may involve this type of contract.

11.2.3 FIXED PRICE PER UNIT DELIVERED CONTRACTS

- It is frequently used in conjunction with function points (FP).
- At the start of the project, the size of the system to be delivered is calculated or estimated.
- The size could be estimated in terms of lines of code, but FPs can be derived more easily from requirements documents.
- There is also a price per unit mentioned.
- The unit price is then calculated by multiplying the number of units to arrive at the final price.
- Consider the following table of schedule of charges per function point

Table 11.1 - Schedule of charges per FP			
Function Point Count	Function Design Cost per FP	Implementation Cost per FP	Total Cost per FP
Up to 2000	\$242	\$725	\$967
2001 – 2500	\$255	\$764	\$1019
2501 – 3000	\$265	\$793	\$1058
3001 – 3500	\$274	\$820	\$1094
3501 – 4000	\$284	\$850	\$1134

- For example, a system to be implemented contains 2,600 FPs.
- The overall charge is calculated as for first 2000 FPs = $\$967 \times 2000 = \1934000 , for the next 500 FPs = $\$1019 \times 500 = \509500 and for the last 100 FPs = $\$1058 \times 100 = \105800 .
- So, charge for all 2600 FPs is sum of these which amounts to \$2549300.
- In the above scenario, the development factor i.e., FP count was fixed.
- But there can be cases where we need to negotiate a series of contracts, each covering a different stage of system development.
- For example, a software supplier might first carry out system design at 1000 FPs. Later the design was then implemented and the actual software delivered has 1000 FPs. The scope grew and new requirements amounted to 100 FPs.
- In this case the overall charge is calculated as for designed system = $\$242 \times 1000 = \242000 , the for implemented system = $\$725 \times 1000 = \725000 and then for new requirements = $\$967 \times 100 = \96700

- So, the total charge for all operation amounts to \$1063700.
- The advantages of this scheme are customer understanding, comparability, emerging functionality, supplier efficiency and life-cycle range.
- The disadvantages are difficulties with software size measurement and changing requirements.
- To address the last issue, one suggestion has been to vary the charge based on the point at which they are requested.

Table 11.2 – Additional Charges for changed functionality

	Pre acceptance testing handover	Post acceptance testing handover
Additional FPs	100%	100%
Changed FPs	130%	150%
Deleted FPs	25%	50%

- For example, a contract stipulates that a computer application is to be designed, constructed, and delivered at a cost of \$600 per FP. After acceptance testing, the customer asks for changes to some of the functions in the system amounting to 500 FPs and some new functions which amount to 200 additional FPs.
- In this case the additional charges are calculated as for changed FPs= \$600 x 500 x 150 / 100 = \$450000 and for the additional FPs=\$600 x 200 x 100 / 100 = \$120000.
- So, the total charge would be \$570000.
- There are additional payment options and permutations.
- The implementation of a specification could be done at a fixed price, with any additions or changes to the requirements charged on a per-FP basis.

11.2.4 OPEN TENDERING PROCESS

- In response to the invitation to tender, any supplier may submit a bid.
- Tenders must all be evaluated in the same manner.
- Local/international law (including EU and WTO, World Trade Organization, requirements) may compel government bodies to do so.
- When client is a public body, then open tendering is a compulsory choice.

11.2.5 RESTRICTED TENDERING PROCESS

- Only bids from suppliers who have been specifically invited by the customer are accepted in this case.
- It has the potential to reduce the number of suppliers considered at any stage.
- This is the best approach to implement.

11.2.6 NEGOTIATED TENDERING PROCESS

- Nevertheless, there may be some valid reasons why the restricted tendering process is not appropriate in certain situations.
- In these cases, a single supplier approach may be justified.
- We negotiate with a single supplier, for example, for extensions to previously supplied software.
- Approaching a single supplier, on the other hand, may expose the customer to accusations of favoritism and should be done only with a sufficient explanation.

11.3 STAGES IN CONTRACT PLACEMENT

- Contract placement requires following four stages namely requirement analysis, evaluation plan, invitation to tender and finally evaluation of proposals.

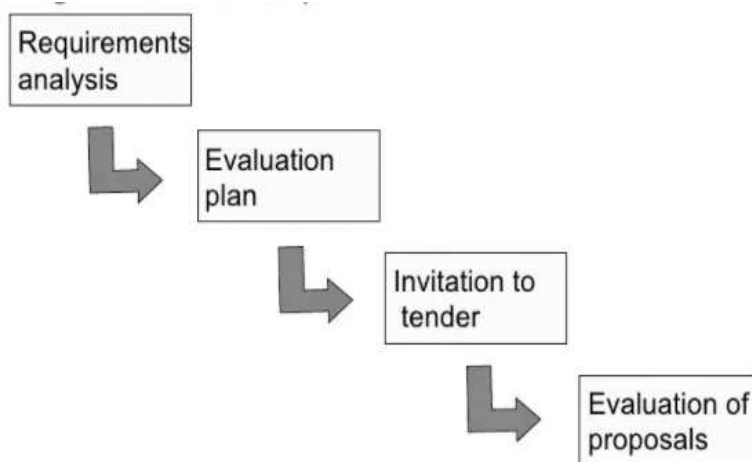
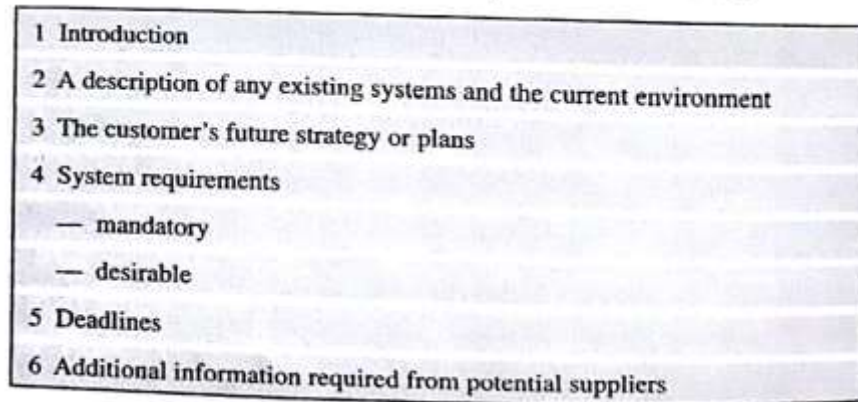


Figure 11.1 – Stages in Contract Placement

11.3.1 REQUIREMENT ANALYSIS

- Before we could even approach potential suppliers, we must have a concise set of requirements.
- A requirements document could even be created by an outside consultant.

- “The lack of or defects in specification are probably at the heart of most disputes resulting from the acquisition of computer equipment and software,” writes David Bainbridge.
- A sample requirements document would contain the following sections



1	Introduction
2	A description of any existing systems and the current environment
3	The customer's future strategy or plans
4	System requirements
	— mandatory
	— desirable
5	Deadlines
6	Additional information required from potential suppliers

Figure 11.2 – Sample Requirements Document

- The requirements carefully define the functions that must be performed by the new application, as well as all of the necessary inputs and outputs for these functions.
- The requirements should also include any standards that must be met, as well as any existing systems that must be compatible with the new system.
- In addition to these functional requirements, there will be operational and quality requirements concerning the new system's required response times, reliability, usability, and maintainability.
- Each requirement must be classified as either mandatory or desirable.
 - **Mandatory** - If a proposal does not meet this requirement, it will be rejected immediately. There would be no need for additional testing.
 - **Desirable** - A proposal may be lacking in this area, but other aspects of the proposal may compensate.
- Requests for any information needed to help us judge the standing of the organization itself should be included among the other details that will be included in the requirements document to be provided to potential suppliers.
- This could include financial reports, customer references, and the CVs of key development personnel.

11.3.2 EVALUATION PLAN

- After developing the requirements, we need to devise a strategy for evaluating proposals.
- The scenario will be different if the contract is for a custom-written system instead of an off-the-shelf package.
- In the latter case, it is application evaluation, whereas in the former, it is proposal evaluation.
- Methodologies for ensuring that the mandatory requirements are met must be identified.
- The next point to consider is how the desirable requirements can be assessed. The issue here is balancing the importance of one quality against another.
- The ISO 9126 standard can help us determine whether one system has more of a certain quality than another, but if there is a price difference between the two, we must determine whether the increase in quality is worth the extra cost.
- The requirement to evaluate value for money (VFM) for each desirable feature.
- The VFM approach improves on the previous emphasis on accepting the lowest bid.
- For example, a feeder file saves data input. The clerical effort is 4 hours work a month saved at £20 an hour and system to be used for 4 years. Cost of feature £1000.
- So, would it be worth it?
- We calculate the cost over a 4-year period as $£20 \times 4 \times 48 = £3840$.
- So, system A with this feature costs £1000 more than system B.
- But this gives an added advantage to system A.
- The costs to be considered are those for the entire lifetime of the proposed system, not just the costs of acquisition.
- Furthermore, where the relationship with the supplier is likely to be ongoing, the supplier organization as well as its products must be evaluated.

11.3.3 INVITATION TO TENDER (ITT)

- After completing the requirements and the evaluation plan, the invitation to tender can be issued to prospective suppliers.

- Essentially, this will be the requirement document accompanied by a supporting letter that may include additional information about how responses to the invitation should be submitted.
- A deadline will be set, and it is hoped that by that time, several proposals with price quotes will have been received.
- In English law, there should be a deal on one side that is accepted by the other side for a contract to exist.
- The invitation to tender is not an offer in and of itself, but rather a request for prospective suppliers to make an offer.
- Certain new issues have emerged. The requirements that have been established may be met in a variety of ways.
- The customer needs to know not only a potential supplier's price, but also how they intend to meet the requirements - this is especially important if the contract is to build a new system from the ground up.
- In the first stage, potential suppliers are asked to submit technical proposals.
- This approach does not necessitate quoting any prices at this time. Some of these proposals may be recommended by the but rejected outright because they do not meet the mandatory requirements.
- Certain aspects of the suppliers' proposals may be required to be demonstrated.
- If flaws in the proposal are discovered, the supplier may be given the opportunity to correct them.
- These discussions may result in a Memorandum of Agreement (MoA) with each prospective supplier.
- This is the customer's acknowledgement that the proposed solution offered by the supplier meets the customer's requirements satisfactorily.
- Tenders are welcomed from all suppliers who have signed independent Memorandums of Agreement in the second stage.
- The MoA would be included in the tender, which would be concerned with the financial terms of a potential contract.
- If a design must be produced as part of a supplier's proposal in response to an invitation to tender, the supplier will have to do a significant amount of detailed design work with only a small chance of being paid for it.

- One way to alleviate this burden is for the customer to select a small number of likely candidates who will be compensated for producing design proposals.
- These can then be compared, and the most appealing proposal will be awarded the final construction contract.
- ISO 12207 takes a different approach.
- Once a contract for software development is signed, the supplier creates a design that must be approved by the customer.

11.3.4 EVALUATION OF PROPOSALS

- This must be done methodically and strategically.
- We have already mentioned the need for an evaluation plan, which will outline how each proposal will be scrutinized to ensure that it meets all of the requirements.
- This reduces the possibility of missing requirements and ensures that all proposals are treated consistently.
- Otherwise, there is a risk that a proposal will be unfairly favored because it includes a feature that was not specified in the original requirement.
- It should be remembered that an application can be bespoke, off-the-shelf, or customized.
- In the case of off-the-shelf packages, the software itself would be evaluated, and it may be possible to combine some of the evaluation with acceptance testing.
- In the case of bespoke development, a proposal is evaluated, whereas COTS may include elements of both.
- As a result, different approaches would be required in each case.
- The process of evaluation may include scrutiny of the proposal documents, interviewing suppliers' representatives, demonstrations and practical tests.
- Supplier proposal documents can be scrutinized to see if they contain features that meet all of the original requirements.
- Clarification on certain points may be required. Any true statements made by a supplier imply a legal commitment on their part if they persuade the customer to award the contract to that supplier.
- As a result, it is critical to obtain a written, agreed-upon record of these clarifications.

- If the delivered product is to be based on an existing product, a demonstration may be possible.
- Demonstrations pose the risk of being controlled by the supplier, and as a passive observer, it is often difficult to maintain full attention.
- As a result, the customer should create a schedule of what needs to be demonstrated, ensuring that all key features are seen in action.
- It should be possible to gain actual access to the application using off-the-shelf software.
- A common issue is that while an existing application performs well on one platform with a certain level of operations, it does not work satisfactorily on the customer's platform or at the level of throughput that it would be subjected to in the customer's work environment.
- Demonstrations, in general, will not reveal this issue.
- Visits to operational sites where the system is already in use will be more informative in this regard.
- As a last resort, a special volume test could be performed.
- A decision will be made eventually to award the contract to one of the suppliers. One of the primary reasons for using a structured and, to the greatest extent possible, objective approach to evaluation is to be able to demonstrate that the decision was made impartially and on merit.
- In most large organizations, placing a contract requires the involvement of a third party within the organization, such as a contracts department, who can ensure that the proper procedures are followed.
- In addition, the final legal format of a contract will almost certainly necessitate the use of legal expertise.
- In any scenario, not only should the successful candidate be alerted, but so should the unsuccessful candidates.
- Legal advice on contract terms is essential when large sums of money are involved.

11.4 TYPICAL TERMS OF A CONTRACT

- It is unrealistic to expect a project manager to be a legal expert.
- He requires assistance.
- However, he must ensure that the contract reflects the true requirements and expectations of both the supplier and the client.
- So, the various terms used in contract checklist are as follows

- **Definitions** - The contract document's terminology, for example, who is meant by the words 'client' and 'supplier' may need to be defined.
- **Form of agreement** - Is it, for example, a contract of sale, a lease, or a licence? Also, can the contract's subject matter, such as a licence to use a software application, be transferred to a third party?
- **Goods and services to be supplied** - It is necessary to provide equipment and software. This includes a detailed list of the individual pieces of equipment that will be delivered, complete with model numbers. Services to be rendered. Documentation, installation, conversion of existing files, maintenance agreements, and transitional insurance arrangements are all covered.
- **Ownership of software** - Who is the owner of the software? There are two major issues to consider here: first, whether the customer can sell the software to others, and second, whether the supplier can sell the software to others. When it comes to off-the-shelf software, the supplier frequently simply grants you a licence to use the software. When software is written specifically for a customer, that customer will usually want exclusive use of the software; they may object to software that they hoped would give them a competitive advantage being sold to their competitors. When software is written by an employee as part of a job contract, it is assumed that the employer owns the copyright. Where the customer organisation has contracted an external supplier to write software, the contract must specify who will retain the copyright - it cannot be assumed in this case that it is the customer.
- **Environment** - When installing physical equipment, the demarcation line between the supplier's and customer's responsibilities for things like accommodation and electrical supply must be specified. Where software is delivered, the software's compatibility with existing hardware and operating system platforms must be confirmed.
- **Customer commitments** - Even when work is done by outside contractors, the customer must still be involved in the development process. The customer will be required to provide accommodation for the suppliers as well as possibly other facilities such as phone lines.
- **Acceptance procedures** - It is best practise to accept a delivered system only after it has undergone user acceptance testing. This section of the contract would include information such as how much time the customer will have to conduct the tests, the deliverables on which the acceptance tests will be

based, and the procedure for signing off on the testing as completed.

- **Standards** - This covers the standards that the goods and services must meet. Some customers complain that the supplier does not thoroughly test specially written or modified software before delivery. Some suppliers appear to believe that it is less expensive to have the customer do the testing for them.
- **Project and quality management** - The project management arrangements must be agreed upon. These would include the frequency and nature of progress meetings, as well as the progress information to be provided to the customer. The contract could specify that ISO 9000-series standards be followed. The ISO 12207 standard requires the customer to have access to quality documentation generated internally by the supplier in order to ensure standard adherence.
- **Timetable** - This provides a timeline for when the major components of the project should be completed. This schedule binds both the supplier and the customer.
- **Price & Payment method** - Obviously, the cost is critical! When the payments are to be made must also be agreed upon. The supplier's preference to meet costs as they arise must be balanced against the customer's commitment to protect those goods and services are satisfactory before dividing with their money.
- **Miscellaneous legal requirements** - This is the legal stipulation. Contracts frequently include clauses that address issues such as the legal jurisdiction that will apply to the contract, the conditions that will apply to the subcontracting of the work, liability for third-party damage, and liquidated damages. Liquidated damages are estimates of the customer's financial losses if the supplier fails to meet their obligations. If there is a disagreement, resorting to litigation, while profitable for the lawyers involved, is both time-consuming and costly. Another option is to agree to have disputes resolved through arbitration. This necessitates the referral of any dispute to an expert third party whose decision on the facts of the case is binding. Even this procedure is rarely quick and cheap, and another option is alternative dispute resolution, in which a third-party act as a mediator with only advisory powers and attempts to broker an agreement between the two parties.

11.5 CONTRACT MANAGEMENT

- We must now consider the communications between the supplier and the customer while the work is being completed.

- It would probably be in everyone's best interests if the contractor could be left alone to finish the job.
- However, at certain decision points, the customer must review previous work and make decisions about the project's future direction.
- The project will necessitate interactions between supplier and customer representatives at various stages of the development cycle.
- This interaction, or other external factors, frequently necessitate changes, which effectively change the terms of the contract, necessitating a careful change control procedure.
- When negotiating a contract, certain key points in the project can be identified where customer approval is required before the project can move forward.
- When work is outsourced, there will be widespread concern about the quality of that work.
- The ISO 12207 standard allows for the possibility of agents working independently of the supplier or customer to perform verification, validation, and quality assurance.
- It also permits joint reviews of project processes and products, the nature of which must be clearly agreed upon when the contract is negotiated, or the supplier may claim unwarranted interference in their work.
- As the system evolves, it is common for the need to change some of the requirements.
- To record requests for changes, as well as the supplier's agreement to them and any fees for the additional work, an effective change control procedure is required.
- It is possible that the supplier will fail to meet one or more of their legal obligations.
- As a result, the customer should protect their legal rights by alerting the supplier as quickly as possible that the failure has been identified.

11.6 ACCEPTANCE

- When the work is finished, the customer must take action to perform acceptance testing.
- Because the contract may specify a time limit for how long acceptance testing can take, the customer must be prepared to complete this testing before the time limit for requesting corrections expires.

- Some software companies are rather hurried with their pre-acceptance testing, with the implication that they would rather have users spend their time testing than they do.
- This imposition can be mitigated by requesting approval of the supplier's internal test plans.
- A related stumbling block is that once the main development work is completed, the supplier will naturally want to reassign the most productive employees to other projects.
- All of the customer's problem reports may be handled by relatively junior members of the supplier's staff who may not be familiar with all aspects of the delivered system.
- Part or all of the payment to the supplier will be contingent on the results of this acceptance testing.
- A portion of the final payment is sometimes retained for a period of operational running and is eventually paid over if the levels of reliability are as contracted.
- There is usually a warranty period during which the supplier should fix any errors discovered at no cost.
- The supplier may propose a very short warranty period, such as 30 days.
- It is in the best interests of the customer to negotiate a more realistic period of at least 120 days.

11.7 SUMMARY

- Successful contracting out of work necessitates a significant amount of management time.
- Before a contract is signed, it is easier to obtain concessions from a supplier than afterward.
- Alternative proposals should be evaluated as thoroughly as possible by comparing costs over the system's entire lifetime rather than just the acquisition costs.
- A contract imposes obligations on both the customer and the supplier.
- Contract negotiations should include reaching an agreement on how the supplier -customer relationship will be managed during the project's execution.

11.8 LIST OF REFERENCES

- “Software Project Management”, Sixth Edition, Bob Hughes, Mike Cotterell, Rajib Mall, Mc Graw Hill Education.
- <https://www.aresprism.com/guides/contract-management-guide>
- <https://www.upcounsel.com/types-of-contracts-in-software-project-management>
- <https://www.greycampus.com/blog/project-management/different-types-of-contract-&-project-management>

11.9 UNIT END EXERCISES

1. A system to be designed and implemented is counted as comprising 3200 FPs. What would be the total charge according to the schedule in table?
2. A contract stipulates that a computer application is to be designed, constructed, and delivered at a cost of \$650 per FP. After acceptance testing, the customer asks for changes to some of the functions in the system amounting to 550 FPs and some new functions which amount to 250 additional FPs. Using the table, calculate the additional charge.
3. How would you evaluate the following aspects of a proposal?
 - a. The usability of an existing software application
 - b. The usability of a software application which is yet to be designed and constructed.
 - c. The maintenance costs of hardware to be supplied
 - d. The time taken to respond to requests for software support
 - e. Training
4. List the various typical terms of a contract?
5. Explain the different stages in contract placement.

