Practical 4

Scheduling a Software Project

Lab Work:

- Draw the Work breakdown Structure, Draw Dependency Diagram and find the critical path for your project.

Plan, Estimate, Schedule

What's the difference?

- Plan : Identify activities. No specific start and end dates.
- Estimate :Determine the size and duration of activities.
- Schedule: Add specific start and end dates, relationships, and resources.

Project Planning: A 12 Step Program

1) Set goal and scope
2) Select lifecycle
3) Set org./team form
4) Start team selection
5) Determine risks
6) Create WBS
7) Identify tasks
8) Estimate size
9) Estimate effort
10) Identify task dependencies
11) Assign resources
12) Schedule work

How To Schedule

- 1. Identify "what" needs to be done
  - Work Breakdown Structure (WBS)
- 2. Identify "how much" (the size)
  - Size estimation techniques
- 3. Identify the dependency between tasks
  - Dependency graph, network diagram
- 4. Estimate total duration of the work to be done
  - The actual schedule

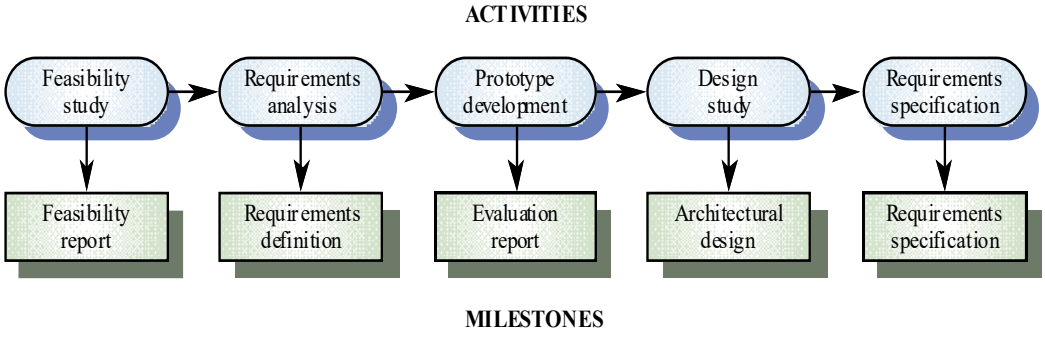
WBS & Estimation

"How long will your project take?"
- Not an easy answer
- Especially not on a real project

- How to manage that issue?

The main steps to scheduling the project are:

1. Identify all task/activities of a project and the milestones to be produced.
2. Estimate the time span of each of these activities.
3. Identify and establish dependencies between these activities.
4. Represent the dependencies among the activities using a dependency diagram say activity graph.
5. Estimate the resources a project team should use to complete the project.
6. Identify the critical path(s). (a critical path is a chain of activities that determines the duration of the project)
7. Identify the slack time of non-critical paths.
8. Control and manage the changes to the project schedule.



What is the Problem?

Your boss: "How long will this take?"

You: "Between 1 and 6 months."

"As long as I can do it within 6 months, I keep my promise."

"With hard work, he can do it in 1 month."

What is the real Problem?

Your boss: "How long will this take?"

You: "Between 1 and 6 months."

"I have not he slightest clue, if this is possible at all."

