Review Questions

Q. 1 Explain the terms : measure, metric and indicator.

Q. 2 List various types of metrics.

Q. 3 What is Defect Removal Efficiency ?

Q. 4 Explain Function point metric with an example.

Q. 5 What is the need of OO metrics and explain in brief.

Q. 6 Brief about metric of maintenance.

Q. 7 Explain metrics of software Quality.

Q. 8 Based on which two parameters, the software is measured. Explain.

❑❑❑

UNIT II

CHAPTER 7 Software Project Management Estimation

Syllabus :

Estimation in Project Planning Process – Software Scope and Feasibility, Resource Estimation, Empirical Estimation Models – COCOMO II, Estimation for Agile Development, The Make/Buy Decision

Syllabus Topic : Estimation in Project Planning Process

7.1 Estimation in Project Planning Process

- Planning and estimating are iterative processes which continue throughout the course of a project.
- Software costs often dominate computer system costs. The costs of software are often greater than the hardware cost.
- Software costs more to maintain than it does to develop. For systems with a long life, maintenance costs may be several times development costs.
- The total cost required in developing a software product can be categorized as below :
  - i. 60% of development costs.
  - ii. 40% are testing costs.

Cost estimating problems occur due to following uncertainties

- Inability to accurately size a software project, Inability to accurately specify a software development and support environment.
- Improper assessment of staffing levels and skills, and Lack of well-defined requirements for the specific software activity being estimated.

Four types of cost estimates represent various levels of Reliability

- **Conceptual Estimate :** Often inaccurate because there are too many unknowns.
- **Preliminary Estimate :** Used to develop initial budget, more precise.
- **Detailed Estimate :** Serves as a basis for daily project control.
- **Definitive Estimate :** Accuracy should be within 10% of final cost.

Cost Estimations are constructed based on the following tasks

- Identifying the purpose and scope of the new system - New software development, software reuse, COTS integration, etc.
- Choosing an estimate type - Conceptual, preliminary, detailed, or definitive type estimates.
- Identifying system performance and/or technical goals.
- Laying out a program schedule.
- Collecting, evaluating, and verifying data.
- Choosing, applying, cross-checking estimating methods to develop the cost estimate.
- Performing risk and sensitivity analysis.
- Providing full documentation.

Syllabus Topic : Software Scope and Feasibility

7.1.1 Software Scope and Feasibility

- Software scope is defined as :
  - The functions and features that are to be delivered to end users
  - The input and output data of the system
  - The performance, constraints, interfaces, and reliability of the system
- Software Scope can be defined using two techniques :
  1. A narrative description
  2. A set of use cases
- Once the scope is decided, the feasibility is addressed
- Software feasibility has four dimensions
  1. **Technology :** Is the project technically feasible so as to reduce the defects.
  2. **Finance :** Is the project financially feasible i.e. can it be developed in the cost estimated by the software organization and its client.
  3. **Time :** Will the project be delivered in the estimated time.
  4. **Resources :** Does the software organization have all those resources required for successfully accomplishing the project?

Syllabus Topic : Resource Estimation

7.2 Resource Estimation

Software engineering resources can be categorized into three kinds :

1. People
   - Number of people required
   - Skills required in them
   - Geographical location from where they will work
2. Development environment
   - Software tools required in Project process

   - Computer hardware
   - Network resources
3. Reusable software components
   - Off-the-shelf components
     - Such components are either from a third party or available from a previous project
     - Such components are ready to use i.e. fully validated and documented
   - Full-experience components
     - Components are similar to the software that needs to be built
   - Partial-experience components
     - Components are somewhat related to the software that needs substantial modification to be built as required and also have substantial risks.
   - New components
     - Components must be built from scratch and can have a high degree of risk.

Each of the above resources is specified with :

- A description about the resource
- A statement of availability
- The time when the resource will be required
- The duration of time that the resource will be applied

7.2.1 Software Cost Estimation Process

Cost Estimation process comprises of 4 main steps :

Step 1 : **Estimate the size of the development product**

Size of the software may depend upon : lines of code, inputs, outputs, functions, transactions, features of the module and etc.