12

MANAGING PEOPLE IN SOFTWARE ENVIRONMENT

Unit Structure

- 12.0 Objectives
- 12.1 Introduction
- 12.2 Understanding Behaviour
- 12.3 Organizational Behavior: A Background
- 12.4 Selecting the Right Person for the Job
 - 12.4.1 Recruitment Process
- 12.5 Instruction in the Best Methods
- 12.6 Motivation
 - 12.6.1 Taylorist Model
 - 12.6.2 Maslow's hierarchy of needs
 - 12.6.3 Herzberg's two factor theory
 - 12.6.4 Expectancy theory of Motivation
- 12.7 Oldham-Hackman Job Characteristics Model
 - 12.7.1 Methods of improving motivation
- 12.8 Stress
- 12.9 Stress Management
 - 12.9.1 Categories of Stress Management
- 12.10 Health and Safety
- 12.11 Some Ethical and Professional Concerns
- 12.12 Summary
- 12.13 List of References
- 12.14 Unit End Exercise

12.0 OBJECTIVES

After going through this chapter, you will be able to understand

- The impact of people behavior in project
- Selection between eligible and suitable candidate
- How to conduct recruitment process
- How to handle stress
- Ethical and professional concerns related to people and project

- There are four major issues to be concerned about namely staff selection, staff development, staff motivation and maintaining staff well-being throughout the course of a project.
- The Step Wise Framework approach has impact of people in several step.
- Some project objectives can address health and safety (Step 1).
- Although project leaders may have little control over organisational structure, they must be aware of its implications (Step 2).
- The scope and nature of activities can be set in such a way that staff motivation is enhanced (Step 4).
- Many risks to project success are related to staffing (Step 6).
- When allocating staff to activities, the qualities of individual members of staff should be considered (Step 7).

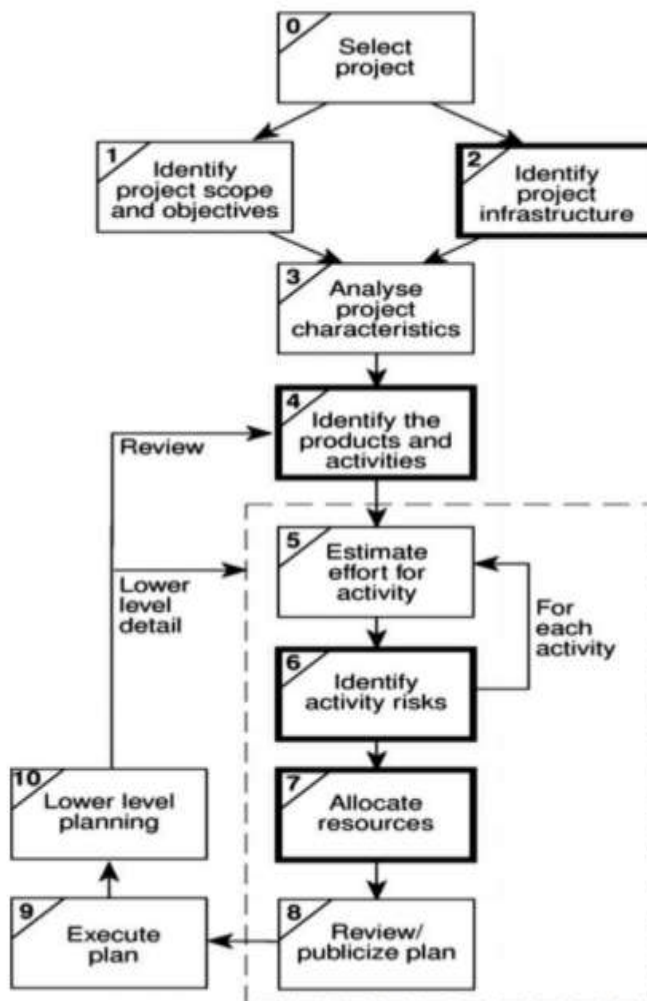


Figure 12.1 – Staffing Concerns in Step Wise Framework

12.2 UNDERSTANDING BEHAVIOUR

- People with practical project management experience undoubtedly identify people management as one of the most essential parts of project management.
- Organizational behaviour (OB), a branch of social science, can be useful.
- This has resulted in theories that attempt to explain people's behaviour and are typically structured as 'If A is the situation, then B is likely to result.'
- A key concern is that in the actual world, there will almost certainly be a wide range of influences on a situation, many of which will be invisible to the observer.
- As a result, determining which set of research findings is pertinent is difficult.
- The risk is that we will end up with a set of dictums that are no better than superstitions.
- However, it is hoped that by investigating these questions, people will become more sensitive and thoughtful about the issues at hand.
- Individual and group behaviour research in software and ICT development environments must use social science research methods.
- The Positivist approach and the Interpretivist Approach are two approaches to establishing the relationship.
- The positivist approach is characterised by a mindset that favours experimentation as a means of establishing relationships between inputs and outputs.
- This model has been attempted to be extended to the social system.
- An interpretivist school of thought differs from a positivist one, particularly in terms of applying quantitative and experimental methods from the physical sciences to people and organisations.
- Many concepts, according to interpretivists, are not objective but rather inter-subjective creations of humans.
- Both positivist and interpretivist perspectives can be valid and useful.
- The quantitative type predominates in the types of research that underpin the material on individuals in the workplace.

12.3 ORGANIZATIONAL BEHAVIOR: A BACKGROUND

- The origins of OB research can be traced back to Frederick Taylor's work in the late nineteenth and early twentieth centuries.
- He attempted to figure out the most productive way to do these tasks by studying how manual workers did them.
- The workers were then taught how to do the work in this manner.
- Taylor had three main goals:
 - to choose the best person for the job
 - to train such people in the best methods
 - to reward the best workers with higher wages.
- Taylorism is frequently portrayed as crude and mechanistic these days.
- However, the concern about identifying best practises is legitimate such as structured and agile methods.
- During the 1920s, OB scientists found, while conducting a now-famous set of tests on the conditions for which staff worked best, that not only did a group of workers whose conditions were improved increase their work-rates, but so did a control group whose conditions remained unchanged.
- Simply caring about what employees did increased productivity.
- This demonstrated how workers' mental states influenced their productivity.
- Some managers' cash-oriented view of work can thus be contrasted with a more refined view of people in their workplace.
- Donald McGregor labelled the two attitudes as Theory X and Theory Y.
- According to Theory X, the average human has an innate dislike of work; as a result, there is a need for coercion, direction, and control; and people tend to avoid responsibility.
- Theory Y, on the other hand, holds that: work is as intuitive as rest or play; external control and coercion should not be the only ways to elicit effort directed toward the company's goals; commitment to goals is a function of the rewards associated with their accomplishment; the average human can try to accept and seek

responsibility; and the ability to exercise creativeness and other creative abilities.

- One way to determine whether a manager believes in Theory X or Theory Y is to perceive how the manager's staff reacts when the boss is absent: if there is no perceivable change, the environment is Theory Y; if everyone visibly relaxes, the environment is Theory X.
- McGregor's distinction between the two theories also emphasises how expectations influence behaviour.
- If a manager (or teacher) expects you to work hard and do well, you are more likely to try to meet those expectations.

12.4 SELECTING THE RIGHT PERSON FOR THE JOB

- Taylor emphasised the importance of hiring the right person for the job. Many factors influence programming productivity, including the use of software tools and methodologies.
- Individual differences in software development performance, on the other hand, are one of the most pronounced.
- A contrast of experienced professional programmers working along the same programming task in 1968 discovered a ratio of 1:25 between the shortest and longest time to code the code and, perhaps more importantly, a ratio of 1:28 for the time required to debug it.
- Questions in mind while selecting the right person could be
 - What kinds of qualities should they be looking for?
 - Should they hire an experienced programmer or a recent graduate with a first-rate mathematics degree, for example?
- It is highly risky to generalise, but when particularly looking at behavioural traits, American researcher Cheney discovered that the most significant effect on programmer productivity appeared to be experience.
- In comparison, mathematical competence had a negligible influence.
- Couger and Zawacki, two American researchers, discovered that computer workers appear to have significantly lower social needs than people in other professions.
- They quote Gerald Weinberg, who says, 'If asked, most programmers probably say they chose to work alone where they are not disturbed by other people.'
- This is reflected in the issue that people who are drawn to and skilled at writing software would not make good managers later in their professions.
- Later polls found no statistically significant discrepancies between IS and other staff.

- This could be explained by the fact that in recent years, IS has become wider and less solely technical.

12.4.1 RECRUITMENT PROCESS

- Recruitment may be regarded as an organizational responsibility.
- We may be currently seeking someone who will work in many different parts of the same organization over time.
- Meredith Belbin makes an important distinction between eligible (qualified) and suitable candidates.
- A qualified candidate is one whose curriculum vitae also referred as resume shows, for example, the right number of years in some previous position and the right paper qualifications.
- Suitable candidates are those who are capable of performing the job well.
- A common misperception is to choose a qualified candidate who is not actually suitable.
- Suitable candidates who are not technically eligible can on the other hand, be ideal candidates because they are more likely to remain loyal to the organization once in position.
- Belbin suggests that selection methods that focus on real skills rather than past experience, as well as an eagerness to provide training to fill minor gaps in expertise, can be more effective in placing suitable staff.
- It also appears to demonstrate that policies that prohibit discrimination on the basis of race, gender, age, or irrelevant disabilities can be both socially responsible and effective recruitment strategies.
- As the name implies, the recruitment and selection process begin with recruiting candidates and ends with selecting a candidate to hire.
- Being thorough and following each step can result in more successful hires and retention rates.
- Examine the following steps in the recruitment and selection process:
 - **Obtain a job order** - When we obtain a job order from a client, we can begin the recruiting process. A job order should include information about the position being filled as well as a well-written job description. The job description should inform potential candidates about everything they need to know about the position, including job title, job description in detail, qualifications that are required and preferred, location and salary scale. Consider rewriting the job description if it does not provide enough information or is not written in a way that will attract top talent.

- **Source candidates** - The next step in the recruitment and selection process is to source candidates once you have a thorough understanding of the open position. We can find passive and active candidates in a variety of ways. Active candidates are those who are actively looking for work, whereas passive candidates are those who are not. Successful recruiters can find both types of candidates. We can find candidates using the appropriate tools and recruitment sources: Online job boards, social media, a recruiting database, and referrals.
- **Screen applicants** - Applicants must be screened as part of the recruitment and selection process. This is where you can learn more about each applicant, allowing us to narrow down the pool. We can perform telephone screenings with a variety of pre-screening interview questions. Ask behavioral interview questions during screenings to learn more about the candidate's personality and how they would function in the open position. Inquire about the candidates' backgrounds, including their work history and career goals. Check to make sure they understand the job description and are qualified. Interviews over the phone should last about 30 minutes. Even if they are not as long as a full interview, we can learn enough to help narrow down candidates. To rank candidates and keep track of their responses, create a candidate scorecard. Take notes so you can compare candidates after spoken with them all.
- **Shortlist candidates** - The procedure of advancing a few candidates from the pool is known as recruitment shortlisting. The shortlisted candidates should consist of no more than three people. These are the contenders we want to invite for client's face-to-face interview. Narrowing down the candidate pool can be difficult because we don't want to advance the wrong candidates. Take the time to learn about each candidate's experiences, qualifications, and personality so we can be confident we have chosen the right people to shortlist.
- **Interview candidates** - After we have narrowed down the candidates, we must provide their contact information to the client. The candidates will then be interviewed by the client. During interviews, we should typically be present to take notes, ask questions, and give opinion afterward. The face-to-face interview allows client to get to know the candidates on a more personal level. We can observe their body language and conduct additional behavioral interviews with them. The interview process allows to gauge a candidate's work ethic. Again, use an interview scorecard to rank and compare candidates. Candidates should be ranked based on their experience, education, and skills.

- **Conduct testing** - The client may want to consider conducting job-fit tests to further assess a candidate's abilities. A job-fit assessment test assists the client in determining how well the candidate would fit into the company. A job-fit test can last from 30 minutes to an hour. It asks a series of questions to which candidates must honestly respond. Each candidate should also be subjected to a background check. Furthermore, we must check references to confirm information and learn more about their character and work ethic.
- **Extend a job offer** - The final step in the selection process is to choose a candidate. Extend the job offer to the candidate they are interested in hiring. The candidate may attempt to negotiate the salary offered by the client. Consult with the client to see if the requested salary is feasible. If the candidate falls the job offer, we will need to either return to the other top candidates or restart the recruitment and selection process.

12.5 INSTRUCTION IN THE BEST METHODS

- This is the second of Taylor's concerns that we have adopted.
- When a new team member is hired, the team leader must carefully plan that person's integration into the team.
- This may be difficult if the project is already well underway.
- However, the steps should be taken because it will eventually pay off because the new recruit will become a fully functional member of the team more rapidly.
- The team leader should also be able to understand the importance of constantly assessing the training requirements of their team members.
- Just as we create a user requirement before considering a new system and a job holder profile before hiring a new employee, we also create a training needs profile for each staff member before considering specific courses.
- Commercial training companies could provide some training.
- Where money is limited, other sources of training should be considered; however, training should not be abandoned entirely, even if it consists of a team member being told to learn about a new software tool and then demonstrate it to colleagues.
- Of course, the methods learned must be put into practise.
- This should be ensured through reviews and inspections.

12.6 MOTIVATION

- Motivation is the third factor as projected by Taylor.
- It is the driving force by which people achieve their goals.
- There are several models for motivation followed in software development practices such as Taylorist model, Maslow hierarchy of needs, Herzberg two factor theory and Expected theory of motivation.

12.6.1 TAYLORIST MODEL

- Taylor's point of view is represented in the use of piece-rates in manufacturing and sales bonuses among sales forces.
- A problem that projects leader must be aware of is that piece-rates frequently cause difficulties when implementing a new system that changes work practices.
- If new technology improves productivity, the question of lowering piece rates to reflect this will be a delicate one.
- Workers on piece-rates are paid a set amount for each item produced.
- Payment for time worked is referred to as a day rate.
- Typically, drastic changes in work practices must be preceded by a shift from piece-rates to day-rates.
- The norms of the group even when work practices are stable and output can be easily linked to reward, as discussed further under people paid by the amount they produce, group decision making will not be maximized.
- In order to maximize their income, they must maximize their output.
- The amount of output is frequently constrained by 'group norms,' which are informal, even unspoken, agreements among colleagues about how much to produce.
- Rewards must be directly related to the work produced in a simple and direct manner.
- This is not an easy task when creating a computer system.
- It is difficult to isolate and quantify work done, particularly because system development and support is a collaborative effort.
- A reward system that makes inordinate distinctions between work colleagues can be detrimental to morale and, eventually, productivity in this type of environment.

12.6.2 MASLOW'S HIERARCHY OF NEEDS

- Maslow's hierarchy of needs is a motivational theory in psychology that consists of a five-tier model of human needs, which is frequently depicted as hierarchical levels within a pyramid.
- The needs are listed in the following order: physiological (food and clothing), safety (job security), love and belonging needs (social), esteem, and self-actualization.
- Lower-level needs must be met before individuals can attend to higher-level needs.
- This five-stage model is divided into two parts: deficiency needs and growth needs. The first four levels are known as deficiency needs (D-needs), while the top level is known as growth or being needs (B-needs).
- Deficiency needs arise as a result of deprivation and are said to motivate people when they go unsatisfied.
- Furthermore, the motivation to meet such needs grows stronger the longer they are denied.
- Growth needs arise from a desire to grow as a person, not from a lack of something.
- Once these growth needs are reasonably met, one may be able to achieve the highest level known as self-actualization.
- Maslow observed that the hierarchy of needs may be flexible depending on external circumstances or individual differences. surpass even the most basic requirements
- Maslow also stated that most behavior is multi-motivated and that “Any behavior tends to be determined by several or all of the basic needs concurrently rather than by only one of them.”



Figure 12.2 – Maslow’s Hierarchy of Needs

12.6.3 HERZBERG'S TWO FACTOR THEORY

- Certain aspects of a job may make you unhappy.
- If the sources of this dissatisfaction are addressed, the job will not necessarily become more exciting.
- According to Herzberg and his associates research into job satisfaction, there appeared to be two sets of factors about a job that were vital
 - **Hygiene or maintenance factors** can make you displeased if they are not right, such as the level of pay or working conditions.
 - **Motivators** can make you feel that the job is worthwhile such as a sense of accomplishment.

12.6.4 EXPECTANCY THEORY OF MOTIVATION

- This is illustrated by a motivation method designed by Vroom and his colleagues.
- It outlines three motivational influences:
 - **Expectancy** – It is the presumption that working harder will result in improved performance.
 - **Instrumentality** – It is the presumption that improved performance will be rewarded.
 - **Perceived value** – It is the perceived value of the resulting reward.
- When all three factors are present, motivation will be high.
- A zero for any of the factors can result in a lack of motivation.

12.7 OLDHAM-HACKMAN JOB CHARACTERISTICS MODEL

- Managers should attempt to group the elements of the tasks that must be completed so that they form meaningful and satisfying assignments.
- According to Oldham and Hackman, job satisfaction is based on five factors.
- The first three factors contribute to the job being meaningful to the person doing it:
- **Skill variety** – It represents the myriad of alternative skills that the job holder has the opportunity to exercise.

- **Task identity** - the intensity to which your work and its outcomes can be identified as yours
- **Task significance** -The degree to which your job has an impact on others.
- The other two factors are:
 - **Autonomy** – It is the freedom to choose how you do your job.
 - **Feedback** - The information you receive about the outcomes of your work.
- In practise, activities should be designed in such a way that employees can track the progress of a specific product and feel personally connected to it.

12.7.1 METHODS OF IMPROVING MOTIVATION

- To improve motivation, the manager does
 - **Selling specific objectives** - These objectives must be challenging while also being acceptable to the staff. Involving employees in the development of goals aids in their acceptance.
 - **Providing feedback** - Goals must be set, but employees must also receive regular feedback on how they are progressing.
 - **Job creation** - Jobs can be changed to make them more interesting and to give employees a greater sense of responsibility.
- Job enlargement and job enrichment are required to measure and enhance job design.
- In job enlargement, as employees perform a broader range of activities, the job's scope expands.
- It is the inverse of increasing enrichment, which is based on specialisation.
- In the case of job enrichment, the job is altered so that the holder performs tasks that would normally be performed at a relatively high, managerial level.
- Employees may be assigned responsibility for ordering consumables, scheduling their work, or quality control.
- A programmer on a maintenance team may be given authority to accept requests for changes that require less than five days of work without requiring their manager's approval.

12.8 STRESS

- Projects are about overcoming adversity and reaching goals.
- Both the project manager and the members of the team will be under duress.
- According to Edward Yourdon, a project manager, “Once a project gets going, you should expect members to put in at least 60 hours per week.” The project manager should plan on working as many hours as possible”.
- Some stress is actually beneficial.
- Many jobs can be soul-destroying due to boredom.
- However, once you reach a certain level of stress, the quality of your work suffers, and your health suffers.
- In a 1960 study in the United States, it was discovered that people under the age of 45 who worked more than 48 hours per week had twice the risk of dying from coronary heart disease.
- Many software developers are required to work extra hours on projects for no extra pay.
- In these cases, a decrease in productivity is more than offset by the fact that the work is essentially free to the employer.
- Result from good project management include
 - Rational effort estimates.
 - Results in fewer unexpected crisis.
 - Making it clear what each team member is expected to do – this reduces role ambiguity.
 - Reduced role conflict, which occurs when a person is torn between competing responsibilities.
- Bullying tactics are a sign of ineffective project management.
- This, however, is the total opposite of competent project management, which seeks to create complex products in a rational, orderly, and careful manner.

12.9 STRESS MANAGEMENT

- Stress is a normal part of almost everyone's life, and it is generally agreed that a certain amount of stress can be beneficial by making a person more focused and productive.

