Software Estimation Metrics

- **Software Metrics**
- **Software Project Estimation**
  - **LOC**
  - **FP**
  - **COCOMO II**
- **Project Scheduling and Tracking**

Software Metrics

❖ What is Software Metrics?

- A software metric is a measure of software characteristics which are measurable or countable.

- It can be defined as 'the continuous application of measurement-based techniques to the software development process and its products to supply meaningful and timely management information, together with the use of those techniques to improve that process and its products.'

- The IEEE Standard Glossary of Software Engineering Terms defines a metric as 'a quantitative measure of the degree to which. a system component or process possesses a given attribute.'

Software Metrics …. Contd.

- 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, there are many metrics that are all connected.

- Software metrics are similar to the four functions of management:
  - Planning, Organization, Control, or Improvement.
  - Software metrics

Terminologies used in Software Metrics

- **Measure**
  - Quantitative indication of the extent, amount, dimension, or size of some attribute of a product or process.

- **Metrics**
  - The degree to which a system, component, or process possesses a given attribute. Relates several measures (e.g. average number of errors found per person hour)

- **Indicators**
  - A combination of metrics that provides insight into the software process, project or product.

Terminologies used in Software Metrics

- **Direct Metrics**
  - Immediately measurable attributes (e.g. lines of code, execution speed, defects reported).

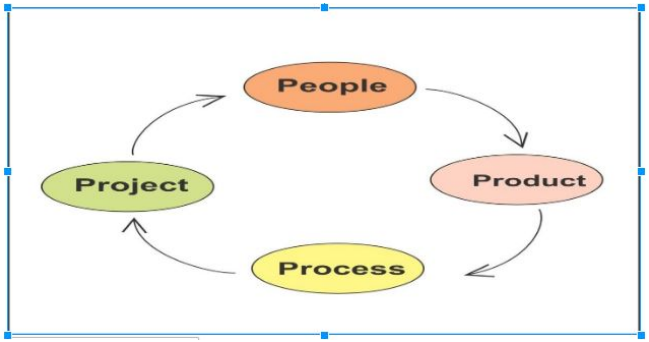
- **Indirect Metrics**
  - Aspects that are not immediately quantifiable (e.g. functionality, quantity, reliability, etc.)

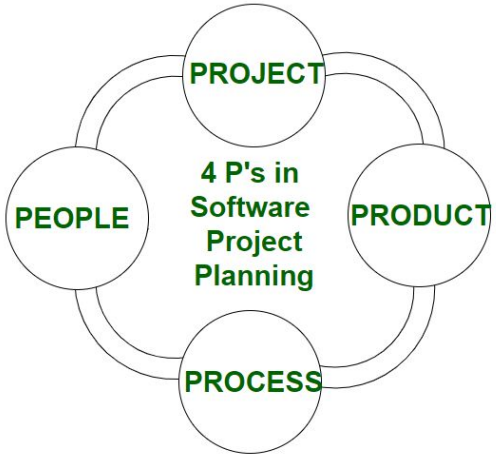
- **Faults**
  - **Errors**
    - Faults found by the practitioners during software development.

  - **Defects**
    - Faults found by the customers after release.

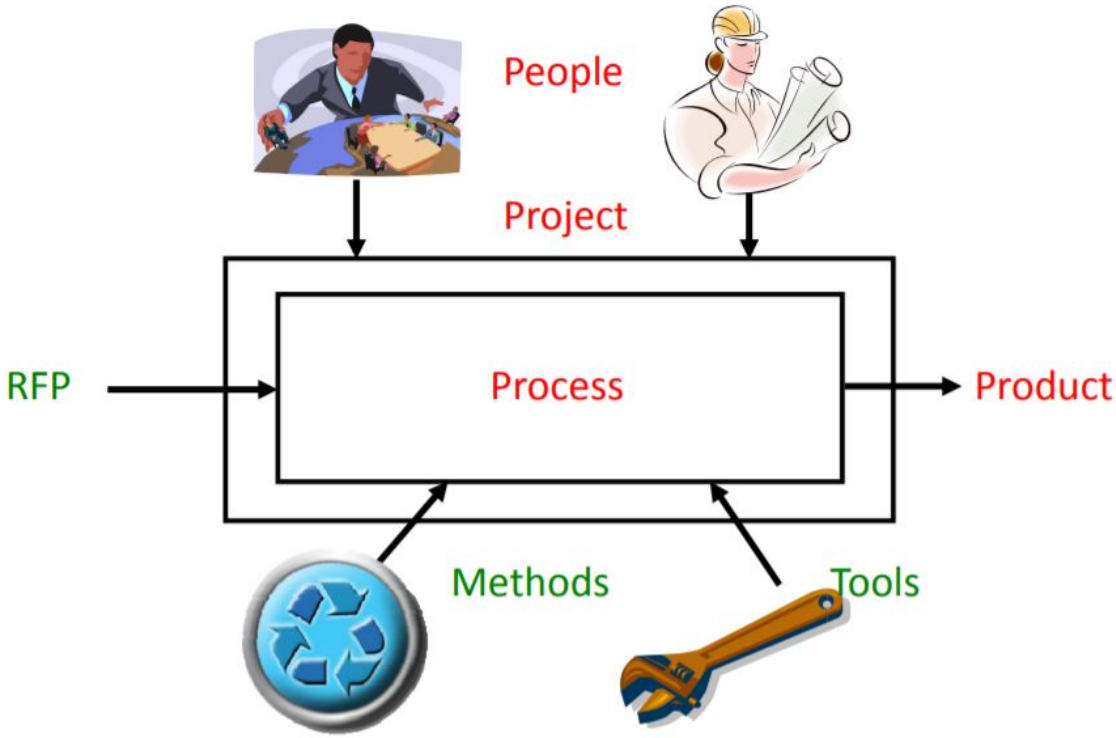
Management Spectrum

- The management spectrum describes the management of a software project or how to make a project successful.
- It focuses on the four P's; people, product, process and project.
- Here, the manager of the project has to control all these P's to have a smooth flow in the project progress and to reach the goal.





Management Spectrum …. Contd.



Management Spectrum …. Contd.

- The four P's of management spectrum are:

A. The People:

- People of a project includes from manager to developer, from customer to end user.
- But mainly people of a project highlight the developers.
- It is so important to have highly skilled and motivated developers that the Software Engineering Institute has developed a People Management Capability Maturity Model (PM-CMM), "to enhance the readiness of software organizations to undertake increasingly complex applications by helping to attract, grow, motivate, deploy, and retain the talent needed to improve their software development capability".
- Organizations that achieve high levels of maturity in the people management area have a higher likelihood of implementing effective software engineering practices.

Management Spectrum …. Contd.

B. The Product:

- Product is any software that has to be developed.
- To develop successfully, product objectives and scope should be established, alternative solutions should be considered, and technical and management constraints should be identified.
- Without this information, it is impossible to define reasonable and accurate estimates of the cost, an effective assessment of risk, a realistic breakdown of project tasks or a manageable project schedule that provides a meaningful indication of progress.

Management Spectrum …. Contd.

C. The Process:

- A software process provides the framework from which a comprehensive plan for software development can be established.
- A number of different tasks sets— tasks, milestones, work products, and quality assurance points—enable the framework activities to be adapted to the characteristics of the software project and the requirements of the project team.
- Finally, umbrella activities overlay the process model.
- Umbrella activities are independent of any one framework activity and occur throughout the process.

D. The Project:

- Here, the manager has to do some job.
- The project includes everything of the total development process.
- To avoid project failure, the manager has to take some steps, has to be concerned about some common warnings etc.

Metric Classification

The software metrics can be classified in three types:

1) **Processes**

- They are the activities related to production of software.
  - Process metrics are collected across all projects and over long periods of time. Their intent is to provide a set of process indicators that lead to long-term software process improvement.
  - Focus on quality achieved as a consequence of a repeatable or managed process.
  - They are strategic and long term.
  - Error categorization and analysis:
    - » All errors and defects are categorized by origin.
    - » The cost to correct each error and defect is recorded.
    - » The number of errors and defects in each category are computed.
    - » Data is analyzed to find categories that result in highest cost to the organization.
    - » Plans are developed to modify the process.

Metric Classification ….Contd.

2) Products

- They are explicit results of software development activities.
- Ex. Deliverables, documentation , by product.
  - They focus on the quality of deliverables.
  - Product metrics are combined across several projects to produce process metrics.
  - Metrics for the product:
    - » Measures of the Analysis Model.
    - » Complexity of the Design Model.
    - » Internal algorithmic complexity.
    - » Architectural complexity.
    - » Data flow complexity.
    - » Code metrics

Metric Classification ….Contd.

3) Project

- They are inputs into the software development activities.
- They can be in the form of hardware, knowledge, people, etc.
  - Project metrics enables a software project manager to assess
    - » the status of an on going project,
    - » track potential risks uncover problem areas before they go “critical”,
    - » adjust work flow or tasks and
    - » evaluate the project team’s ability to control quality of software work products.
  - It occurs during estimation for most software projects.

Metric Classification ….Contd.

- Metrics collected from past projects are used as a basis from which effort and time estimates are made for current software work.
- Production rates represented in terms of models created, review hours, function points, and delivered source lines are measured.
- These metrics are used to minimize the development schedule by making the adjustments necessary to avoid delays and mitigate potential problems and risks.
- Project metrics are used to assess product quality on an ongoing basis, and when necessary, modify the technical approach to improve quality.

Software Measurement

- Measurement of software (types of metrics) can be done in following different ways:

  - **Direct Measure/ Internal metrics (Internal attributes)**
    - Internal metrics are the metrics used for measuring properties that are viewed to be of greater importance to a software developer.
    - Ex. Cost, effort, Lines of Code (LOC), speed, memory

  - **Indirect Measure/ External metrics (External attributes)**
    - External metrics are the metrics used for measuring properties that are viewed to be of greater importance to the user.
    - Ex. Functionality, portability, usability, quality, complexity, efficiency, reliability, maintainability

Software Metrics

- **Advantages of Software Metrics**:-
  - Comparative study of various design methodology of software systems.
  - 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 providing feedback to software managers about the progress and quality during various phases of the software development life cycle.

