Software Engineering

T.E – B

Unit 3

Faculty Incharge: Ms. Drashti Shrimal

Software Project Estimation:

Software development estimation is a process by which one can accurately determine the amount of effort, as in time and money, necessary to deliver or maintain a software-based project. It also approximates effort in human resources required to develop the project.
Various types of Estimation are: Line of Code, Function Points, and COCOMO.

Project Scheduling:

Project-task scheduling is a significant project planning activity. It comprises deciding which functions would be taken up when. To schedule the project plan, a software project manager wants to do the following:

1. Identify all the functions required to complete the project.
2. Break down large functions into small activities.
3. Determine the dependency among various activities.
4. Establish the most likely size for the time duration required to complete the activities.
5. Allocate resources to activities.
6. Plan the beginning and ending dates for different activities.
7. Determine the critical path. A critical way is the group of activities that decide the duration of the project.

Advantages of Project Scheduling:

- Assists with tracking, reporting, and communicating progress.
- Ensures everyone is on the same page with tasks, dependencies, and deadlines.
- Highlights issues and concerns, such as a lack of resources.
- Identifies task relationships.

Q.1 LOC for Software Project Estimation.

A **line of code (LOC)** is any line of text in a code that is not a comment or blank line, and also header lines, in any case of the number of statements or fragments of statements on the line. LOC clearly consists of all lines containing the declaration of any variable, and executable and non-executable statements. As Lines of Code (LOC) only counts the volume of code, you can

only use it to compare or estimate projects that use the same language and are coded using the same coding standards.

Features:

- Variations such as “source lines of code”, are used to set out a codebase.
- LOC is frequently used in some kinds of arguments.
- They are used in assessing a project’s performance or efficiency.

Advantages:

- Most used metric in cost estimation.
- Its alternates have many problems as compared to this metric.
- It is very easy in estimating the efforts.

Disadvantages:

- Very difficult to estimate the LOC of the final program from the problem specification.
- It correlates poorly with quality and efficiency of code.
- It doesn’t consider complexity.

Research has shown a rough correlation between LOC and the overall cost and length of developing a project/ product in Software Development, and between LOC and the number of defects. This means the lower your LOC measurement is, the better off you probably are in the development of your product.

Q.2 FP for Software Project Estimation.

The function point is a "unit of measurement" to express the amount of business functionality an information system provides to a user. Function points are used to compute a functional size measurement of software. The cost of a single unit is calculated from past projects.

Function points **measure the size of an application system based on the functional view of the system**. The size is determined by counting the number of inputs, outputs, queries, internal files and external files in the system and adjusting that total for the functional complexity of the system.

Q.3 COCOMO Model.

Cocomo (Constructive Cost Model) is a regression model based on LOC, i.e. **number of Lines of Code**. It is a procedural cost estimate model for software projects and is often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time, and quality. It was proposed by Barry Boehm in 1981 and is based on the study of 63 projects, which makes it one of the best-documented models. The key parameters which define the quality of any software products, which are also an outcome of the Cocomo, are primarily Effort & Schedule:

- **Effort:** Amount of labor that will be required to complete a task. It is measured in person-months units.

- **Schedule:** Simply means the amount of time required for the completion of the job, which is, of course, proportional to the effort put in. It is measured in the units of time such as weeks, and months.

Different models of Cocomo have been proposed to predict the cost estimation at different levels, based on the amount of accuracy and correctness required. All of these models can be applied to a variety of projects, whose characteristics determine the value of the constant to be used in subsequent calculations. These characteristics pertaining to different system types are mentioned below. Boehm's definition of organic, semidetached, and embedded systems:

1. **Organic –** A software project is said to be an organic type if the team size required is adequately small, the problem is well understood and has been solved in the past and also the team members have a nominal experience regarding the problem.
2. **Semi-detached –** A software project is said to be a Semi-detached type if the vital characteristics such as team size, experience, and knowledge of the various programming environment lie in between that of organic and embedded. The projects classified as Semi-Detached are comparatively less familiar and difficult to develop compared to the organic ones and require more experience and better guidance and creativity. Eg: Compilers or different Embedded Systems can be considered Semi-Detached types.
3. **Embedded –** A software project requiring the highest level of complexity, creativity, and experience requirement fall under this category. Such software requires a larger team size than the other two models and also the developers need to be sufficiently experienced and creative to develop such complex models.
    1. Basic COCOMO Model
    2. Intermediate COCOMO Model
    3. Detailed COCOMO Model

4. **Basic Model –**

$$E = a(KLOC)^b$$

$$time = c(Effort)^d$$

$$Personrequired = Effort/time$$

1. The above formula is used for the cost estimation of for the basic COCOMO model, and also is used in the subsequent models. The constant values a,b,c, and d for the Basic Model for the different categories of the system:

Software Projects	a	b	c	d
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

Advantages of the COCOMO model:

1. Provides a systematic way to estimate the cost and effort of a software project.
2. Can be used to estimate the cost and effort of a software project at different stages of the development process.
3. Helps in identifying the factors that have the greatest impact on the cost and effort of a software project.
4. Can be used to evaluate the feasibility of a software project by estimating the cost and effort required to complete it.