- Low stress is typically associated with boredom, which leads to a decrease in performance.
- When stress becomes too much, performance suffers as a result of cognitive, emotional, and physical strains.
- When an individual's stress level exceeds a certain threshold, appropriate stress management techniques should be used.
- Worrying, forgetting, and lack of concentration are all symptoms of cognitive strain.
- Anxiety, restlessness, panic, interpersonal problems, losing touch with friends, irritability, and anger can all result from emotional strains.
- Physical strains may emerge as shallow breathing, nausea, fatigue, headache, shoulder and back pain, sleep disturbances, and hypertension.

12.9.1 CATEGORIES OF STRESS MANAGEMENT

- There are three important categories of stress management techniques.
 - **Imagery, relaxation, and meditation** - Deep breathing, relaxation, physical exercise, guided imagery, yoga, progressive muscle relaxation, and massage therapy are all used in these techniques. Rolling the head from side to side is an example of a simple relaxation technique. Guided imagery encompasses a wide range of techniques, including simple visualisation. To instil a positive feeling, use metaphor and story-telling.
 - **Cognitive behavioural approaches** - These techniques entail the development of emotion-focused cognitive coping skills such as stress concentration self-monitoring, thought record-keeping and rewriting, time management, assertiveness training, and increased social interactions.
 - **Systemic approach** - Systemic approaches concentrate on modifying the factors that contribute to stress. For example, if a team member finds it stressful to work with certain tools and techniques, a job role change may be suggested so that the team member does not have to deal with the stressful tools and techniques.

12.10 HEALTH AND SAFETY

- Health and safety refer to programmes, policies, and procedures that protect the safety, welfare, and health of anyone who works or is employed.

- The overall goal of any health and safety programme is to create the safest working environment possible and to reduce the risk of workplace accidents, injuries, and fatalities.
- The primary goal of workplace safety and health programmes is to prevent workplace injuries, illnesses, and deaths, as well as the pain and financial hardship that these events can cause for workers, their families, and employers.
- Every organisation, no matter what they do, is responsible for maintaining workplace health and safety standards.
- If a company does not follow these guidelines, they may be held liable for any damages or accidents that occur.
- Not only would the company have a legal and financial obligation to comply with health and safety standards, but it also has a moral obligation to care for the well-being of its employees.
- The project manager should treat safety objective as any other objective.
- Safety management should be part of project management.
- Certain issues to be considered in implementing safety at all levels are as follows
 - Top management must be dedicated to the safety policy.
 - Delegation of safety responsibilities should be clear. Job descriptions should include definitions of safety-related duties.
 - Those to whom responsibilities are delegated must understand and accept those responsibilities.
 - Deployment of a safety officer and assistance from experts in specific technical areas.
 - Safety consultation.
 - A sufficient budget for safety costs.
- Employees must be made aware of safety procedures, and appropriate training must be provided where necessary.

12.11 SOME ETHICAL AND PROFESSIONAL CONCERNS

- There is now a legal requirement to take action to reduce workplace threats to employees' health and safety.
- Even if such a law did not exist, there would be few who would not at least acknowledge the moral obligation to protect those at work from foreseeable harm.

- This would be an ethical decision.
- There are three groups of ethical responsibilities:
 - Everyone has responsibilities.
 - People in organisations have responsibilities.
 - Responsibilities associated with your profession
- Another reason for commercial organisations reduced or at least unusual ethical responsibilities is that they compete with other businesses.
- If a company wins some aspect of this game the competitors must lose investors may lose money, and employees may lose their jobs.
- However, it is argued that this is how the market works, and as a result, consumers benefit from lower prices.
- However, in the long run, competition that destroys competitors leads to monopoly dominance and higher prices.
- Most organisations, on the other hand, will recognise that they do have ethical responsibilities.
- This could be motivated solely by self-interest.
- As a potential customer, you may be wary of entrusting your business to organisations that are clearly motivated solely by greed.
- Organizations frequently express their goals and aspirations, perhaps in the form of a mission statement, and these often include goals related to the general public good, such as concern for the environment.
- Professionals have technical knowledge that the general public does not have.
- The expert's ethical duty to warn laypeople of the risks involved in a particular course of action.
- Many professions or would-be professions, have member codes of conduct presented in
 - <http://www.bcs.org/upload/pdf/cop.pdf>
 - <http://www.ieee.org/web/aboutus/ethics>
 - http://www.apm.org/about/se_code

12.12 SUMMARY

- People are influenced by profit, but they are also motivated by other factors.
- Both personnel selection and identifying training needs should be done in a systematic, structured manner, with requirements clearly defined first.
- A well-thought-out job design can boost employee motivation.
- Overburdening employees may yield short-term benefits, but it is detrimental to both productivity and personal health in the long run.
- Objectives of the project should include, as appropriate, health and safety objectives.

12.13 LIST OF REFERENCES

- “Software Project Management”, Sixth Edition, Bob Hughes, Mike Cotterell, Rajib Mall, Mc Graw Hill Education.
- <https://www.iedunote.com/organizational-behavior>
- https://www.tutorialspoint.com/organizational_behavior/organizational_behavior_motivation.htm
- <https://www.techwell.com/techwell-insights/2018/05/stress-and-project-management-5-ways-relieve-project-pressure>
- <https://pmtips.net/article/project-managers-care-health-safety>

12.14 UNIT END EXERCISES

1. Explore the possible disadvantages of job enlargement for employers and staff.
2. What is motivation under the Taylor’s model?
3. Explain how new staff can be selected and inducted into a project.
4. Explain the methods to increase staff motivation.
5. Give a brief note on health and safety issues.

WORKING IN TEAMS

Unit Structure

- 13.0 Objectives
- 13.1 Introduction
- 13.2 An Overview
 - 13.2.1 Becoming a Team
 - 13.2.2 Decision Making
 - 13.2.3 Organization and Team Structures
 - 13.2.4 Coordination Dependencies
 - 13.2.5 Dispersed and Virtual Teams
 - 13.2.6 Communication Genres
 - 13.2.7 Communication Plans
 - 13.2.8 Leadership
- 13.3 Summary
- 13.4 Exercise
- 13.5 List of Books and References

13.0 OBJECTIVES

After going through this chapter, you will be able to know:

- Introduction of Working in Teams Becoming a Team how to build effective software development team and types of teams, Decision Making What Is Agile Decision-Making in Project Management? Organization and Team Structures What Is an Organizational Structure? And types of organizational structure
- Coordination, what is coordination in a team? And types, Dependencies **and types of dependencies** Dispersed and Virtual Team Types of Virtual Teams and its Advantages and Disadvantages
- Communication Genres, Communication Plans What is a communication plan? The benefits of a communication plan, How to make a project management communication plan?
- Leadership and different leadership styles

13.1 INTRODUCTION

A team is a group of people who work together toward a common goal. Teams have defined membership (which can be either large or small) and a set of activities to take part in. People on a team collaborate on sets of related tasks that are required to achieve an objective. Each member is responsible for contributing to the team, but the group as a whole is responsible for the team's success.

Project teams do the work of the project. Team building is well-known, focuses on team attitudes and teamwork. Transferring the learning from team building and teamwork to working as a team is tough. Little thought or effort is given to the work of the team. Become a more effective project manager. Understand project team development, teamwork, and the work of the project team. Know that project work and project management work is not teamwork, nor the work of the team, nor the development of the team. Be a better project leader by understanding teams and ensuring the team works.

13.2.1 Becoming a Team

Five Steps to Building an Effective Software Development Team

When kicking off a new software development project, naturally we all intend for it to be a success. But to be one you must rely on a strong core — your software development team. While it is an undeniable truth that all teams are different in terms of their work style and the unique ecosystem within, there are some common elements to it. We all expect our team to consist of highly experienced and skilled individuals. But is it the only prerequisite? And, in general, how to build an effective software development team? To make a group of professions a truly effective software development team you need to remember about some elements to consider:

- 1. Define the kind of development team type that fits for your project**

First things first, before we dive deeper into what pay attention to when building a development team, you need to decide on the kind of team you want to create. Establishing a clear software development team structure is an important first step into an overall success of your project.

Here you must choose from the three main types:

- Generalists

They are Jack of all trades, so to say. Generalists possess a broad range of knowledge and expertise. Generally, these types of teams are designed to handle end-to-end solutions. The advantage of generalists is in the fact that they can provide a complete solution to the problem. However, they

also have some drawbacks — if your project requires a higher level of expertise in some area, generalists will find themselves at a loss as they may lack the knowledge and skills.

- **Specialists**

Unlike generalists, this type of teams consists of members that are highly skilled in a particular field. The advantage of specialists is clear — they can address a specific matter with all their knowledge and expertise, resulting in more efficient and effective work. On the other hand, communication is not exactly a forte of this type of software development team. Often being a very narrow specialist, team members may lack general understanding of what are the roles of other team members, and thus making communication between them somewhat ineffective.

- **Hybrid team**

As you've probably guessed, this type of team is a mix of the previous two. It seems like this approach combines the best from the two worlds, where specialists focus on functional parts, and generalists are responsible for the communication and cooperation inside the team. This dream of a team, however, comes with a constraint — usually this type of software development team is more difficult to gather in terms of time and financial resources.

So, which one is it? Well, that's for you to decide. Ideally, strive for the balance of generalists and specialists within your team to achieve better results.

2. **Decide on the software development team size**

Now that you've established what kind of team you want to build, let's talk about the size. There really isn't a magical number when it comes to the size of your software development team. Smaller teams are easier to manage on the one hand, but in this case, every team member plays a crucial role in the project and losing even one person can have a significant impact on the overall result. With bigger teams the challenge is in managing the communication.

But when it comes to assembling the team, it all boils down to the following key factors:

- Complexity of your project
- Budget
- Deadline
- Available resources

Based on these important elements you can decide what kind of team size is your solution. According to Scrum methodology, the optimal team size is between 3 and 9 members with 7 being the most perfect fit. But that doesn't mean that you must strictly follow this rule. If your software project requires a bigger team, it doesn't mean you are bound to have problems in managing it or establishing the communication. The key here is to carefully manage your team in accordance to your project requirements.

3. Establish clear roles and goals

Now this one seems like an obvious one — the roles inside your team are clear. There are designers, developers, and probably a tester, right? Yes but no. The roles of an effective software development team are more versatile and complex than that.

The general development team structure looks and includes the following roles:

Now let's look more closely into each of software development team roles:

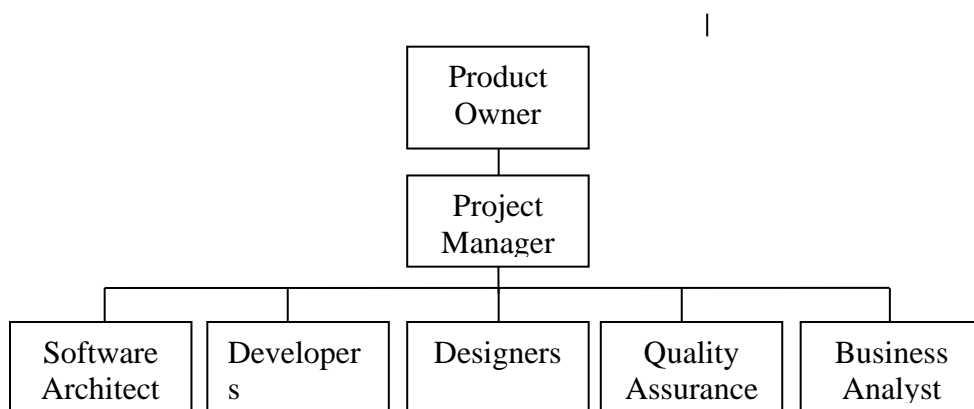


Figure 13.1 General development team structure

Product owner, in the case of an outsourced project, this is the client with a vision of how the end-product should look, who are the end-users and what it should do.

Project manager is a person responsible for managing and leading the whole team. Their role is to efficiently optimize the work of the team, ensure the product is meeting the requirements and identify the goals for the team.

Software architect is a highly skilled software developer that has to think through all the aspects of the project and is responsible for making high level design choices, as well as select technical standards (for instance, determines the technology stack to use).

Developers or product engineers are team members that apply their knowledge of engineering and programming languages in software development.

Experience designers ensure that the product is easy and pleasant to use. They conduct user interviews, market research, and design a product with end-users in mind.

QA or tester is responsible for the Quality Assurance and makes sure the product is ready to use.

Business Analyst's role is to uncover the ways to improve the product. They interact with stakeholders to understand their problems and needs, and later document and analyze them to find a solution.

4. **Build an agile software development team**

Agile methodology makes your team more flexible; it allows to adapt and respond to some unforeseen changes that might come without damaging the whole process.

There's no single right formula how to build your agile software development team. Even within this approach to software development there are two types of methodology: some prefer Scrum, while others use Kanban.

The later practices real-time communication and full work transparency. It relies greatly on visualization. It's in the core of this framework. Work is constantly visually represented on Kanban board to help the team understand at what stage they currently are. On top of that, by using visualization, they can work out what are they stumbling blocks, what slows them down and find ways to overcome them.

Scrum is the most popular Agile framework that breaks down a large project into smaller chunks (sprints) and reviews and adapts them along the way. Sprints can last from a week to a month in duration. If you are adopting Scrum methodology, the structure of your software development team will include a very important element — Scrum master. He or she makes sure that the team sticks to the agile values and principles and follows the process that the team has agreed on.

Regardless of the framework you choose, agile will help your team to deliver faster and efficiently.

5. **Make communication a priority**

And the last but definitely not least — communication. This is the last piece of the puzzle that very often is either the reason of success or failure software development team. You can have all the other elements right on track and properly functioning but if your team

lacks this last element, it can sufficiently harm your product and the whole process.

As hard as your team is at work daily, they spend a sufficient amount of time interacting with each other. And if your team members can clearly communicate their needs and project demands to one another, it can boost their collaboration and improve the work process overall. It is also capable of driving creativity and innovation inside the team. It comes in handy when discussing the project's details and requirements with stakeholders, negotiating timelines and generally, when trying to understand what is it that the stakeholders want.

Great communication skills are an important soft skill for any software engineer. But you should treat their ability to communicate as a given. Your task is to nurture and encourage the communication inside your software development team and make it a part of your normal process by practicing daily standups, design, and code reviews, writing documentation, presentations and also some social events.

13.2.2 Decision Making

What Is Agile Decision-Making in Project Management?

Information is available faster and from more sources than ever before, which makes decision-making *seem* easier.

But if you combine this information onslaught with traditional types of business decision-making, it will make your process slower and more frustrating. A new means of gathering input and data calls for a new way of making decisions: **agile decision-making**.

Making decisions in an agile way means working iteratively, collaboratively, and with transparency. Pretty different from the older trend of having a task force craft a solution, the project manager presents it, and executives come to a consensus, right? In agile project teams, managers, leaders, and executives need to release the reins and empower project teams to make decisions.

What is agile decision-making?

As stated above, agile decision-making is a process that is collaborative, iterative, and transparent. It means all stakeholders are updated on assigned tasks at regular intervals, they give feedback, and then the team knows what needs to be changed or improved. The team discusses issues together and comes up with a solution together.

Agile decision-making doesn't mean rushed, on-the-fly changes at the whim of one project stakeholder. It also doesn't mean that the project team makes all the decisions amongst themselves and then hands over a final product at the end.

Why you should adopt agile decision-making

Maintaining a course of action because that's what has been successful for you in the past is no longer enough to be competitive. It can even lead to your downfall.

For example, Kodak infamously played it safe, underestimated the pace of change, and paid the price. Of course, their bankruptcy was rooted in compounding issues, but their late adoption of digital transformation and lack of taking chances on new products were driving forces. They took the mindset that you don't fix what isn't broken and were left behind by competitors that were willing to try new things and new ways to operate.

But don't equate agile project management with being as fast as possible and blowing up a budget with overtime to complete endless requirements. Working in an agile method means making iterative progress and changes as an empowered and self-organizing team.

Empower the project team to deliver what the customer wants (or senior leadership) in the manner they decide is the most efficient and successful. They won't go down some rabbit hole, wasting hours or days, because an agile team checks in with the customer on a regular basis.

Five tips for agile decision-making

Iterative delivery and feedback are a key component in agile projects. Though this approach makes sense to businesses—adjust and make changes when needed, easy enough right?—it's the most difficult aspect of the agile method for people to get comfortable with. Here are a few tips to help you and your team be successful in agile decision-making during a project.

1. Gather iterative feedback

When you're showing the work and getting feedback regularly, it reduces the chance of a major disruption that requires a complex decision-making technique with a formal committee. The team can make incremental pivots and adjustments as needed, without formal oversight or stress about deviating from a lengthy process document.

2. Balance alignment and autonomy

Your agile team needs to be empowered and well-informed when making decisions. But this doesn't mean it can deviate from what the goals of the project are. Leaders often don't give teams the autonomy needed because they fear the team going rogue and producing something completely off the wall.

An agile team will be showing their progress on a regular basis and getting feedback for changes. So, for example, if a construction company tasks a team to build a new website, an agile team won't decide one sprint that the website would be better if you could book

travel arrangements on it instead and then proudly hand over a new “cheap flights” site to the construction manager.

3. Get comfortable with good enough

One huge mindset change that is needed is getting comfortable with “good enough.” The requirements are documented “good enough.” The timing to start is “good enough.”

Then, you can work toward “great” during the execution process. Yes, there will be work scrapped while sharpening what “great” looks like and that’s OK. When the whole team is accountable for the project’s success, it won’t be such an emotional hit if some of one person’s work must be cut because of time. The team will support its members.

4. Place time limits on decisions

As project managers, we love timelines and due dates. And agile decision-making needs timelines. Set deadlines for when analysis must be cut off—is good enough. This helps alleviate analysis paralysis because you’re forced to move forward with a decision when it’s good enough rather than when it’s perfect.

5. Don’t get sloppy

Plan project meetings at a regular cadence to give updates, brainstorm for improvements, plan the next block of tasks for a sprint, and get iterative feedback on what’s been delivered. This cadence is what builds trust with senior leadership, improves the strength of the team, and builds the team’s confidence in their decision-making skills.

13.2.3 Organization and Team Structures

What Is an Organizational Structure?

An organizational structure is a system that outlines how certain activities are directed to achieve the goals of an organization. These activities can include rules, roles, and responsibilities.

The organizational structure also determines how information flows between levels within the company. For example, in a centralized structure, decisions flow from the top down, while in a decentralized structure, decision-making power is distributed among various levels of the organization.

Having an organizational structure in place allows companies to remain efficient and focused.

Understanding an Organizational Structure

Businesses of all shapes and sizes use organizational structures heavily. They define a specific hierarchy within an organization. A successful

organizational structure defines each employee's job and how it fits within the overall system. Put simply, the organizational structure lays out who does what so the company can meet its objectives.

This structuring provides a company with a visual representation of how it is shaped and how it can best move forward in achieving its goals. Organizational structures are normally illustrated in some sort of chart or diagram like a pyramid, where the most powerful members of the organization sit at the top, while those with the least amount of power are at the bottom.

Not having a formal structure in place may prove difficult for certain organizations. For instance, employees may have difficulty knowing to whom they should report. That can lead to uncertainty as to who is responsible for what in the organization.

Having a structure in place can help with efficiency and provide clarity for everyone at every level. That also means each and every department can be more productive, as they are likely to be more focused on energy and time.

Centralized vs. Decentralized Organizational Structures

An organizational structure is either centralized or decentralized. Traditionally, organizations have been structured with centralized leadership and a defined chain of command. The military is an organization famous for its highly centralized structure, with a long and specific hierarchy of superiors and subordinates. In a centralized organizational system, there are very clear responsibilities for each role, with subordinate roles defaulting to the guidance of their superiors.

There has been a rise in decentralized organizations, as is the case with many technology start-ups. This allows companies to remain fast, agile, and adaptable, with almost every employee receiving a high level of personal agency. For example, Johnson & Johnson is a company that's known for its decentralized structure. As a large company with over 200 business units and brands that function in sometimes very different industries, each operates autonomously. Even in decentralized companies, there are still usually built-in hierarchies (such as the chief operating officer operating at a higher level than an entry-level associate). However, teams are empowered to make their own decisions and come to the best conclusion without necessarily getting "approval" from up top.