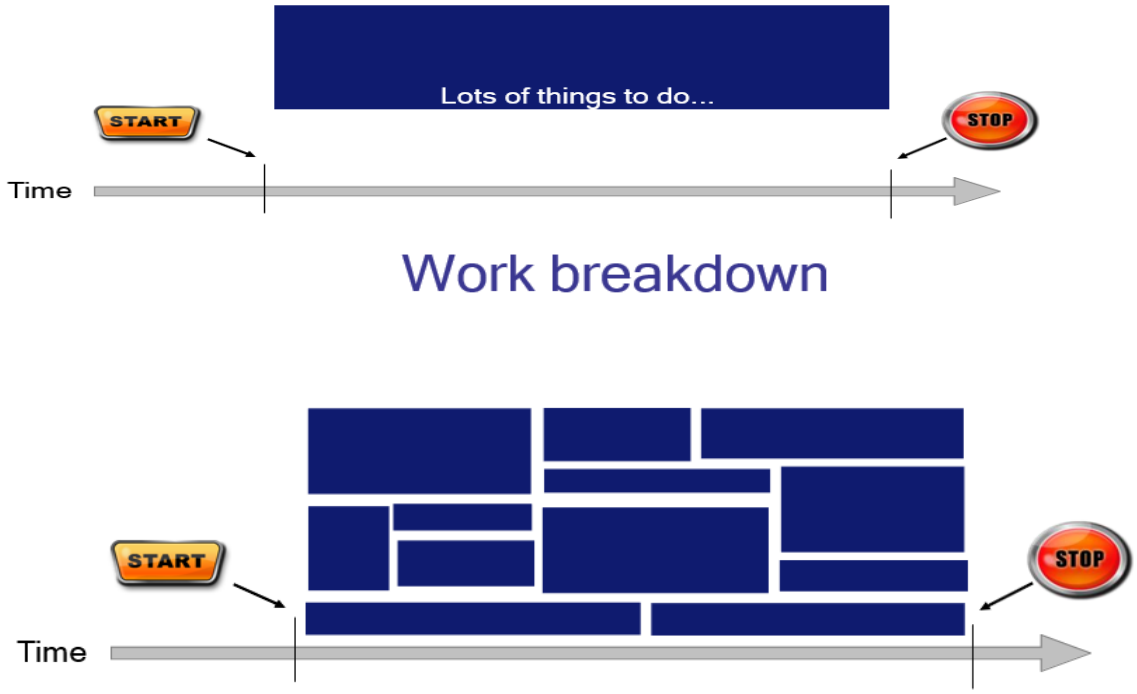
"Even if it is possible, I don't know, how long it will take."

Solution: Use divide and conquer

- To give a good answer you have to break the work down into activities for which you try to get timing estimates
- Only if with good estimates can you estimate the overall project duration

Work Breakdown Structure

A work breakdown structure (WBS) is a visual, hierarchical and deliverable-oriented deconstruction of a project. It is a helpful diagram for project managers because it allows them to break down their project scope and visualize all the tasks required to complete their projects.



Why Use a WBS In Project Management?

Making a WBS is the first step in developing a project schedule. It defines all the work that needs to be completed (and in what order) to achieve the project goals and objectives. By visualizing your project in this manner, you can understand your project scope, and allocate resources for all your project tasks.

A well-constructed work breakdown structure helps with important project management process groups and knowledge areas such as:

- Project Planning, Project Scheduling and Project Budgeting
- Risk Management, Resource Management, Task Management and Team Management

In addition, a WBS helps avoid common project management issues such as missed deadlines, scope creep and cost overrun, among others.

Types of WBS

There are two main types of WBS: deliverable-based, and phase-based. They depend on whether you want to divide your project in terms of time or scope.

Deliverable-Based Work Breakdown Structure

A deliverable-based WBS first breaks down the project into all the major areas of the project scope as control accounts, and then divides those into project deliverables and work packages.

Phase-Based Work Breakdown Structure

The phase-based WBS displays the final deliverable on top, with the WBS levels below showing the five phases of a project (initiation, planning, execution, control and closeout). Just as in the deliverable-based WBS, the project phases are divided into project deliverables and work packages.

Procedure:

A) How to Create a Work Breakdown Structure and Why You Should?

1. Include 100% of the work necessary to complete the goal.

2. Don't account for any amount of work twice.

3. Focus on outcomes, not actions.

4. A work package should take no less than 8 hours and no more than 80 hours of effort.

5. Include about three levels of detail.

6. Assign each work package to a specific team or individual.

B) What is a work breakdown structure?

A work breakdown structure starts with a large project or objective and breaks it down into smaller, more manageable pieces that you can reasonably evaluate and assign to teams. Rather than focusing on individual actions that must be taken to accomplish a project, a WBS generally focuses on deliverables or concrete, measurable milestones.

C) Why use a WBS in project management?

1. Estimate the cost of a project.

2. Establish dependencies.

3. Determine a project timeline and develop a schedule.

4. Write a statement of work (or SOW, one of your other acronyms).

5. Assign responsibilities and clarify roles.

6. Track the progress of a project.

7. Identify risk.

D) How to create a work breakdown structure?

1. Record the overarching objective you are trying to accomplish. This objective could be anything from developing a new software feature to building a missile.

2. Divide the overarching project into smaller and smaller pieces, but stop before you get to the point of listing out every action that must be taken. Remember to focus on concrete deliverables rather than actions.

