Project Planning Scheduling & Tracking

Marwadi University

Computer Engineering Diploma

Unit 3:-
Project Planning Scheduling & Tracking

Software Engineering (09CE2402)

Software Scope

- First step in project planning is to define a clear software scope.
- Project scope simply defines what your software will do and will not do.
- If your project scope is poorly defined then there are chances that your project will eventually end up in failure.

Software Scope includes

- The functions and features that are to be delivered to the end users.
- The data that are to and output from the system.
- The “content” that is presented to the users as a consequence of using the software.
- The performance, constraints, interfaces and reliability that bound the system.

How to define Software Scope?

- A narrative description of software scope is developed after communication with all stockholders.
- A set of use cases are developed by analyst.

How to define Software Scope?

- Define the software requirements.
- Involve all the stackholders.
- Define the functionalities of software.
- Identify the limitations. (out of scope)
- Never leave ambiguity. (doubt)

Points to be covered in Software Scope

❖ **Project Aim & Purpose**

- Short statement that captures exactly what project has to accomplished.
- (e.g. implement a new education management system which will be used throughout the institute to manage everything online)

Points to be covered in Software Scope

❖ **Project Objective (Scope)**

- Include the details what will be delivered as a result of project.
- (e.g. software will include the system for attendance, e-content management, student performance, student result, etc.)

Points to be covered in Software Scope

❖ **Out of Scope**

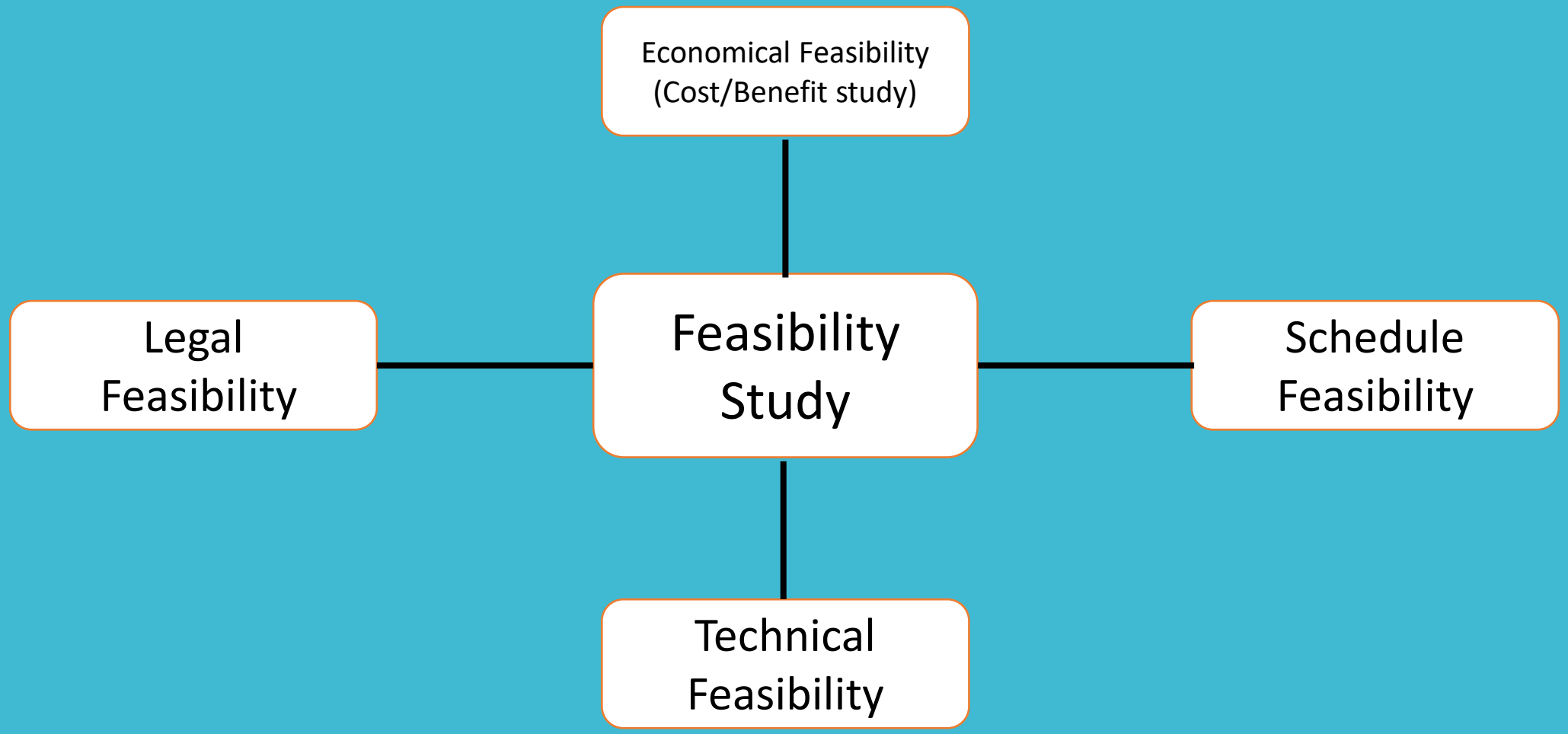
- Define what will not be delivered as a part of the system.
- E.g. the system will not include the functionality to send the attendance SMS to parents.
- E.g. Software does not include the functionality to compare multiple products.