Step 2 : **Estimate the effort in person-hours**

The effort of various Project tasks expressed in person-hours is influenced by various factors such as :

- Experience/Capability of the Team members.
- Technical resources.
- Familiarity with the Development Tools and Technology Platform.

Step 3 : **Estimate the schedule in calendar months**

The Project Planners work closely with the Technical Leads, Project Manager and other stakeholders and create a Project schedule. Tight Schedules may impact the Cost needed to develop the Application.

Step 4 : **Estimate the project cost in dollars (or other currency)**

Based on the above information the project effort is expressed in dollars or any other currency.

7.3 Cost Estimation Techniques

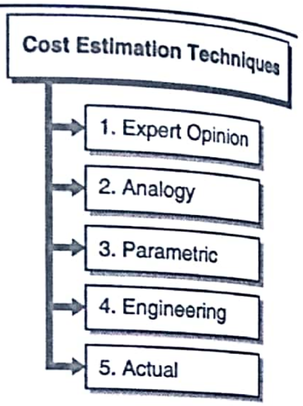


Fig. C7.1 : Cost Estimation Techniques

→ 1. Expert Opinion

Also called as Delphi method, proposed by Dr. Barry Boehm is useful in assessing differences between past projects and new ones for which no historical precedent exists.

☞ **Advantages**

- Little or no historical data needed.
- Suitable for new or unique projects.

☞ **Disadvantages**

- Very subjective.
- Experts may do partiality
- Qualification of experts may be questioned.

→ 2. Analogy

- Estimates costs by comparing proposed programs with similar, previously completed programs for which historical data is available.
- Actual costs of similar existing system are adjusted for complexity, technical, or physical differences to derive new cost estimates
- Analogies are used early in a program cycle when there is insufficient actual cost data to use as a detailed approach
- Compares similarities and differences
- Good choice for a new system that is derived from an existing subsystem.

☞ **Advantages**

- Inexpensive – Easily changed – Based on actual experience (of the analogous system)

☞ **Disadvantages**

- Very Subjective
- Large amount of uncertainty
- Truly similar projects must exist and can be hard to find
- Must have detailed technical knowledge of program and analogous system

→ 3. Parametric :

Utilizes statistical techniques. Can be used prior to development.

☞ **Advantages**

- Can be excellent predictors when implemented correctly
- Easily changed
- Objective
- Once created, CERs are fast and simple to use
- Useful early on in a program

☞ **Disadvantages**

- Often lack of data on software intensive systems for statistically significant CER

- Does not provide access to subtle changes
- Top level; lower level may be not visible
- Need to be properly validated and relevant to system

→ 4. **Engineering :**

Also referred to as ***bottoms up*** or detailed method.

- o Start at the component level and estimate the effort required for each component. Add these efforts to reach a final estimate.
- o Future costs for a system are predicted with a great deal of accuracy from historical costs of that system.
- o Involves examining separate work segments in detail.
- o Estimate is built up from the lowest level of system costs.
- o Includes all components and functions.
- o Can be used during development and production.

☞ **Advantages**

- Objective – Reduced uncertainty

☞ **Disadvantages**

- Expensive – Time Consuming – Not useful early on
- May leave out software integration efforts

→ 5. **Actual**

- Decides future costs on recent historical costs of same system.
- Used later in development or production.
- Costs are calibrated to actual development or production productivity for your organization.

☞ **Advantages**

- Most accurate – Most objective of the five methodologies

☞ **Disadvantages**

- Data not available early
- Time consuming
- Labour intensive to collect all the data necessary

☞ **Choice of methodology depends upon**

- **Type of system** - software, hardware, etc
- **Phase of program** - Development, Production, Support
- **Available data** - Historical data points from earlier system versions or similar system or Technical parameters of system.