Software Metrics ….Contd.

- **Disadvantages 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.

Size Oriented Metrics

- Size-oriented software metrics are derived by normalizing quality and/or productivity measures by considering the size of the software that has been produced.

- Ex. Lines of Code (LOC)
  - It is one of the earliest and simpler metrics for calculating the size of the computer program.
  - It is generally used in calculating and comparing the productivity of programmers.
  - These metrics are derived by normalizing the quality and productivity measures by considering the size of the product as a metric.

Size Oriented Metrics …. Contd.

- Following are the points regarding LOC measures:
  - In size-oriented metrics, LOC is considered to be the normalization value.
  - It is an older method that was developed when FORTRAN and COBOL programming were very popular.
  - Productivity is defined as KLOC / EFFORT, where effort is measured in person-months.
  - Size-oriented metrics depend on the programming language used.
  - As productivity depends on KLOC, so assembly language code will have more productivity.
  - LOC measure requires a level of detail which may not be practically achievable.

Size Oriented Metrics …. Contd.

- The more expressive is the programming language, the lower is the productivity.
- LOC method of measurement does not apply to projects that deal with visual (GUI-based) programming.
  - Graphical User Interfaces (GUIs) use forms basically. LOC metric is not applicable here.
- It requires that all organizations must use the same method for counting LOC.
  - This is so because some organizations use only executable statements, some useful comments, and some do not. Thus, the standard needs to be established.
- These metrics are not universally accepted.

Size Oriented Metrics …. Contd.

- **Advantages of LOC:**
  - Using these metrics, it is very simple to measure size.
  - Artifact of Software development which is easily counted.
  - LOC is used by many methods that are already existing as a key input.
  - Estimation is closer to developer's perspective.

- **Disadvantage of LOC:**
  - It cannot measure size of specification as it is defined on code.
  - It characterizes only one specific view of size, namely length, it takes no account of functionality or complexity.
  - Bad software design may cause an excessive line of code.
  - It is language dependent.
  - Users cannot easily understand it.
  - At times, it is very difficult to estimate LOC in early stage of development.

Size Oriented Metrics …. Contd.

- Based on the LOC/KLOC count of software, many other metrics can be computed:
  - Errors/KLOC.
  - $/ KLOC.
  - Defects/KLOC.
  - Pages of documentation/KLOC.
  - Errors/PM.
  - Productivity = KLOC/PM (effort is measured in person-months).
  - $/ Page of documentation.

Cost Estimation using LOC

- Assume estimated lines of code of a system is 33,200 LOC. Average productivity for system of this type is 620 LOC/person-month. There are 6 developers. Labor rate is $800 per person-month.
- Calculate the total effort and cost required to complete the above project.

  - Method I:

    Total Effort = Total LOC/Productivity = 33200/620=53.54 ≈ 54 person-months

    6 developers ➔Effort = Total Effort/6 = 54/6 = 9 months

    Total Cost = Total Effort * Labor Rate = 54 * 800 ≈ $43,200

  - Method II:

    Cost per LOC = Labor Rate /Productivity=800/620=$1.29 ≈ $1.3

    Total Cost = Total LOC * Cost per LOC = 33,200* 1.3=$43160 ≈ $43,200

Function Oriented Metrics

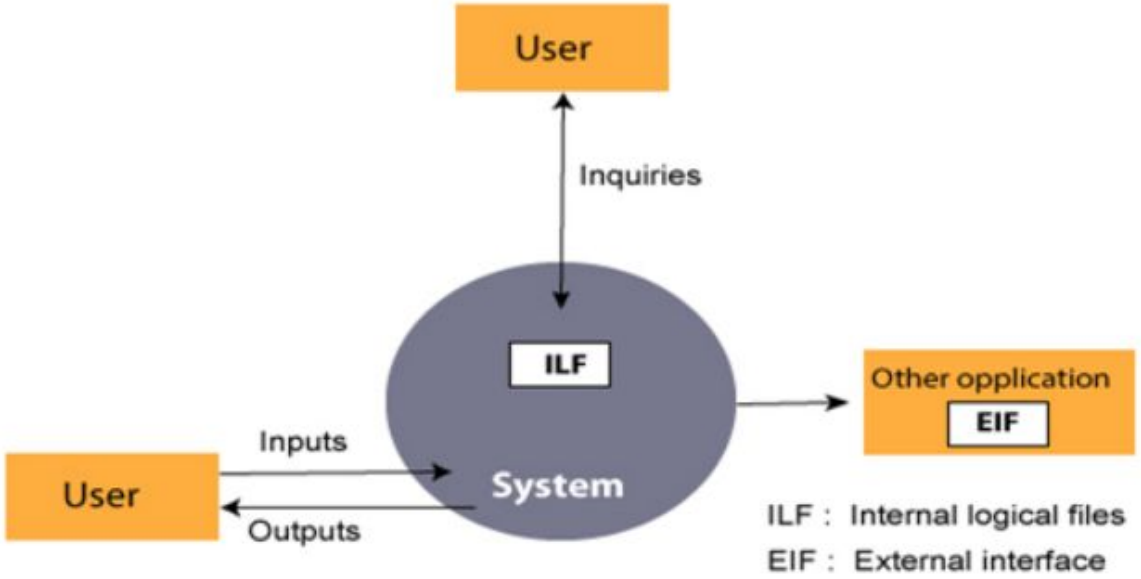
- Function-oriented software metrics use a measure of the functionality delivered by the application as a normalization value.
- The most widely used function-oriented metric is the Function/ Feature Point (FP).
  - Function point metrics provide a standardized method for measuring the various functions of a software application.
  - Computation of the function point is based on the characteristics of the software's information domain and complexity.
  - It measures functionality from the user's point of view, i.e.; on the basis of what the user requests and receives in return.

Function/ Feature Point (FP)

- Information domain values:
  - Number of user/ external inputs – Distinct input from user
    - Ex. Input screen, tables
  - Number of user/ external outputs – Output screens, reports, error messages, etc.
  - Number of user/ external inquiries - Online input that generates some result (Prompts, interrupts, etc.)
  - Number of internal files – Logical files (database, directories, etc.)
  - Number of external interfaces – Data files/connections as interface to other systems (shared databases, shared routines, etc.).

- All these parameters are then individually assessed for complexity.

Function Point (FP) ….Contd.



Function Point (FP) ….Contd.

•Formula to compute FP is:

$$FP = Total\ Count * [0.65 + 0.01 * \sum(Fi)]$$

where, Total-Count is all the counts a weighting factor that is determined for each organization via empirical data.
Fi (i = 1 to 14) are complexity adjustment values.

Value Adjustment Factors (Fi=> 1 to 14):

F1. Data Communication	F2. Distributed Data Processing	F3. Performance
F4. Heavily Used Configuration	F5. Transaction Role	F6. Online Data Entry
F7. End-User Efficiency	F8. Online Update	F9. Complex Processing
F10. Reusability	F11. Installation Ease	F12. Operational Ease
F13. Multiple Sites	F14. Facilitate Change	

Function Point (FP) ….Contd.

Information Domain Value	Count		Weighting factor Simple	Average	Complex		
External Inputs (EIs)		×	3	4	6	=	
External Outputs (EOs)		×	4	5	7	=	
External Inquiries (EQs)		×	3	4	6	=	
Internal Logical Files (ILFs)		×	7	10	15	=	
External Interface Files (EIFs)		×	5	7	10	=	
Count total →							

Measurement Parameter	Count		Weighing factor Simple	Average	Complex		
1. Number of external inputs (EI)	—	*	3	4	6	=	—
2. Number of external Output (EO)	—	*	4	5	7	=	—
3. Number of external Inquiries (EQ)	—	*	3	4	6	=	—
4. Number of internal Files (ILF)	—	*	7	10	15	=	—
5. Number of external interfaces(EIF)	—	*	5	7	10	=	—
Count-total →							

Function Point (FP) ….Contd.

- Fi (i=1 to 14) are “complexity adjustment values” depending upon responses to the following given fourteen questions:
  1) Is there a need of reliable backup and recovery to the system?
  2) Is there any requirement of communications?
  3) Are there distributed processing functions?
  4) Is performance critical?
  5) Will it be possible to execute the system in current, greatly utilized operational environment?
  6) Is there a need of on-line data entry to the system?
  7) Is there need of input transaction to on-line data entry so as to build over multiple screens or operations?
  8) Is the updation of master files done online?
  9) Is there complexity in the inputs, outputs, files or inquiries?
  10) Is there complexity in internal processing?

Function Point (FP) ….Contd.

11) Can the code designed be reusable?
12) Is there involvement of conversion and installation in the design?
13) Is the system designed for more than one installation in various organizations?
14) Does the design of application facilitates change and ease of use by the user?

- All these questions are usually answered with the help of a scale in the range of 0-5.
- The constant values which are present in the Function Point equation and the weighting factors which are applied to the information domain counts are decided based on the facts.

Function Point (FP) ….Contd.

- Example:

inputs

3 simple X 3 = 9

4 average X 4 = 16

1 complex X 6 = 6

outputs

6 average X 5 = 30

2 complex X 7 = 14

files

5 complex X 15 = 75

inquiries

8 average X 4 = 32

interfaces

3 average X 7 = 21

4 complex X 10 = 40

Unadjusted function points 243

- The functional complexities are multiplied with the corresponding weights against each function, and the values are added up to determine the UFP (Unadjusted Function Point) of the subsystem.

Function Point (FP) ….Contd.

F09. Complex internal processing	= 3
F10. Code to be reusable	= 2
F03. High performance	= 4
F13. Multiple sites	= 3
F02. Distributed processing	= 5
Project adjustment factor	= 17

Adjustment calculation:

Adjusted FP = Unadjusted FP X [0.65 + (adjustment factor X 0.01)]

= 243 X [0.65 + (17 X 0.01)]

= 243 X [0.82]

= 199.26 Adjusted function points

FP = Count-total * [0.65 + 0.01 * ∑(fi)]

= Count-total * CAF (Complexity Adjustment Factor)

where Count-total is obtained from the above Table.

CAF = [0.65 + 0.01 *∑(fi)]

Function Point (FP) ….Contd.