Feasibility Study

Feasibility Study

- The objective behind the feasibility study is to create the reasons for developing the software that is acceptable to users, flexible to change and conformable (similar) to established standards.

Feasibility Study



- A feasibility study is an analysis that takes all of a project's relevant factors into account—including economic, technical, legal, and scheduling.

Feasibility Study – First Step

- Roughly understand what customer wants
- Data which would be the input to the system
- Processing needed on those data
- Output data to be produced by the system
- Various constraints on the behaviour of system

Activities During Feasibility Study

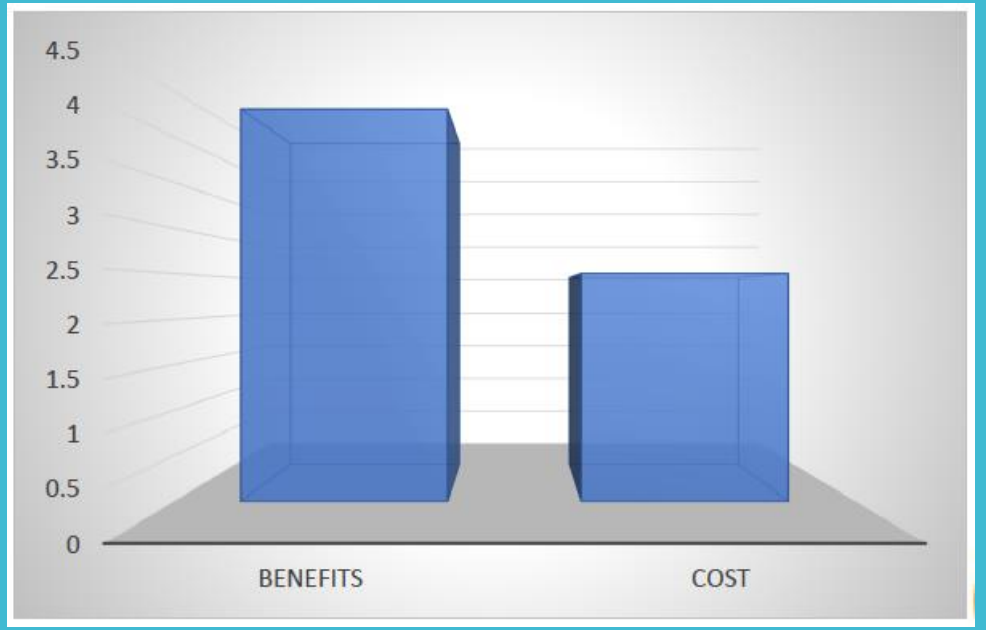
- Workout on overall understanding of the problem
- Formulate different solution strategies
- Examine alternate solution strategies in terms of
    1. Resource required
    2. Cost of development
    3. Development time

Cost/Benefit Analysis

- Need to identify all costs – these could be:-
    - Development cost
    - Set up costs
    - Operational costs
- Identify the value of benefits
- Check benefits weather benefits are greater than costs

Cost/Benefit Analysis

- Benefits of delivered project must outweigh cost
- **Cost includes:-**
  - Development
  - Operation
- **Benefits:-**
  - Quantifiable (Which can be measure)
  - Non-Quantifiable (Which can’t be measure)



Types of Feasibility

Types of Feasibility

1. Technical Feasibility
2. Scheduled Feasibility
3. Legal Feasibility
4. Operational Feasibility
5. Economic Feasibility

Technical Feasibility

- Check weather the software is technically feasible to develop or not.
- Technical feasibility may include
  - What are the different technologies available to develop the software?
  - Are all the technology available to develop or not?
  - Are the manpower have technical skills?
  - Can it be achieved by training?

Scheduled Feasibility

- Scheduled feasibility includes whether the software is possible to deliver in time expected by the customer or not?

Legal Feasibility

- In this part, it is checked whether the requirements like data protection acts or social media laws of software are legal to make or not according to current jurisdiction.