7.4 Cost Estimation Parameters

- Various models (such as COCOMO, Co-star etc.) are available for estimating the cost of software development.

- All these cost estimating models can be represented on the basis of five basic parameters:

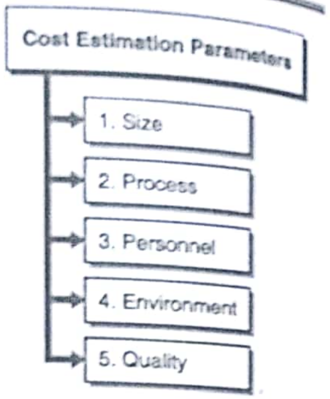


Fig. C7.2 : Cost Estimation Parameters

→ 1. Size

The size of the proposed software product is weighed in terms of components i.e. ultimately in terms of the number of source code instructions or the number of functions required in developing the proposed product.

→ 2. Process

The process includes the phases and activities carried out in each phase. So whatever process used to produce the proposed product is measured on the ability of the process to avoid unnecessary activities such as rework, bureaucratic delays, communications overhead and such other overhead activities which may delay the delivery of the product.

→ 3. Personnel

This deals with the capabilities and experience of software engineering *personnel* (team members) in the field of computer science issues and the applications domain issues of the project. It also depends upon the personnel's familiarity with the Development Tools and Technology Platform.

→ 4. Environment

The environment constitutes of the tools and techniques that are required to develop efficient software and also to automate the process.

→ 5. Quality

- The required quality of the proposed product depends upon the features, performance, reliability, and adaptability of the software.
- The above described five parameters (size, process, personnel, environment and quality) can be related with each other so as to calculate the estimated cost for the proposed software development.

$$\boxed{\text{Effort/Cost} = (\text{Personnel})(\text{Environment})(\text{Quality})(\text{Size}^{\text{process}})}$$

7.5 Pragmatic Software Cost Estimation

Developers and customers always had an argument on software cost estimation models and tools.

Three most common topics of their argument

1. Selection of Cost Estimation Model

   There are various cost estimation models (COCOMO, COSTXPERT, CHECKPOINT, SLIM, ESTIMACS, Knowledge Plan, SEER, SOFTCOST, Co-star, REVIC, Price-S, ProQMS etc.) that are based on statistically derived *cost* estimating relationships (CERs) and various estimating methodologies.

2. Whether to measure software size on the basis of lines of code (LOC) or function points (FP) ?
   - Most of the cost estimation models are bottom-up i.e. substantiating a target cost rather than *top-down* i.e. estimating the 'should' cost.
   - The Fig. 7.5.1 depicts predominant cost estimation practices where :

- o First, project manager defines the target cost of the software.
- o Later, he manipulates the parameters and does the sizing until the target cost is justified.
- o The target cost is defined so as :
- o to win the proposal,
- o to solicit the customer funding,
- o to attain the internal corporate funding, or
- o to achieve some other goal.
- The process described in Fig. 7.5.1 forces the software project manager to check the risks associated in achieving the target costs and also discuss this problem with other stakeholders.

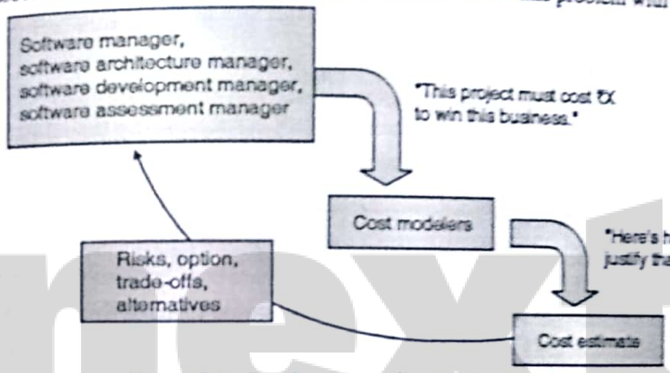

Software manager, software architecture manager, software development manager, software assessment manager
"This project must cost $X to win this business."
Cost modelers
"Here's how justify that cost."
Risks, option, trade-offs, alternatives
Cost estimate