3. Depending on the nature of your project, start dividing by project phases, specific large deliverables, or sub-tasks.

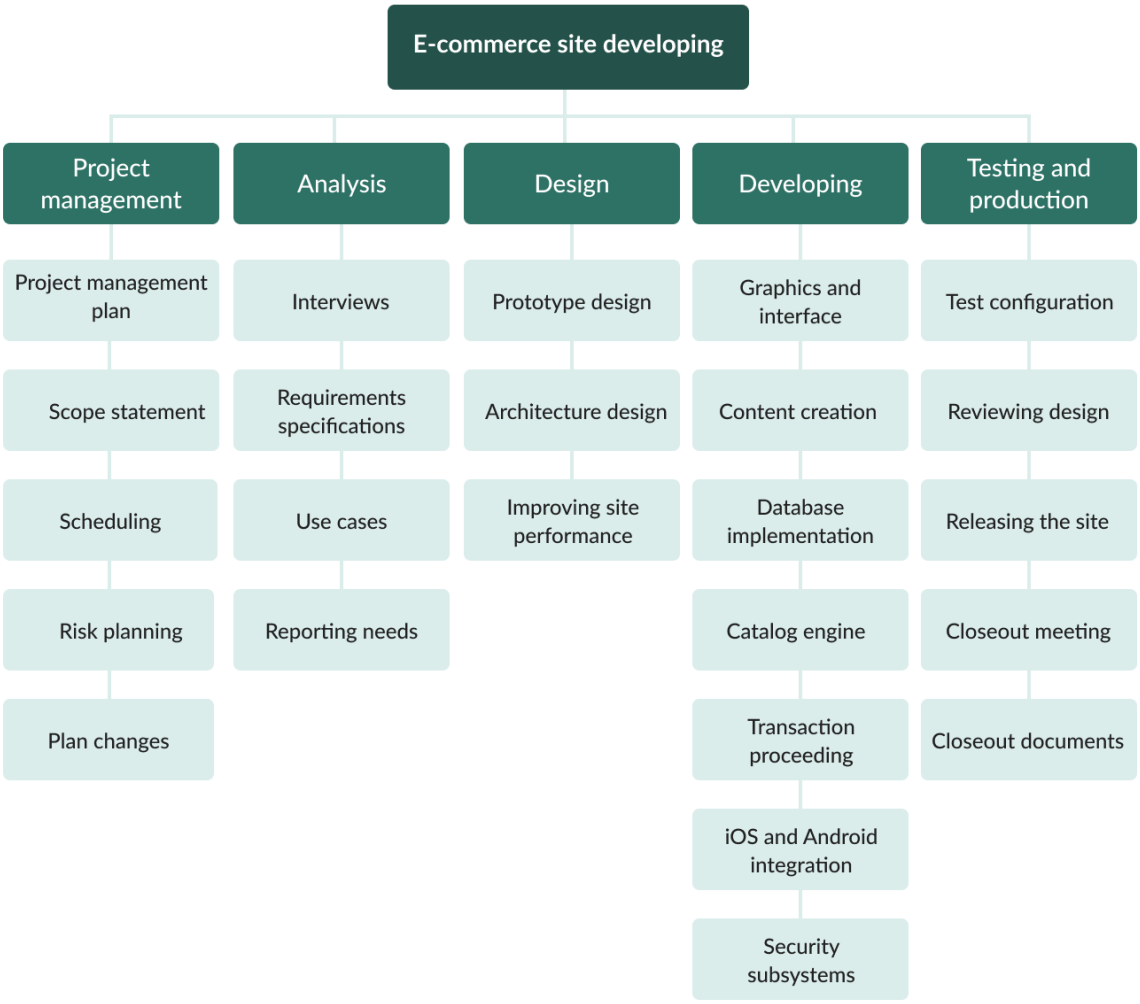
E) Tips for making a work breakdown structure