Operational Feasibility

- Operational feasibility assesses the range in which the required software performs a series of levels to solve business problems and customer requirements.

Economic Feasibility

- Economic feasibility decides whether the necessary software can generate financial profits for an organization.

Estimation of Project Cost & Effort

Estimation of Project Cost & Effort

Cost in project includes:-

- Hardware and Software costs
- Training cost
- Effort cost

Estimation of Project Cost & Effort

- **Hardware Cost:-**
  - Computer cost, Server cost, Networking cost, etc
- **Software Cost:-**
  - Software cost includes cost for various tools used for development and deployment of software.
- **Training Cost:-**
  - Sometimes team need some training to develop the software.
  - This training cost should be included in the estimation of project cost.

Estimation of Project Cost & Effort

- **Effort Cost:-**
  - Costs of providing lighting to office space.
  - Costs of support staff such as accountants, administrators, system managers, cleaners and technicians.
  - Costs to operate central facilities such as library or recreational facilities.
  - Cost of Social security and employee benefits such as pension and health insurance.

Empirical Estimation Model

Empirical Estimation Model

- A typical estimation model is derived using regression analysis on data collected from past software projects.
- The empirical data that support most estimation model is derived from a limited sample of projects.
- Hence, no estimation model is appropriate for all classes of software and in all development environments.

Empirical Estimation Model

- It includes three approaches for Estimation:-

  (1) LOC Based estimation

  (2) FP Based estimation

  (3) COCOMO Model

LOC Based Model

LOC Based Model

- Software project estimation is a form of problem solving (i.e, developing a cost and effort estimate for a software project)
- For this reason, you should decompose the problem, characterizing it as a set of smaller problems.

LOC Based Model

- One commonly used function for effort estimation is

  Effort =a*(size)^{b}

  •

  Where **a** and **b** are some constants and size of the project
- Which is in terms of kilo Lines of code (KLOC)
- Values of a and b are constants i.e.,

  Effort = 3*(size)$^{1.2}$
- The effort is denoted in terms of Person/month (PM)
- i.e., One person's working time for a month

LOC Based Model - Example

- To develop a CAD (Computer Aided Design) like application for mechanical engineering company
- First estimate the total KLOC of the project.
- To calculate total KLOC of the project, decompose the software into many small parts and estimate the total KLOC of project by adding KLOC of each module of the project.

LOC Based Model - Example

Function	Estimated LOC
User interface and control facilities (UICF)	2,300
Two-dimensional geometric analysis (2DGA)	5,300
Three-dimensional geometric analysis (3DGA)	6,800
Database management (DBM)	3,350
Computer graphics display facilities (CGDF)	4,950
Peripheral control function (PCF)	2,100
Design analysis modules (DAM)	8,400
Estimated lines of code	*33,200*

LOC Based Model - Example

Effort = 3*(size)^0.91

Effort = 3*(33.2)^0.91

Effort = 72.67

Effort = ~73 (Person-Month)

LOC Based Model - Example

- This means it will take 73 month if developed by 1 developer.
- If we have a team of 25 developer, we can calculate the estimate time by dividing per month effort by team size.
- Time = 73/25 = 2.92 Month

LOC Based Model - Example

- Now its time to estimate the cost.
- Take average monthly salary cost of 1 developer per month (e.g. 22,000).
- So cost of 25 developer for a month will be
- 22,000*25=5,50,000 per month
- Multiply it to estimated time,
- Total Cost = 5,50,000*2.92
- Total Cost = 16,06,000/-

LOC Based Model - Practice

- Below is the LOC for Online shopping website
  - GUI = 3200
  - Backend = 2200
  - Database = 800
  - Payment = 500
- If team size is 5 person and average cost to operate one person per month is 20000, find the cost and time to develop an online shopping website.
- Value of a=3.1 and b=0.9

Practice Answer

- Total Code = 3200+2200+800+500 =6700 = 6.7 KLOC
- Effort 3.1*6.7 ^0.9 = 17.17 =~17 PM
- Cost to operate team = 20000*5 = 100000
- Time = 17/5 = 3.4 Month
- Cost = 100000*3.4 = 3.4 Lakhs

LOC Based Model – Example-2

- Below is the LOC for Management System
  - GUI = 4300
  - Backend = 3500
  - Database = 700
  - Reports = 500
  - User Records = 400
- If team size is 4 person and average cost to operate one person per month is 21000, find the cost and time to develop an online shopping website.
- Value of a=3.0 and b=1.2

Practice Answer

- Total Code = 4300+3500+700+500+400 =9400 = 9.4 KLOC
- Effort 3.0*9.4 ^1.2 = 44.14 =~44 PM
- Cost to operate team = 21000*4 = 84000
- Time = 44/4 = 11 Month
- Cost = 84000*11 = 9.24 Lakhs

FP(Functional Point) Based Model

Functional Point Analysis

- Function Point Analysis (FPA) is a method or set of rules of Functional Size Measurement.
- It assesses the functionality delivered to its users, based on the user's external view of the functional requirements.

