

Unit:4

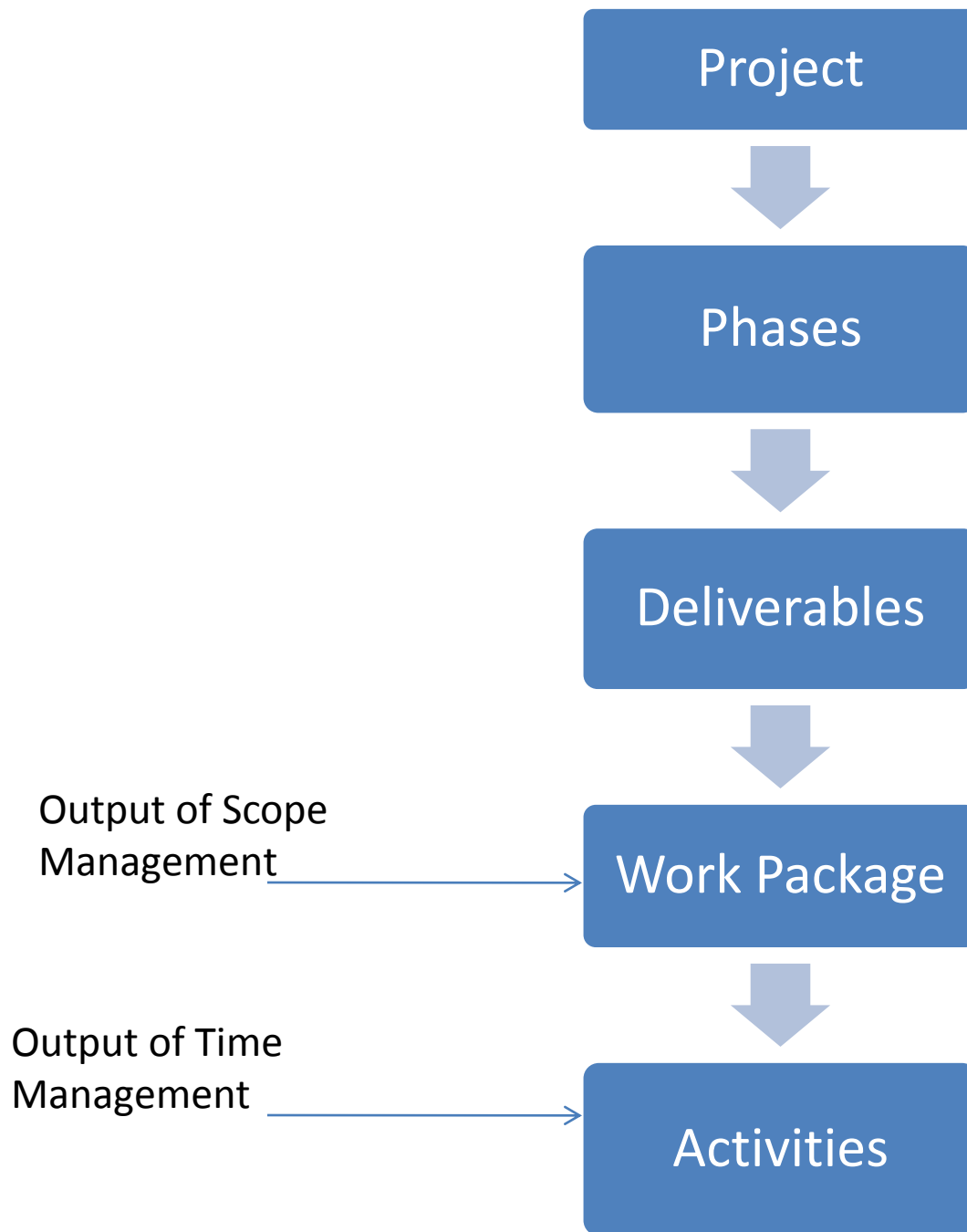
Project Economics, Project scheduling and Tracking