Fig. 7.5.1 : Predominant Cost Estimation Process

3. Factors that lead to good cost estimation

Good software cost estimation may be based on the following attributes :

- The cost is estimated collectively by the project manager, architectural team, development team, and test team.
- The estimated cost is acknowledged and supported by all stakeholders.
- The cost estimation is based on some well-defined model.
- The cost estimation is based on the databases of similar type of project experiences that use similar methodologies, similar technologies, and similar environment with similar type of stakeholders.
- The key risk areas are detected and accordingly the probability of success is determined.

Syllabus Topic : Empirical Estimation Models

7.6 Empirical Estimation Models

- Estimation models use empirically derived formulas such as LOC or FP to predict the required efforts.
- The computed values for LOC or FP are entered into an estimation model.
- The empirical data used in deriving the LOC or FP for these models are derived from the sample projects.

7.6.1 COCOMO Model

- **Constructive Cost Model** (COCOMO) is one of the earliest cost models widely used by the cost estimating community.
- COCOMO was originally published in Software Engineering Economics by Dr. Barry Boehm in 1981.
- COCOMO is a regression-based model that considers various historical programs software size and multipliers.
- COCOMO's most fundamental calculation is the use of the Effort Equation to estimate the number of Person-Months required in developing a project.
- COCOMO stands for Constructive Cost Model. It is the oldest cost estimation model that is popularly used in the process of cost estimation.
- COCOMO model estimates the cost by considering the size and other quality aspects of the similar type of historical (previously developed) programs.
- COCOMO calculates Efforts i.e. it estimates the number of Person-Months required in developing a project.

Number of person months * loaded labour rate = Estimated Cost

- Most of the other estimates (requirements, maintenance, etc) are derived from this quantity.
- COCOMO requires as input the project's estimated size in Source Lines of Code (SLOC).
- Initial version published in 1981 was COCOMO-81 and then, through various instantiations came COCOMO2.
- COCOMO-81 was developed with the assumption that a waterfall process would be used and that all software would be developed from scratch.
- Since then, there have been many changes in software engineering practice and COCOMO2 is designed to accommodate different approaches to software development.

The COCOMO model is based on the relationships between the two formulate

Formulae 1 : Development effort is based on system size.

MM = a.KDSI b

where,

MM is the effort measured in Man per Moths

KDSI is the number of Source Instructions Delivered in a Kilo (thousands).

Formulae 2 : Effort (MM) and DEVelopment Time.

TDEV = c.MM d

where,

TDEV is the development time.

In both the above formulas, we have used the coefficients a, b, c and d which are dependent upon the 'mode of development'.

According to Boehm, the mode of development can be classified into following 3 distinct modes :

1. **Organic** mode of development – talks about the projects that involve small development teams whose team members are familiar with the project and work to achieve stable environments. This category includes the projects like the *payroll systems.*

2. **Semi-detached** mode of development – talks about the projects that involve mixture of experienced team members in the project. This category includes the projects like the interactive banking system.
3. **Embedded** mode of development – talks about the complex projects that are developed under tight constraints with innovations in it and have a high volatility of requirements. This category includes the projects like the *nuclear reactor control systems.*

Drawbacks

- It is difficult to accurately estimate the KDSI in early phases of the project when most effort estimates are still not decided yet.
- Easily thrown misclassification of the development mode.
- Its success largely depends on tuning the model to the needs of the organization and this is done based upon the historical data which is not always available.

Advantages

- COCOMO is transparent that means we can se it working.
- Allows the estimator to analyse the different factors that affect the project costs.