- Compute the function point, productivity, documentation, cost per function for the following data:

Number of user inputs = 24
Number of user outputs = 46
Number of inquiries = 8
Number of files = 4
Number of external interfaces = 2
Effort = 36.9 p-m
Technical documents = 265 pages
User documents = 122 pages
Cost = $7744/ month
Various processing complexity factors are: 4, 1, 0, 3, 3, 5, 4, 4, 3, 3, 2, 2, 4, 5.

Function Point (FP) ….Contd.

Note:
Weighting factor is varying for individual parameter (from simple to average to complex)

Measurement Parameter	Count		Weighing factor
1. Number of external inputs (EI)	24	*	4 = 96
2. Number of external outputs (EO)	46	*	4 = 184
3. Number of external inquiries (EQ)	8	*	6 = 48
4. Number of internal files (ILF)	4	*	10 = 40
5. Number of external interfaces (EIF)	2	*	5 = 10

Function Point (FP) ….Contd.

So, sum of all Fi =
(i ← 1 to 14) $= 4 + 1 + 0 + 3 + 5 + 4 + 4 + 3 + 3 + 2 + 2 + 4 + 5$
$= 43$

FP = Count-total * [0.65 + 0.01 *∑(Fi)]
$= 378 * [0.65 + 0.01 * 43]$
$= 378 * [0.65 + 0.43]$
$= 378 * 1.08$
$= 408$

$$\text{Productivity} = \frac{\text{FP}}{\text{Effort}} = \frac{408}{36.9} = 11.1$$

Function Point (FP) ….Contd.

- Total pages of documentation
  = technical document + user document
  = 265 + 122
  = 387 pages

- Documentation = Pages of documentation/FP
  = 387/408
  = 0.94

$$\text{Cost per function} = \frac{\text{cost}}{\text{productivity}} = \frac{7744}{11.1} = \$700$$

Function Point Vs Lines of Code

Function Point (FP)	Lines of Code (LOC)
1. FP is specification based.	1. LOC is an analogy based.
2. FP is language independent.	2. LOC is language dependent.
3. FP is user-oriented.	3. LOC is design-oriented.
4. It is extendible to LOC.	4. It is convertible to FP (i.e. Backfiring)
5. Function Point is used for data processing systems.	5. LOC is used for calculating the size of the computer program.
6. Function Point can be used to portray the project time.	6. LOC is used for calculating and comparing the productivity of programmers.

- In general, people prefer the functional size of software (FP), because the size expressed using the FP metric stays constant in any case and whichever language or languages are used.

Software Project Estimation

- Software project estimation is a form of problem solving.
  - In most cases, the problem to be solved (i.e. developing a cost and effort estimate for a software project) is too complex to be considered in one piece.
  - Estimation of resources, cost and schedule for the software engineering effort requires experience, access to good historical information (metrics), and the courage to commit to quantitative predictions when qualitative information is all that exists.
  - For this reason, one should decompose the problem, characterizing it as a set of smaller and hopefully more manageable problems.

Problem Based Estimation

1. Start with a bounded statement of scope.
2. Decompose the software into problem functions that can each be estimated individually.
3. Compute LOC or FP value for each function.
4. Derive cost or effort estimates by applying the LOC or FP values to your baseline productivity metrics (e.g. LOC/person-month, FP/person-month).
5. Combine function estimates to produce an overall estimate for the entire project.

Software Project Estimation – COCOMO

- Barry Boehm proposed COCOMO (Constructive Cost Estimation Model) in 1981.
- It is one of the most generally used software estimation models in the world.
- It predicts the efforts and schedule of a software product based on the size of the software (number of lines of code – LOC).
- It is a procedural cost estimate model for software projects and 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 is based on the study of 63 projects, which makes it one of the best-documented models.
- The key parameters of COCOMO are:
  - 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 proportional to the effort put.
    - ✔ It is measured in the units of time such as weeks, months.

Software Project Estimation – COCOMO

- In COCOMO, projects are categorized into three types:

i. **Organic**:
   - A development project can be treated of the organic type, if
     - the project deals with developing a well-understood application program,
     - the problem has been solved in the past,
     - 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.

COCOMO …. Contd.

ii. Semidetached:

- 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.
- It requires more experience and better guidance and creativity
- Example of Semidetached system includes developing a new operating system (OS), a Database Management System (DBMS), and complex inventory management system.

COCOMO …. Contd.

iii. Embedded:

- A software project requiring the highest level of complexity, creativity, and experience requirement falls under this category.
- A development project is treated to be of an embedded type, if
  - the team size is larger than the other two models.
  - the developers are sufficiently experienced and creative to develop such complex models.
  - the software being developed is strongly coupled to complex hardware, or
  - the stringent regulations on the operational method exist.
- Example: ATM, Air Traffic control.

COCOMO …. Contd.

All the system types utilize different values of the constants used in Effort Calculations.

- Types of Models:
  - COCOMO consists of a hierarchy of three increasingly detailed and accurate forms.
  - Any of the three forms can be adopted according to our requirements.

  - Types of COCOMO model:
    I. Basic COCOMO Model
    II. Intermediate COCOMO Model
    III. Detailed COCOMO Model

COCOMO …. Contd.

I. Basic COCOMO Model:-

- The first level, Basic COCOMO can be used for quick and slightly rough calculations of Software Costs.
- Its accuracy is somewhat restricted due to the absence of sufficient factor considerations.
- Estimation of Effort:

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

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

$$Personrequired = Effort/time$$

- The constant values a, b, c and d for the Basic Model for the different categories of system.

COCOMO …. Contd.

I. Basic COCOMO Model ……

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

- The effort is measured in Person-Months and as evident from the formula is dependent on Kilo-Lines of code.
- The development time is measured in Months.
- These formulas are used as such in the Basic Model calculations, as not much consideration of different factors such as reliability, expertise is taken into account, henceforth the estimate is rough.

COCOMO …. Contd.

II. 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, there are 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:

(i) Product attributes –

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

COCOMO …. Contd.

II. Intermediate COCOMO Model ……

(ii) Hardware attributes –

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

(iii) Personnel attributes –

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

(iv) Project attributes –

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

COCOMO …. Contd.

Cost Drivers	Very Low	Low	Normal	High	Very High
Product Attributes					
Required Software Reliability	0.75	0.88	1.00	1.15	1.40
Size of Application Database		0.94	1.00	1.08	1.16
Complexity of The Product	0.70	0.85	1.00	1.15	1.30
Hardware Attributes					
Runtime Performance Constraints			1.00	1.11	1.30
Memory Constraints			1.00	1.06	1.21
Volatility of the virtual machine environment		0.87	1.00	1.15	1.30
Required turnabout time		0.94	1.00	1.07	1.15

Cost Drivers	Very Low	Low	Normal	High	Very High
Personnel attributes					
Analyst capability	1.46	1.19	1.00	0.86	0.71
Applications experience	1.29	1.13	1.00	0.91	0.82
Software engineer capability	1.42	1.17	1.00	0.86	0.70
Virtual machine experience	1.21	1.10	1.00	0.90	
Programming language experience	1.14	1.07	1.00	0.95	
Project Attributes					
Application of software engineering methods	1.24	1.10	1.00	0.91	0.82
Use of software tools	1.24	1.10	1.00	0.91	0.83
Required development schedule	1.23	1.08	1.00	1.04	1.10

COCOMO …. Contd.

II. Intermediate COCOMO Model ……..

- The project manager is to rate these 15 different parameters for a particular project on a scale of one to three.
- Then, depending on these ratings, appropriate cost driver values are taken from the above table.
- These 15 values are then multiplied to calculate the EAF (Effort Adjustment Factor).
- The Intermediate COCOMO formula now takes the form:

$$\mathrm{E}= (\mathrm{a}(\mathrm{KLOC})^{b}) * EAF$$

- The values of a and b in case of the intermediate model are as follows:

Software Projects	a	b
Organic	3.2	1.05
Semi Detached	3.0	1.12
Embedded	2.8	1.20

COCOMO …. Contd.

II. Detailed COCOMO Model:-

- Detailed COCOMO incorporates all characteristics of the intermediate version with an assessment of the cost driver's impact on each step of the software engineering process.
- The detailed model uses different effort multipliers for each cost driver attribute.
  - In detailed COCOMO, the whole software is divided into different modules and then we apply COCOMO in different modules to estimate effort and then sum the effort.
  - The Six phases of detailed COCOMO are:
    a. Planning and requirements
    b. System design
    c. Detailed design
    d. Module code and test
    e. Integration and test
    f. Cost Constructive model
  - The effort is calculated as a function of program size and a set of cost drivers are given according to each phase of the software lifecycle.

COCOMO II

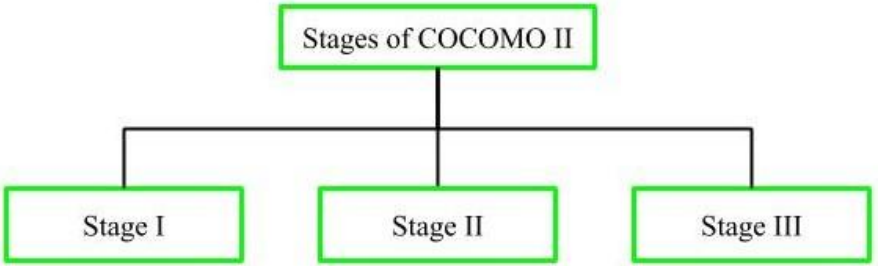
- The Constructive Cost Model (COCOMO) is an algorithmic software cost estimation model developed by Barry Boehm in 1981.
- It is an empirical model that was derived with the help of gathering data from a large number of software projects.
- These data were analyzed to determine formulae that are best fit to the observations.
  - The model uses a basic regression formula, with parameters that are derived from historical project data and current project characteristics.
- These formulae link the size of the system and product, project and team factors to the effort to develop the system.
- We can select to use the COCOMO model for a number of reasons:
  - It is well documented, accessible in the public domain and supported by public domain and commercial tools.
  - It has been widely used and evaluated in several organizations.

COCOMO II …. Contd.

- 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 does not 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 …. Contd.