Disadvantages of the COCOMO model:

1. Assumes that the size of the software is the main factor that determines the cost and effort of a software project, which may not always be the case.
2. Does not take into account the specific characteristics of the development team, which can have a significant impact on the cost and effort of a software project.
3. Does not provide a precise estimate of the cost and effort of a software project, as it is based on assumptions and averages.

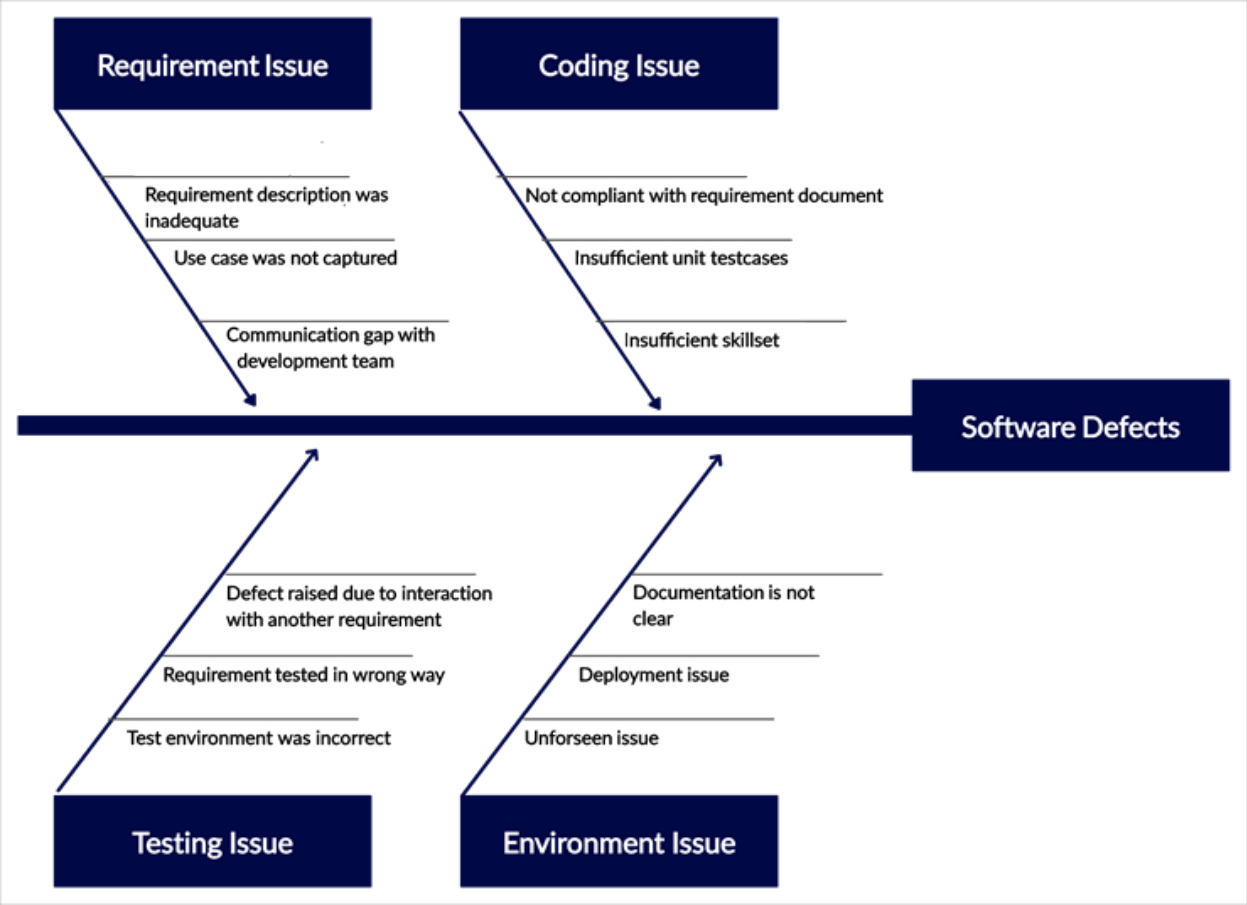
Q.4 Fishbone Diagram.

A fishbone diagram is a visual way to look at cause and effect. It is a more structured approach than some other tools available for brainstorming causes of a problem (e.g., the Five Whys tool). The problem or effect is displayed at the head or mouth of the fish.

Steps to develop fishbone diagram:

1. Agree on a problem statement (effect). Write it at the center right of the flipchart or whiteboard. Draw a box around it and draw a horizontal arrow running to it.
2. Brainstorm the major categories of causes of the problem. If this is difficult use generic headings:
   - Methods
   - Machines (equipment)
   - People (manpower)
   - Materials
   - Measurement
   - Environment
3. Write the categories of causes as branches from the main arrow.
4. Brainstorm all the possible causes of the problem. Ask "Why does this happen?" As each idea is given, the facilitator writes it as a branch from the appropriate category. Causes can be written in several places if they relate to several categories.
5. Again ask "Why does this happen?" about each cause. Write sub-causes branching off the causes. Continue to ask "Why?" and generate deeper levels of causes. Layers of branches indicate causal relationships.
6. When the group runs out of ideas, focus attention to places on the chart where ideas are few.

Example:



Module Takeaway:

1. Define Software Project Estimation.
2. Define Project scheduling and state advantages.
3. Explain Fishbone diagram with example and steps.
4. Explain the COCOMO model.
5. State and explain LOC with example.
6. Define FP in Software engineering.