Objectives of Functional Point Analysis

- The objective of FPA is to measure functionality that the user requests and receives.
- The objective of FPA is to measure software development and maintenance independently of technology used for implementation.
- Further, it is used to measure the software project development along with its maintenance, consistently throughout the project irrespective of the tools and the technologies.

Types of Functional Point Analysis

Types of FPA

- There are two types of FPA (Functional Point Analysis)

  (1) Transactional Functional Type

  (2) Data Functional Type

TRANSACTIONAL FUNCTIONAL TYPE

(i) External Input (EI):

EI processes data or control information that comes from outside the application's boundary. The EI is an elementary process.

(ii) External Output (EO):

EO is an elementary process that generates data or control information sent outside the application's boundary.

(iii) External Inquiries (EQ):

EQ is an elementary process made up of an input-output combination that results in data retrieval.

Data Functional Type

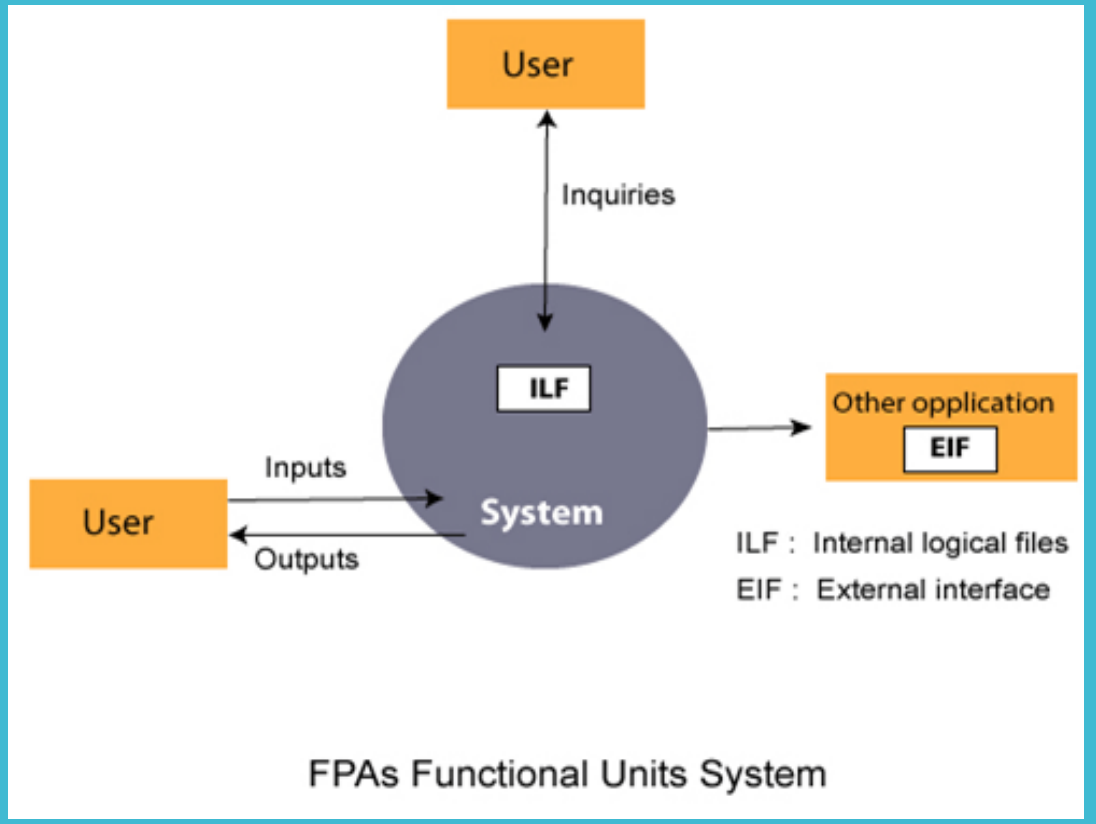
(i) Internal Logical File (ILF):

A user identifiable group of logically related data or control information maintained within the boundary of the application.

(ii) External Interface File (EIF):

A group of user recognizable logically related data allusion to the software but maintained within the boundary of another software.

FPA Functional Units



FPAs Functional Units System

Measurement Parameters

Sr. No.	Measurement Parameters	Examples
1	Number of External Inputs(EI)	Input screen and tables
2	Number of External Output (EO)	Output screens and reports
3	Number of external inquiries (EQ)	Prompts and interrupts.
4	Number of internal files (ILF)	Databases and directories
5	Number of external interfaces (EIF)	Shared databases and shared routines.