- **The 100% rule.** The work represented by your WBS must include 100% of the work necessary to complete the overarching goal without including any extraneous or unrelated work. Also, child tasks on any level must account for all of the work necessary to complete the parent task.
- **Mutually exclusive.** Do not include a sub-task twice or account for any amount of work twice. Doing so would violate the 100% rule and will result in miscalculations as you try to determine the resources necessary to complete a project.
- **Outcomes, not actions.** Remember to focus on deliverables and outcomes rather than actions. For example, if you were building a bike, a deliverable might be "the braking system" while actions would include "calibrate the brake pads."
- **The 8/80 rule.** There are several ways to decide when a work package is small enough without being too small. This rule is one of the most common suggestions—a work package should take no less than eight hours of effort, but no more than 80. Other rules suggest no more than ten days (which is the same as 80 hours if you work full time) or no more than a standard reporting period. In other words, if you report on your work every month, a work package should take no more than a month to complete. When in doubt, apply the "if it makes sense" rule and use your best judgment.
- **Three levels.** Generally speaking, a WBS should include about three levels of detail. Some branches of the WBS will be more subdivided than others, but if most branches have about three levels, the scope of your project and the level of detail in your WBS are about right.
- **Make assignments.** Every work package should be assigned to a specific team or individual. If you have made your WBS well, there will be no work overlap so responsibilities will be clear.

Examples of WBS

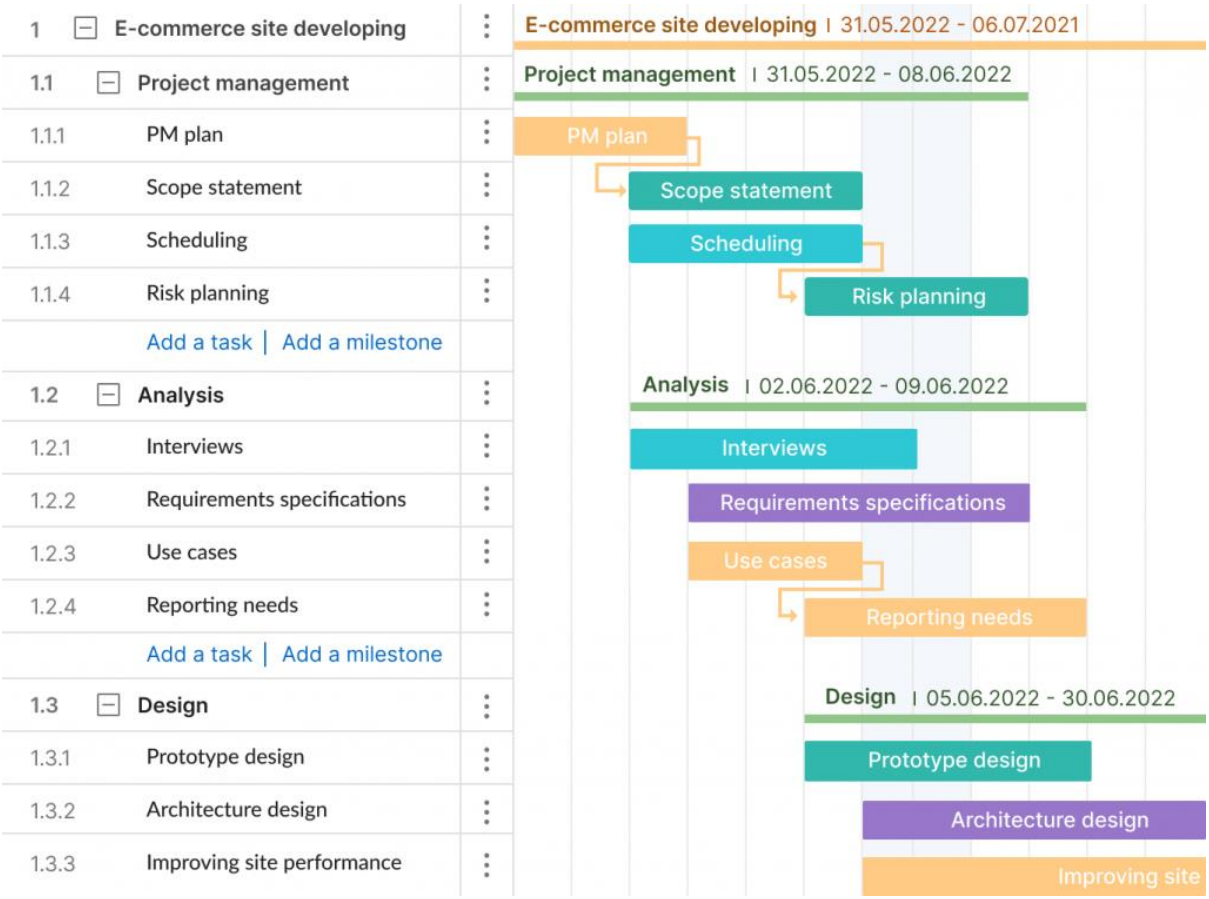
Work breakdown structure example related to creating a new e-commerce application

A new e-commerce site must be executed impeccably and clearly, since there are so many competitors on the market. Therefore the work breakdown structure software development example must also be clear, concise, and detailed.