COCOMO II has three different levels/ models:



1.Stage-I :The Application Composition Model

- It assumes that systems are generated from reusable components, scripting or database programming.
- It is designed to create estimates of prototype development.
- This model was established into COCOMO II to maintain the estimation of effort needed for prototyping projects.
- It depends on an estimate of weighted application points called as object points separated by a standard estimate of application-point productivity.

COCOMO II …. Contd.

- The estimate is then changed according to the complexity of development of every object point.
- Programmer productivity is also based on the experience and capability as well as the capability of the CASE tools used to support development.

- Formula for effort calculation :

  PM = (NAP *(1 * %reuse /100))/PROD

  Where, PM is effort estimate in person-months

  NAP is total number of application points in the delivered system

  %reuse is estimate of the amount of reused code in the development

  PROD is object-point productivity

COCOMO II …. Contd.

2. Stage-II :The Early Design Model

- This model is used at the time of early stages of the system design after the requirements have been established.
- Estimates are depending on function points, which are then changed to number of lines of source code.
- Estimates can be completed after the user requirements have been agreed.

- Formula

  E = A * SizeB * M

  - The multiplier M in COCOMO II depends on a simplified collection of seven projects and process characteristics that influence the estimate.
  - These can increase or decrease the effort needed.

COCOMO II …. Contd.

- The characteristics used in the early design model are product reliability and complexity (RCPX), reuse required (RUSE), platform difficulty (PDIF), personnel capability (PERS), personnel experience (PREX), schedule (SCED) and support facilities (FCIL).

- This results in an effort computation as follows:

  PM = 2.94 * SizeB * M

  where,

  M = PERS * RCPX * RUSE * PDIF * PREX * FCIL * SCED

COCOMO II …. Contd.

3. Stage-III :The Post-Architecture Model

- This model is used at the time of early stages of the system design after the requirements have been established.
- Estimates are depending on function points, which are then changed to number of lines of source code.
- Estimates can be completed after the user requirements have been agreed.

- Formula

  E = A * SizeB * M

  - The multiplier M in COCOMO II depends on a simplified collection of seven projects and process characteristics that influence the estimate.
  - These can increase or decrease the effort needed.

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:**
   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.

COCOMO II

2. The Early Design Model

3. 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.

COCOMO II

COCOMO II can be used for the following major decision situations:

- Making investment or other financial decisions involving a software development effort
- Setting project budgets and schedules as a basis for planning and control
- Deciding on or negotiating tradeoffs among software cost, schedule, functionality, performance or quality factors
- Making software cost and schedule risk management decisions
- Deciding which parts of a software system to develop, reuse, lease, or purchase
- Making legacy software inventory decisions: what parts to modify, phase out, outsource, etc
- Setting mixed investment strategies to improve organization's software capability, via reuse, tools, process maturity, outsourcing, etc.

Empirical Estimation Models

- Empirical estimation techniques are usually founded on making a studied guess regarding the project parameters.
  - If similar products are developed previously, then that experience is always helpful.
- Even though the base of empirical estimation techniques is common sense, several activities involved in estimation have been formalized over the years.
- There are two frequently used empirical estimation techniques:

1) **Expert Judgement technique**
   - In this approach, an expert analyzes the problem in detail and makes guess regarding the problem size.
   - Usually the expert makes an estimation of the cost regarding the different components (such as modules or subsystems) of the respective system and then integrates them to find out the overall estimate.
     - Yet, this technique is exposed to human errors.
     - Additionally, it is likely that the expert may miss few factors accidently.

Empirical Estimation Models ….

- Also, an expert who is making the estimate may lack experience as well as knowledge of all aspects of a project.
  - Ex. Expert may be familiar with the database and user interface components but may not be conversant regarding the computer communication part.
- Estimation made by group of experts can be considered as more refined form of expert judgement.
  - It minimizes factors like individual oversight, lack of familiarity with a specific aspect regarding the project, etc.

3) Delphi Cost estimation

- In this approach, some shortcomings of the expert judgement approach are tried to resolve.
- In this, a team which involves a group of experts and a coordinator takes part.

Empirical Estimation Models ….

- The coordinator gives a copy of the Software Requirement Specification (SRS) document to all of the estimators and a particular form for the purpose of recording their cost estimates.
- Individual estimates are completed by all of the estimators and submitted to the coordinator.
- In individual estimates, the estimators point out any unusual characteristics regarding the product which has great impact on the estimation.
- The coordinator then prepares a summary of the responses by considering all the estimators, and incorporates any unusual rationale that may be noted by any of the estimators.
- By considering this summary, all of the estimators re-estimate. This method is repeated for several rounds.
- However, estimators cannot communicate with each other regarding the estimates. The reason is that the estimate of more experienced or senior estimator may influence other estimators.
- After several iterations of estimations are done, the coordinator compiles the results and prepares the final estimate.

Specialized Estimation Techniques

In general, the decomposition techniques can be used for any software project.

❖However, when there is very short period of time for project completion (weeks instead of months), which is probable to have a continuing series of changes, estimation in particular should be abbreviated.

❖Following are the two specialized estimation techniques:

A. Estimation for Agile Development

- As the requirements for an agile project are set by user scenarios, an estimation approach can be developed for each software increment.
- Approach of decomposition is used in the estimation for agile projects which encompasses the following steps:
  i. Each and every user scenario is considered independently for estimation purposes.

Specialized Estimation Techniques

ii. The scenario is divided into a group of software engineering tasks which will be necessary to develop it.

iii. (a) The estimation of effort needed for each task is done separately.

iii. (b) On the other hand, the estimation of "volume" of the scenario can be done in LOC (Lines of Code), FP (Function Point), or in any other volume-oriented measure (e.g. use case count)

iv. (a) Estimates of all the tasks are combined to set an estimate for the scenario.

iv. (b) Alternatively, historical data is used to translate the volume estimate for the scenario into effort.

v. The effort estimates for each and every scenario which is to be implemented for a provided software increment are

Specialized Estimation Techniques

- Since the project duration essential for the software increment development is relatively short (3-6 weeks), two purposes are served by this estimation:
  - ✔ To make sure that the number of scenarios which are to be considered in the increment conforms to the existing resources
  - ✔ To set a basis for assigning effort as the increment is developed.

B. Estimation of WebApp Projects

- WebApp projects usually like to refer the agile process model.
- An updated FP measure helps to develop an estimate for the WebApp.
- Following approach is suggested when adapting FP for WebApp estimation
  - Inputs are nothing but the input screen or form (ex. CGI or

Specialized Estimation Techniques

- Since the project duration essential for the software increment development is relatively short (3-6 weeks), two purposes are served by this estimation:
  - ✔ To make sure that the number of scenarios which are to be considered in the increment conforms to the existing resources
  - ✔ To set a basis for assigning effort as the increment is developed.

B. Estimation of WebApp Projects

- WebApp projects usually like to refer the agile process model.
- An updated FP measure helps to develop an estimate for the WebApp.
- Following approach is suggested when adapting FP for WebApp estimation
  - a) Inputs are nothing but the input screen or form (ex. CGI or Java), every maintenance screen, each tab (if used).

Specialized Estimation Techniques

b) Outputs are very static. Web page, every dynamic Web page script (ex. ASP, PHP, or other DHTML script), and every report.

c) Tables are very logical in the DB plus, every XML object, if used.

d) Interfaces preserve their definition as logical files (such as unique record formats) into the out-of-the-system boundaries.

e) Queries are the ones which are published externally or take help of a message-oriented interface such DCOM or COM as external references.

- Function Points (FPs) are considered as logical indicator of volume for a WebApp.

Specialized Estimation Techniques

- It is possible to determine the volume for a WebApp by gathering measures (known as “predictor variables”) allied with the application (such as page count, media count, function count), its characteristics (such as page complexity, linking complexity, graphic complexity), characteristics of media (such as media duration), and functional characteritics (such as code length, reused code length).

- These measures help to establish empirical estimation models for the entire project effort, page and media authoring effort, and scripting effort.

Project Scheduling and Tracking

- Project schedule is a mechanism that is used to communicate and know that tasks are needed and has to be done or performed.
- It is a technique that describes the sequence of tasks to be done and assigns organizational resources to complete that tasks in pre-defined time frame.
- A project is nothing but a group of many tasks, and each task has scheduling start and end dates.
- Thus, project scheduling is a kind of document that summarizes the work required to deliver the project on time.
- It also distributes the estimated efforts across the planned project period.
- It establishes the “Road map” for the project manager.

Software Project Planning

- The objective of software project planning is to provide a framework that enables the manager to make reasonable estimates of resources, cost, and schedule.
- In addition, estimates should attempt to define best-case and worst-case scenarios so that project outcomes can be bounded.
- Although there is an inherent degree of uncertainty, the software team embarks on a plan that has been established as a consequence of these tasks.

- Therefore, the plan must be adapted and updated as the project proceeds.

Task set for Project planning

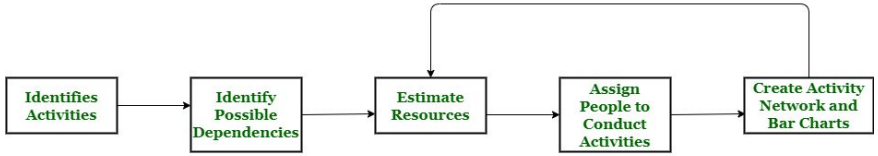
i. Establish project scope
ii. Determine feasibility
iii. Analyze risks
iv. Define required resources
    a) Determine required human resources
    b) Define reusable software resources
    c) Identify environmental resources
v. Estimate cost and effort
    a) Decompose the problem
    b) Develop two or more estimates using size, function points, process tasks, or use cases
    c) Reconcile the estimates
vi. Develop a project schedule
    a) Establish a meaningful task set
    b) Define a task network
    c) Use scheduling tools to develop a time-line chart
    d) Define schedule tracking mechanisms

Project Delivery

- Reasons for delay in project delivery:
  a) An unrealistic deadline established by someone outside the software engineering group and forced on managers and practitioners within the group.
  b) Changing customer requirements that are not reflected in scheduled changes.
  c) An honest under estimate of the amount of effort and/ or the number of resources that will be required to do the job.
  d) Predictable and/ or unpredictable risks that are not considered when the project commenced.
  e) Technical difficulties that could not have been foreseen in advance.
  f) Human difficulties that could not have been foreseen in advance.
  g) Miscommunication among project staff that results in delays.
  h) A failure by project management to recognize that the project is falling behind schedule and the lack of action to correct the problem.