All these parameters are then individually assessed for complexity.

Benefits of FPA

- FPA is a tool to determine the size of a purchased application package by counting all the functions included in the package.
- It is a tool to help users discover the benefit of an application package to their organization by counting functions that specifically match their requirements.
- It is a tool to measure the units of a software product to support quality and productivity analysis.
- It s a vehicle to estimate cost and resources required for software development and maintenance.
- It is a normalization factor for software comparison.

Differentiate between FP and LOC

Sr. No.	Function Point (FP)	Line of Code (LOC)
1	FP is specification based	LOC is an analogy based
2	FP is language independent	LOC is language dependent
3	FP is user-oriented	LOC is design-oriented
4	It is extensible to LOC	It is convertible to FP (backfiring)

COCOMO Model

COCOMO Model

- Boehm proposed COCOMO (Constructive Cost Estimation Model) in 1981.
- COCOMO is one of the most generally used software estimation models in the world.
- COCOMO predicts the efforts and schedule of a software product based on the size of the software.

Necessary Steps in COCOMO Model

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 multiplying 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 step1 and step2.

- Ei=a*(KDLOC)b
- The value of the constant a and b are depends on the project type.

Project Category in COCOMO Model

- (1) Organic
- (2) Semidetached
- (3) Embedded

(1) Organic Category - COCOMO Model

- A development project can be treated of the organic type, if the project deals with developing a well-understood application program, the size of the development team is reasonably small, and the team members are experienced in developing similar methods of projects.
- **Examples of this type of projects are simple business systems, simple inventory management systems, and data processing systems.**

(2) Semidetached Category - COCOMO Model

- A development project can be treated with semidetached type if the development consists of a mixture of experienced and inexperienced staff.
- Team members may have finite experience in related systems but may be unfamiliar with some aspects of the order being developed.
- **Example of Semidetached system includes developing a new operating system (OS), a Database Management System (DBMS), and complex inventory management system.**

(3) Embedded Category - COCOMO Model

- A development project is treated to be of an embedded type, if the software being developed is strongly coupled to complex hardware, or if the stringent regulations on the operational method exist.
- **For Example: ATM, Air Traffic control**

Types of COCOMO Model

- For three product categories, Bohem provides a different set of expression to predict effort (in a unit of person month) and development time from the size of estimation in KLOC(Kilo Line of code) efforts estimation takes into account the productivity loss due to holidays, weekly off, coffee breaks, etc.

- **According to Boehm, software cost estimation should be done through three stages:**

1) **Basic Model**
2) **Intermediate Model**
3) **Detailed Model**

(1) Basic COCOMO Model

The basic COCOMO model provide an accurate size of the project parameters.

The following expressions give the basic COCOMO estimation model:

Effort=a1*(KLOC) a2 PM
Tdev=b1*(efforts)b2 Months

Where,

- KLOC is the estimated size of the software product indicate in Kilo Lines of Code,
- a1,a2,b1,b2 are constants for each group of software products,
- Tdev is the estimated time to develop the software, expressed in months,
- Effort is the total effort required to develop the software product, expressed in person months (PMs).

(1) Basic COCOMO Model

The values of a, b and c are as follows:

Software Project	a	b	c
Organic	2.4	1.05	0.38
Semidetached	3.0	1.12	0.35
Embeded	3.6	1.20	0.32

Estimation of development effort

For the three classes of software products, the formulas for estimating the effort based on the code size are shown below:

Organic: Effort = 2.4(KLOC) 1.05 PM

Semi-detached: Effort = 3.0(KLOC) 1.12 PM

Embedded: Effort = 3.6(KLOC) 1.20 PM

(1) Basic COCOMO Model

The values of a and b for Tdev are as follows:

Software Project	a	b
Organic	2.5	0.38
Semidetached	2.5	0.35
Embeded	2.5	0.32

Estimation of development time

For the three classes of software products, the formulas for estimating the development time based on the effort are given below:

Organic: Tdev = 2.5(Effort) 0.38 Months

Semi-detached: Tdev = 2.5(Effort) 0.35 Months

Embedded: Tdev = 2.5(Effort) 0.32 Months

Example 1 – Basic Model

- Suppose a project was estimated to be 400 KLOC. Calculate the effort and development time for each of the three model i.e., organic, semi-detached & embedded.