As you see, this result-oriented tree work breakdown structure example of a software project focuses on the project management, analysis, designing, developing, and testing stages.

- **Project management**. The first stage of our work breakdown structure for software project example includes planning, defining scope, scheduling, risk management, and work with possible plan changes.
- **Analysis.** At this stage, project teams conduct required interviews, work on requirements specifications, and prepare use cases.
- **Design** is one of the most essential parts of our software development work breakdown structure example. Here, you should care about the prototype design, architecture design, and site performance improvements.
- **Developing.** This is typically one of the most active phases of software development, so you will need to thoroughly work on developing the new e-commerce site and care about all the details, meaning graphics and interface, content creation, database implementation, catalog engine, transaction processing, iOS and Android integration, security, and other important issues.
- **Testing and production** are what end the process. This is when test configuration, reviewing design, releasing the site, closeout meetings, and preparing closeout documents happen.



Schedule all the activities and sub-activities using the PERT/CPM charts

Procedure:

1. Introduction

Basically, CPM (Critical Path Method) and PERT (Programme Evaluation Review Technique) are project management techniques, which have been created out of the need of Western industrial and military establishments to plan, schedule and control complex projects.

Planning, Scheduling & Control

Planning, Scheduling (or organizing) and Control are considered to be basic Managerial functions, and CPM/PERT has been rightfully accorded due importance in the literature on Operations Research and Quantitative Analysis.

Far more than the technical benefits, it was found that PERT/CPM provided a focus around which managers could brain-storm and put their ideas together. It proved to be a great communication medium by which thinkers and planners at one level could communicate their ideas, their doubts and fears to another level. Most important, it became a useful tool for evaluating the performance of individuals and teams.

The Framework for PERT and CPM

Essentially, there are six steps which are common to both the techniques. The procedure is listed below:

1. Define the Project and all of its significant activities or tasks. The Project (made up of several tasks) should have only a single start activity and a single finish activity.

2. Develop the relationships among the activities. Decide which activities must precede and which must follow others.

3. Draw the "Network" connecting all the activities. Each Activity should have unique event numbers. Dummy arrows are used where required to avoid giving the same numbering to two activities.

4. Assign time and/or cost estimates to each activity

5. Compute the longest time path through the network. This is called the critical path.

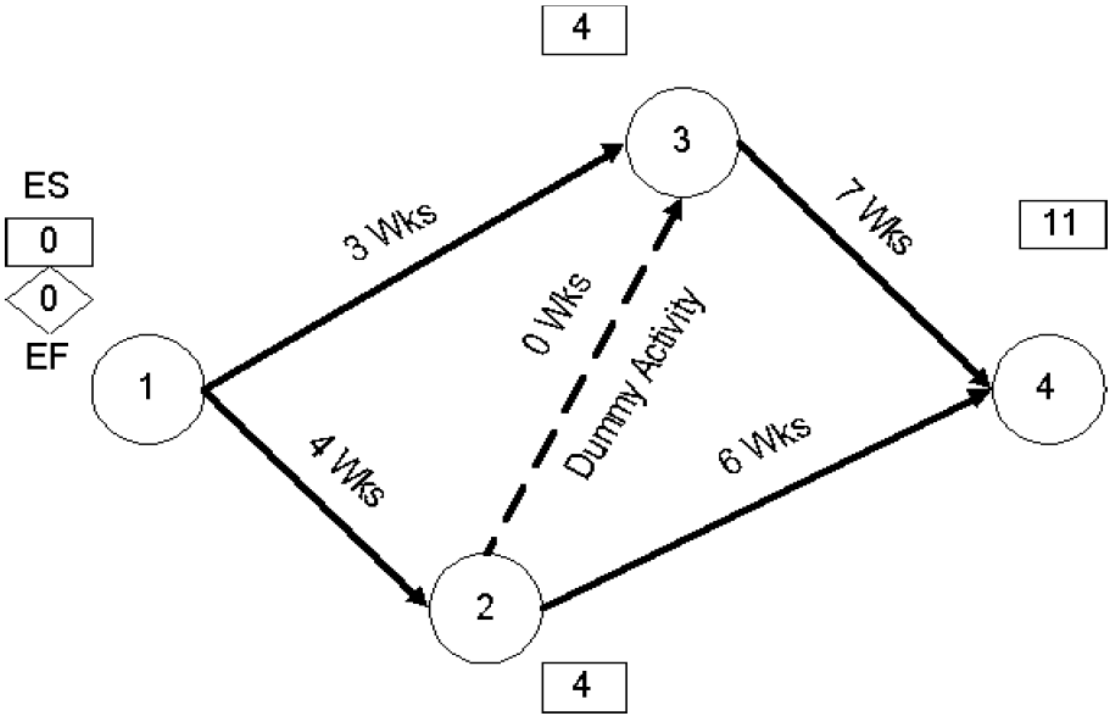
6. Use the Network to help plan, schedule, and monitor and control the project.