Syllabus Topic : COCOMO II Model

7.6.2 COCOMO II Model

- COCOMO II constitutes of sub-models so as to produce in-detail software estimates. The sub models are listed as follows :
  - **Application composition model :** It is applied when software is being developed from existing parts.
  - **Early design model :** It is applied when system requirements are gathered and concluded but design has not yet started.
  - **Reuse model :** It is applied when software is being developed using the reusable components so as to compute the effort of integrating these components.
  - **Post-architecture model :** It is applied when once the system architecture has been designed and when more system information is gathered.

Multipliers reflect the capability of the developers, the non-functional requirements, the familiarity with the development platform, etc.

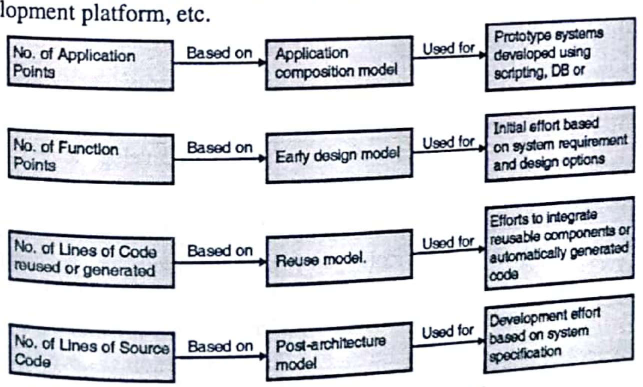

No. of Application Points
Based on
Application composition model
Used for
Prototype systems developed using scripting, DB or
No. of Function Points
Based on
Early design model
Used for
Initial effort based on system requirement and design options
No. of Lines of Code reused or generated
Based on
Reuse model.
Used for
Efforts to integrate reusable components or automatically generated code
No. of Lines of Source Code
Based on
Post-architecture model
Used for
Development effort based on system specification

Fig. 7.6.1 : Use of COCOMO2 Model

- o **Product attributes** describe the required characteristics of the software product being developed.
- o **Computer attributes** describe the constraints imposed on the software product by the hardware platform.
- o **Personnel attributes** describe the experiences and capabilities (of the project development team members) that are taken into account.
- o **Project attributes** describe particular characteristics of the project.

– Estimating the calendar time required to complete a project and when staff will be required using a COCOMO2 formula

$$TDEV = 3\ ´\ (PM)^{(0.33+0.2*(B-1.01))}$$

– PM is the effort computation and B is the exponent (B is 1 for the early prototyping model). This computation predicts the nominal schedule for the project.

– The time required is independent of the number of people working on the project.

☞ Improving Software Economics

– The software economics can be improved by using a ‘balanced’ approach as the key.

– The Five Key parameters that can help in improving the software economics are :
  - Reducing the product size (Number of Lines of Code) *and* complexity of the software
  - Improving the software development *process*
  - Using more-skilled personnel i.e. improving the team effectiveness.
  - Creating better environment by improving the *automation* with more appropriate tools and technologies.
  - Achieving required quality by peer inspections

Table 7.6.1 : Trends on which the Five Key Parameters depend

Five Key parameters that effect the Cost Model	Trends
Size : describes abstraction and component based development technologies	High level programming Languages (C++, Java, VB, Object Oriented Methods and Visual modelling (analysis, design and programming) Reusability Commercial Exponents Packages
Process : involves methods and techniques	Iterative Development Process Maturity Models such as CMM Architecture-first development
Personnel : describes the effectiveness of the development team	Training to develop the personnel skills Team work Win-win culture
Environment : involves automated tools and technologies.	Integrated tools such as compiler, editors, and debuggers. Hardware performance Automated coding, documentation, testing and analyses.
Quality : describes the Performance, reliability, accuracy issues	Hardware platform performance Peer inspections Statistical quality control

Syllabus Topic : Estimation for Agile Development

7.6.3 Estimation for Agile Development

In Agile development, the Project Management process is highly iterative and incremental, where all the project stakeholders actively work together to understand the domain, to identify what needs to be built, and to prioritize the functionality.