Types of Organizational Structures

Functional Structure

Four types of common organizational structures are implemented in the real world. The first and most common is a functional structure. This is also referred to as a bureaucratic organizational structure and breaks up a company based on the specialization of its workforce. Most small-to-medium-sized businesses implement a functional structure. Dividing the

firm into departments consisting of marketing, sales, and operations is the act of using a bureaucratic organizational structure.

Divisional or Multidivisional Structure

The second type is common among large companies with many business units. Called the divisional or multidivisional structure, a company that uses this method structures its leadership team based on the products, projects, or subsidiaries they operate. A good example of this structure is Johnson & Johnson. With thousands of products and lines of business, the company structures itself, so each business unit operates as its own company with its own president.

Flatarchy Structure

Flatarchy, a newer structure, is the third type and is used among many startups. As the name alludes, it flattens the hierarchy and chain of command and gives its employees a lot of autonomy. Companies that use this type of structure have a high speed of implementation.

Matrix Structure

The fourth and final organizational structure is a matrix structure. It is also the most confusing and the least used. This structure matrixes employees across different superiors, divisions, or departments. An employee working for a matrixed company, for example, may have duties in both sales and customer service.

Benefits of Organizational Structures

Putting an organizational structure in place can be very beneficial to a company. The structure not only defines a company's hierarchy but also allows the firm to layout the pay structure for its employees. By putting the organizational structure in place, the firm can decide salary grades and ranges for each position.

The structure also makes operations more efficient and much more effective. By separating employees and functions into different departments, the company can perform different operations at once seamlessly.

In addition, a very clear organizational structure informs employees how best to get their jobs done. For example, in a hierarchical organization, employees will have to work harder at buying favor or courting those with decision-making power. In a decentralized organization, employees must take on more initiative and bring creative problem solving to the table. This can also help set expectations for how employees can track their own growth within a company and emphasize a certain set of skills—as well as for potential employees to gauge if such a company would be a good fit with their own interests and work styles.

There are many ways to organize the project team. Some important ways are as follows :

- Hierarchical team organization
- Chief-programmer team organization
- Matrix team, organization
- Egoless team organization
- Democratic team organization

Hierarchical team organization

In this, the people of organization at different levels following a tree structure. People at bottom level generally possess most detailed knowledge about the system. People at higher levels have broader appreciation of the whole project.

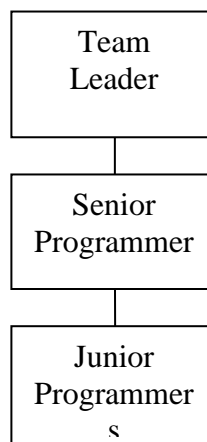


Figure 13.2 Hierarchical team organization

Benefits of hierarchical team organization :

It limits the number of communication paths and stills allows for the needed communication.

It can be expanded over multiple levels.

It is well suited for the development of the hierarchical software products.

Large software projects may have several levels.

Limitations of hierarchical team organization :

As information has to be travel up the levels, it may get distorted.

Levels in the hierarchy often judges people socially and financially.

Most technical competent programmers tend to be promoted to the management positions which may result in loss of good programmer and also bad manager.

Chief-programmer team organization :

This team organization is composed of a small team consisting of the following team members :

The Chief programmer : It is the person who is actively involved in the planning, specification, and design process and ideally in the implementation process as well.

The project assistant : It is the closest technical co-worker of the chief programmer.

The project secretary : It relieves the chief programmer and all other programmers of administration tools.

Specialists : These people select the implementation language, implement individual system components, and employ software tools and carry out tasks.

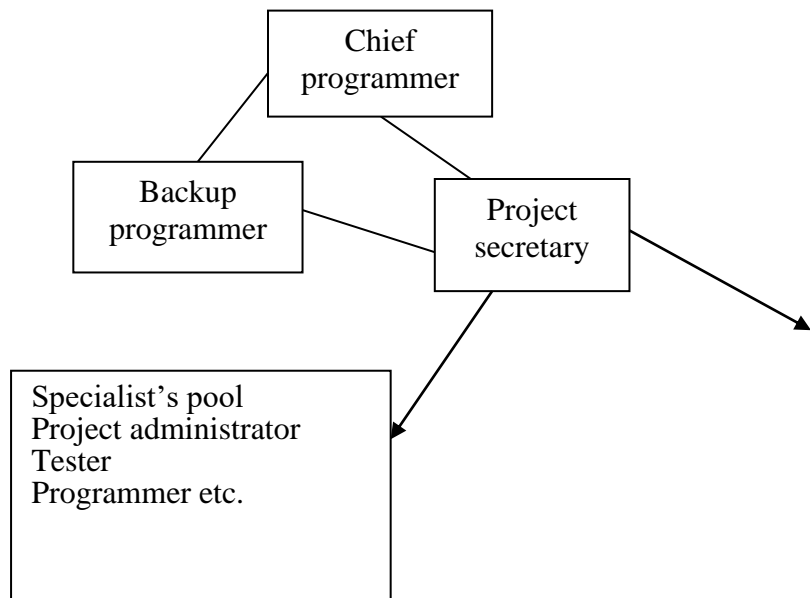


Figure 13.3 Chief-programmer team organization

Advantages of Chief-programmer team organization :

Centralized decision-making

Reduced communication paths

Small teams are more productive than large teams

The chief programmer is directly involved in system development and can exercise the better control function.

Disadvantages of Chief-programmer team organization :

Project survival depends on one person only.

Can cause the psychological problems as the “chief programmer” is like the “king” who takes all the credit and other members are resentful.

Team organization is limited to only small team and small team cannot handle every project.

Effectiveness of team is very sensitive to Chief programmer’s technical and managerial activities.

Matrix Team Organization :

In matrix team organization, people are divided into specialist groups. Each group has a manager. Example of Matrix team organization is as follows :

Egoless Team Organization :

Egoless programming is a state of mind in which programmer are supposed to separate themselves from their product. In this team organization goals are set, and decisions are made by group consensus. Here group, ‘leadership’ rotates based on tasks to be performed and differing abilities of members.

In this organization work products are discussed openly, and all freely examined all team members. There is a major risk which such organization if teams are composed of inexperienced or incompetent members.

Democratic Team Organization :

It is quite like the egoless team organization, but one member is the team leader with some responsibilities :

Coordination

Final decisions when consensus cannot be reached.

Advantages of Democratic Team Organization :

Each member can contribute to decisions.

Members can learn from each other.

Improved job satisfaction.

Disadvantages of Democratic Team Organization :

Communication overhead increased.

Need for compatibility of members.

Less individual responsibility and authority.

Software development team structure

1. Project Manager

The project manager plays the main role in managing, planning, implementing, controlling, and closing the project. The project manager guides the team through different phases of the project and is also responsible for ensuring that it runs on budget, on time and within scope.

2. Architect

The architect, or software architect, oversees designing and developing the software product. This individual determines the technical standard and design, as well as other high-level decisions related to the project. The architect takes care of the systems, solutions, database, security, and integrations involved in the product development.

3. Designer

This can be a UI or UX designer, and their role is to ensure overall user satisfaction of the product. The designer creates user-friendly interfaces for the software application and takes care of the branding, usability, design and function of the product.

4. Developer/s

The type and number of developers in a team depends largely on the type of project. However, most projects require an API, mobile, backend and frontend developer. The developer will identify, design, install and test the software product according to the final design.

5. Quality Assurance (QA) Engineer

You can either have a manual QA Engineer or engage an automated performance test after creating the software product. Either way, the QA Engineer oversees checking the quality of the product and giving helpful feedback.

Before setting up your software development teams structure, it's always important to troubleshoot your current team structure. If you don't know the problem with your current team structure, you won't know how to improve it or rearrange it so it can be effective.

When businesses set up their development team, they usually make common mistakes that end up reducing its effectiveness. For example, when setting up a development team for a project with a new strategy or purpose, many businesses end up trying to work with their old structure even if it does not suit or serve the new purpose.

Another common mistake businesses make is employing individuals in multiple roles without understanding the responsibilities for each unique role.

Based on these problems, here are some tips to ensure an effective development team structure:

- **Ensure the structure is relevant to the project**

Consider the time, complexity and budget of your project before giving attention to setting up a team. As mentioned earlier, a team structure for a previous development project may not work effectively for a brand-new project with different goals. The structure should perfectly fit the new project's aim.

- **Define and establish the size of the team**

The first step in setting up a development team is to define and establish the size of the group. First, identify exactly what needs to be done to complete the project; you will then have a better idea of the size of the team you require.

- **Establish leadership and define clear roles**

One of the main elements in a team is its leadership. It is vital to establish the leadership, as well as other individual roles in the first stage of structuring the team. This will avoid confusion and miscommunication during the working stages.

- **Match responsibilities to individuals**

Once roles are defined, everyone's tasks and responsibilities should be established. Having individual responsibilities will help the team to stay focused and aid in building a solid structure.

- **Put workflow processes in place**

Teams cannot operate optimally if they are not completely sure about processes. Put road maps, communication funnels and smaller targets in place to ensure smooth workflows.

- **Avoid a large and complex team**

Large teams can quickly become difficult to manage and roles and responsibilities can easily become blurry. Smaller software development teams are easier to coordinate and allow you to identify good or bad efforts more efficiently. For large-scale projects, you can divide your team into sub-teams and place team leaders on each one.

- **Use project management software**

These days, software can save you a lot of time and effort and take some strain off your development team. When using project management software, time consuming tasks can be automated, and your team can better keep track of their individual responsibilities. You can also include communication apps for better interaction. This is specifically effective when working with remote teams.

- **Have a good reporting structure**

Quality reporting is at the core of any project's final success. Reporting reflects how effective your team works together and helps you to keep track of the progress of your project. Reporting can be done by individual members or by the project members on a daily, weekly, or monthly basis.

Once you understand how to troubleshoot your current team and have a better idea of how to set up a new structure, you are ready to put your team together. To ensure you have access to the best talent, it is advisable to make use of staff augmentation services. Mobilunity is an experienced vendor who can assist with team augmentation.

13.2.4 Coordination Dependencies

What is coordination in a team?

Image result for Coordination among team in software project management

Team coordination refers to the processes and strategies organizations use to help their teams collaborate more effectively on their individual and collective goals. ... Getting a team (or multiple teams) of people with unique feelings, ideas, and ways of doing things to work together seamlessly is not a small task

There are two types of team coordination: explicit and implicit.

Many of the strategies that leaders implement to drive effective team coordination fall into the “**explicit coordination**” bucket. These include things like building effective work processes, delegation, planning, and direct communication. You can think of these as things leaders actively do to organize their company.

However, while explicit coordination has been studied for decades, research has only recently started to explore “implicit coordination.”

Implicit coordination refers to the ways teams adapt to the needs of their organization and other departments proactively on their own, apart from direction from superiors or others. An example of it could be your sales team sending customer feedback proactively to your product team without being asked.

Needless to say, implicit coordination is kind of the holy grail for leaders who are trying to get their teams to work together seamlessly. If you can get your teams to anticipate the needs of other teams with little input or motivation from you, it makes everything easier.

A dependency describes the relationship among activities and specifies the order in which they need to be performed. Dependencies arise in every decision making, planning, and developing process and are ideally predetermined. Tasks can be successors and predecessors to other tasks whereby the instant of each execution may be aligned accordingly.

There are 4 types of dependencies in project management viz. Mandatory, Discretionary, External, & Internal

Mandatory Dependency

Mandatory Dependencies are either required by the law or contract and sometimes they are inherent in the nature of the work. Due to these, the work must be done in a certain order. They are also called Hard Logic.

As an example, consider 2 activities A and B. If B has a Mandatory Dependency on A then it means action on B cannot be performed until Action on A has been completed. Let us look at following examples to understand:

A: Requirements Documentation; B: Client Approval – Client cannot approve requirements until documentation is complete

A: Lay Building Foundation; B: Construct a Floor – A floor of a building cannot be constructed until foundation is laid.

A: Build Car Prototype; B: Perform Crash Testing – Crash testing on a care prototype cannot be performed unless the prototype itself is available.

Discretionary Dependency

Discretionary Dependencies are defined by the Project Team as a certain order of activities is more suitable for the nature of work. These are also called Preferred Logic, Preferential Logic, or Soft Logic.

Sometimes, there is more than one way to define a sequence between 2 activities, but the Project Team decides to take one sequence over the

other. They choose a particular sequence because of the best practices or lessons learned from prior experiences.

As an example, consider 2 activities A and B. A and B can be independently performed, or one can be performed after the other. The Project Team can choose to make B dependent on A. Let us look at following examples to understand:

A:-Develop System Module X; B: Develop System Module Y – The project team can develop either X first or Y first, but they decide to develop X first.

A: Furnish Room R; B: Furnish Room S – The project team can furnish either room R first or S first, but they decide to furnish S first.

A: Book Airline Ticket; B: Buy Travel Insurance – The project team can book an airline ticket before buying travel insurance or do it other way around.

External Dependency

External Dependencies are defined between project activities and non-project activities. The project activities are done by the Project Team. The non-project activities are done by people who are external to the Project Team e.g., representatives from Client's organization, Vendors' organization, or any other external groups within the same organization.

Generally, the Project Team has no control over non-project activities and hence the project schedule can be disrupted because of nonperformance of non-project activities.

As an example, consider 2 activities A and B. If B has an External Dependency on A then it would signify that B is a project activity while A is a non-project activity. Following examples will be helpful in understanding:

A: Client Go-Ahead; D: Initiate Project: A project cannot be initiated before the client gives a go-ahead.

A: Delivery of Equipment; D: Start Development: Project development cannot start until equipment is delivered.

A: Approval of Building Plans; D: Start Construction: Construction of a building cannot be started unless the building plans are approved.

Internal Dependency

Internal Dependencies are defined between two project activities. Generally, the Project Team has complete control over the project activities.

As an example, consider 2 activities A and B. If B has an Internal Dependency on A, then it would signify that both A and B are project activities. These are performed by the Project Team members. There is no

involvement of any external party. Let us look at following examples to understand:

A: Develop System; B: Test System

A: Construct Wall; B: Paint Wall

A: Assemble Machine; B: Pack Machine

Mandatory vs Discretionary and External vs Internal Dependencies

There are a total of 4 Types of Project Dependencies. However, only 2 are applicable at the same time. A dependency between 2 activities could be any one of the following:

Mandatory-External

Discretionary-External

Mandatory-Internal

Discretionary-Internal

Two activities cannot have Mandatory and discretionary at the same time. Similarly External and Internal cannot happen at the same time.

For example, the first example from the mandatory dependencies section above is also an external dependency, whereas the second example is an internal dependency.

13.2.5 Dispersed and Virtual Teams

A **virtual team** (also known as a geographically **dispersed team**, **distributed team**, or **remote team**) usually refers to a group of individuals who work together from different geographic locations and rely on communication technology such as email, instant messaging, and video or voice conferencing services to collaborate. The term can also refer to groups or teams that work together asynchronously or across organizational levels. Powell, Piccoli and Ives (2004) define virtual teams as "groups of geographically, organizationally and/or time dispersed workers brought together by information and telecommunication technologies to accomplish one or more organizational tasks. According to Ale Ebrahim et. al. (2009), virtual teams can also be defined as "small temporary groups of geographically, organizationally and/or time dispersed knowledge workers who coordinate their work predominantly with electronic information and communication technologies to accomplish one or more organization tasks.

A virtual team is a team where the primary method of interaction is done through electronic mediums. When it comes to the medium, it could range from e-mail communications to video conferencing.

Some virtual teams do not interact face-to-face (when team members reside in different demographics) and some virtual teams physically meet up occasionally.

Think of an online business for web development. Someone can start such a business and hire developers, QA engineers, UI engineers and project managers from different parts of the globe.

Since web development does not involve in physical delivery of goods and all the deliveries are done electronically, such a company can exist on the Internet.

Team meetings can be held through conference voice calls or video calls. This virtual team can work towards their company goals and act as a single entity just by telecommuting.

Why Virtual Teams?

There are many reasons for having a virtual team. First, it is the technology.

The Internet and related technologies helped enhancing the communication across the globe, where certain industries that do not require the person to be present in physical sense could make much use of it. A good example is a web development team.

Following are some of the top reasons for having virtual teams:

- Team members are not located in the same demography.
- The transportation cost and time is quite an overhead.
- Team members may work in different times.
- The company does not require a physical office, so the logistics and related costs are minimum.
- The type of work done may require high level of creativity, so the employees will have better creativity when they work from a place, they are comfortable with (home).

Types of Virtual Teams

There are many types of virtual teams operating at present. Following are a few of those teams:

- Entire companies that operate virtually
- Tasks teams, responsible of carrying out a specific task
- Friendship teams such as groups in Facebook or any other social network
- Command teams, such as a sales team of a company distributed throughout the US
- Interest teams where the members share a common interest

The Technology

The technology plays a vital role for virtual teams. Without the use of advanced technology, virtual teams cannot be effective.

The Internet is the primary technology used by the virtual teams. The Internet offers many facilities for the virtual teams. Some of them are:

- E-mail
- VoIP (Voice Over IP) - voice conferencing
- Video conferencing
- Groupware software programs such as Google Docs where teams can work collaboratively.
- Software for conducting demonstrations and trainings such as Microsoft Live Meeting and WebEx.

When it comes to the technology, not only the software matters, but the virtual teams should also be equipped with necessary hardware as well.

As an example, for a video conference, the team members should be equipped with a web camera and a microphone.

Advantages and Disadvantages

First, let's look at the **advantages** of operating as a virtual team.

- Team members can work from anywhere and anytime of the day. They can choose the place they work based on the mood and the comfort.
- You can recruit people for their skills and suitability to the job. The location does not matter.
- There is no time and money wasted for commuting and clothing.
- Physical handicaps are not an issue.
- The company does not have to have a physical office maintained. This reduces a lot of costs to the company. By saving this money, the company can better compensate the employees.

disadvantages of using virtual team:

- Since team members do not frequently meet or do not meet at all, the teamwork spirit may not be present.
- Some people prefer to be in a physical office when working. These people will be less productive in virtual environments.
- To work for virtual teams, individuals need to have a lot of self-discipline. If the individual is not disciplined, he or she may be less productive.

13.2.6 Communication Genres

Communications genres:

- Communication is a critical factor in project management. There are instances where projects have failed because of miscommunication and communication gaps.

- Project managers fill this gap by devising a good communication mechanism that will help him to communicate with the team members as well as stakeholders, sponsors, top-tier management and all the people who are connected to the project.
- If an effective communication methodology is not followed by the project manager, it may lead to many discrepancies and ultimately may also lead to project failure, which is not appropriate for the organization.
- It is also important that the right information is delivered to the right person.
- So, project managers have the responsibility to properly channelize the communication process, so that the right persons receive the right information.
- Another important point that project managers must make a note of is that the information sent must be clear, concise, and informative.

13.2.7 Communication Plans

What is a communication plan?

In project management, a communication plan is an outline of how you're going to communicate important, ongoing project information to key stakeholders. Your communication plan will help your team understand who should be getting which notifications and when to loop in project stakeholders. As part of your communication plan, you'll clarify which channel stakeholders should use and when, how frequently different details should be communicated, and who is responsible for each of the different channels.

Sharing a communication plan can give your team clarity about which tools to use when and who to contact with each of those tools. Without a communication plan, you might have one team member trying to ask questions about work in a tool that another team member rarely checks. Rather than being able to clearly communicate and move forward with work, each team member would end up frustrated, confused, and disconnected from the work that matters. Then, if they don't have clear insight into who is responsible for each channel, they might end up reaching out to an executive stakeholder with questions that person can't answer. What started out as a simple miscommunication has spiraled into three frustrated team members—and all the while, work isn't moving forward.

The benefits of a communication plan

Poor communication contributes to project failure, and therefore, it could spell massive financial loss to the company. At the opposite end of the spectrum, high-performing businesses communicate more frequently and do so more effectively than their low-performing counterparts.

A project management communication plan will keep your project on track because it:

- Creates written documentation that the team can reference
- Sets expectations of when stakeholders will receive updates
- Increases stakeholders' visibility into the project and its status
- Provides the opportunity for stakeholders to give feedback, which can help the team detect issues early on and decrease wasted work
- Increases productivity during meetings or eliminates them altogether

So, if you want your project to be completed successfully and on time, make sure you know how to create an effective communication plan.