Concept of Project Management:

- Project management is the discipline of planning, organizing, motivating, and controlling resources to achieve specific goals.
- A project is a temporary endeavor with a defined beginning and end undertaken to meet unique goals and objectives, typically to bring about beneficial change or added value.
- The primary challenge of project management is to achieve all of the project goals and objectives while honouring the preconceived constraints.
- The primary constraints are scope, time, quality and budget. The secondary and more ambitious challenge is to optimize the allocation of necessary inputs and integrate them to meet pre-defined objectives.
- Two forefathers of project management are Henry Gantt, called the father of planning and control techniques, who is famous for his use of the Gantt chart as a project management tool and Henri Fayol for his creation of the five management functions that form the foundation of the body of knowledge associated with project and program management.

DECOMPOSITION TECHNIQUES:

- Decomposition involves dividing each project deliverable into smaller and smaller pieces until there is enough detail to support scheduling, estimating and control.
- Software project estimation is a form of problem solving and in most cases, the problem to be solved is too complex to be considered in one piece. For this reason, we decompose the problem, re-characterizing it as a set of smaller problems.
- Decomposition of project scope generally involves the following activities.
 1. Determine your main project deliverables.
 2. Create a high-level Work Breakdown Structure(WBS) by 'chunking' work into smaller tasks.
 3. Continue to break down high-level tasks into smaller tasks.
 4. Create a system for tracking each task.
 5. Verify that the resulting tasks are manageable



- As you decompose requirements, keep in mind that you can structure and manage the project deliverables in various ways.

- Some of the most common approaches for different types of organizations include:

1. Spreading deliverables across project phases : This is typical for projects conducted in a waterfall fashion and for schedule-oriented projects.

2. Spreading deliverables across knowledge areas : Project teams organized by areas of expertise (such as operations, legal or finance) usually use the technique in their project.