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 should do the following:



Project Scheduling Process

I. Identify all the functions required to complete the project.
   - The first method in scheduling a software plan involves identifying all the functions required to complete the project.
   - A good judgment of the intricacies of the project and the development process helps the supervisor to identify the critical role of the project effectively.

Project Scheduling …. Contd.

II. Break down large functions into small activities.
- Next, the large functions are broken down into a valid set of small activities which would be assigned to various engineers.

III. Determine the dependency among various activities.
- The work breakdown structure formalism supports the manager to breakdown the function systematically after the project manager has broken down the purpose and constructs the work breakdown structure; he has to find the dependency among the activities.
- Dependency among the various activities determines the order in which the various events would be carried out.
- If an activity A necessitates the results of another activity B, then activity A must be scheduled after activity B.

Project Scheduling …. Contd.

- In general, the function dependencies describe a partial ordering among functions, i.e., each service may precede a subset of other functions, but some functions might not have any precedence ordering describe between them (called concurrent function).
- The dependency among the activities is defined in the pattern of an activity network.

V. Establish the most likely size for the time duration required to complete the activities.

VI. Allocate resources to activities.

- Once the activity network representation has been processed out, resources are allocated to every activity.
- Resource allocation is usually done using a Gantt chart.
- After resource allocation is completed, a PERT chart representation is developed.

Project Scheduling …. Contd.

- The PERT chart representation is useful for program monitoring and control.

VII. Plan the beginning and ending dates for different activities.

- For task scheduling, the project plan needs to decompose the project functions into a set of activities.
- The time frame when every activity is to be performed is to be determined.
- The end of every action is called a milestone.
- The project manager tracks the function of a project by auditing the timely completion of the milestones.
- If he examines that the milestones start getting delayed, then he has to handle the activities carefully so that the complete deadline can still be met.

VIII. Determine the critical path. A critical way is the group of activities that decide the duration of the project.

Project Scheduling Guidelines

Basic principles for software project scheduling are:

1. Compartmentalization
   - The project must be divided into a number of manageable activities, actions and tasks.
   - To accomplish this, both the product and the process are decomposed.

2. Interdependency
   - The interdependency of each decomposed activity, action or task must be determined.
   - Some tasks must occur in sequence, while others can occur in parallel.
   - Dependent tasks cannot commence until the work product produced by another task is available.

Project Scheduling Guidelines ….

3. Time allocation
   - Each task to be scheduled must be allocated some number of work units (i.e. effort by person in days/months.
   - Each task must be assigned a start date and a completion date.

4. Effort Validation
   - Every project must have a defined number of people working on a specific project as a team.
   - As time allocation occurs, project manager should check no more than allocated number of people have been scheduled at any given time.

5. Defined Responsibilities
   - Every task that is scheduled should be assigned to a specific team member.

Project Scheduling Guidelines ….

6. Defined Outcomes
   - Each task that is scheduled should have a defined outcome. i.e. a work product (design of a module) or a part of a work product.

7. Defined Milestones
   - Every task or a group of tasks should be associated with a project milestone.
   - A milestone is accomplished when one or more work products have been reviewed for quality and been approved.

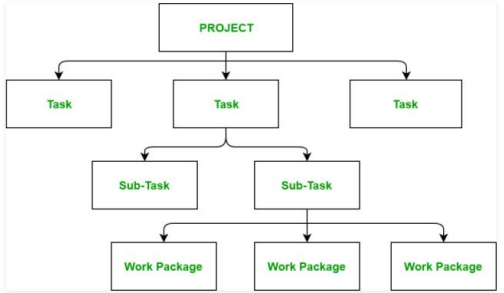
Project Effort Distribution

- The effort distribution rule is as follows:
  - 40-20-40 rule
    - 40% front-end analysis and design
    - 20% coding
    - 40% back-end testing

- Generally accepted guidelines are:
  - 02-03% planning
  - 10-25% requirements analysis
  - 20-25% design
  - 15-20% coding
  - 30-40% testing and debugging

Work Breakdown Structure (WBS)

- A Work Breakdown Structure includes dividing a large and complex project into simpler, manageable and independent tasks.
- The root of this tree (structure) is labelled by the Project name itself.
- For constructing a work breakdown structure, each node is recursively decomposed into smaller sub-activities, until at the leaf level, the activities becomes undividable and independent.
- It follows a Top-Down approach.



- Steps:
  - Step-1: Identify the major activities of the project.
  - Step-2: Identify the sub-activities of the major activities.
  - Step-3: Repeat till undividable, simple and independent activities are created.

WBS …. Contd.

- The work breakdown structure (WBS) defines the work that is required in order to produce the product or deliverables.
- It is represented as a hierarchical subdivision of a project into work areas with the lowest generally being a work package or sometimes even an activity.
- The lowest level of the WBS should be consistent and agreed at the outset of the creation of the WBS.
- The WBS provides the foundation for all project management work, including planning, cost and effort estimation, resource allocation, and scheduling.

WBS …. Contd.

- Reasons for creating a WBS:
  - enables the definition of the total scope of work
  - provides the ability to assign work to people responsible for carrying out the work
  - establishes a control baseline
  - measures accomplishments objectively when the work is done
  - defines, collects and reports information at the appropriate level required
  - defines the relationships between work, organization and cost.

Construction of WBS

- First, the project managers and top level management identifies the main deliverables of the project.
- Next, these main deliverables are broken down into smaller higher-level tasks and this complete process is done recursively to produce much smaller independent tasks.
  - It depends on the project manager and team that up to which level of detail they want to break down their project.
- Generally the lowest level tasks are the most simplest and independent tasks and takes less than two weeks worth of work.
- Hence, there is no rule for up to which level we may build the WBS of the project as it totally depends upon the type of project we are working on and the management of the company.
- The efficiency and success of the whole project majorly depends on the quality of the WBS of the project and hence, it implies its importance.

WBS – Work Packages

- The WBS decomposes the project into smaller components and more manageable units of work called work packages.
- Work packages provide a logical basis for defining the project activities and assigning the resources to those activities so that all the project work is identified.
- A work package makes it possible to develop a project plan, schedule, and budget and then later monitor the project's progress.

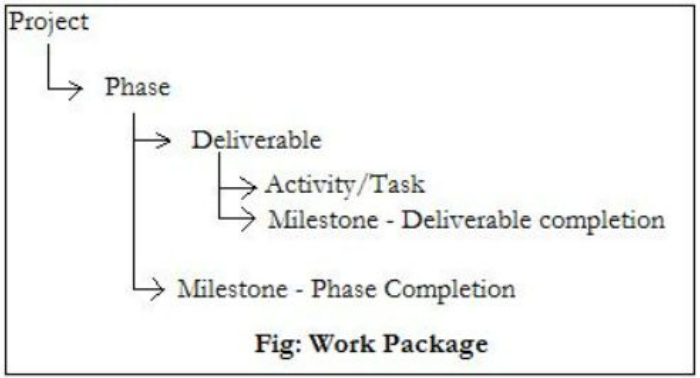


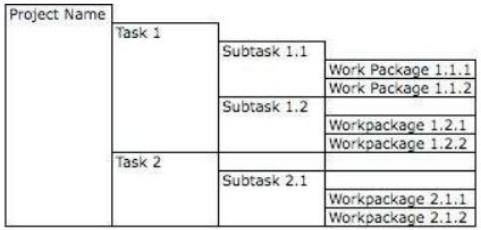
Fig: Work Package

Work Packages …. Contd.

- Deliverables and Milestones:
  - A milestone is a significant event or achievement that provides evidence that the deliverable has been completed or that a phase is formally over.
  - While a deliverable includes such things as presentations, report, plans and final application system, a milestone must focus on achievement.
  - If a project team succeeds in meeting all of its scheduled milestones, the project should finish as planned.
  - The project risk is reduced by having a milestone.

WBS …. Contd.

- The main purpose of a WBS is to reduce complicated activities to a collection of tasks.
- This is important for the project manager because he/ she can oversee the tasks more effectively than the complex activities.
- Tasks must be measurable and independent, with clearly defined limits.



- **Uses:**
  - It allows to do a precise cost estimation of each activity.
  - It allows to estimate the time that each activity will take more precisely.
  - It allows easy management of the project.
  - It helps in proper organization of the project by the top management.

Scheduling Techniques

- Project scheduling is concerned with the techniques that can be employed to manage the activities that need to be undertaken during the development of a project.
- Scheduling is carried out in advance of the project commencing and involves:
  - identifying the tasks that need to be carried out;
  - estimating how long they will take;
  - allocating resources (mainly personnel);
  - scheduling when the tasks will occur.
- Once the project is underway, control needs to be exerted to ensure that the plan continues to represent the best prediction of what will occur in the future:
  - based on what occurs during the development;
  - often necessitates revision of the plan.

Scheduling Techniques ….

- Effective project planning will help to ensure that the systems are delivered:
  - within cost;
  - within the time constraint;
  - to a specific standard of quality.

- Following are the Project Scheduling Techniques:
  - Critical Path Method (CPM)
  - Program Evaluation and Review Technique (PERT)

- They are the two most commonly used techniques by project managers.
- These methods are used to calculate the time span of the project through the scope of the project.

Critical Path Method (CPM)

- Critical Path Method (CPM) is a method used in project planning, generally for project scheduling for the on-time completion of the project.
- It actually helps in the determination of the earliest time by which the whole project can be completed.

- There are two main concepts in this method namely critical task and critical path.
  - Critical task is the task/activity which can not be delayed otherwise the completion of the whole project will be delayed.
    - It must be completed on-time before starting the other dependent tasks.
  - Critical path is a sequence of critical tasks/activities and is the largest path in the project network.
    - It gives us the minimum time which is required to complete the whole project.
    - The activities in the critical path are known as critical activities and if these activities are delayed then the completion of the whole project is also delayed.

Critical Path Method …. Contd.