How to make a project management communication plan?

Based on the benefits explained above, we're sure you're anxious to start your own project management communication plan. Follow these steps to get started.

1. Choose a format

Choose a platform where it will be easy to gather feedback on your communication plan and to share or store the plan for your team and stakeholders to reference.

Many project managers create their communication plan on a word document or a spreadsheet, starting from a project communication plan template, but you might also consider choosing a more visual option, such as a timeline or a flowchart, to clearly explain the frequency of communication or the best method to use based on the stakeholder.

2. Set a communication goal

Whatever you hope to achieve, the first step to crafting a successful communication plan is to write that goal down. Referring back to the importance of a communication plan, your goal will likely be to keep stakeholders updated on the project status or even to keep stakeholders mindful of the project's benefits, so they'll continue to advocate for it.

3. Identify stakeholders

Most projects have many stakeholders, most of whom have different levels of interest in and influence on the project. You'll need to identify the stakeholders with whom you'll communicate throughout the project and list them.

4. Identify methods of communication

Your CTO never checks his email but is on Slack all day. On the other hand, your head designer never installed Slack but checks her email constantly. And you'll need to hire a skywriter to communicate with your art director.

One purpose of your communication plan is to get the right eyes on the right information, so along with listing who your stakeholders are, your communication plan should also list how you intend on communicating with those stakeholders.

Consider the following methods depending on what your stakeholders are most likely to see or attend:

- Weekly check-ins
- Meetings, whether in person, over the phone, or through video conferencing
- Meeting summaries
- Status reports
- Formal presentations
- Surveys
- To-do lists
- Project dashboards
- Collaboration apps, such as Slack or Google Hangouts

The communication method you choose may also depend on the information you need to deliver. You likely don't need a formal in-person meeting every week to share updates on the project; you could send a weekly email with updates and hold meetings when the team reaches a major milestone.

5. Determine frequency of communication

List how often you will send out each type of communication (e.g., send a weekly email on Mondays with project progress, links to completed deliverables, current budget, etc.) or how often you need to loop in each stakeholder (e.g., each team member should send daily emails to update the project manager but only include the executive stakeholder on the video conference following each milestone).

In addition to including this information on your project management communications plan, make sure to schedule communication frequency on your calendar or into your task management software.

6. Determine who provides communication updates

Most often, this task will fall on the project manager, but if not, the owner of a specific update needs to be clearly identified in your communications plan.

13.2.8 Leadership

Leadership means directing, motivating and organizing groups of people for performing the set tasks. Leaders must have the quality to lead the groups of people. They must have the capability to inspire others and must make them perform the tasks timely.

Project leadership plays a very important part in project management. Each leader has owned leadership style and the style differs from person to person, which depends upon their experiences, philosophies and their personalities. Leadership styles is the behavior of the leaders towards the team they are leading. The different kinds of leadership style will have different effects on the environment and the output produced by the team.

There are different leadership styles that a project leader must have, to be effective in their role in leading the teams. The different kinds of leadership styles are:

- ***The Autocratic Leadership Style:*** The leaders have all the authority, and they give instructions to the team what all tasks have to be done and how it is to be achieved.
- ***The Democratic Leadership Style:*** In this style the leaders focus to build group consensus and commitments in the team management of decision-making processes. The leaders participate with the team and greatly motivate the team members which is very important for the organization in achieving their goals.
- ***The Coercive Leadership Style:*** This style is used by the project leaders when they issue orders in the manner when there is only one direction of communication. The leaders give a direct order to the team, and they have strong control over the situations.
- ***The Authoritative Leadership Style:*** The leaders share the vision with the team and allows the team to give their input in the decisions. The leaders motivate the team members and the team's contribution is given greater value.
- ***The Affiliative Leadership Style:*** This style is used by the leader to encourage every member, promote cooperation, and team harmony.
- ***The Pacesetting Leadership Style:*** The leadership style in which high performance standards are set and the team members have the capability to achieve the goals with less supervision.
- ***The Coaching Style:*** In this style the leaders teach and allows the team in identifying the strengths and weaknesses. The team is given full support and encouragement, and the mistakes are considered as learning opportunities in the development processes.
- ***The Transformational Leadership Style:*** This kind of style is helpful when the organization is going through some changes and the team members are being replaced. At this stage the leaders can

motivate the team by creating new styles and inspire them with enthusiasm and positive styles.

- ***The Free-rein Leadership Style:*** The leaders do not direct the team members on how to perform a task. They only assign the tasks and the time period by which it has to be achieved. The team has complete freedom to make decision on policies and methodologies to be adopted for the goal achievement.

Thus, the good leaders are those leaders who use the styles proficiently according to the situation.

13.3 SUMMARY

In this Working in Teams, you learned about what is team and how to Building an Effective Software Development Team?, What Is Agile Decision-Making in Project Management?, Organization and Team Structures and Types of Organizational Structures, Software Project Team Organization ways and software development team structure, What is coordination in a team ? and **types of dependencies in project management**, Dispersed and Virtual Teams and its types and advantages and disadvantages , definition of **Communications genres** , What is a communication plan? its benefits, How to make a project management communication plan?, Leadership In Project Management and its styles etc. Still, you had a doubt go through references and bibliography

13.4 EXERCISE

- Q.1 Write Five Steps to Building an Effective Software Development Team and types of teams
- Q.2 What Is Agile Decision-Making in Project Management?
- Q.3 What is agile decision-making?
- Q.4 Explain Software Project Team Organization?
- Q.5 What Is an Organizational Structure? And Types of Organizational Structures
- Q.6 Explain software development team structure
- Q.7 Explain coordination and dependency and types of dependency
- Q.8 Explain virtual teams
- Q.9 What are the Types of Virtual Teams
- Q.10 Explain **Communications genres**
- Q.11 What is a communication plan?

Q.12 Explain The benefits of a communication plan

Q.13 How to make a project management communication plan?

Q.14 Explain leadership and its styles

13.5 LIST OF BOOKS AND REFERENCES

1. Software Project Management Bob Hughes, Mike Cotterell, Rajib Mall TMH 6 th 2018
2. Project Management and Tools & Technologies – An overview Shailesh Mehta SPD 1st 2017
3. Software Project Management Walker Royce Pearson 2005
4. <https://perfectial.com>
5. <https://www.geeksforgeeks.org>
6. <https://www.softwareadvice.com>
7. <https://www.investopedia.com>
8. <https://www.pmbypm.com>
9. <https://www.tutorialspoint.com>
10. <https://en.wikipedia.org>
11. <https://citizenchoice.in>
12. <https://www.lucidchart.com>
13. <https://asana.com/resources>
14. <https://www.whizlabs.com>

SOFTWARE QUALITY

Unit Structure

14.0 Objectives

14.1 Introduction

14.2 An Overview

14.2.1 The Place of Software Quality in Project Planning

14.2.2 Importance of Software Quality

14.2.3 Defining Software Quality

14.2.4 Software Quality Models

14.2.5 ISO 9126

14.2.6 Product and Process Metrics

14.2.7 Product versus Process Quality Management

14.2.8 Quality Management Systems

14.2.9 Process Capability Models

14.2.10 Techniques to Help Enhance Software Quality

14.2.11 Testing, Software Reliability

14.2.12 Quality Plans

14.3 Summary

14.4 Exercise

14.5 List of Books and References

14.0 OBJECTIVES

After going through this chapter, you will be able to know:

- The Place of Software Quality in Project Planning
- Importance of Software Quality, Defining Software Quality, Software Quality Models, ISO 9126
- Product and Process Metrics What is process and product metrics? Types of Metrics Advantages and disadvantages of Software Metrics

- Product versus Process Quality Management What is product quality and process quality? Software quality management Quality Management Systems, Process Capability Models Techniques to Help Enhance Software Quality
- Testing and type of testing, different techniques of Software Testing, Software Reliability, Quality Plans

14.1 INTRODUCTION

Quality software refers to a software which is reasonably bug or defect free, is delivered in time and within the specified budget, meets the requirements and/or expectations, and is maintainable. In the software engineering context, software quality reflects both **functional quality** as well as **structural quality**.

- **Software Functional Quality** – It reflects how well it satisfies a given design, based on the functional requirements or specifications.
- **Software Structural Quality** – It deals with the handling of non-functional requirements that support the delivery of the functional requirements, such as robustness or maintainability, and the degree to which the software was produced correctly.
- **Software Quality Assurance** – Software Quality Assurance (SQA) is a set of activities to ensure the quality in software engineering processes that ultimately result in quality software products. The activities establish and evaluate the processes that produce products. It involves process-focused action.
- **Software Quality Control** – Software Quality Control (SQC) is a set of activities to ensure the quality in software products. These activities focus on determining the defects in the actual products produced. It involves product-focused action.

14.2.1 The Place of Software Quality in Project Planning

Software quality is a nuanced concept that requires a framework that addresses functional, structural and the process of the software delivery understand. Measurement of each aspect is a key tool for understanding whether we are delivering a quality product and whether our efforts to improve quality are having the intended impact. However, measurement can be costly. To balancing the effort required to measure quality versus the benefit, you first need to understand the reasons for measuring quality. Five of reasons quality is important to measure include:

1. **Safety** – Poor quality in software can be hazardous to human life and safety. Quality problems can impact the functionality of the software products. Jon Quigley discussed the impact of configuration defects that effected safety in SPaMCAST 346.

2. Cost – Quality issues cost money to fix. Whether you believe that a defect is 100x more costly to fix late in the development cycle or not, doing work over because it is defective does not deliver more value than doing it right once.
3. Customer Satisfaction (internal) – Poor quality leads stakeholders to look for someone else to do your job or perhaps shipping your job and all your friend’s jobs somewhere else. Recognize that the stakeholders experience as the software is being developed, tested and implemented is just as critical as the raw number of defects.
4. Customer Satisfaction (external) – Software products that don’t work, are hard to use (when they don’t need to be) or are buggy don’t tend not to last long in the marketplace. Remember that in today’s social media driven world every complaint that gets online has a ripple effect. Our goal should be to avoid problems that can be avoided.
5. Future Value – Avoiding quality problems increases the amount of time available for the next project or the next set of features. Increasing quality also improves team morale, improved team morale is directly correlated with increased productivity (which will increase customer satisfaction and reduce cost).

14.2.2 Importance of Software Quality

The Seven Aspects(importance) of Software Quality

Software engineering is a complex field. Good software engineers are capable of balancing opposing forces and working within constraints to create great software. Poor software developers (they really aren’t engineers) are ones who are incapable of perceiving the trade-offs they are making and the implications of their design decisions (or lack thereof).

Every software engineer absolutely must know the seven aspects of software quality:

- Reliability
- Understandability
- Modifiability
- Usability
- Testability
- Portability
- Efficiency

These seven aspects can be measured and judged for any software product. They apply to embedded systems, websites, mobile apps, video games, open-source APIs, internal services, and any other sort of software product. These aspects are entirely business domain independent. A software engineer’s ability can be measured by the quality of the software he creates. Skilled engineers create high-quality software and source code.

For different projects, the prioritization of the aspects of quality will vary. Some projects should be focused on reliability, usability, and understandability, while other projects will place high importance on testability and efficiency. As a software engineer, you must know which aspects of quality are most important to your project. You must apply your best efforts towards the most critical aspects, and not spend excessive time on less important aspects.

Reliability: Software is reliable if it behaves consistently. The functionality of a program should be predictable and repeatable. Errors should occur rarely or not at all. Errors that do occur should be handled gracefully and proactively. Users should never ask themselves whether the software will work correctly.

Understandability: The structure, components, and source code must be understandable. They must be clear. They must be well-organized. They must behave the way a developer would expect. Anything in the code that causes developers confusion reveals that the code is lacking in understandability. High-quality source code always appears simple and obvious.

Modifiability: It should be easy to add or change the behavior of a system. Flexible systems require changing very few lines of code to alter a behavior. For the expected dimensions of change, an application should have plugin points that allow the application to be used with different contextual elements. Tight coupling to an element that is expected to change is unacceptable.

Usability: Software products must be simple and easy to use. The common use cases should be as obvious and clearly presented as possible. Software should not require excessive configuration. Users should feel empowered by your software. They should not need internet searches to discover core application functionality.

Testability: The functionality of software must be verifiable. The process of testing the software must be easy. Each business use case should be directly testable. Clear verification metrics must be available. Highly testable software will ship with a comprehensive automated test suite.

Portability: Portable software is usable in different environments and contexts. It is highly reusable. Portable software is decoupled from specific operating systems, types of hardware, and deployment contexts. Extremely portable software is reusable across projects and problem domains.

Efficiency: Efficient software uses as few physical resources as possible. It is fast. It is memory efficient. It consumes few CPU cycles. It uses little battery life. It makes few external service calls. It minimizes the number of database calls. Efficient software accomplishes as much as possible with the least number of resources.

you absolutely must know these seven aspects of software quality. You must know which of the seven aspects are most important and least important in your current projects. Your code reviews should reference these aspects. Your design meetings and discussions should explicitly involve these aspects. You must know when you are sacrificing one of these dimensions to improve another dimension. You must cultivate a deep awareness of software quality. It should inform and guide your designs.

Indeed, nearly all the software practices, patterns, and methodologies that have been created in recent years are attempts to increase software quality in one or more of these dimensions. The best practices and methodologies are ones that improve multiple aspects (SOLID, TDD, XP, DDD). Those are means to an end. You must know the goal to adequately judge their effectiveness. Quality software is the goal.

14.2.3 Defining Software Quality

What is software quality?

The quality of software can be defined as the ability of the software to function as per user requirement. When it comes to software products it must satisfy all the functionalities written down in the SRS document.

Key aspects that conclude software quality include,

- Good design – It's always important to have a good and aesthetic design to please users
- Reliability – Be it any software it should be able to perform the functionality impeccably without issues
- Durability- Durability is a confusing term, In this context, durability means the ability of the software to work without any issue for a long period of time.
- Consistency – Software should be able to perform consistently over platform and devices
- Maintainability – Bugs associated with any software should be able to capture and fix quickly and new tasks and enhancement must be added without any trouble
- Value for money – customer and companies who make this app should feel that the money spent on this app has not gone to waste.

14.2.4 Software Quality Models

Software Quality Models

Software Quality Models are a standardized way of measuring a software product.

Software quality is the totality of features and characteristics of a product or a service that bears on its ability to satisfy the given needs. Poor quality of the software product in sensitive systems may lead to loss of human life, permanent injury, mission failure, or financial loss. So, the quality of the project should be maintained at appropriate label. To maintain the quality, there are different quality models. "A high-quality product is one which has associated with it a number of quality factors.

Types of Software Quality Models

Below are few quality models from the so-called Quality Management Gurus.

1. McCall's Quality model (1977)

Also called as General Electric's Model. This model was mainly developed for US military to bridge the gap between users and developers. It mainly has 3 major representations for defining and identifying the quality of a software product, namely *Product Revision* : This identifies quality factors that influence the ability to change the software product.

- (1) **Maintainability** : Effort required to locate and fix a fault in the program within its operating environment.
- (2) **Flexibility** : The ease of making changes required as dictated by business by changes in the operating environment.
- (3) **Testability** : The ease of testing program to ensure that it is error-free and meets its specification, i.e, validating the software requirements.

Product Transition : This identifies quality factors that influence the ability to adapt the software to new environments.

- (1) **Portability** : The effort required to transfer a program from one environment to another.
- (2) **Re-usability** : The ease of reusing software in a different context.
- (3) **Interoperability**: The effort required to couple the system to another system.

Product Operations : This identifies quality factors that influence the extent to which the software fulfills its specification.

- (1) **Correctness** : The extent to which a functionality matches its specification.
- (2) **Reliability** : The system's ability not to fail/ the extent to which the system fails.

- (3) **Efficiency** : Further categorized into execution efficiency and storage efficiency and generally means the usage of system resources, example: processor time, memory.
- (4) **Integrity** : The protection of program from unauthorized access.
- (5) **Usability** : The ease of using software.

2. **Boehm's Quality model (1978):**

Boehm's model is similar to the McCall Quality Model in that it also presents a hierarchical quality model structured around high-level characteristics, intermediate level characteristics, primitive characteristics – each of which contributes to the overall quality level.

The high-level characteristics represent basic high-level requirements of actual use to which evaluation of software quality could be put – the general utility of software. The high-level characteristics address three main questions that a buyer of software has:

- As-is utility: How well (easily, reliably, efficiently) can I use it as-is?
- Maintainability: How easy is it to understand, modify and retest?
- Portability: Can I still use it if I change my environment?

The intermediate level characteristic represents Boehm's 7 quality factors that together represent the qualities expected from a software system:

- Portability (General utility characteristics): Code possesses the characteristic portability to the extent that it can be operated easily and well on computer configurations other than its current one.
- Reliability (As-is utility characteristics): Code possesses the characteristic reliability to the extent that it can be expected to perform its intended functions satisfactorily.
- Efficiency (As-is utility characteristics): Code possesses the characteristic efficiency to the extent that it fulfills its purpose without waste of resources.
- Usability (As-is utility characteristics, Human Engineering): Code possesses the characteristic usability to the extent that it is reliable, efficient and human-engineered.
- Testability (Maintainability characteristics): Code possesses the characteristic testability to the extent that it facilitates the establishment of verification criteria and supports evaluation of its performance.
- Understandability (Maintainability characteristics): Code possesses the characteristic understandability to the extent that its purpose is clear to the inspector.

- Flexibility (Maintainability characteristics, Modifiability): Code possesses the characteristic modifiability to the extent that it facilitates the incorporation of changes once the nature of the desired change has been determined.

The lowest level structure of the characteristics hierarchy in Boehm's model is the primitive characteristics metrics hierarchy. The primitive characteristics provide the foundation for defining qualities metrics – which was one of the goals when Boehm constructed his quality model. Consequently, the model presents one more metrics supposedly measuring a given primitive characteristic.

Though Boehm's and McCall's models might appear very similar, the difference is that McCall's model primarily focuses on the precise measurement of the high-level characteristics "As-is utility", whereas Boehm's quality model is based on a wider range of characteristics with an extended and detailed focus on primarily maintainability.

3. Dromey's Quality model(1995):

Dromey has built a quality evaluation framework that analyzes the quality of software components through the measurement of tangible quality properties. Each artifact produced in the software life cycle can be associated with a quality evaluation model. Dromey gives the following examples of what he means by software components for each of the different models:

- Variables, functions, statements, etc. can be considered components of the Implementation model.
- A requirement can be considered a component of the requirements model.
- A module can be considered a component of the design model; According to Dromey, all these components possess intrinsic properties that can be classified into four categories:
 - Correctness: Evaluates if some basic principles are violated.
 - Internal: Measure how well a component has been deployed according to its intended use.
 - Contextual: Deals with the external influences by and on the use of a component.
 - Descriptive: Measure the descriptiveness of a component (for example, does it have a meaningful name.) Dromey proposes a product-based quality model that recognizes that quality evaluation differs for each product and that a more dynamic idea for modeling the process is needed to be wide enough to apply for different systems. Dromey is focusing on the relationship between the quality attributes and the sub-attributes, as well as attempting to connect software product properties with software quality attributes.

This quality model presented by R. Geoff Dromey is most recent model which is also similar to McCall's and Boehm's model. Dromey proposes a product-based quality model that recognizes that quality evaluation differs for each product and that a more dynamic idea for modeling the process is needed to be wide enough to apply for different systems. Dromey focuses on the relationship between the quality attributes and the sub-attributes, as well as attempts to connect software product properties with software quality attributes.

- 1) Product properties that influence quality.
- 2) High level quality attributes.
- 3) Means of linking the product properties with the quality attributes.

Dromey's Quality Model is further structured around a 5-step process:

- 1) Choose a set of high-level quality attributes necessary for the evaluation.
- 2) List components/modules in your system.
- 3) Identify quality-carrying properties for the components/modules (qualities of the component that have the most impact on the product properties from the list above).
- 4) Determine how each property effects the quality attributes.
- 5) Evaluate the model and identify weaknesses.