3. Spreading deliverable across processes : This is typical for process-oriented projects.







4. Spreading deliverables across sub-projects : This is most effective big projects with

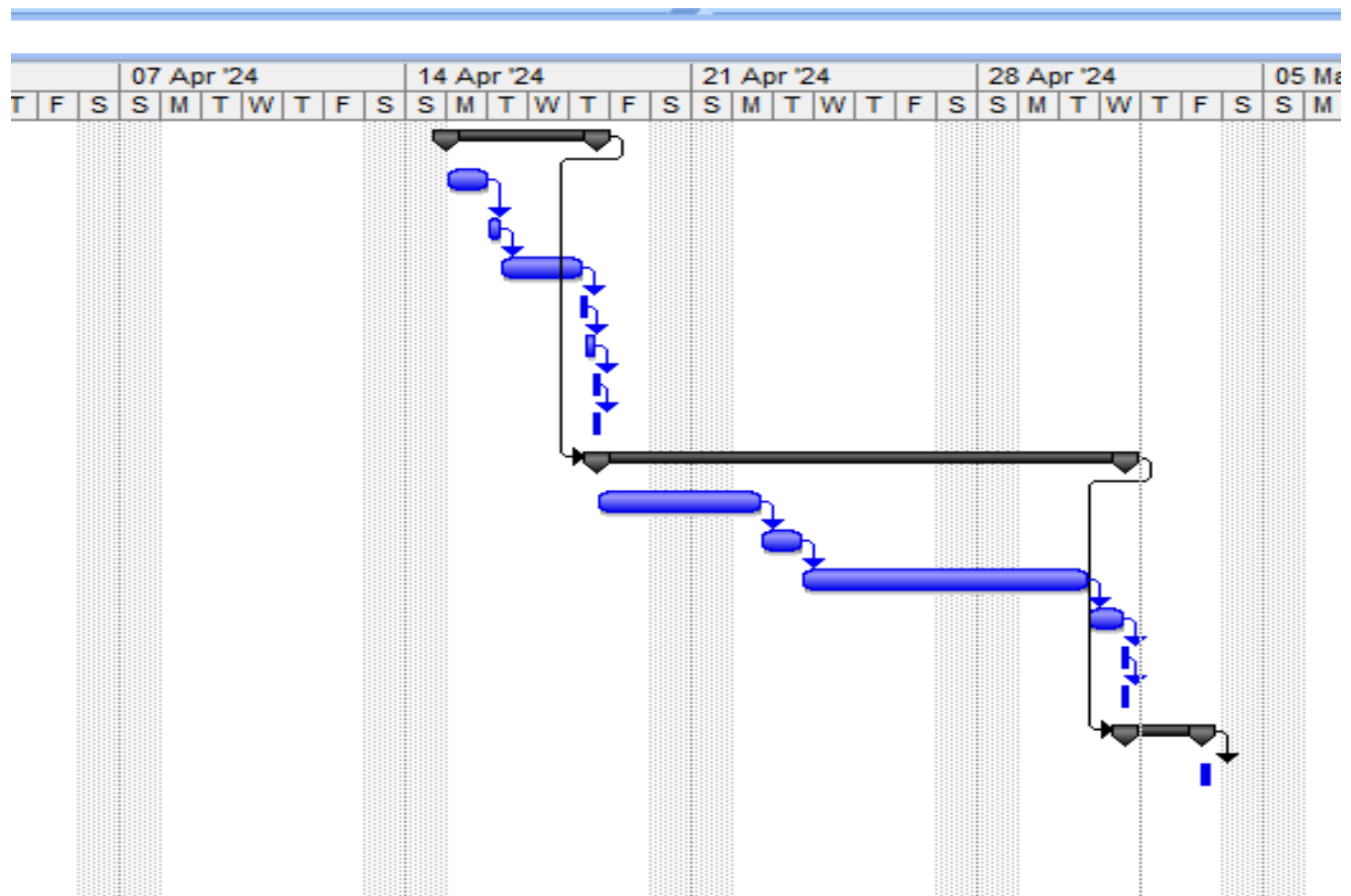
added complexity, where deliverables are specific for parts of the project but not for the project as a whole.

5. Organizing deliverables hierarchically, from major deliverables to sub-deliverables : This is best for product-oriented projects.

TimeLine Chart:

- Timeline chart refers to user items that create a chart where a series of events are arranged on a bar graph. Where by every event could be a single point in time or date range.
- Also called a Gantt chart; invented by Henry Gantt, industrial engineer, 1917.
- All project tasks are listed in the far left column
- The next few columns may list the following for each task:
projected start
- date, projected stop date, projected duration, actual start date, actual stop date, actual duration, task inter-dependencies (i.e., predecessors)
- To the far right are columns representing dates on a calendar
- The length of a horizontal bar on the calendar indicates the duration of the task
- When multiple bars occur at the same time interval on the calendar, this implies task concurrency
- A diamond in the calendar area of a specific task indicates that the task is a milestone; a milestone has a time duration of zero