- Solution: The basic COCOMO equation takes the form:

Effort=a1*(KLOC) a2 PM
Tdev=b1*(efforts)b2 Months
Estimated Size of project= 400 KLOC

(i) Organic Mode

E = 2.4 * (400)1.05 = 1295.31 PM
D = 2.5 * (1295.31)0.38=38.07 PM

(ii) Semidetached Mode

E = 3.0 * (400)1.12=2462.79 PM
D = 2.5 * (2462.79)0.35=38.45 PM

(iii) Embedded Mode

E = 3.6 * (400)1.20 = 4772.81 PM
D = 2.5 * (4772.8)0.32 = 38 PM

Example 2 – Basic Model

- A project size of 200 KLOC is to be developed. Software development team has average experience on similar type of projects. The project schedule is not very tight.
- Calculate the Effort, development time, average staff size, and productivity of the project.

- Solution: The semidetached mode is the most appropriate mode, keeping in view the size, schedule and experience of development time.

Hence,

$E=3.0(200)1.12=1133.12PM$

$D=2.5(1133.12)0.35=29.3PM$

Productivity = KLOC/E
= 200/1133.12
= 0.1765 KLOC/PM

Average Staff Size (SS) =(E/D) persons
= 1133.12/29.3 persons
= 38.61 persons

Example 3 – Basic Model

- A project size of 350 KLOC is to be developed.
- Calculate the Effort, development time, average staff size, and productivity of the project for all the categories.

- Solution: The semidetached mode is the most appropriate mode, keeping in view the size, schedule and experience of development time.

Hence,

$E=3.0(200)1.12=1133.12PM$

$D=2.5(1133.12)0.35=29.3PM$

Productivity = KLOC/E

= 200/1133.12

= 0.1765 KLOC/PM

Average Staff Size (SS) =(E/D) persons

= 1133.12/29.3 persons

= 38.61 persons

(2) Intermediate COCOMO Model

- The basic COCOMO model assumes that the effort is only a function of the number of lines of code and some constants evaluated according to the different software system.
- However, in reality, no system's effort and schedule can be solely calculated on the basis of Lines of Code.
- For that, various other factors such as reliability, experience, Capability.
- These factors are known as Cost Drivers and the Intermediate Model utilizes 15 such drivers for cost estimation.

Classification of Cost Drivers and their attributes:

(1) Product attributes –

- Required software reliability extent
- Size of the application database
- The complexity of the product

(2) Hardware attributes –

- Run-time performance constraints
- Memory constraints
- The volatility of the virtual machine environment
- Required turn aroud time

(3) Personnel attributes –

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

(4) Project attributes –

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

(3) Detailed COCOMO Model

- Detailed COCOMO incorporates all qualities of the standard version with an assessment of the cost driver's effect on each method of the software engineering process.
- The detailed model uses various effort multipliers for each cost driver property.
- In detailed COCOMO, the whole software is differentiated into multiple modules, and then we apply COCOMO in various modules to estimate effort and then sum the effort.
- The Six phases of detailed COCOMO are:

(1) Planning and Requirements
(2) System Design
(3) Detailed Design
(4) Module code and test
(5) Integration and test
(6) Cost Constructive model