4. ISO IEC 9126 Model

As, many software quality models were proposed, the confusion happened, and new standard model was required. Thus, ISO/IEC JTC1 began to develop the required consensus and encourage standardization world-wide. The ISO 9126 is part of the ISO 9000 standard, and it is the most important standard for quality assurance. The first considerations originated in 1978, and in the year 1985 the development of ISO/IEC 9126 was started.

In this model, for software development companies, the totality of software product quality attributes was classified in a hierarchical tree structure of characteristics and sub characteristics. And the highest level of this structure consists of the quality characteristics and the lowest level consists of the software quality criteria. This model specified six characteristics including Functionality, the Reliability, Usability, Efficiency, Maintainability, and the Portability; all of which are further divided into 21 sub characteristics. All these sub characteristics are manifested externally when the software is used as part of a computer system, and thus are the result of internal software attributes. All the defined characteristics are applicable to every kind of software, including computer programs and data contained in firmware and provide consistent terminology for software product quality. And they also provide a framework for making trade-offs between software product capabilities.

5. **FURPS Model** -This model categorizes requirements into functional and non-functional requirements. The term FURPS is an acronym for Functional requirement(F) which relies on expected input and output, and in non functional requirements (U) stands for Usability which includes human factors, aesthetic, documentation of user material of training, (R) stands for reliability(frequency and severity of failure, time among failure), (P) stands for Performance that includes functional requirements, and finally (S) stands for supportability that includes backup, requirement of design and implementation etc.
6. **Ghezzi Model** -This model states that the internal qualities of a software help the software developers to attain a collaborative result both in terms of external and internal qualities of a software. The overall qualities can be accuracy, flexibility, integrity, maintainability, portability, reliability, re-usability and usability.
7. **IEEE Model** –It is a standard which defines various specifications for software maintenance, thus providing a quality model. This model gives a variety of measurement techniques for various qualitative factors like efficiency, functionality, maintainability, portability, reliability and usability.
8. **SATC's Model** -SATC is an acronym for Software Assurance Technology Centre. Its objective is to improve software quality by defining metrics program which helps to meet the basic needs with least expenditure. This model tests a quality model by evaluating the results of the metrics used, and also on the basis of discussions based on the project. This model defines set of goals and process attributes based on the structure of ISO 9126-1 quality model.
9. **Capability Maturity Model** -One of the most important quality models of software quality maintenance. The model lays down a very simple approach to define the quality standards. It has five levels namely – initial, repeatable, defined, managed, optimizing.

At the initial level, the company is quite small, and it solely depends on an individual how he handles the company. The repeatable level states that at least the basic requirements or techniques have been established and the organization has attained a certain level of success. By the next level that is , defined, the company has already established a set of standards for smooth functioning of a software project/process. At the managed level, an organization monitors its own activities through a data collection and analysis. At the fifth level that is the optimizing level, constant improvement of the prevailing process becomes a priority, a lot of innovative approach is applied towards the qualitative enhancement.

14.2.5 ISO 9126

The ISO 9126 software is an international standard software quality model that helps in creating a solid framework for assessing software. This standard way of assessing software can be segregated in four different

ways. These are used to address subjects of different nature. This software is profoundly used in a widespread way to embrace various models and metrics. The recommended features describe externally when software is found to be a result of attributes of internal attributes of software.

The following ways by which a standard software quality model can be calculated are as follows:

1. Quality Model.
2. External Metrics.
3. Internal Metrics.
4. Quality in use Metrics.

Like every software, ISO 926 software model has distinct qualities. These are laid on following basis:

1. **Functionality**

It is a key aspect of any product or service. It is due to this the software can fulfill a task and keep to its purpose. It is defined as a software product that helps to meet the needs of the clients. A functionality of software is dependent on its complexity. For example: an ATM machine. This is further divided in other categories are as follows:

- Suitability.
- Accuracy.
- Interoperability.
- Security.
- Functional compliance.

2. **Reliability**

This characteristic determines the capability of software to sustain its use when put under different circumstances.

3. **Usability**

The usability of software is highly dependent on the functional uses of software. For example: ATM machine is used to withdraw cash. According to the usability of an ATM, the ATM is not affected or influenced by any amounts entered by the user. This is further divided into other sub-categories, and these are as follows:

- Maturity.
- Fault Tolerance.
- Recoverability.
- Reliability Compliance.

4. **Efficiency**

This feature of the model is more concerned by resources of the system when used for providing a desired functionality. This type of feature is defined by amount of disk space, memory, and network. This is further divided into other sub-categories, and these are as follows:

- Understandability.
- Learner ability.
- Operability.
- Attractiveness.
- Usability Compliance.

5. **Maintainability**

This property of maintainability of the software model is used to recognize and fix a defect accordingly. The model is inspected for the faults, and these can be identified easily. In accordance with this the cause and effect of maintainability of software is a concern. This is further divided into other sub-categories, and these are as follows:

- Analyzability.
- Resource Utilization.
- Stability.
- Testability.
- Changeability.

6. **Portability**

According to this feature, capable software should easily adapt to the environmental changes frequently as possible. The designing of an object and the practices of its implementation are highly dependent on this feature. This standard method is further divided in few categories:

- Adaptability.
- Install ability.
- Co-existence.
- Replaceability.
- Portability compliance.

14.2.6 Product and Process Metrics

What is process and product metrics?

Product metrics – Describes the characteristics of the product such as size, complexity, design features, performance, and quality level. Process metrics – These characteristics can be used to improve the development and maintenance activities of the software.

Software Metrics

A software metric is a measure of software characteristics which are measurable or countable. Software metrics are valuable for many reasons, including measuring software performance, planning work items, measuring productivity, and many other uses.

Within the software development process, many metrics are that are all connected. Software metrics are like the four functions of management: Planning, Organization, Control, or Improvement.

Classification of Software Metrics

Software metrics can be classified into two types as follows:

1. Product Metrics: These are the measures of various characteristics of the software product. The two important software characteristics are:

1. Size and complexity of software.
2. Quality and reliability of software.

These metrics can be computed for different stages of SDLC.

3. Process Metrics: These are the measures of various characteristics of the software development process. For example, the efficiency of fault detection. They are used to measure the characteristics of methods, techniques, and tools that are used for developing software.

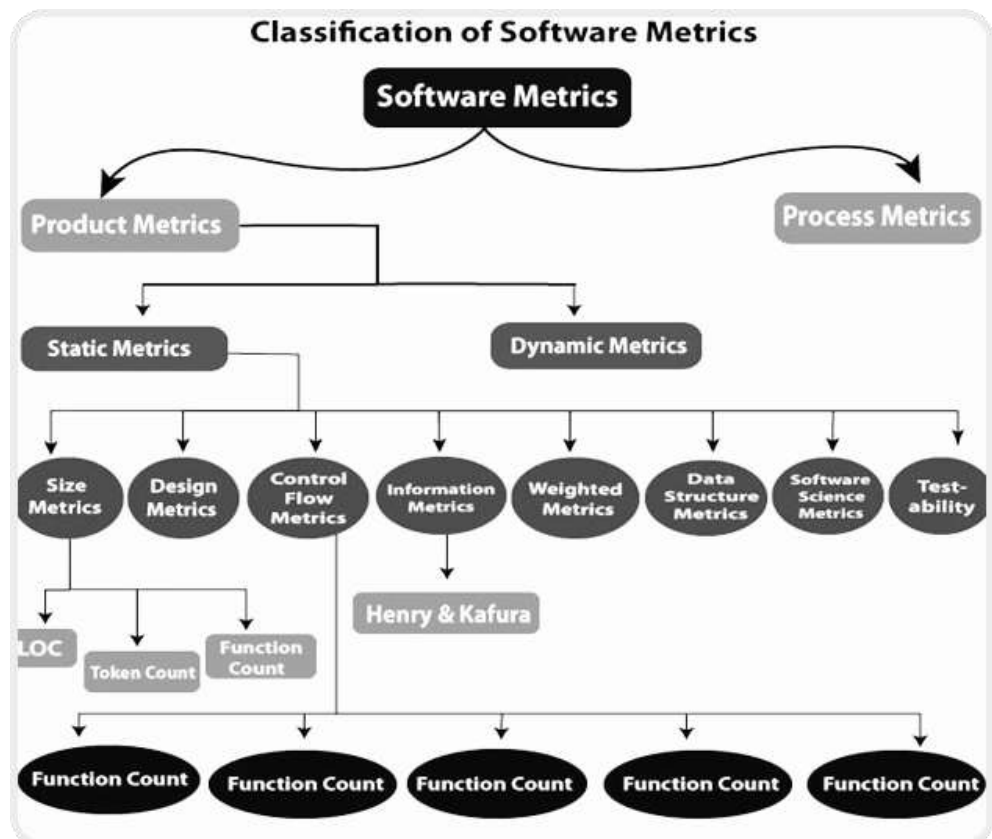


Figure 14.1 classification of software metrics

Types of Metrics

Internal metrics: Internal metrics are the metrics used for measuring properties that are viewed to be of greater importance to a software developer. For example, Lines of Code (LOC) measure.

External metrics: External metrics are the metrics used for measuring properties that are viewed to be of greater importance to the user, e.g., portability, reliability, functionality, usability, etc.

Hybrid metrics: Hybrid metrics are the metrics that combine product, process, and resource metrics. For example, cost per FP where FP stands for Function Point Metric.

Project metrics: Project metrics are the metrics used by the project manager to check the project's progress. Data from the past projects are used to collect various metrics, like time and cost; these estimates are used as a base of new software. Note that as the project proceeds, the project manager will check its progress from time-to-time and will compare the effort, cost, and time with the original effort, cost and time. Also understand that these metrics are used to decrease the development costs, time efforts and risks. The project quality can also be improved. As quality improves, the number of errors and time, as well as cost required, is also reduced.

Advantage of Software Metrics

Comparative study of various design methodology of software systems.

For analysis, comparison, and critical study of different programming language concerning their characteristics.

In comparing and evaluating the capabilities and productivity of people involved in software development.

In the preparation of software quality specifications.

In the verification of compliance of software systems requirements and specifications.

In making inference about the effort to be put in the design and development of the software systems.

In getting an idea about the complexity of the code.

In taking decisions regarding further division of a complex module is to be done or not.

In guiding resource manager for their proper utilization.

In comparison and making design tradeoffs between software development and maintenance cost.

In providing feedback to software managers about the progress and quality during various phases of the software development life cycle.

In the allocation of testing resources for testing the code.

Disadvantage of Software Metrics

The application of software metrics is not always easy, and in some cases, it is difficult and costly.

The verification and justification of software metrics are based on historical/empirical data whose validity is difficult to verify.

These are useful for managing software products but not for evaluating the performance of the technical staff.

The definition and derivation of Software metrics are usually based on assuming which are not standardized and may depend upon tools available and working environment.

Most of the predictive models rely on estimates of certain variables which are often not known precisely.

14.2.7 Product versus Process Quality Management

What is product quality and process quality?

Product Quality: Means that **the product is according to the specifications gathered by Business Analyst AND is bug free.**

Process Quality: Means the process adopted by the company to test the application should be defined well enough that it covers all the aspect of the application

Product:

In the context of software engineering, Product includes any software manufactured based on the customer's request. This can be a problem-solving software or computer based system. It can also be said that this is the result of a project.

Process:

Process is a set of sequence steps that have to be followed to create a project. The main purpose of a process is to improve the quality of the project. The process serves as a template that can be used through the creation of its examples and is used to direct the project.

The main difference between a process and a product is that the process is a set of steps that guide the project to achieve a convenient product. while on the other hand, the product is the result of a project that is manufactured by a wide variety of people.

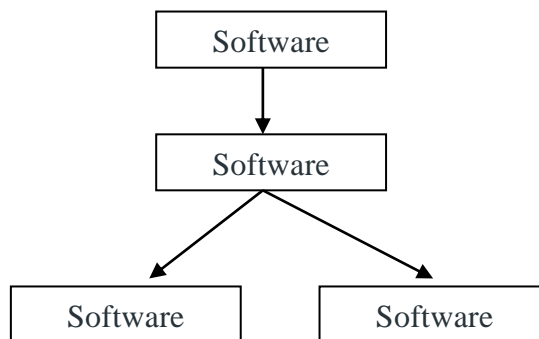


Figure 14.2 Process and product quality

It is general, that the quality of the development process directly affects the quality of delivered products. The quality of the product can be measured, and the process is improved until the proper quality level is achieved. Figure 14.3. illustrates the process of quality assessment based on this approach.

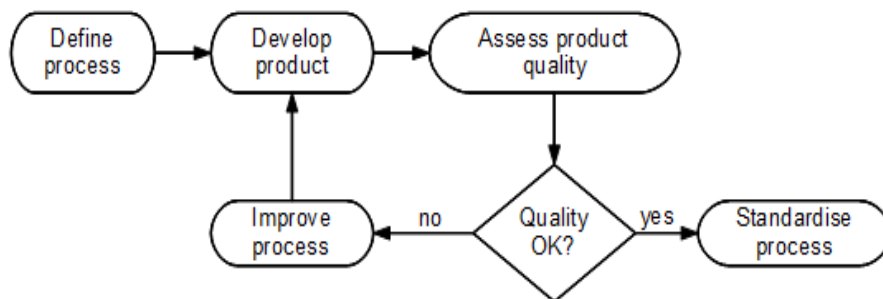


Figure 14.3 Process based quality assessment

In manufacturing systems there is a clear relationship between production process and product quality. However, quality of software is highly influenced by the experience of software engineers. In addition, it is difficult to measure software quality attributes, such as maintainability, reliability, usability, etc., and to tell how process characteristics influence these attributes. However, experience has shown that process quality has a significant influence on the quality of the software.

Process quality management includes the following activities:

1. Defining process standards.
2. Monitoring the development process.
3. Reporting the software.

14.2.8 Quality Management Systems

Software quality management (SQM) is a management process that aims to develop and manage the quality of software in such a way so as to best ensure that the product meets the quality standards expected by the customer while also meeting any necessary regulatory and developer requirements, if any. Software quality managers require software to be tested before it is released to the market, and they do this using a cyclical process-based quality assessment to reveal and fix bugs before release. Their job is not only to ensure their software is in good shape for the consumer but also to encourage a culture of quality throughout the enterprise

Quality management activities

Software quality management activities are generally split up into three core components: quality assurance, quality planning, and quality control. Some like software engineer and author Ian Sommerville don't use the term "quality control" (as quality control is often viewed as more a

manufacturing term than a software development term), rather, linking its associated concepts with the concept of quality assurance. However, the three core components otherwise remain the same.

Quality assurance

Software quality assurance sets up an organized and logical set of organizational processes and deciding on that software development standards — based on industry best practices — that should be paired with those organizational processes, software developers stand a better chance of producing higher quality software. However, linking quality attributes such as "maintainability" and "reliability" to processes is more difficult in software development due to its creative design elements versus the mechanical processes of manufacturing. Additionally, "process standardization can sometimes stifle creativity, which leads to poorer rather than better quality software."

This stage can include:

- encouraging documentation process standards, such as the creation of well-defined engineering documents using standard templates
- mentoring how to conduct standard processes, such as quality reviews
- performing in-process test data recording procedures
- identifying standards, if any, that should be used in software development processes

Quality planning

Quality planning works at a more granular, project-based level, defining the quality attributes to be associated with the output of the project and how those attributes should be assessed. Additionally, any existing organizational standards may also be assigned to the project at this phase. Attributes such as "robustness," "accessibility," and "modularity" may be assigned to the software development project. While this may be a more formalized, integral process, those using a more agile method of quality management may place less emphasis on strict planning structures.^[3] The quality plan may also address intended market, critical release dates, quality goals, expected risks, and risk management policy.

Quality control

The quality control team tests and reviews software at its various stages to ensure quality assurance processes and standards at both the organizational and project level are being followed. Some like Sommerville link these responsibilities to quality assurance rather than call it quality control. These checks are optimally separate from the development team to lend more of an objective view of the product to be tested. However, project managers on the development side must also assist, helping to promote as part of this phase a "culture that provides support without blame when errors are discovered." In software development firms implementing a more agile quality approach, these

activities may be less formal; however, a switch to agile methods from a more formal quality management structure may create problems if management procedures aren't appropriately adapted.

Activities include:

- release testing of software, including proper documentation of the testing process
- examination of software and associated documentation for non-conformance with standards
- follow-up review of software to ensure any required changes detailed in previous testing are addressed
- application of software measurement and metrics for assessment

14.2.9 Process Capability Models

Process Capability Modeling

L. Bauer Published 2002 Computer Science Every production process is subject to variations that limit our ability to produce a defect-free product. Process capability models (PCMs) are used to quantify likely process variations, which can then be included during the analysis of a product design. PCMs facilitate the flowback of capability information from manufacturing/sourcing to design and are an essential element of Design for Six Sigma (DFSS). This paper discusses how process capability can be measured and modeled, how models can be organized and saved, and how models can be retrieved, evaluated, and applied

Capability Maturity Model (CMM) is a methodology used to develop, refine maturity of an organization's software development process. It is developed by SIE in mid 1980. It is a process improvement approach.

To assess an organization against a scale of 5 process maturity levels. It Deals with the what processes should be implemented & not so much with the how processes should be implemented. Each maturity level comprises a predefined set of process areas called KDA (Key Process Area), these KDA – Goals, Commitment, Ability, measurement, verification.

Levels of Capability Maturity Model (CMM) are as following below.

1. Level One : Initial – Work is performed informally. A software development organization at this level is characterized by AD HOC activities (organization is not planned.).

2. Level Two : Repeatable – Work is planned and tracked. This level of software development organization has a basic and consistent project management processes to TRACK COST, SCHEDULE, AND FUNCTIONALITY. The process is in place to repeat the earlier successes on projects with similar applications.

3. Level Three : Defined – Work is well defined. At this level the software process for both management and engineering activities are DEFINED AND DOCUMENTED.

4. Level Four : Managed – Work is quantitatively controlled.

- **Software Quality management** – Management can effectively control the software development effort using precise measurements. At this level, organization set a quantitative quality goal for both software process and software maintenance.
- **Quantitative Process Management** – At this maturity level, the performance of processes is controlled using statistical and other quantitative techniques, and is quantitatively predictable.

5. Level Five : Optimizing – Work is Based Upon Continuous Improvement.

The key characteristic of this level is focusing on CONTINUOUSLY IMPROVING PROCESS performance.

Key features are:

- Process change management
- Technology change management
- Defect prevention

14.2.10 Techniques to Help Enhance Software Quality

Seven Ways to Improve Software Quality Which Will Not Break the Bank

The project management lifecycle is not complete without quality assurance and businesses are shifting their focus to build a high-quality product. If you are somehow related to an application or software, then you are surely looking out for ways to ensure the quality of your product without breaking the bank. Exceptional software quality practices promise cost-effectiveness and enhance the performance of the product being delivered.

It is recommended to start implementing testing strategies at the initial stage as bugs can be easily identified and fixed. Bug identification in the later stage of the project may result in a complication that will eventually out-turn in more cost and time. To ensure that their testing goes in the right direction, consider using defect tracking software from the start of the testing. If issues are highlighted and solved in the initial stage, then it is a win-win situation and efficiency of the results is also increased. Issues can be exacerbated if they are not identified timely, and an expensive exercise would be carried out to fix the damage.

Instead of spending unnecessary time and resources on the project to fix the software issues one should shift maximum focus on the quality of the

software from the start. Below mentioned are seven points which one should consider increasing software quality and follow an assured approach through the development.

1. Test Often and Early

Early testing is paramount for a successful quality product and this aspect neither can be ignored nor delayed. If testing is carried out often and in the early phase, then issues and bugs won't snowball into larger complications. If an issue becomes complex, then it means it will set off to be more expensive and can't be ironed out easily.

Testers need to be involved at earliest so that they remain at top of all issues and cater to issues timely. Adopt an early automation approach from basic UI tests and then accelerate it as the project stabilizes.

2. Implement Quality Controls

Testers and developers should work in partnership from the start and monitor quality controls as it ensures that set standards are being met. This is an ongoing process that starts from the beginning and is carried out until the product is delivered. Testers and developers should work shoulder to shoulder to develop software strategy and deal with bugs in a structured way.

3. Promote Innovation

As much it is appreciated to follow important testing structures along with quality metrics, it is also endorsed to think out of the box. A great way to promote innovation is to automate the monotonous processes and that sufficient time can be saved and used wisely.