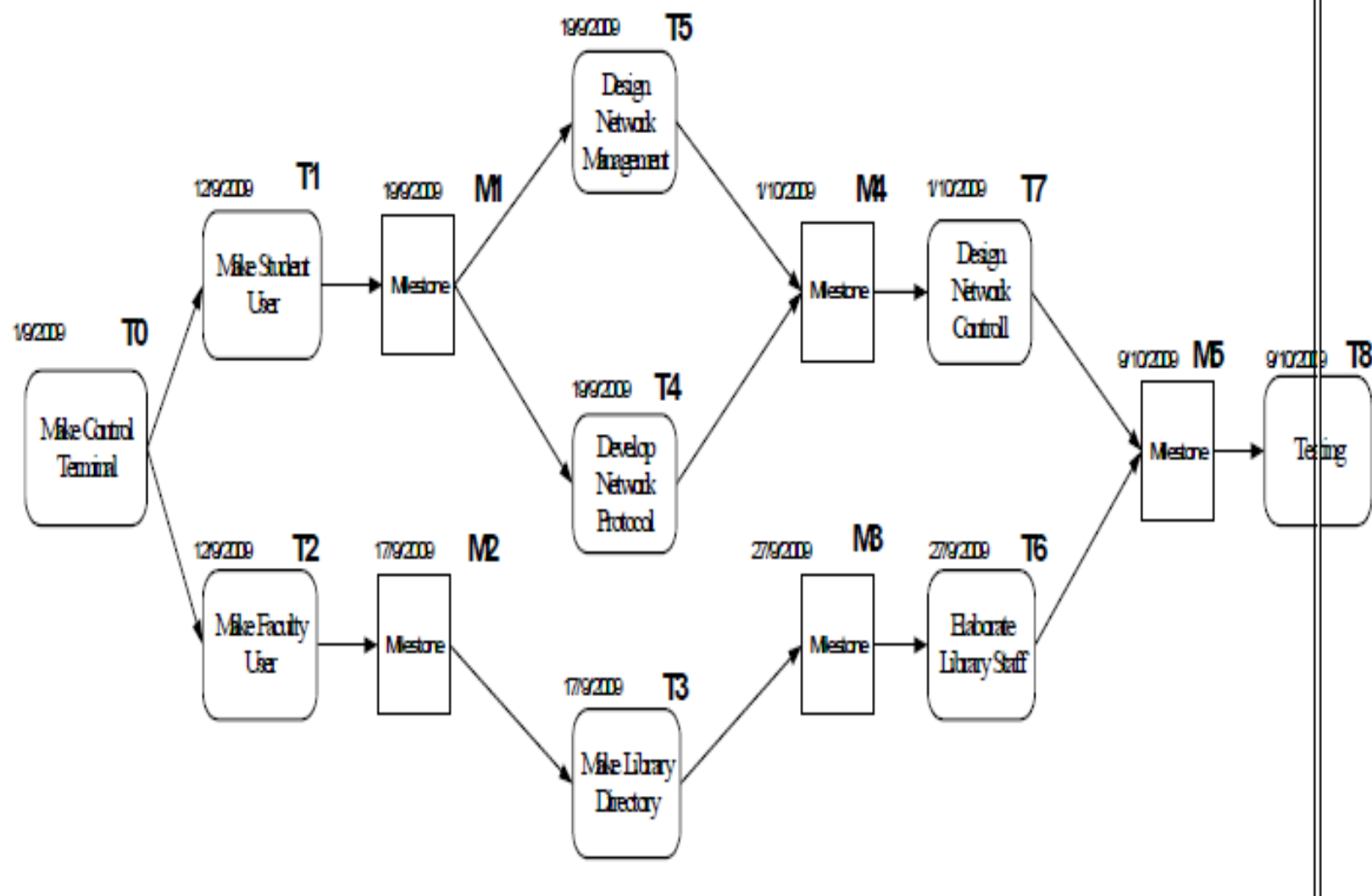
		Task Name	Duration	Start	Finish	Predecessors
1		 Plannig phase	3.96 days	Mon 15-04-24	Thu 18-04-24	
2		Move things to Rent	1 day	Mon 15-04-24	Mon 15-04-24	
3		Take Floor Measurment	0.5 hrs	Tue 16-04-24	Tue 16-04-24	2
4		Hire someone to lay Tiles	2 days	Tue 16-04-24	Thu 18-04-24	3
5		Choose closet design	0.7 hrs	Thu 18-04-24	Thu 18-04-24	4
6		Pick/order tiles for floor	5 hrs	Thu 18-04-24	Thu 18-04-24	5
7		place order on ikea	1 hr	Thu 18-04-24	Thu 18-04-24	6
8		check planning details	0.5 hrs	Thu 18-04-24	Thu 18-04-24	7
9		 tiles phase	8.94 days	Thu 18-04-24	Wed 01-05-24	1
10		remove old tiles	2 days	Thu 18-04-24	Mon 22-04-24	
11		level floors	1 day	Mon 22-04-24	Tue 23-04-24	10
12		place new tiles	5 days	Tue 23-04-24	Tue 30-04-24	11
13		check floor	5 hrs	Tue 30-04-24	Wed 01-05-24	12
14		clean up	0.5 hrs	Wed 01-05-24	Wed 01-05-24	13
15		piant house	2 hrs	Wed 01-05-24	Wed 01-05-24	14
16		 closet phase	1.63 days	Wed 01-05-24	Fri 03-05-24	9
20		bring things back from rental place	2 hrs	Fri 03-05-24	Fri 03-05-24	16