- Major steps of the Critical Path Method:
  1. Identifying the activities
  2. Construct the project network
  3. Perform time estimation using forward and backward pass
  4. Identify the critical path

- The table in the next slide contains the activity label, its respective duration (in weeks) and its precedents.
- We will use critical path method to find the critical path and activities of this project.

Critical Path Method …. Contd.

Activity	Duration (in weeks)	Precedents
A	6	–
B	4	–
C	3	A
D	4	B
E	3	B
F	10	–
G	3	E, F
H	2	C, D

Critical Path Method …. Contd.

- Rules for designing the Activity-on-Node (AON) network diagram:

  - A project network should have only one start node
  - A project network should have only one end node
  - A node has a duration
  - Links normally have no duration
  - “Precedents” are the immediate preceding activities
  - Time moves from left to right in the project network
  - A network should not contain loops
  - A network should not contain dangles

Critical Path Method …. Contd.

Node Representation:

- **Activity label** is the name of the activity represented by that node.
- **Earliest Start** is the date or time at which the activity can be started at the earliest.
- **Earliest Finish** is the date or time at which the activity can completed at the earliest.
- **Latest Start** is the date or time at which the activity can be started at the latest.
- **Latest Finish** is the date or time at which the activity can be finished at the latest.
- **Float** is equal to the difference between earliest start and latest start or earliest finish and latest finish.

Critical Path Method …. Contd.

Forward Pass:

The forward pass is carried out to calculate the earliest dates on which each activity may be started and completed.

1) Activity A may start immediately. Hence, earliest date for its start is zero i.e. ES(A) = 0. It takes 6 weeks to complete its execution. Hence, earliest it can finish is week 6 i.e. EF(A) = 6.

2) Activity B may start immediately. Hence, earliest date for its start is zero i.e. ES(B) = 0. It takes 4 weeks to complete its execution. Hence, earliest it can finish is week 4 i.e. EF(B) = 4.

3) Activity F may start immediately. Hence, earliest date for its start is zero i.e. ES(F) = 0. It takes 10 weeks to complete its execution. Hence, earliest it can finish is week 10 i.e. EF(F) = 10.

Critical Path Method …. Contd.

4) Activity C starts as soon as activity A completes its execution. The earliest week it can start its execution is week 6 i.e. ES(C) = 6. It takes 3 weeks to complete its execution. Hence, earliest it can finish is week 9 i.e. EF(C) = 9.

5) Activity D starts as soon as activity B completes its execution. Earliest week it can start its execution is week 4 i.e. ES(D) = 4. It takes 4 weeks to complete its execution. Hence, earliest it can finish is week 8 i.e. EF(D) = 8.

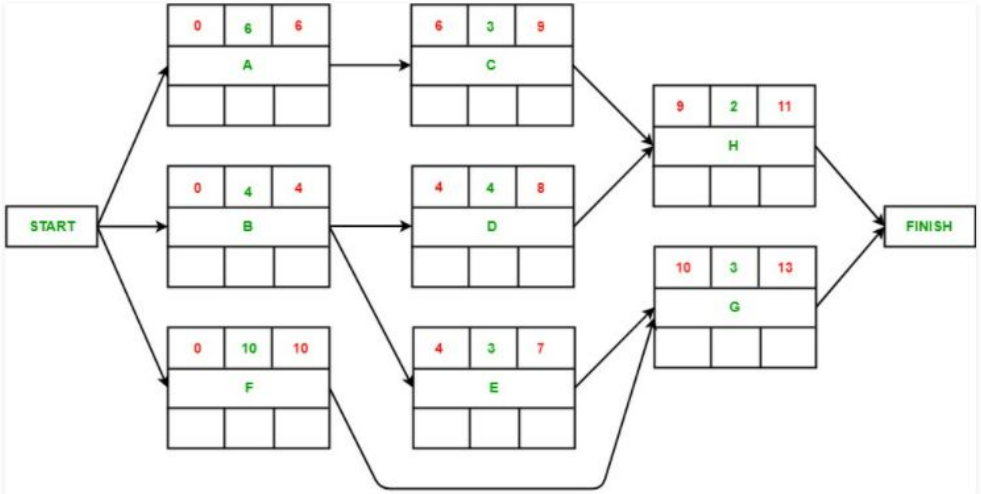
6) Activity E starts as soon as activity B completes its execution. Earliest week it can start its execution is week 4 i.e. ES(E) = 4. It takes 3 weeks to complete its execution. Hence, earliest it can finish is week 7 i.e. EF(E) = 7.

7) Activity G starts as soon as activity E and activity F completes their execution. Since, activity requires the completion of both for starting its execution, we would consider the MAX(ES(E), ES(F)).

Critical Path Method …. Contd.

7) ….. Earliest week it can start its execution is week 10 i.e. ES(G) = 10. It takes 3 weeks to complete its execution. Hence, earliest it can finish is week 13 i.e. EF(G) = 13.

8) Activity H starts as soon as activity C and activity D completes their execution. Since, activity requires the completion of both for starting its execution, we would consider the MAX(ES(C), ES(D)). Hence, earliest week it can start its execution is week 9 i.e. ES(H) = 9. It takes 2 weeks to complete its execution. Hence, earliest it can finish is week 11 i.e. EF(H) = 11.



Critical Path Method …. Contd.

Backward Pass:

- The backward pass is carried out to calculate the latest dates on which each activity may be started and finished without delaying the end date of the project.
- Assumption: Latest finish date = Earliest Finish date (of project).

1. Activity G's latest finish date is equal to the earliest finish date of the precedent activity of finish according to the assumption i.e. LF(G) = 13. It takes 3 weeks to complete its execution. Hence, latest it can start is week 10 i.e. LS(G) = 10.

2. Activity H's latest finish date is equal to the earliest finish date of the precedent activity of finish according to the assumption i.e. LF(H) = 13. It takes 2 weeks to complete its execution. Hence, latest it can start is week 11 i.e. LS(H) = 11.

Critical Path Method …. Contd.

3. The latest end date for activity C would be the latest start date of H i.e. LF(C) = 11. It takes 3 weeks to complete its execution. Hence, latest it can start is week 8 i.e. LS(C) = 8.

4. The latest end date for activity D would be the latest start date of H i.e. LF(D) = 11. It takes 4 weeks to complete its execution. Hence, latest it can start is week 7 i.e. LS(D) = 7.

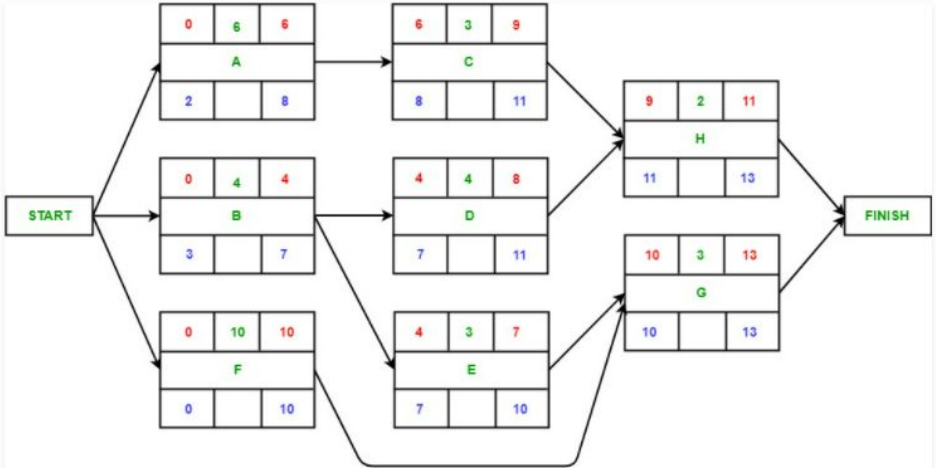
5. The latest end date for activity E would be the latest start date of G i.e. LF(G) = 10. It takes 3 weeks to complete its execution. Hence, latest it can start is week 7 i.e. LS(E) = 7.

6. The latest end date for activity F would be the latest start date of G i.e. LF(G) = 10. It takes 10 weeks to complete its execution. Hence, latest it can start is week 0 i.e. LS(F) = 0.

Critical Path Method …. Contd.

7. The latest end date for activity A would be the latest start date of C i.e. LF(A) = 8. It takes 6 weeks to complete its execution. Hence, latest it can start is week 2 i.e. LS(A) = 2.

8. The latest end date for activity B would be the earliest of the latest start date of D and E i.e. LF(B) = 7. It takes 4 weeks to complete its execution. Hence, latest it can start is week 3 i.e. LS(B) = 3.



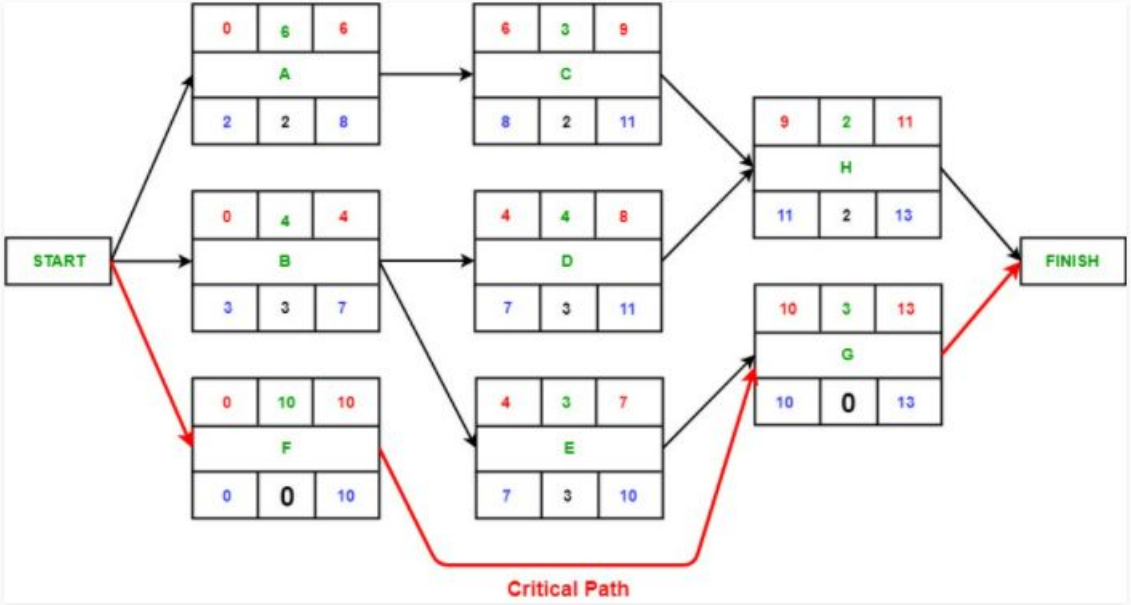
Critical Path Method …. Contd.