4. Incorporate Management Tools

Effective quality management tools can help the team to build a sustainable business. Stakeholders are only concerned about their product and its quality as they expect companies to do whatever it takes to meet the quality standards. With the right tools, transparency is increased, and employees do not get drained while performing day to day operations. Defect tracking software, issue tracking tool, project management software, and test automation tools are among the few which every software testing company should have.

5. Employee Training

Technology is evolving day by day and to cope up with the advancements, it is important to embrace the change and train your team accordingly. Professional training should be promoted, and employees should be encouraged to opt for certifications to work in their best capacity. As tools and techniques upgrade with each passing day, employees should upgrade their skill set in a similar manner.

6. Error Management and Analysis

An organized approach should be adapted to track the issues and manage the risk. For successful software development, a risk register plays a vital role and so does defect tracking software. These tools help to keep everyone informed regarding the risks and defects in the systems so everyone can play their part in proposing a convenient solution and resolving ambiguities.

7. Review, Revise and Remember

A high-quality product is not a coincidence, and it can't happen overnight. It goes through a repeated process of review, revise, and remember which are explained as follows.

Review: Testers constantly review the code and ensure that set quality standards are being met.

Revise: To study the software process and understand which aspects worked for the project and which require more improvement. Analyze if innovation can transcend and changes can be made.

Remember: While delivering a quality product, note down what worked for you and what areas didn't align. List down all positives and negatives of a successful project.

Final Thoughts

The success of your software product highly depends on the way you deal with it from the initial phase and how intelligently you manage the issues encountered. It is imperative that your team knows what they are working on and what is expected from them to be delivered. The testing team should come up with a clear plan to start with the entire process.

iTexico and Improving's nearshore delivery model for software quality assurance and collaborative, results-driven approach is designed around the success of your project. We'll make use of the right technologies and tools for your strategic goals and work closely with your in-house team to ensure a timely delivery.

14.2.11 Testing and Software Reliability

Software testing can be stated as the process of verifying and validating that software or application is bug-free, meets the technical requirements as guided by its design and development, and meets the user requirements effectively and efficiently with handling all the exceptional and boundary cases.

The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy, and usability. It mainly aims at measuring the specification, functionality, and performance of a software program or application.

Software testing can be divided into two steps:

1. **Verification:** it refers to the set of tasks that ensure that software correctly implements a specific function.
2. **Validation:** it refers to a different set of tasks that ensure that the software that has been built is traceable to customer requirements.

Verification: “Are we building the product, right?”

Validation: “Are we building the right product?”

What are different types of software testing?

Software Testing can be broadly classified into two types:

1. **Manual Testing:** Manual testing includes testing software manually, i.e., without using any automated tool or any script. In this type, the tester takes over the role of an end-user and tests the software to identify any unexpected behavior or bug. There are different stages for manual testing such as unit testing, integration testing, system testing, and user acceptance testing.

Testers use test plans, test cases, or test scenarios to test software to ensure the completeness of testing. Manual testing also includes exploratory testing, as testers explore the software to identify errors in it.

2. **Automation Testing:** Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses another software to test the product. This process involves the automation of a manual process. Automation Testing is used to re-run the test scenarios that were performed manually, quickly, and repeatedly.

Apart from regression testing, automation testing is also used to test the application from a load, performance, and stress point of view. It increases the test coverage, improves accuracy, and saves time and money in comparison to manual testing.

What are the different techniques of Software Testing?

Software techniques can be majorly classified into three categories:

1. **Black Box Testing:** The technique of testing in which the tester doesn't have access to the source code of the software and is conducted at the software interface without concern with the internal logical structure of the software is known as black-box testing.
2. **White-Box Testing:** The technique of testing in which the tester is aware of the internal workings of the product, has access to its source code, and is conducted by making sure that all internal operations are performed according to the specifications is known as white box testing.

Black Box Testing	White Box Testing
Internal workings of an application are not required.	Knowledge of the internal workings is a must.
Also known as closed box/data-driven testing.	Also known as clear box/structural testing.
End users, testers, and developers.	Normally done by testers and developers.
This can only be done by a trial and error method.	Data domains and internal boundaries can be better tested.

Table 14.1 Comparison between black box and white box testing

3. Gray Box testing

Gray box testing is a combination of white box and Black box testing. It can be performed by a person who knew both coding and testing. And if the single person performs white box, as well as black-box testing for the application, is known as Gray box testing.

What are different levels of software testing?

Software level testing can be majorly classified into 4 levels:

1. **Unit Testing:** A level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed.
2. **Integration Testing:** A level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units.
3. **System Testing:** A level of the software testing process where a complete, integrated system/software is tested. The purpose of this test is to evaluate the system’s compliance with the specified requirements.
4. **Acceptance Testing:** A level of the software testing process where a system is tested for acceptability. The purpose of this test is to evaluate the system’s compliance with the business requirements and assess whether it is acceptable for delivery

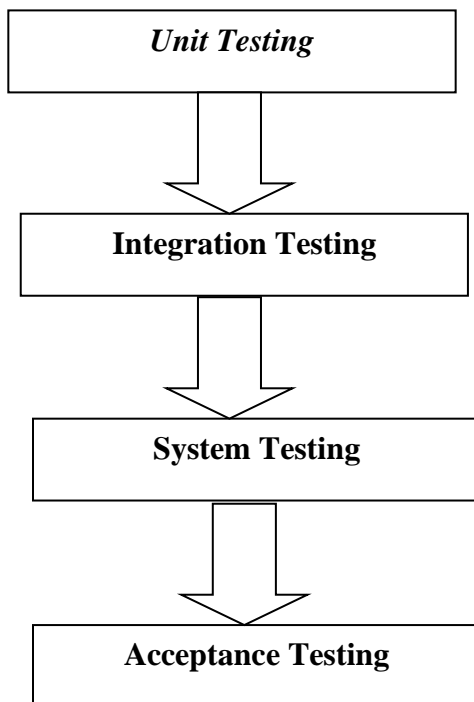


Figure14.4 Different levels of software testing

Software Reliability

Software Reliability means Operational reliability. It is described as the ability of a system or component to perform its required functions under static conditions for a specific period.

Software reliability is also defined as the probability that a software system fulfills its assigned task in a given environment for a predefined number of input cases, assuming that the hardware and the input are free of error.

Software Reliability is an essential connect of software quality, composed with functionality, usability, performance, serviceability, capability, install ability, maintainability, and documentation. Software Reliability is hard to achieve because the complexity of software turn to be high. While any system with a high degree of complexity, containing software, will be hard to reach a certain level of reliability, system developers tend to push complexity into the software layer, with the speedy growth of system size and ease of doing so by upgrading the software.

For example, large next-generation aircraft will have over 1 million source lines of software on-board; next-generation air traffic control systems will contain between one and two million lines; the upcoming International Space Station will have over two million lines on-board and over 10 million lines of ground support software; several significant life-critical defense systems will have over 5 million source lines of software. While the complexity of software is inversely associated with software reliability, it is directly related to other vital factors in software quality, especially functionality, capability, etc.

A good software reliability engineering program, introduced early in the development cycle, will mitigate these problems by: Preparing program management in advance for the testing effort and allowing them to plan both schedule and budget to cover the required testing.

Continuous review of requirements throughout the life cycle, particularly for handling of exception conditions. If requirements are incomplete there will be no testing of the exception conditions.

SoHaR software reliability engineers are experienced in all the stages and tasks required in a comprehensive software reliability program. We can support or lead tasks such as:

- 1) Reliability Allocation
- 2) Defining and Analyzing Operational Profiles
- 3) Test Preparation and Plan
- 4) Software Reliability Models

1. Reliability Allocation:-

Reliability allocation is the task of defining the necessary reliability of a software item. The item may be part of an integrated hardware/software system, may be a relatively independent software application, or, more and more rarely, a standalone software program. In either of these cases our goal is to bring system reliability within either a strict constraint required by a customer or an internally perceived readiness level or optimize reliability within schedule and cost constraints.

SoHaR will assist your organization in the following tasks:

Derive software reliability requirements from overall system reliability requirements.

When possible, depending on lifecycle stage and historical data, estimate schedule and cost dependence on software reliability goals.

Optimize reliability/schedule/cost based on your constraints and your customer's requirements,

2. Defining and Analyzing Operational Profiles:-

The reliability of software, much more so than the reliability of hardware, is strongly tied to the operational usage of an application. A software fault may lead to system failure only if that fault is encountered during operational usage. If a fault is not accessed in a specific operational mode, it will not cause failures at all. It will cause failure more often if it is in code that is part of an often used "operation" (An operation is defined as a major logical task, usually repeated multiple times within an hour of application usage). Therefore, in software reliability engineering we focus on the

operational profile of the software which weighs the occurrence probabilities of each operation. Unless safety requirements indicate a modification of this approach, we will prioritize our testing according to this profile.

SoHaR will work with your system and software engineers to complete the following tasks required to generate a useable operational profile:

Determine the operational modes (high traffic, low traffic, high maintenance, remote use, local use etc).

Determine operation initiators (components that initiate the operations in the system).

Determine and group "Operations" so that the list includes only operations that are significantly different from each other (and therefore may present different faults).

Determine occurrence rates for the different operations.

Construct the operational profile based on the individual operation probabilities of occurrence.

3. Test Preparation and Plan:-

Test preparation is a crucial step in the implementation of an effective software reliability program. A test plan that is based on the operational profile on the one hand, and subject to the reliability allocation constraints on the other, will be effective at bringing the program to its reliability goals in the least amount of time and cost.

Software Reliability Engineering is concerned not only with feature and regression test, but also with load test and performance test. All these should be planned based on the activities outlined above.

The reliability program will inform and often determine the following test preparation activities:

Assessing the number of new test cases required for the current release.

New test case allocation among the systems (if multi-system).

New test case allocation for each system among its new operations.

Specifying new test cases

Adding the new test cases to the test cases from previous releases.

4. Software Reliability Models:-

Software reliability engineering is often identified with reliability models, in particular reliability growth models. These, when applied correctly, are successful at providing guidance to management decisions such as:

- Test schedule
- Test resource allocation
- Time to market
- Maintenance resource allocation

14.2.12 Quality Plans

Quality planning is the process of developing a quality plan for a project. The quality plan defines the quality requirements of software and describes how these are to be assessed. The quality plan selects those organizational standards that are appropriate to a particular product and development process. Quality plan has the following parts:

1. Introduction of product.
2. Product plans.
3. Quality goals.
4. Risks and risk management.

The quality plan defines the most important quality attributes for the software and includes a definition of the quality assessment process. Table 14.2. shows generally used software quality attributes that can be considered during the quality planning process

Safety	Understand ability	Portability
Security	Testability	Usability
Reliability	Adaptability	Reusability
Resilience	Modularity	Efficiency
Robustness	Complexity	Learnability
Maintainability		

Table 14.2 Software quality attributes

14.3 SUMMARY

In this software quality chapter, you learned about The Place of Software Quality in Project Planning, Importance of Software Quality, Defining Software Quality, Software Quality Models, ISO 9126, Product and Process Metrics, Product versus Process Quality Management, Quality Management Systems, Process Capability Models, Techniques to Help Enhance Software Quality, Testing and its types, Software Reliability, Quality Plans etc. Still, you had a doubt go through references and bibliography

14.4 EXERCISE

- Q.1 Explain place of Software Quality in Project Planning
- Q.2 Write about Importance of Software Quality

- Q.3 Explain in detail Software Quality Models and it's types.
- Q.4 What is process and product metrics?
- Q.5 Write short notes on software metrics
- Q.6 Write about types of software metrics
- Q.7 What are the advantages and disadvantages of software metrics?
- Q.8 Explain in detail Product versus Process Quality Management
- Q.9 Write short notes on Quality Management Systems
- Q.10 Explain in detail Process Capability Models
- Q.11 Write about Techniques to Help Enhance Software Quality
- Q.12 Explain different levels of software testing
- Q.13 Compare black box testing and white box testing
- Q.14 Explain software testing and it's types in detail
- Q.15 Explain different levels of software testing
- Q.16 What are the Quality Plans?
- Q.17 Explain Software Reliability

14.5 LIST OF BOOKS AND REFERENCES

1. Software Project Management Bob Hughes, Mike Cotterell, Rajib Mall TMH 6 th 2018
2. Project Management and Tools & Technologies – An overview Shailesh Mehta SPD 1st 2017
3. Software Project Management Walker Royce Pearson 2005
4. <https://smartbear.com>
5. <https://www.tutorialspoint.com>
6. <https://www.professionalqa.com>
7. <https://msritse2012.wordpress.com>
8. <http://moodle.autolab.uni-pannon.hu>
9. <https://en.wikipedia.org>
10. <https://www.semanticscholar.org>
11. <https://www.itexico.com>
12. <https://www.geeksforgeeks.org>
13. <https://www.javatpoint.com>

PROJECT CLOSEOUT

Unit Structure

15.0 Objectives

15.1 Introduction

15.2 An Overview

15.2.1 Project closure

15.2.1.1 What is closing a project?

15.2.1.2 When do you close a project (Reasons)?

15.2.1.3 How to close a project?

15.2.1.4 Types of project closure

15.2.1.5 Steps to closing a project and project closure checklist(Process)

15.2.1.6 Importance of closing a project

15.2.2 Financing and Financial Closure

15.2.2.1 Importance of Project Financing

15.2.2.2 Important issues in Financing Of Project

15.2.2.3 The keys to a faster close (Financial)

15.2.3 Project Closeout Report

15.2.3.1 What is a Project Closure Report?

15.2.3.2 When do I use a Project Closure Report?

15.2.3.3 Eight Steps to Writing a Project Closure Report

15.3 Summary

15.4 Exercise

15.5 List of Books and References

15.0 OBJECTIVES

After going through this chapter, you will be able to:

- What is project closeout?
- Reasons for Project Closure
- Importance of closing a project
- Project Closure Process
- Performing a Financial Closure
- Project Closeout Report

15.1 INTRODUCTION

The Project Closure Phase is the fourth and last phase in the project life cycle. In this phase, you will formally close your project and then report its overall level of success to your sponsor. Project Closure involves handing over the deliverables to your customer, passing the documentation to the business, cancelling supplier contracts, releasing staff and equipment, and informing stakeholders of the closure of the project. After the project has been closed, a Post Implementation Review is completed to determine the project's success and identify the lessons learned

15.2 AN OVERVIEW

15.2.1 Project closure

The project lifecycle consists of five groups:

- Initiating process group
- Planning process group
- Executing process group
- Monitoring and controlling process group
- Closing process group

The closing phase of project management is the final phase of the project lifecycle. This is the stage where all deliverables are finalized and formally transferred, and all documentation is signed off, approved, and archived.

The project closure process ensures that:

- All work has been completed according to the project plan and scope.
- All project management processes have been executed.
- You have received final sign-off and approval from all parties.

The project management closure process also gives the team the opportunity to review and evaluate the project's performance to ensure future projects' success.

15.2.1.1 What Is closing a project?

It's has been established that every project has a start date and an end date. So, the process of completing the work on the project to an end is exactly what closing a project is. Nothing more, nothing less. If you are not closing an exercise or ending an exercise, then you need to know that the exercise isn't a project.

It could be an operation. One of the differences between a project and an operation is that while projects are temporary, operations are ongoing and continuous. No matter how long the duration of a project is, it must end.

15.2.1.2 When do you close a project(Reasons)?

Well, there are three circumstances under which a project can be closed. Yeah, THREE! One out of every ten readers of this article will find this surprising. Just give me a second and I will explain the three times.

i. When the Project has delivered all the objectives and/or result.

This is probably the most popular and most desirous time when a project should be closed. At the beginning of the project, a set of objectives, deliverables, and results were set. The PM and the whole organization rolled up their sleeves and started working towards building and delivering the objectives and deliverables for the project. Once the objective is met and the deliverables completed and accepted by the Project Sponsor/owner, it is time to close the project. Reason? The PM and the team have completed all they committed to, and there isn't anything left to do on the project. What if there is a modification or an addition to the deliverables of the project. Well, that's called scope creep, if there is no adjustment to other components that may be affected, and once the addition didn't go through change control for approval, and no re-baseline, then it must be initiated as a new project, after closing the current project.

For PRINCE2, Once the project has delivered what was specified in the business case of the project mandate, while for PMI, once it has developed the content of the approved scope.

ii. When the objectives of the project can't be met again

This is often called TERMINATION. Hopefully, it can be done early.

This is usually not a very pleasant one, but it's usually a reality. It could be because of the complexity of the project or a combination of many other things. I recall working on a project with some client in the financial services industry a few years back. The business case and feasibility were highly optimistic, and the organization decided to invest in the project hoping for the objective to be realized. However, after series of attempts, efforts, and re-baselining by the project team and the stakeholder, it became clear to even the blind stranger that the approach and the technology chosen for the project couldn't meet the objective, the organization had to terminate the project for that same reason.

At times it could be that the organization had spent so much money on the project, yet they haven't realized any benefit and at every point in time, there was no sign that the project would still deliver the objective.

Please note that in most cases, the project manager and the team are not responsible for this, and often should not be blamed.

iii. **When the objective of the project is no longer needed**

This is often called early termination

We live in a dynamic and constantly changing environment, and there are many actors and factors driving the market and corporate space. Most of these factors could be a technological change, a governmental policy, a change in strategy and direction for the organization etc. PRINCE2 advises that at every point on the project, the project organization should continually check for business justification, to ensure the business case is still valid. If at any point the organization realizes that the business case has become invalid, the only thing left is to close the project. A typical example was a project I was working on for a client to penetrate a new market and release a product that would solve a particular challenge. We were about 45% into the project when the government introduced an initiative that would address a similar challenge at no cost. It became obvious that the product would not make many sales, hence the objective and business case became invalid. We had to terminate the project immediately. This is better than going head to complete the project. Organizations don't just initiate or complete a project for the mere sake of completing a project, the product of the project must be valid all through and the end product must be desired and desirable at all times.

Another example was a consulting project I was working on with a startup to rebrand the company. Midway into the project, there was a merger and acquisition with another company it became instantly known that there was no need to rebrand the company anymore. Again, this could be at no fault of the project team.

In the next section, I would highlight how to close a project, irrespective of when it is closed.

15.2.1.3 How to close a project

This process will be broken that into two different sections. The first section will highlight steps to closing a project whose objectives have been met, while the second would highlight the steps to close a project whose objective is either not needed again or can't be met.

How to close a project whose objective has been met.

1. Confirm that all the deliverables have been completed and accepted by the appropriate approving authority. Here you will need to reference your plan, specifically your approved scope and acceptance criteria
2. Obtain formal acceptance and approval (this could be by way of formal SIGN-OFF or whatever method has been agreed upon).

3. Close any procurement component of the project
4. Gather the team and update lessons learned on the project
5. Release all resources and provide feedback as required
6. Complete end of project report and archive project information
7. Celebrate. And I mean it.

How to close a project whose objective can't be met or whose objective is not needed again

1. Validate the reason for the early termination
2. Determine all the deliverables that have been created so far, and ensure they are accepted
3. Obtain formal closure notification from the Sponsor
4. Close any procurement engagement
5. Update lessons learned with the team
6. Release resources and complete end of project report, capturing the stage the project was at when it was closed, the reason it was closed, and the lessons learned and archive project information.
7. Celebrate (if you have the courage to)

15.2.1.4 Types of project closure

There are 5 types of Project Closure

1. Normal: the project goes to completion.
2. Premature: the project is completed early and meets performance specifications.
3. Perpetual: the project keeps getting extended primarily because of changes in the constraints.
4. Change in Priorities: the project is cancelled due to changes in the constraints.
5. Failed Project: the project is:
 - Cancelled.
 - Completed but was considered a failure because it didn't perform as expected.

15.2.1.5 Steps to closing a project and project closure checklist(Process)

The close of the project is the final phase of your job, it's the last turn of the project life cycle, and like any other aspect of a project, it requires a process. The following are five steps you should take to make sure you've dotted all the I's and crossed all the T's, as well as taken full advantage of the experience.

1. Arrange a Postmortem

Managing a project isn't only about tasks and resources, budget, and deadlines, it's an experience you can constantly learn from. While you should have been learning throughout the project, now is a great time to look back without the pressure and distractions that might have dulled your focus.