Gantt Chart

PERT chart (Program Evaluation Review Technique)

- A PERT chart is a project management tool used to schedule, organize, and coordinate tasks within a project. PERT stands for *Program Evaluation Review Technique*, a methodology developed by the U.S. Navy in the 1950s to manage the Polaris submarine missile program.
- A similar methodology, the *Critical Path Method (CPM)* was developed for project management in the private sector at about the same time.
- sequence of tasks, which tasks can be performed Simultaneously, and the critical path of tasks that must be completed on time in order for the project to meet its completion deadline.
- The chart can be constructed with a variety of attributes, such as earliest and latest start dates for each task, earliest and latest finish dates for Each task and slack time between tasks.
- A PERT chart can document an entire project or a key phase of a project.



Project Scheduling (Basic Principles):

- Software project scheduling is an activity that distributes estimated effort across the planned project duration by allocating the effort to specific software engineering tasks.
- First, a macroscopic schedule is developed.
- A detailed schedule is redefined for each entry in the macroscopic schedule.
- A schedule evolves over time.
- Basic principles guide software project scheduling:
 - Compartmentalization
 - Interdependency
 - Time allocation
 - Effort allocation
 - Effort validation
 - Defined responsibilities
 - Defined outcomes
 - Defined milestones

Scheduling:

- Task networks (activity networks) are graphic representations can be of the task interdependencies and can help define a rough schedule for particular project.
- Scheduling tools should be used to schedule any non-trivial project.
- *Program evaluation and review technique (PERT) and critical path method (CPM)) are quantitative techniques that allow software planners to identify the chain of dependent tasks in the project work breakdown structure (WBS) that determine the project duration time.*
- *Timeline (Gantt) charts enable software planners to determine what tasks will be need to be conducted at a given point in time (based on estimates for effort, start time, and duration for each task).*
- The best indicator of progress is the completion and successful review of a defined software work product.
- Time-boxing is the practice of deciding a priori the fixed amount of time that can be spent on each task. When the task's time limit is exceeded, development moves on to the next task (with the hope that a majority of the critical work was completed before time ran out).

Automated Estimation Tools:

- The right tools can make all the difference in helping to provide better software. We should focus on providing effective tools to help estimate, measure and report software project performance, manage function point data and utilize historical and industry benchmark data.
- Some of the popular tools are given below.
 - Benchmark data
 - PQM plus
 - Function point workbench
 - SMRe
 - Q estimating

(1) Benchmark data:

- Q/P management group has established the world's largest size based software metrics benchmark database.
- Q/P has been collecting data since 1990. project and application data are added to the database annually after rigorous analysis and verification to ensure the highest degree of data integrity in the industry, the database consist of statistics on thousand of project and application.
- The given statistics include:
- Project productivity for new development and enhancement efforts.
- Project cost and labour rates.
- Application maintenance productivity.
- Application support cost.
- Application and project quality.
- Time to market-schedule duration.
- Project staffing.