☞ Estimation Scales in Agile development

- o 1,2,3,5,8 (Fibonacci series)
- o 1,2,4,8

☞ Common techniques for Estimation in Agile development

- o Expert opinion
- o Analogy
- o Disaggregation

☞ Estimation Process

– User stories
  - o Users break down the functionality into “user stories”
  - o User stories are kept small and these include the acceptance criteria

– High level estimation
  - o After all the user stories are written, do a high level of estimation for setting the priorities.

– Story points
  - o Break down user stories to units of relative size so that you can compare their features which will be useful in measuring the size/complexity.

– Product backlog
  - o All story points which actually represent the project tasks are put into a bucket that acts as a Backlog which will have an item and its estimate
  - o This backlog is not time based, but point based

– Velocity
  - o Velocity is the number of story points completed per sprint
  - o Velocity is computed to predict how much work to commit to in a sprint and this only works if you estimate your story points’ consistency

– Re-estimation
  - o As you complete more sprints, your velocity will go on changing. This change in Velocity is due to minor inconsistencies in the story point estimates
  - o Re-estimation of the entire project happens after each sprint.

Syllabus Topic : The Make/Buy Decision

7.6.4 The Make/Buy Decision

- Software engineering Managers face many problems in Make and Buy decisions that later have many number of acquisition options such as :
  - o should software be purchased off the shelf ?
  - o should it use "Full-experience" or "partial-experience" software components ?
  - o should the software be custom built by an outside contractor ?
- The make/buy decision can be made based on the following conditions
  - o Will the software product be available sooner if given to outside contractor than internally developed software?
  - o Will the cost of acquisition plus the cost of customization be less than the cost of developing the software internally?
  - o Will the cost of outside maintenance support be less than the cost of internal support?

Review Questions

Q. 1 Describe the cost estimation process.

Q. 2 Write short notes on cost estimation techniques.

Q. 3 List and describe the cost estimation parameters.

Q. 4 What is cost estimation and explain COCOMO model.

Q. 5 Explain the concept of estimation in agile development.

Q. 6 Explain the concept of Make/Buy decision

Q. 7 Explain in brief about : Software scope and feasibility

□□□

CHAPTER 8 Project Scheduling

UNIT II

Syllabus :

Basic Principles, Relationship between People and Effort, Effort Distribution, Time-Line Charts

8.1 Project Scheduling

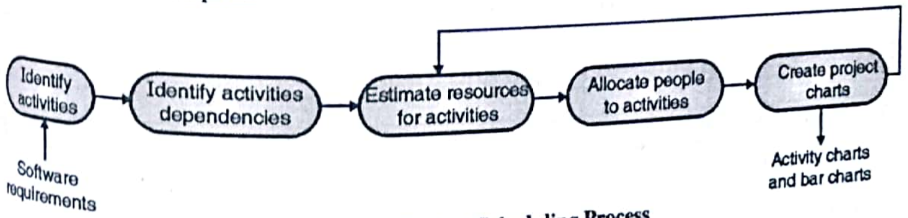
Q. Explain the project scheduling process.

The *project schedule* is a calendar that is used to associate the tasks to be performed with the resources that will perform them. Before a project schedule is estimated, the project manager designs a work breakdown structure (WBS) which is an attempt to estimate the time needed to implement each task and the resources that are available for accomplishing each task.

Project Scheduling is dependent on project managers' intuition and experience.

Project Scheduling Process

- Divide the project into various tasks and estimate the duration and resources required to complete each task.
- Arrange the tasks so as to make optimal use of workforce.
- Reduce the task dependencies so as to avoid delays that might be caused by some task i.e. waiting for another to complete.



Identify activities
Identify activities dependencies
Estimate resources for activities
Allocate people to activities
Create project charts
Software requirements
Activity charts and bar charts

Fig. 8.1.1: Project Scheduling Process

Scheduling Problems

- Detecting the complexity of the problems and accordingly estimating the cost of developing a solution is difficult.