The Key Concept used by CPM/PERT is that a small set of activities, which make up the longest path through the activity network control the entire project. If these "critical" activities could be identified and assigned to responsible persons, management resources could be optimally used by concentrating on the few activities which determine the fate of the entire project.

Non-critical activities can be preplanned, rescheduled and resources for them can be reallocated flexibly, without affecting the whole project.

Drawing the CPM/PERT Network

Each activity (or sub-project) in a PERT/CPM Network is represented by an arrow symbol. Each activity is preceded and succeeded by an event, represented as a circle and numbered.

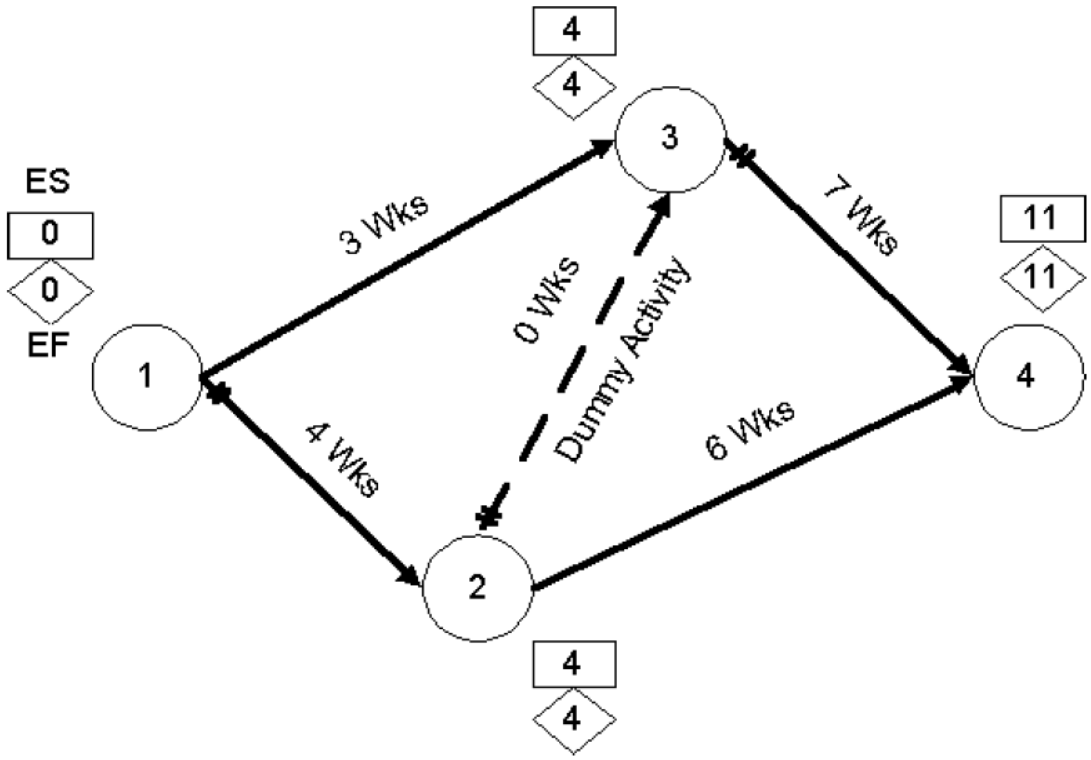


At Event 3, we have to evaluate two predecessor activities – Activity 1-3 and Activity 2- 3, both of which are predecessor activities. Activity 1-3 gives us an Earliest Start of 3 weeks at Event 3. However, Activity 2-3 also has to be completed before Event 3 can begin. Along this route, the Earliest Start would be 4+0=4. The rule is to take the longer (bigger) of the two Earliest Starts. So the Earliest Start at event 3 is 4.