(2) SMRe(software measurement, reporting and estimating):

- SMRe is a tool that automates the software project estimating and the reporting of project performance metrics.
- Organizations can use SMRe to estimate project size, effort , schedule and staffing early in the lifecycle using in-house and/or industry benchmarks.
- Once the project is complete SMRe is used to capture project data, report and the performance of development project, and compare the performance to in-house and/or industry benchmarks.
- SMRe has been designed to work with PQMPLUS and the function point WORKBENCH in order to share relevant data to aid in the production of software measurement reports.

(3) PQM plus:

- The intelligent software measurement and estimating tools
- PQM plus is a productivity/quality measurement system developed for software development project managers and measurement specialists.
- PQM plus is a benchmarking and measurement tool with a robust function point repository that provides project estimating based on historical data, project scheduling and risk assessments.
- PQM plus and SMR have been designed to work together to share relevant data to aid in the production of software measurement reports.

Algorithmic Method:

- The algorithmic method is designed to provide some mathematical equations to perform software estimation. These mathematical equations are based on research and historical data and use inputs such as Source Lines of Code (SLOC), number of functions to perform, and other cost drivers such as language, design methodology, skill-levels, risk assessments, etc.
- The algorithmic methods have been largely studied and there are a lot of models have been developed, such as COCOMO models, Putnam model, and function points based models.

Advantages:

- It is able to generate repeatable estimations.
- It is easy to modify input data, refine and customize formulas.

Disadvantages:

- It is unable to deal with exceptional conditions, such as exceptional personnel in any software cost estimating
- exercises, exceptional teamwork, and an exceptional match between skill-levels and tasks.

(1) COCOMO Model:

- Software cost estimation is an important part of the software development process. The COCOMO offers a powerful instrument to predict software costs.
- The Constructive Cost Model is an algorithmic software cost estimation model developed by Barry W. Boehm. The model uses a basic regression formula with parameters that are derived from historical project data and current as well as future project characteristics.
- COCOMO provides more support for modern software development processes and an updated project database. The need for the new model came as software development technology moved from mainframe and overnight batch processing to desktop development code reusability and the use off-the-shelf software components.
- Boehm proposed three levels of the model:
 - Basic
 - Intermediate
 - Detailed

- The first level, basic COCOMO is good for quick, early, rough order of magnitude estimates of software costs, but its accuracy is limited due to its lack of factors to account for difference in project attributes.
- The intermediate COCOMO model computes software development effort as a function of program size and a set of fifteen “cost drivers” that include subjective assessments of product hardware, personnel and project attributes.
- The advanced or detailed COCOMO model incorporates all characteristics of the intermediate version with an assessments of the cost driver’s impact on each step(analysis, design etc.) of the software engineering process.
- **Advantages of COCOMO’8:**
 1. COCOMO is transparent – one can see how it works unlike other models such as SLIM.
 2. Drivers are particularly helpful to the estimator to understand the impact of different factors that affect project costs.

- **Disadvantages of COCOMO'81:**

1. It is hard to accurately estimate KDSI (Thousand Delivered Source.) early on in the project, when most effort estimates are required.
2. KDSI, actually is not a size measured it is a length measured.
3. Extremely vulnerable to mis-classification of the development mode.
4. Success depends largely on tuning the model to the needs of the organization, using historical data which is not always available.

- COCOMO II is the latest major extension to the original COCOMO81 model published in 1981.

2. SLIM (Software Lifecycle Management) Model :

- SLIM is one of the first algorithmic cost model. It is based on the Norden/Rayleigh function and generally known as a macro estimation model (It is for large projects.)
- SLIM also uses historical data from past projects for estimation. It also uses and considers other project parameters, characteristics, attributes and KLOC for its estimation calculation.
- SLIM enables a software cost estimator to perform the following functions :
- Calibration: Fine tuning the model to represent the local software development environment by interpreting a historical database of past projects.
- Build: An information model of the software system, collecting software characteristics, personal attributes, computer attributes etc.
- Software Sizing: SLIM uses an automated version of the lines of code (LOC) costing technique.