Gather the core team to invite feedback about what worked, and what didn't. Encourage honesty. By documenting the mistakes and the successes of the project, you're building a catalog that offers historic data. You can go back and look over the information for precedents when planning for new projects.

Projects are never standalone things, but part of a continuum, where the specifics might vary, but the general methods usually remain the same. There's a wealth of knowledge produced after any project closes.

2. Complete Paperwork

As noted, projects generate reams of documents. These documents are going to have to get sign off and approval from stakeholders. Everything needs attention and must be signed for, which is the legal proof that in fact these documents have concluded. That includes closing all contracts you might have made with internal partners or vendors or any other resources you contracted with.

This includes addressing all outstanding payments. You want to make sure that all invoices, commissions, fees, bonus, what have you, are paid. Complete all the costs involved with the project. It's not done if it's not paid for.

Project management software can help you organize all these documents. ProjectManager acts like a hub for all your project files. You can track them on our list view, which is more than the usual to-do list app. For one thing, you can see the percentage complete for each item on the list. Now you know if that contractor has been paid and whether you can sign off on the contract. You can even set up notifications to make sure your payments are delivered on time. Try it out for yourself with this free trial.

3. Release Resources

You assemble a team for the project, and now you must cut them loose. It's a formal process, and a crucial one, which frees them for the next project. Each team is brought together for the mix of skills and experience they bring to a project. The project determines the team members you'll want to work with, and each project is going to be a little bit different, which will be reflected in the team hired to execute it.

This is true for internal as well as external resources. The external ones might be more obvious, as you contracted with them, and that contract is going to have a duration. When it's over, make sure they're all paid in full so they can sign off and leave. But internal resources remain, so you have to remind yourself that their time on the project is also limited, and you might be blocking other team's projects if you don't release your resources once the project is done.

4. Archive Documents

There are lessons to be learned from old projects, which is why you meet with your team regularly during the project and look back on the process afterwards. However, if you don't have an archive in which to pull the old records, then whatever knowledge you gain is lost because of poor organization and management. You worked hard to have great project documentation, don't lose it.

Before you close a project, archive all the documents and any notes and data that could prove useful. Even if you never access it, there's a need to keep a paper trail of the work done on any project for other people in the organization. This might include legal teams, or HR teams, or even your successor. You never know when someone might have to go back and respond to a question or want to learn how an old issue was resolved. Consider it like putting away provisions for the winter.

5. Celebrate Success

If it sounds silly to you, then you're not doing your job. There's nothing silly about rewarding your team to acknowledge a job well done. It creates closure, which is what this part of the project is all about, but it also plants a seed that will bloom in later projects when you work with members of the old team.

That's because when you note a job well done, you're building morale. It makes team members feel better. You might have been a hard taskmaster in the project, but you give them their due for a job well done. That creates loyalty, and they're going to work even harder for you the next time. And there will be the next time because a happy team is a team that you retain. Why would you want to close a project and lose the very resources that made it a success? Loosen up!

Project Closure Checklist

To make sure that every i is dotted and t crossed, follow this step-by-step project closure checklist.

1. Start at the beginning with the project scope document you created and make sure that you've met all the requirements listed there.
2. Make sure that all deliverables have been handed off and signed by stakeholders, getting their approval and satisfaction. Keeping track

of all those deliverables can be confusing unless you're using project management software. Project Manager has a board view that gives you transparency into the process so you can see that everything has been handed off. Customizable columns allow you to add sign-off as a step to make sure stakeholders have approved the deliverable.

3. Other project documents must also be signed by the appropriate person; this includes any outstanding contracts and agreements with vendors and other contractors.
4. Once documents are signed off on, then process them and pay off all invoices and close out any project-related contracts.
5. Add all documents together, including finalizing all project reports, then organize and archive them as historical data to be used for future reference.
6. Use collected paperwork to identify and document the lessons learned over the course of the project, including any feedback from stakeholders, so you don't make the same mistakes in future projects.
7. Assign a transition support person to shepherd the project after completion so that the project closure is thorough.
8. Release or reassign the project resources, which includes your team and other project personnel and any equipment or site rentals used for the project.
9. If you've not used a project management software, get one, as it helps control not only the life cycle of the project but also the process of closing the project thoroughly.
10. Finally, but perhaps most importantly, celebrate with your project team. They did the work and deserved credit and an opportunity to blow off steam until the next project is started.

15.2 .1.6 Importance of closing a project

At first glance, it might seem like completing the first four phases of the project lifecycle would be all you need to do to tie up your project and call it good.

However, without a formal closing process, you risk letting crucial details fall through the cracks, which can result in confusion, a never-ending project, dissatisfied clients, and even liability issues.

Project closure helps avoid:

- Repeating mistakes on future projects and objectives
- Having final products or deliverables without dedicated support and resources
- Failing to identify the team or individuals who will own and maintain the solution following final delivery

- Creating liability issues resulting from incomplete payments, contracts, or deliverables

Following a clear project closure plan helps you properly transition your solution to the client or end-user. This process ensures the final stakeholders have the information, resources, and training to successfully manage and use the product.

The project closure process also ensures the project is formally completed and is no longer considered a project, allowing you to hand the reins over to the correct team in charge of managing and maintaining the project's outputs.

By officially closing a project, you minimize risks, increase client satisfaction, and ensure all parties are on the same page. In other words, project closure is a process you can't afford to skip.

15.2.2 Financing and Financial Closure

Financial close occurs when all the project and financing agreements have been signed, all conditions on those agreements have been met, and the private party to the PPP can start drawing down the financing to start work on the project. As noted in **Yescombe**, financial close conditions are often circular—the PPP contract does not become effective until funding is available for draw down (that is, funding availability is a Condition Precedent for contract effectiveness), and vice versa.

- One of the important stages in project life cycle is to decide the 'sources of finance' and finalize the 'financial closure'.
- During project life cycle various sources of finance are available which carry different terms, conditions, rules, maturity period, repayment schedules so study of these factors and

thereby deciding the sources of finance is very crucial.

15.2.2.1 Importance of Project Financing

- Project Cost
- Cost of project will be affected by sources of funds carrying lower cost, easy and timely availability.
- Cost over-run and time
- Cheap and easily available over-run finance will facilitate timely completion of project.
- Better profitability
- Low cost of finance will improve profitability.
- Quicker financial closure
- Easy mix of sources of financing will help quicker financial closure.

- Low interest burden, low
- Other benefits depreciation, income tax, pricing of product, repayment of loan.

15.2.2.2 Important issues in Financing of Project

- Modes of Finance
- Cost of Capital & Capital Structure
- Lending policies and appraisal norms of financial institutions
- Project financing and venture capital
- Dividend policies
- Corporate taxation and its impact
- Exchange risk management
- International Financial Management
- Leverage analysis and financial decisions etc.

The financial close is an arduous process that finance and accounting staff dread. Tracking down information from other departments, ensuring all transactions have been recorded, and identifying and correcting errors are time-consuming, labor-intensive tasks requiring long hours and overtime. Pressure to close the books quickly so financial statements can be delivered on time, combined with the need for accuracy and intense focus on compliance only add to the stress.

Fortunately, there are several steps companies can take to save time, reduce errors and increase efficiency.

15.2.2.3 The keys to a faster close (Financial)

1. **Define and assign.** Document every step in the process and the tasks required to complete them. Your list should indicate when each task needs to be completed and, where there are dependencies, in what order. Next, assign responsibility for completing individual tasks to specific people within the department and hold them to a deadline. This is the easiest and most basic step a company can take to improve the close process and should be standard practice. Unfortunately, it typically isn't.

Accounting leaders often assume people on their team know what they need to do, so tasks are managed informally. This approach can work in a small company with only one or two bookkeepers, but it won't scale as a company grows and the number of transactions being processed each month increases.

2. **Reconcile accounts more frequently.** Account reconciliation is a fundamental part of the accounting process and helps to ensure the transactions are recorded accurately to the correct accounts and in accordance with generally accepted accounting principles (GAAP).

But while reconciliation is one of the easiest ways to identify errors, many organizations only review their accounts monthly. That's partly because more frequent reconciliation isn't mandated, and partly because it's usually a time consuming, largely manual task.

Reconciling accounts at the end of the month, however, only delays the inevitable, as any errors that are found must be corrected. This increases the effort required to close the books and ultimately delays the process. Performing reconciliations more frequently means errors are spotted sooner and can be addressed before the close process begins, saving time. The challenge is finding a way to do so. Automating the process is the key.

- 3. Minimize data entry.** Accounting demands accuracy, yet most accounting departments continue to enter vendor invoices, customer payments and other data manually. And the risk of errors increases with every keystroke. Since computer keyboards aren't going away anytime soon, it's impossible to eliminate data entry errors. They can be reduced, however.

Start by avoiding paper wherever possible. Request that vendors send invoices electronically, in XML format for instance, so they can be imported directly into your accounting system. For suppliers that lack these capabilities, consider purchasing a dedicated scanner with optical character recognition (OCR) software.

Next, stop using spreadsheets to manage allocations, depreciation, and other calculations. Sure, they're convenient and relatively easy to use, but spreadsheet data still must be entered into your accounting system manually, which not only increases the risk of errors, but also takes time.

- 4. Simplify the chart of accounts.** A bloated chart of accounts invites errors. While a bookkeeper who's spent decades with the same company might be able to remember hundreds of different account codes and all their permutations, most people can't.

Coding errors may not happen every day, but they happen, and probably more often than anyone realizes. And, as the number of monthly transactions increases, the more significant those occasional errors become. A 0.5% error rate isn't a big deal when there are only 1,000 transactions. At 10,000, however, it means 50 transactions recorded incorrectly. Correcting that many mistakes take hours, if someone even spots the errors.

Of course, a company's chart of accounts becomes complex for a reason – usually because the business needs to track operational performance and the accounting system is the only way to do it. So, before the chart of accounts can be cleaned up, you need to find another way of tracking performance.

- 5. Improve access to information.** Waiting for information from other departments is a common source of frustration and delay. To close the books, for instance, accounting needs to know how much

revenue was generated. This requires gathering data from sales, project management, shipping, and anyone else who influences revenue. They may also need information on fixed assets, inventory, or other data for reporting.

Unless accounting staff have access to the systems where this information is stored, which is seldom the case, they must rely on their peers for updates. Managers in other parts of the organization don't always share the accounting team's sense of urgency or attention to detail, however. Information frequently comes in at the last minute, requires clarification or is incomplete.

6. **Automate intercompany consolidation.** Managing the close process is even more challenging for companies with subsidiaries, especially if there are multiple ERP or accounting systems to contend with. The close process can't be completed at the corporate level until each subsidiary has closed its own books. Then, data from each of those businesses must be pulled together, put into a common format and mapped to the correct fields to allow accurate reporting. Intercompany transactions must also be identified and eliminated. Only then can the parent company finish closing its own books and produce consolidated financial results.

The intercompany consolidation process is by far the most difficult and time-consuming aspect of the accounting cycle. It is often done manually, using spreadsheets. Once again, spreadsheets aren't the ideal way to manage complex calculations. They're also a poor choice for handling large volumes of data and, more importantly, they don't provide a record of data or formula changes. If mistakes are made, spreadsheet can't be audited to see what went wrong.

15.2.3 Project Closeout Report

Project Closure report helps you take the steps needed to formally wind-up your project.

The report helps you undertake the *Project Closure* phase within a project, by documenting all of the tasks needed to complete your project and hand over the deliverables to your customer.

It is critical that you complete the Project Closure phase properly, as the manner within which these closure steps are taken will determine the final success of your project.

*Using this **Project Closure Report** you can perform Project Closure by:*

- Identifying the project completion criteria
- Listing any outstanding activities or deliverables
- Creating a plan for passing deliverables to your customer
- Planning the handover of project documentation
- Closing supplier contracts and agreements

- Releasing projects resources to the business
- Communicating the closure of the project

This Project Closure Report is unique because it:

- Includes pre-formatted sections and tables
- Lists all of the key activities needed to close a project
- Contains step-by-step instructions to help you complete it
- Has lots of practical examples, tips and hints
- Is pre-completed to save you time and effort

Written by project experts, this Project Closure Report helps you to document all of the steps needed to close your projects quickly and efficiently.

15.2.3.1 What is a Project Closure Report?

A *Project Closure Report* describes how you intend to close your projects. The Project Closure Report confirms that the objectives have been met, the deliverables have been handed over to the customer and that project closure can commence. Every Project Manager needs to complete a Project Closure Report to gain agreement from their Sponsor that the project is ready for closure. Once the Project Closure Report has been approved, the Manager can proceed with the actions needed to close the project swiftly.

15.2.3.2 When do I use a Project Closure Report?

A *Project Closure Report* should be documented any time that a project is ready for closure. Using this Project Closure Report, you can document the actions needed to perform project closure immediately. This Project Closure Report already includes the sections, tables, and practical examples you need, to save you time.

15.2.3.3 Eight Steps to Writing a Project Closure Report

Every project no matter how complex it may be will eventually come to the end of its lifecycle. One of the most significant documents that have to be submitted once a project reaches its end is the Project Closure Report.

The Project Closure Report is the final document produced upon the completion of a project. The report details everything to do with the project is often used by the various stakeholders involved in the project to assess the success of the project. Besides the assessment of the project's success, the document is also an invaluable tool to use for identifying the best practices to ensure that all future projects go on smoothly.

Writing a Project Closure Report is not as simple as it seems. There are key steps to be followed. The steps relate to a specific part of the project and must be followed to the dot for effective results. Here are the steps to help you write your Project Closure Report.

1. *Give the Project Overview Including A Summary Statement*

The first step to writing this project closing document is to give your general overview of the whole project and the summary statement. An overview statement is a brief description of what the project was about. It looks at the ‘what’ side of a project. It looks at what needed to be done during the project and how it was done. In addition, an overview investigates and describes things like the Opportunity/Problem, Goal, and Objective, Success Criteria and any risks or assumptions about the project.

On the other hand, the summary statement of the project in a Project Closure report will be looking at the overall summary of what’s in the report. One important thing to note is the key difference between the project overview and the summary statement. The overview is about the project, its scope, and the activities that were done and the summary statement is about the report itself and things contained in the report.

2. *Describe the Results and Outcomes Of The Project*

Before you set out to do your project, chances are, you first wrote down your key performance and indicators and key targets. In addition to the KPI’s, another thing you probably had was outcome targets. On this section, the goal is to look at the whole project in relation to the Key Performance Indicators that you would have set and see the outcomes achieved from that.

What are the project outcomes? Project outcomes refer to the level of performance or achievement that would have occurred due to the activities of the teams on the project. However, measuring project outcomes correctly is not an easy task. There are three metrics that you can use to determine if your project outcomes were positive or negative. The metrics are stakeholder satisfaction, project cost, and overall quality of the project.

3. *Describe the Project Scope, Project Schedule, And Project Cost*

This step is closely related to the above but independent in its own way when it comes to your project closure report. Defined, the project scope is the part of a project where you document the specific goals, deliverables, features, deadlines, and the tasks of a project. It looks at everything that’s needed to get a project through from beginning to completion. This part of the Project Closure Report will look at the overall scope of the Project in relation to the actual project schedule and ultimately the cost.

When a Scope analysis is initially done, everything including the project costs is factored into the initial analysis. These figures, however, will vary and shift as the project goes which is why it’s important to do such a comparison of the actual vs the targeted costs to see whether you ended up going over budget or remained under budget.

4. *Project Performance Analysis*

The project performance analysis can easily qualify as the most important step of the whole project closure report. The performance analysis expands on step 3 and really dives deep into the budget and compare the actual costs and schedule of the project with the set baseline. To be effective, the performance analysis must be subdivided into three parts namely, the Goals and Objective Performance, the Success Criteria Performance and the Schedule and Budget Performance.

i. *Goals and Objectives Performance*

Before you began the project, in the project outline, what were the set goals and objectives for the project? What did you hope to have achieved by the end of the project? How many of those goals have actually been achieved? In addition, how many of those have had to be revised as the conditions on the ground changed?

The questions above are some of the key questions that should be asked when looking at the Goals and Objectives Performance Analysis of the project.

ii. *Success Criteria Performance*

The success Criteria is the one that deals directly with the KPI's. One thing great project manager do before they embark on a new project is to define success before the project begins. The definition of success for a project can differ from one project to the next. Therefore, you must look at how you defined the success of your project and check to see whether you got there.

iii. *Schedule and Budget Performance*

Lastly, under performance analysis, you will have to look at your Schedule and Budget Performance. On your project Scope, what were the set deadlines? Did you meet those deadlines? If not, what were the main reasons for the failure to meet the deadlines? The same questions will apply when you look at your budget. Was it enough or did you have to go to the bank or client for more funding?

In any case, the key thing would be to analyze and compare your actual performance with your set targets.

5. *Project Highlights (Important Aspects of The Project)*

The project Highlight Section looks at the highlights of the whole project throughout the whole timeline. It usually includes high-level project information such as the requests and any other issues that arose within the project.

Compiling the highlight report and adding it to the project closure report should not be hard. It is recommended that you should at least make a highlight report at the end of each week throughout the

course of a project updating the different stakeholders involved in the project of the project's current progress.

If you have those reports, then you can just refer to them and pick the key points from each of the weekly reports to combine them into one master report to include in your final Project Closure Report.

6. *Write and Outline the Challenges Faced and Risks*

Every Project has its challenges and risks. This section will enable you to highlight all the challenges that might have been faced throughout the course of the project. One thing about challenges especially in relation to projects and project management is that they can be difficult to foresee. Apart from that, no matter how carefully you plan at the inception of the project, you can never plan around every potential challenge.

For future reference and presentation to stakeholders, you should use this section to highlight every challenge you faced throughout the course of the project. In addition to listing down the challenges, you should also highlight how the challenge affected other aspects of your project including your budget and schedule.

Besides the challenges, you should also highlight the risks faced. Risks can be anything from the weather, workplace safety, or even money.

7. *Write About the Lessons Learned During Implementation*

One source of valuable lessons for any project are the challenges. When you overcome the challenges faced when doing a process, chances are, you will learn one or two things. Use this section of your Project Closure Report to highlight what you learned.

During the project, you will also be working with different stakeholders from different industries. Sometimes these stakeholders can teach you different techniques to help work get done faster which is valuable. If you learned such techniques from the various stakeholders, you would have worked with throughout the project implementation, then you use this section to those lessons.

The reason why it's important to note down the lessons learned in this project closing document is that later on when doing another project, you can always reference the report of your previous project to look for common pitfalls and how you can avoid those pitfalls.

Conclusion

A Project Closure Report is an important document that signifies the formal project closing. One thing to remember when working on the report is to pay attention to detail especially on performance analysis. Paying

attention to detail especially when a project goes over budget will help you avoid falling into the same pitfalls in the future.

15.3 SUMMARY

In this Project Closeout chapter, you learned about what is project closeout ?, HOW To Close a Project? Reasons for Project Closure, Project Closure Process, performing a Financial Closure, Project Closeout Report You also saw the Importance of closing a project, along with that, Steps to Writing a Project Closure Report ,you saw Types of Project Closure, Project closure helps avoid mistake etc. Still, you had a doubt go through references and bibliography

15.4 EXERCISE: -

- Q.1 What are the five major activities for closing a project?
- Q.2 Write Reasons for project closure.
- Q.3 Write Importance of project closure.
- Q.4 Write types of Project Closure.
- Q.5 Write Steps to Closing a Project.
- Q.6 What is Financial Closure?
- Q.7 Write Important issues in Financing of Project.
- Q.8 Write the Keys to a Faster financial Close
- Q.9 *What is a Project Closure Report?*
- Q.10 Write steps to writing a Project Closure Report.

15.5 LIST OF BOOKS AND REFERENCES

- 1. Software Project Management Bob Hughes, Mike Cotterell, Rajib Mall TMH 6 th 2018
- 2. Project Management and Tools & Technologies – An overview Shailesh Mehta SPD 1st 2017
- 3. Software Project Management Walker Royce Pearson 2005
- 4. <https://blog.spendsesk.com>
- 5. <https://www.netsuite.com>
- 6. <https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Technology/gx-cons-tech-sap-five-signs-financial-close-process-broken.pdf> (for financial closure)
- 7. <https://www.method123.com>