Similarly, at Event 4, we find we have to evaluate two predecessor activities – Activity 2-4 and Activity 3-4. Along Activity 2-4, the Earliest Start at Event 4 would be 10 wks, but along Activity 3-4, the Earliest Start at Event 4 would be 11 wks. Since 11 wks is larger than 10 wks, we select it as the Earliest Start at Event 4.We have now found the longest path through the Network. It will take 11 weeks along activities 1-2, 2-3 and 3-4. This is the Critical Path.

The Backward Pass – Latest Finish Time Rule

To make the Backward Pass, we begin at the sink or the final event and work backwards to the first event.



Tabulation & Analysis of Activities

We are now ready to tabulate the various events and calculate the Earliest and Latest Start and Finish times. We are also now ready to compute the SLACK or TOTAL FLOAT, which is defined as the difference between the Latest Start and Earliest Start.

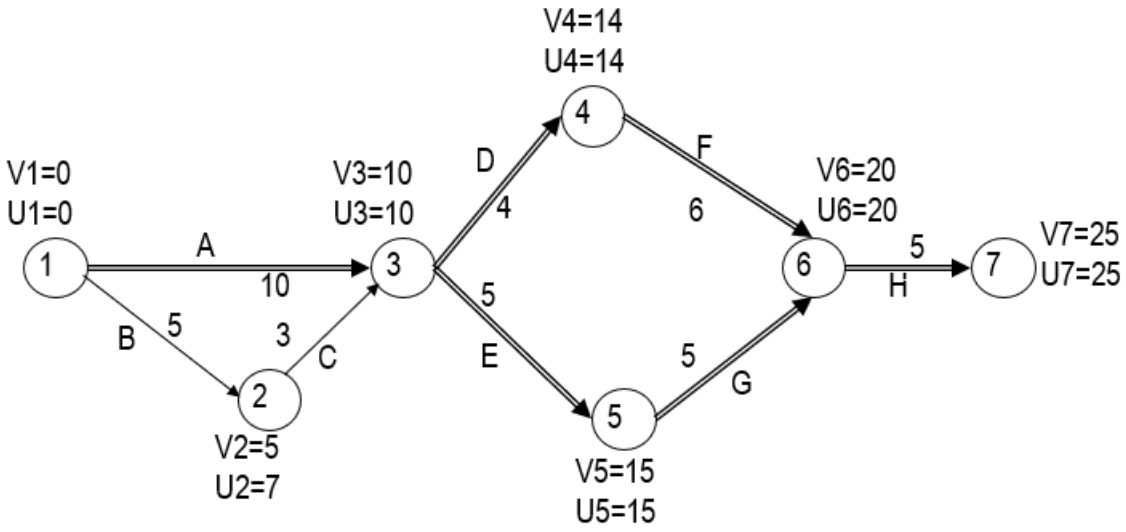
Event	Duration(Weeks)	Earliest Start	Earliest Finish	Latest Start	Latest Finish	Total Float
1-2	4	0	4	0	4	0
2-3	0	4	4	4	4	0
3-4	7	4	11	4	11	0
1-3	3	0	3	1	4	1
2-4	6	4	10	5	11	1

- The Earliest Start is the value in the rectangle near the tail of each activity
- The Earliest Finish is = Earliest Start + Duration
- The Latest Finish is the value in the diamond at the head of each activity
- The Latest Start is = Latest Finish – Duration

There are two important types of Float or Slack. These are Total Float and Free Float.
TOTAL FLOAT is the spare time available when all preceding activities occur at the earliest possible times and all succeeding activities occur at the latest possible times. Total Float = Latest Start – Earliest Start
Activities with zero Total float are on the Critical Path

=======

Job	Job Description	Predecessor	Normal Time (days)
A	Analysis-1 & SRS-1	-	10
B	Analysis-2	-	5
C	SRS-2	B	3
D	GUI Design	A, C	4
E	DB Design	A, C	5
F	GUI Coding & Unit Testing	D	6
G	DB Coding & Unit Testing	E	5
H	Integration & System Testing	F, G	5