Advantages:

- It provides a set of software development management tools that support the entire program life cycle.
- Offers value-added planning
- It simplifies strategic decision making.
- Supports “what-if” analysis.
- It allows report and graph generation.

Disadvantages:

- It works best on large projects.
- The software size needs to be estimated in advance in order to use this model.
- Estimates are extremely sensitive to the technology factor.
- Model is also sensitive to size estimate.
- Tool is considered to be fairly complex.
- It works for Waterfall life cycle that doesn't cover up spiral model.

Effort Estimation:

- **Estimating:** The process of forecasting or approximating the time and cost of completing project deliverables.
- The task of balancing the expectations of stakeholders and the need for control while the project is implemented

Types of Estimates:

- Top-down (macro) estimates: analogy, group consensus, or mathematical relationships
- Bottom-up (micro) estimates: estimates of elements of the work breakdown

Estimating Techniques

- The following estimating techniques fit into either the top-down or bottom-up approach. No one estimating technique is ideal for all situations; each has its own strengths and weaknesses. When estimating a project, you need to decide which technique is appropriate and what adjustments, if any, are needed.

1. Ballpark Estimating

- With this estimating technique you use a combination of time, effort, peak staff, and derived from the QSM SLIM completed projects database.
- Each row represents a consistent set of estimates that may be determined based on any one of the variables estimated using expert judgment.
- This technique can be used at any point in the lifecycle. It can be used early in the lifecycle and when no historical information is available. Once the estimate is developed, a comparative estimate can be developed using a proportional technique. The final estimates can be compared to other estimates for analysis.
- Using the Business Case documentation, a proposed solution is visualized, and using expert judgment, the Modules, Interfaces, Configuration Items, or Programs in the visualized solution can be identified and entered into Top Down Estimate by CI worksheet.

2. Proportional Percentage Estimating

- With this estimating technique you use the size of one component to proportionally estimate the size of another. For example, the Design effort might be estimated as 22% of the Requirements effort; Construction 45% of Requirements effort, and Testing/Pilot 33% of Requirements effort.
- This technique is very effective when used appropriately, when the estimated value really does depend proportionally on another factor. There are different proportional models for different types of life cycles, which must be considered in developing proportional estimates.
- Consideration needs to be given to whether the current estimate is for an effort that is more like a Development/Enhancement effort or a Maintenance effort. If any portion of the labour distribution is estimated, it can be used to expand the known portion into a total estimate.

Estimation Technique	Strengths	Weaknesses
Comparative	Estimate can be very accurate if a suitable analogy can be identified.	Historical data repository required. Often difficult to find comparable projects.
Expert Judgment	Estimate can be extremely accurate. Identifies areas where requirements clarification is needed. Identifies requirements tradeoffs.	Must be verified by another method. High risk; may not be repeatable by anyone other than the “expert”. Single data point.
Proportional	Effective when estimated value really does depend proportionally on another factor (e.g., software management, quality assurance, configuration management).	Requires previous personal experience or experience based guideline metrics for proportionality factors. Can magnify estimating errors made in other areas.
Feature Point	Same strengths as Function Point Analysis, with added benefit of accounting for algorithms and internal processing complexity	. Does not yet have overall acceptance. Can be complicated to administer.

3. Function Point Analysis

- This estimating technique is suited for projects that are based on straightforward database input, output, maintenance, and inquiry, with low algorithmic processing complexity. Function Point Analysis is the basis for several automated estimating tools.
- The basic steps involved in this estimating technique include:
 1. Decomposing the project or application into a defined set of function types, described below.
 2. Assigning a complexity to each of these function types.
 3. Tallying the function types and applying pre-defined weighting factors to these totals to drive a single unadjusted function point count.
 4. Adjusting this function point count based on the overall project complexity.
 5. Translating the function point count to an effort estimate based on a function point delivery rate. (This is probably the most difficult step.)