- **Identifying Critical Path:**
  - Critical path is the path which gives us or helps us to estimate the earliest time in which the whole project can be completed.
  - Any delay to an activity on this critical path will lead to a delay in the completion of the whole project.
  - In order to identify the critical path, we need to calculate the activity float for each activity.

- Activity float is actually the difference between an activity's Earliest start and its latest start date or the difference between the activity's Earliest finish and its latest finish date.
  - It indicates how much the activity can be delayed without delaying the completion of the whole project.

Critical Path Method …. Contd.

- If the float of an activity is zero, then the activity is an critical activity and must be added to the critical path of the project network.
- In this example, activity F and G have zero float and hence, are critical activities.



Critical Path Method …. Contd.

- **Advantages of Critical Path Method (CPM):**
  - It figures out the activities which can run parallel to each other.
  - It helps the project manager in identifying the most critical elements of the project.
  - It gives a practical and disciplined base which helps in determining how to reach the objectives.
  - CPM is effective in new project management.
  - CPM can strengthen a team perception if it is applied properly.
  - CPM provides demonstration of dependencies which helps in the scheduling of individual activities.
  - It shows the activities and their outcomes as a network diagram.
  - It gives a fair and concise procedure of documenting of project.
  - It helps in determining the slack time.

Critical Path Method …. Contd.

- An explicit and clear approach of communicating project plans, schedules, time and cost performance is developed.
- It is extensively used in industry.
- It helps in optimization by determining the project duration.

- **Disadvantages of Critical Path Method (CPM):**
  - The scheduling of personnel is not handled by the CPM.
  - In CPM, it is difficult to estimate the completion time of an activity.
  - The critical path is not always clear in CPM.
  - For bigger projects, CPM networks can be complicated too.
  - It also does not handle the scheduling of the resource allocation.
  - In CPM, critical path needs to be calculated precisely.

Program Evaluation and Review Technique

- Project Evaluation and Review Technique (PERT) is a procedure through which activities of a project are represented in its appropriate sequence and timing.
  - It is a scheduling technique used to schedule, organize and integrate tasks within a project.
  - PERT is basically a mechanism for management planning and control which provides blueprint for a particular project.
  - All of the primary elements or events of a project are finally identified by the PERT.

  - In this technique, a PERT Chart is made which represents a schedule for all the specified tasks in the project.
  - The reporting levels of the tasks or events in the PERT Charts is somewhat same as defined in the work breakdown structure (WBS).

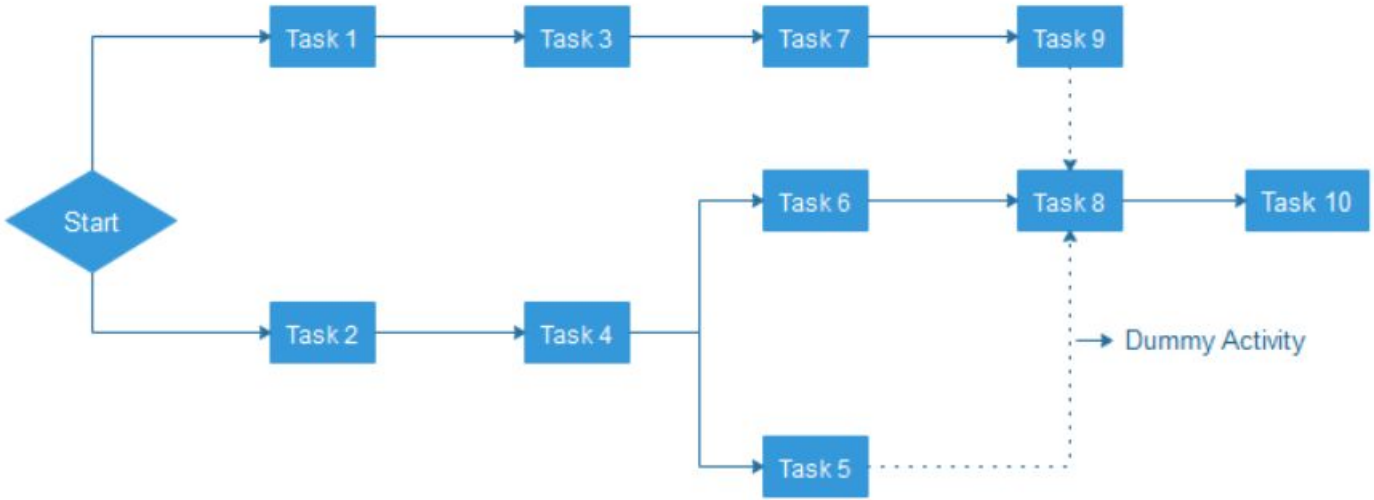
PERT Characteristics

- PERT is an acronym of Program Evaluation Review Technique.
- In the 1950s, it is developed by the U.S. Navy to handle the Polaris submarine missile program.

- The main characteristics of PERT are as following :
  - It serves as a base for obtaining the important facts for implementing the decision-making.
  - It forms the basis for all the planning activities.
  - PERT helps management in deciding the best possible resource utilization method.
  - PERT takes advantage by using time network analysis technique.
  - PERT presents the structure for reporting information.
  - It helps the management in identifying the essential elements for the completion of the project within time.

PERT …. Contd.

- PERT chart is represented as a network diagram concerning the number of nodes, which represents events.



  - The direction of the lines indicates the sequence of the task.
  - Here, tasks between "Task 1 to Task 9" must complete, and these are known as a dependent or serial task.

PERT …. Contd.

- Between Task 4 and 5, and Task 4 and 6, nodes are not dependent and can undertake simultaneously. These are known as Parallel or concurrent tasks.
- Without resource or completion time, the task must complete in the sequence which is considered as event dependency, and these are known as Dummy activity and represented by dotted lines.

- Logic Network
  - The Logic Network shows the order of activities over time.
  - It shows the sequence in which activities are to be done.
  - Distinguishing events and pinning down the project are the two primary uses.
  - Moreover, it will help with understanding task dependencies, a timescale, and overall project workflow.

PERT …. Contd.

- There are three estimates involved in PERT:
  a. TOPT (Optimistic Time Estimate)
     - This is the fastest time during which an activity can be finished.
     - The assumption is made that all the required resources are available and all the previous activities are completed as planned.
  b. TLIKELY (Most Likely Time Estimate)
     - Generally, project managers are asked to present one estimate during the initial project period.
     - In that case, this is the estimate which goes to the upper management.
  c. TPESS (Pessimistic Time Estimate)
     - This is the maximum time required to complete an activity.
     - It is assumed that many things related to project activity will go wrong.
     - A lot of rework will need to be done and resource unavailability is considered when such estimate is derived.

PERT …..Contd.

- **Advantages of PERT**:
  - Estimation of completion time of project is given by the PERT.
  - It supports the identification of the activities with slack time.
  - The start and end dates of the activities of a specific project is determined.
  - It helps project manager in identifying the critical path activities.
  - PERT makes well organized diagram for the representation of large amount of data.

PERT …..Contd.

- **Disadvantages of PERT:**
  - The complexity of PERT is more which leads to the problem in implementation.
  - The estimation of activity time are subjective in PERT which is a major disadvantage.
  - Maintenance of PERT is also expensive and complex.
  - The actual distribution may be different from the PERT beta distribution which causes wrong assumptions.
  - It under estimates the expected project completion time as there are chances that other paths can become the critical path if their related activities are deferred.

PERT Vs CPM

PERT	CPM
PERT is a project management technique, used to manage uncertain activities of a project.	CPM is a statistical technique of project management that manages well defined activities of a project.
It is a technique of planning and control of time.	It is a method to control cost and time.
It is a probability model.	It is a deterministic model.
It is event oriented technique which means that network is constructed on the basis of event.	It is activity oriented technique which means that network is constructed on the basis of activities.
It is appropriate for high precision time estimation.	It is appropriate for reasonable time estimation.
It has non-repetitive nature of job.	It has repetitive nature of job.

PERT Vs CPM

PERT	CPM
It uses three estimates.	It uses one estimate.
There is no chance of crashing as there is no certainty of time.	There may be crashing because of certain time bound.
It uses dummy activities for representing sequence of activities.	It does not use any dummy activities.
It is suitable for projects which require research and development.	It is suitable for construction projects.

Project Tracking

- The project schedule is a road map which defines the tasks and milestones that are to be tracked and controlled as the project proceeds.
- Tracking can be done in a number of different ways:
  i. By calling project status meetings periodically in which each team member reports progress of assignments/ tasks.
  ii. Evaluating the reviews during the software engineering process.
  iii. Setting the tentative project deadlines that have been completed by the scheduled dates.
  iv. Comparing the real start date to the intended start date for every project.
  v. Meeting can be conducted informally with resources to obtain their progress on the given assignment.

- All of these tracking techniques are used by experienced project managers.

Project Tracking

- If things are going well (i.e. the project is on schedule and within allocated budget, reviews progress is also going well and milestones are being reached), control is normal.
- But, if problems occur, the project manager has to control them as quickly as possible.
  - After the problem has been identified, additional resources may be focused on problem areas, staff may be redeployed or the project schedule can be changed.
- If the project manager is facing deadline pressure, experienced project managers sometimes have to take the decision about the project scheduling and this control technique is called as time-boxing.
  - Time-boxing approach identifies that the whole product may not be deliverable by the pre-defined target.
  - Hence, an incremental software process is taken into consideration and a schedule is preceded for each incremental delivery.