Job	Expected Duration	Earliest Start	Latest Start	Earliest Finish	Latest Finish	Slack Time (Max.delay)	Remarks
A	10	0	0	10	10	0	Critical
B	5	0	2	5	7	2	Non-critical
C	3	5	7	8	10	3	Non-critical
D	4	10	10	14	14	0	Critical
E	5	10	10	15	15	0	Critical
F	6	14	14	20	20	0	Critical
G	5	15	15	20	20	0	Critical
H	5	20	20	25	25	0	Critical

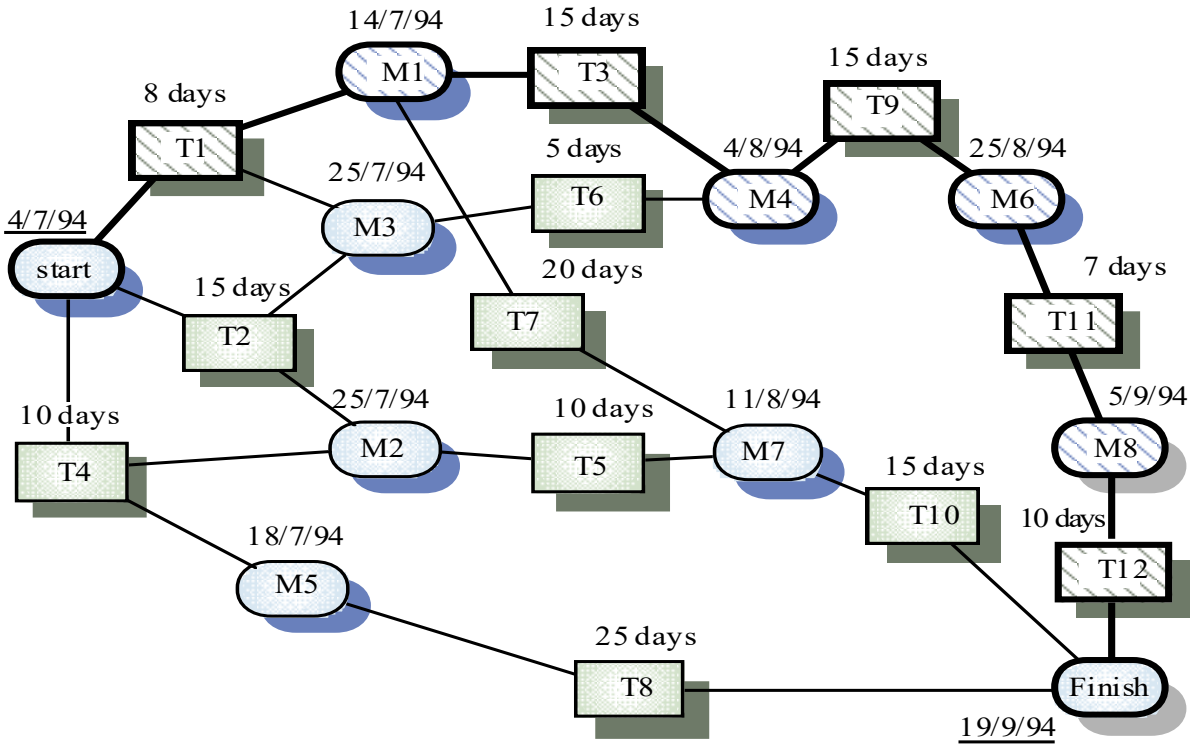
If all the jobs are done at their normal times, the project duration (length of the longest path) is 25 days.

Critical path A → D → F → H = 25 days
or A → E → G → H = 25 days

Task durations and dependencies

Task	Duration (days)	Dependencies
T1	8	
T2	15	
T3	15	T1
T4	10	
T5	10	T2, T4
T6	5	T1, T2
T7	20	T1
T8	25	T4
T9	15	T3, T6
T10	15	T5, T7
T11	7	T9
T12	10	T11

Activity Network


14/7/94
15 days
M1
T3
15 days
T9
8 days
T1
5 days
4/8/94
25/8/94
25/7/94
T6
M4
M6
4/7/94
M3
start
20 days
7 days
15 days
T7
T11
T2
5/9/94
25/7/94
11/8/94
10 days
10 days
M8
M2
M7
T4
T5
15 days
T10
10 days
18/7/94
T12
M5
25 days
T8
Finish
19/9/94