Gantt Chart

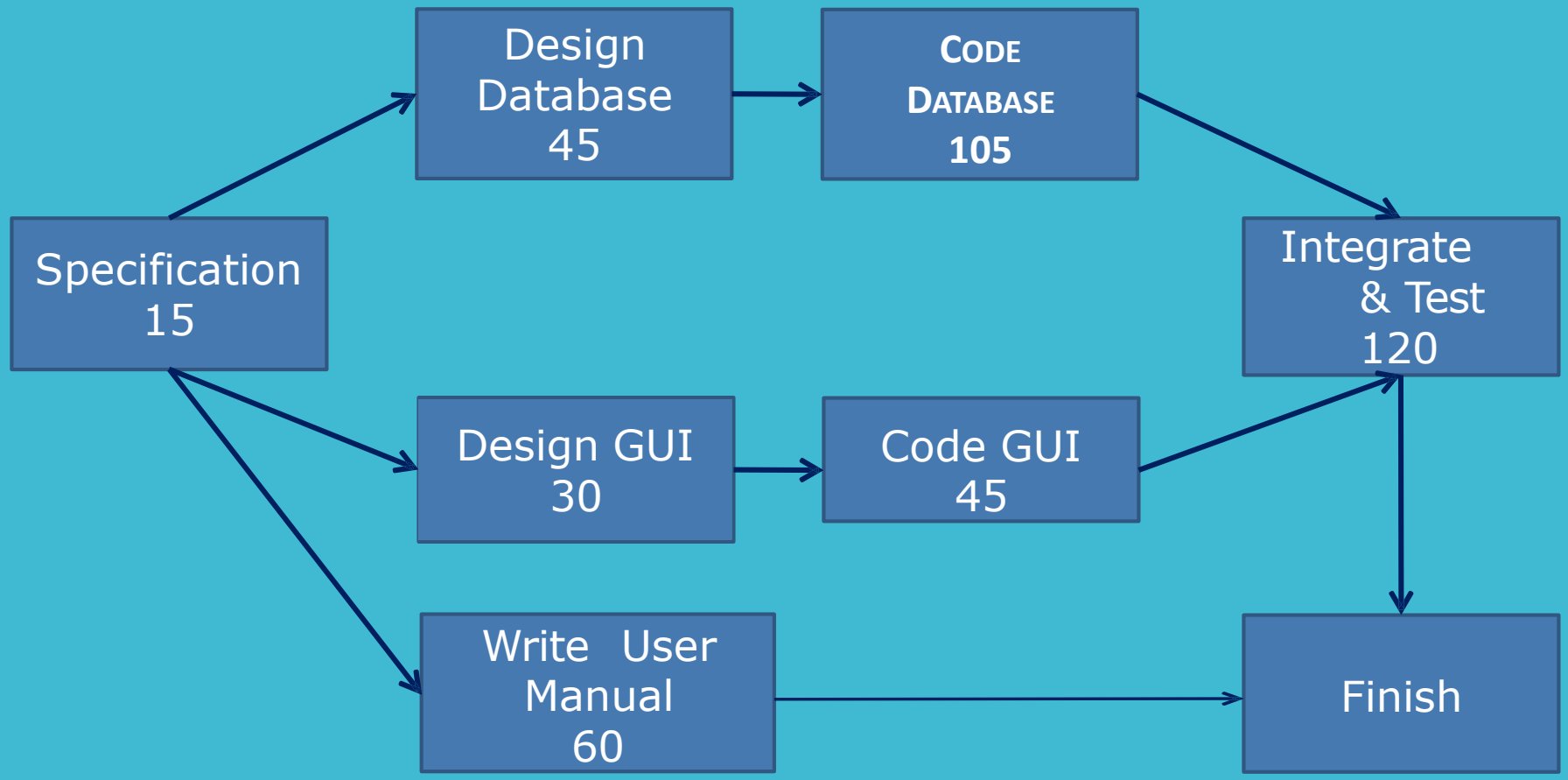
GANTT CHARTS

- Generalized Activity Normalization Time Table (GANTT) chart is type of chart in which series of horizontal lines are present that show the amount of work done or production completed in given period of time in relation to amount planned for those projects.
- Gantt charts have been named from their developer Henry Gantt.
- They are mainly used to allocate resources to activities.
- The resources include:
  - Staff,
  - Hardware, and
  - Software.
- They are useful for resource planning.