TimeLine Charts/ Gantt Charts

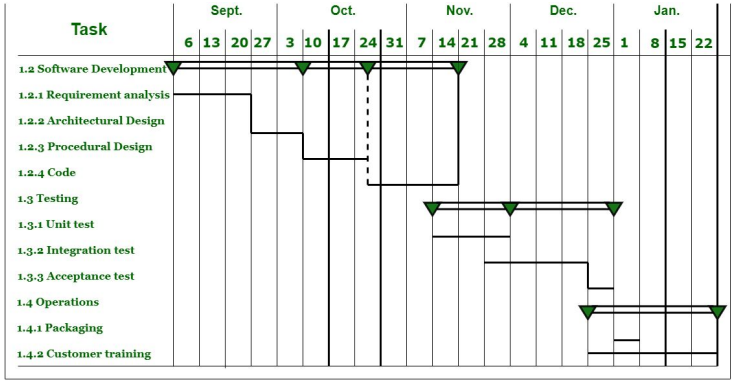
- **Generalized Activity Normalization Time Table** (GANTT) chart is type of chart in which series of horizontal lines are present.
  - These lines show the amount of work done or production completed in given period of time in relation to amount planned for those projects.

- It is horizontal bar chart developed by Henry L. Gantt (American engineer and social scientist) in 1917 as production control tool.

- It is simply used for graphical representation of schedule that helps to plan in an efficient way, coordinate, and track some particular tasks in project.

Gantt Charts …. Contd.

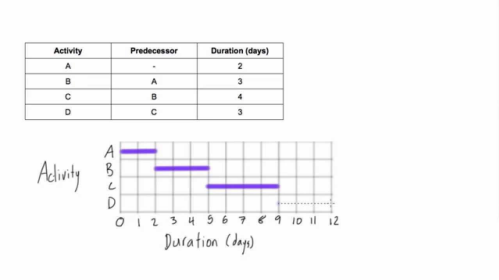
- **Gantt chart represents following things :**
  - ✔ All the tasks are listed at leftmost column.
  - ✔ The horizontal bars indicate or represent required time by corresponding particular task.
  - ✔ When occurring of multiple horizontal bars takes place at same time on calendar, then that means concurrency can be applied for performing particular tasks.
  - ✔ The diamonds indicate milestones.



Gantt Chart

Gantt Charts …. Contd.

- The purpose of Gantt chart is to emphasize scope of individual tasks. Hence set of tasks is given as input to Gantt chart.
- Gantt chart is also known as timeline chart.
  - It can be developed for entire project or it can be developed for individual functions.
  - In most of projects, after generation of timeline chart, project tables are prepared.
  - In project tables, all tasks are listed in proper manner along with start date and end date and information related to it.



Gantt Charts …. Contd.

Advantages :

- Simplify Project
  - Gantt charts are generally used for simplifying complex projects.
- Establish Schedule
  - It simply establishes initial project schedule in which it mentions who is going to do what, when, and how much time it will take to complete it.
- Provide Efficiency
  - It brings efficiency in planning and allows team to better coordinate project activities.
- Emphasize on scope
  - It helps in emphasizing i.e., gives importance to scope of individual tasks.

Gantt Charts …. Contd.

- Ease at understanding
  - It makes it for easy stakeholders to understand timeline and brings clarity of dates.
- Visualize project
  - It helps in clearly visualizing project management, project tasks involved.
- Organize thoughts and Highly visible
  - It organizes your thoughts and can be highly visible so that everyone in enterprises can have basic level of understanding and have knowledge about what is happening in project even if they are not involved in working.
- Make Practical and Realistic planning
  - It makes the project planning practical and realistic as realistic planning generally helps to avoid any kind of delays and losses of many that can arise.

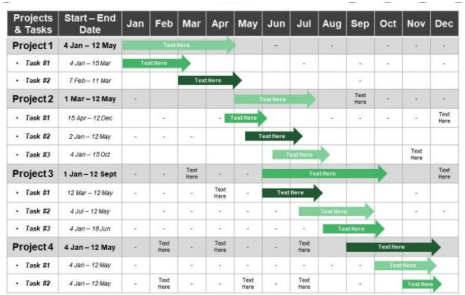
Gantt Charts …. Contd.

Disadvantages :

- Sometimes, using Gantt chart makes project more complex.
- The size of bar chart dost not necessarily indicate amount of work done in project.
- Gantt charts and projects are needed to be updated on regular basis.
- It is not possible or difficult to view this chart on one sheet of paper. The software products that produce Gantt chart needed to be viewed on computer screen so that whole project can be seen easily.

Project Timetable

Name of the task/ activity	Scheduled Start Date	Actual Start Date	Scheduled End Date	Actual End Date	Effort/ Task Assigned
Requirement Analysis	21-07-2021	25-07-2021	28-07-2021	03-08-2021	Athrav, Saili
Architectural Design	30-07-2021	04-08-2021	04-08-2021	06-08-2021	Hamza, Muaaz
Procedural Design	06-08-2021	16-08-2021	07-08-2021	16-08-2021	Omkar, Krutika, Kunal
Coding	18-08-2021	25-08-2021	18-08-2021	27-08-2021	Pratiksha, Janvi, Aditya
Testing	27-08-2021	03-09-2021	29-08-2021	07-09-2021	Harsh, Omkar, Asmita



Earned Value Analysis (EVA)

- During the project tracking, each developer gives his/ her project or task progress status to manager. But this assessment of information is subjective.
- There should be some quantitative technique for assessing progress of the project.
- Earned Value Analysis (EVA) is one of the key tools and techniques used in Project Management, to have an understanding of how the project is progressing.
- EVA implies gauging the progress based on earnings or money. Both, schedule and cost are calculated on the basis of EVA.

EVA …. Contd.

Features of EVA:-

- ✔ Earned Value Analysis is an objective method to measure project performance in terms of scope, time and cost.
- ✔ EVA metrics are used to measure project health and project performance.
- ✔ Earned Value Analysis is a quantitative technique for assessing progress as the software project team moves through the work tasks, allocated to the Project Schedule.
- ✔ EVA provides a common value scale for every project task.
- ✔ Total hours to complete the project are estimated and every task is given an Earned Value, based on its estimated (%) of the total.
- ✔ Earned Value is a measure of ‘Progress’ to assess ‘Percentage of Completeness’.

EVA …. Contd.

Need for EVA:-

- EVA provides different measures of progress for different types of tasks.
  - It is the single way for measuring everything in a project.
- Provides an ‘Early Warning’ signal for prompt corrective action. The types of signals can be the following:

a) Bad news does not age well –
   - Holding on to the bad news does not help. The project manager needs to take an immediate action.

b) Still time to recover –
   - In case, the project is not going as per schedule and may get delayed, the situation is needed to be taken care of by finding out the reasons that are causing delay and taking the required corrective action.

EVA …. Contd.

c) Timely request for additional funds –
   - While there is time to recover, the need for additional resources or funds can be escalated with an early warning.

- It allows ‘rolling up’ the progress of many tasks into an overall project status.
- It provides with a uniform unit of measure (dollars or work-hours) for the progress.

Key Elements of EVA:-

II. Planned Value (PV) –

- The approved cost baseline for the work package.
- It was earlier known as Budgeted Cost of Work Scheduled (BCWS).
- PV = Percent Complete (planned) x Task Budget or
  Planned Value = (Planned % Complete) X (BAC)
  where, BAC is Budget at Completion

EVA …. Contd.

II. Earned Value (EV) –

- The budgeted value of the completed work packages.
- It used to be known as Budgeted Cost of Work Performance at a specified point (BCWP).
- EV = Percent complete (actual) x Task Budget.

III. Actual Cost (AC) –

- The actual cost incurred during the execution of work packages up to a specified point in time.
- It was previously called Actual Cost of Work Performed (ACWP).
- AC = Actual Cost of the Task

Earned Value Calculations

- With the terms PV, EV, and AC defined, along with how to determine progress, some key calculations can easily be done, which provide important information on how the project is doing.

- The formulas for earned value calculations are:

a. Cost Variance (CV) : CV = EV – AC

b. Cost Performance Index (CPI) : CPI = EV/AC

- ✔ CPI measures the value of work completed against the actual cost.
- ✔ A CPI value <1.0 indicates costs were higher than budgeted. CPI >1.0 indicates costs were less than budgeted.

Earned Value Calculations ….

c. Schedule Performance Index (SPI): SPI = EV/PV

- ✔ SPI measures progress achieved against progress planned.
- ✔ A SPI value <1.0 indicates less work was completed than was planned.
- ✔ SPI >1.0 indicates more work was completed than was planned.

d. Estimated at Completion (EAC): EAC = (Total Project Budget)/CPI

- ✔ EAC is a forecast of how much the total project will cost.

e. Schedule Variance (SV) = EV– PV

Earned Value Example

Assume we are halfway through a year-long project that has a total budget of $100,000. The amount budgeted through this six-month mark is $55,000. The actual cost through this six-month mark is $45,000.

So,

- Planned Value (PV) = $55,000
- Actual Cost (AC) = $45,000
- Earned Value (EV) = ($100,000 * 0.5)
  = $50,000
- Schedule Variance (SV) = EV– PV
  = $50,000 - $55,000
  = - $5,000 (bad because <0)
- Schedule Performance Index (SPI) = EV/PV
  = $50,000/$55,000
  = 0.91 (bad because <1)

Earned Value Example ….

- Cost Variance (CV) = EV– AC
  = $50,000 - $45,000
  = $5,000 (good because >0)
- Cost Performance Index (CPI) = EV/AC
  = $50,000/$45,000
  = 1.11 (good because >1)
- Estimated at Completion (EAC) = (Total Project Budget)/CPI
  = $100,000/1.11
  = $90,000

Earned Value Example …..

- Because SV is negative and SPI is <1, the project is considered behind schedule.
  - We are 50% of the way through the project but have planned for 55% of the costs to be used.
  - There will have to be some catch-up in the second half of the project.
- Because CV is positive and CPI is >1, the project is considered to be under budget.
  - We are 50% of the way through the project, but our costs so far are only 45% of our budget.
  - If the project continues at this pace, then the total cost of the project (EAC) will be only $90,000, as opposed to our original budget of $100,000.