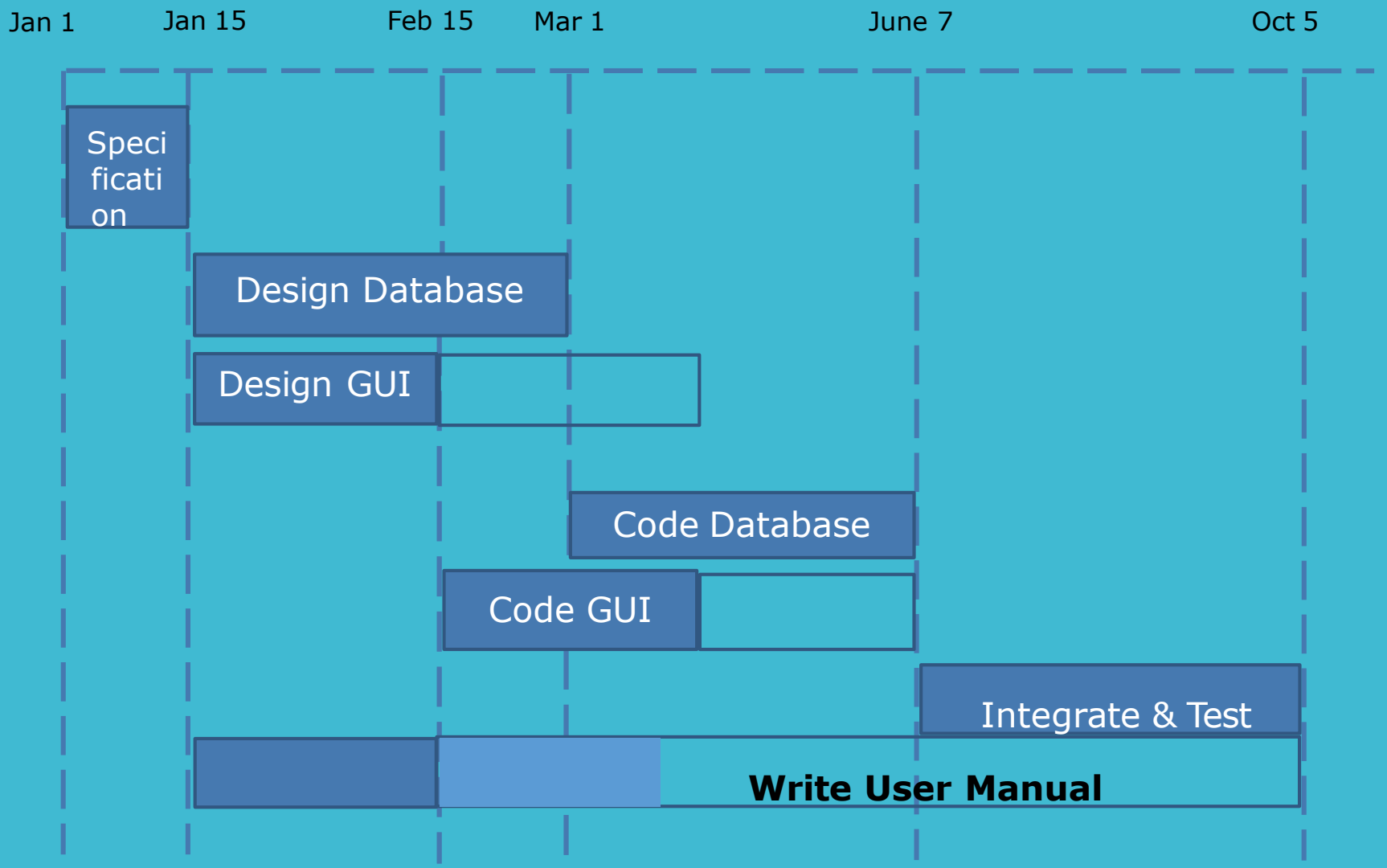
GANTT CHARTS

- A Gantt chart is a special type of bar chart where each bar represents an activity.
- The bars are drawn along a timeline.
- The length of each bar is proportional to the duration of time planned for that activity.
- Each bar consists of a shaded part and a white part.
- The shaded part shows the length of the time each task is estimated to take.
- The white part shows the slack time.

EXAMPLE


Design Database 45
CODE DATABASE 105
Specification 15
Integrate & Test 120
Design GUI 30
Code GUI 45
Write User Manual 60
Finish

Example: Gantt Chart



What is Scheduling ?

- Scheduling is an important part of project planning activity
- It involves deciding which tasks would be taken up when.

WHY SCHEDULING?

- The majority of projects are 'completed' late, if at all.
- A project schedule is required to ensure that required project commitments are met.
- A schedule is required to track progress towards achieving these commitments.

Why Software Delivered Late?

- An unrealistic deadline
- Changing but unpredicted customer requirements
- Underestimation of efforts needed
- Risks not considered at the project start
- Unforeseen technical difficulties
- Unforeseen human difficulties
- Miscommunication among project staff
- Failure to recognize that project is falling behind schedule

Project Scheduling

Project Scheduling

- On large projects, hundreds of small tasks must occur to accomplish a larger goal
- Some of these tasks lie outside the mainstream and may be completed without worry of impacting on the project completion date
- Other tasks lie on the critical path; if these tasks fall behind schedule, the completion date of the entire project is put into failure.

PROJECT SCHEDULING

To schedule the project, a project manager must do the following:

- Define all project tasks
- Build a network that depicts their interdependence.
- Identify the critical tasks
- Track the progress of these tasks
- Recognize the delay “one day at a time” (to deal with each day's problems as they come)

Principle for Project scheduling

- **Compartmentalization –** define distinct tasks
- **Interdependency-**sequence and parallel tasks
  - The interdependency of each activity must be determined
  - Some tasks must occur in sequence while others can occur in parallel
- **Time allocation** - assigned person days, start time, ending time
  - Each task to be scheduled must be allocated some number of work units
  - In addition, each task must be assigned a start date and a completion date

Principle for Project scheduling

- **Effort validation** - be sure resources are available
  - Every project has a defined number of people on the team
  - As time allocation occurs, the project manager must ensure that no more than the allocated number of people have been scheduled at any given time
- **Defined responsibilities** — people must be assigned
  - Every task that is scheduled should be assigned to a specific team member

Principle for Project scheduling

- **Defined Outcomes-** each task must have an output

  Every task that is scheduled should have a defined outcome for software projects such as a work product or part of a work product

- **Defined milestones -** review for quality
  - Every task or group of tasks should be associated with a project milestone
  - A milestone is accomplished when one or more work products has been reviewed for quality and has been approved

Thank You!!!!