Project Scheduling

Why care about project management?

- 10% of projects successful between 1998 and 2004

Software project scheduling is an action that distributes estimated effort across theplanned project duration by allocating the effort to specific software engineering tasks.

Basic Principles

Compartmentalization

Interdependency.

Time allocation

Effort validation

Defined responsibilities

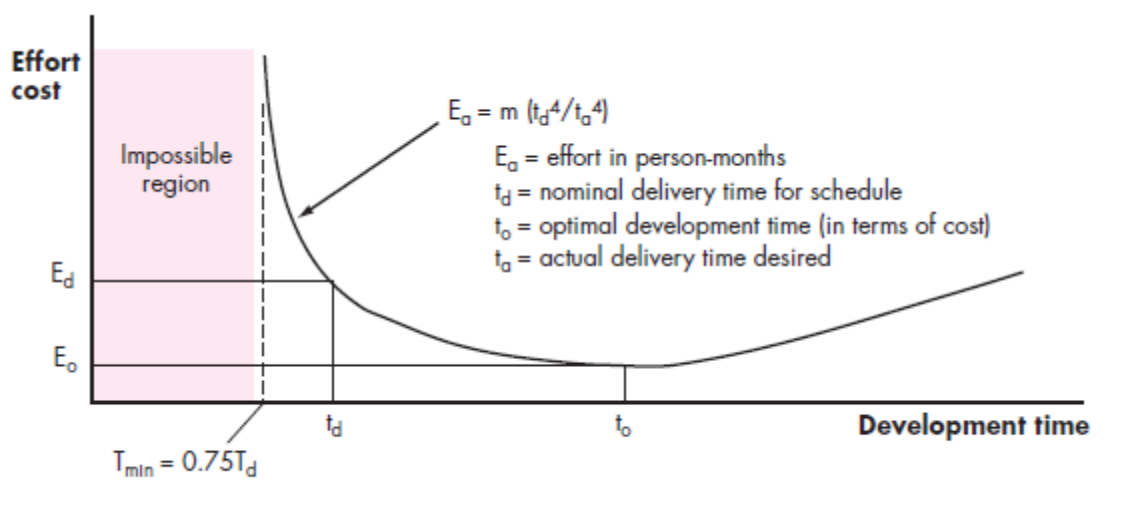
Defined outcomes

Defined milestones

A milestone is accomplished when one or more work products
has been reviewed for quality (Chapter 15) and has been approved.

The Relationship Between People and Effort

The relationship between effort and delivery time



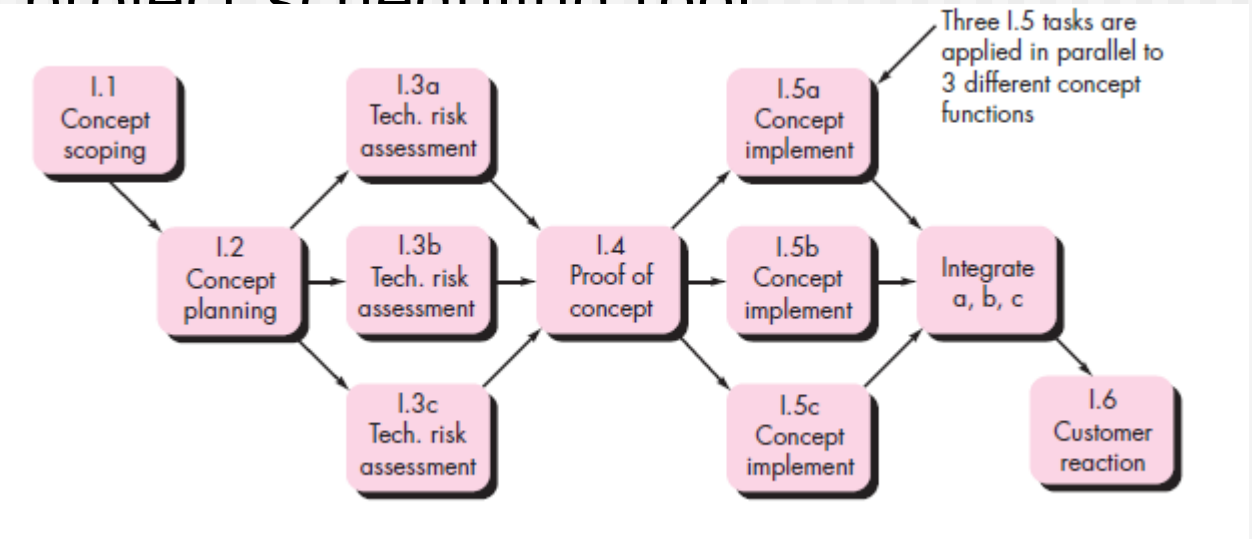
DEFINING A TASK SET FOR THE SOFTWARE PROJECT

most software organizations encounter the following projects:

1. *Concept development projects* that are initiated to explore some new business concept or application of some new technology.
2. *New application development* projects that are undertaken as a consequenceof a specific customer request.
3. *Application enhancement* projects that occur when existing software undergoes major modifications to function, performance, or interfaces that are observable by the end user.
4. *Application maintenance projects* that correct, adapt, or extend existing softwarein ways that may not be immediately obvious to the end user.
5. *Reengineering projects* that are undertaken withthe in tent of rebuilding an existing (legacy) system in whole or in part.

Activity network

A *task network*, also called an *activity network*, is a graphic representation of the task flow for a project. It is sometimes used as the mechanism through which task sequence and dependencies are input to an automated project scheduling tool

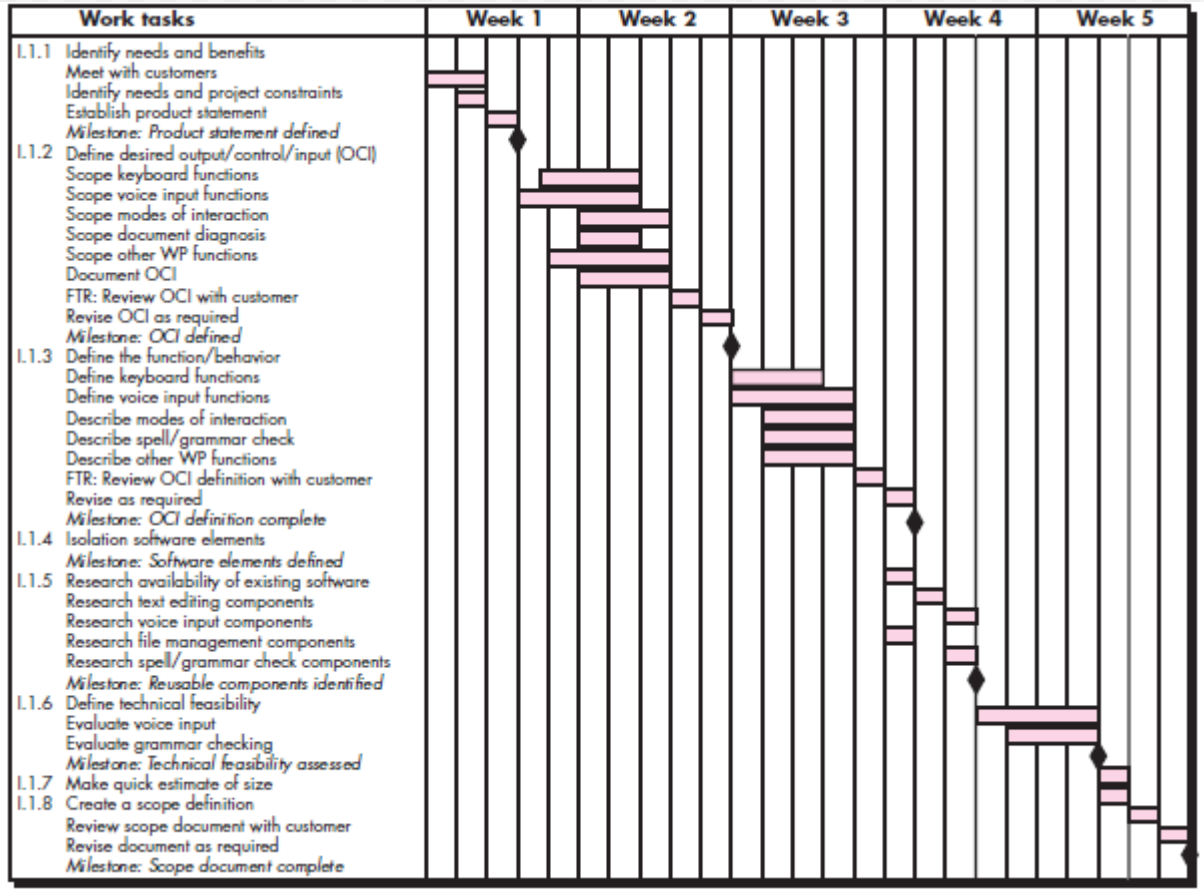


SCHEDULING

Program evaluation and review technique (PERT) and the *critical path method* (CPM)are project scheduling methods that can be applied to software development.

Both techniques are driven by information already developed in earlier project planning activities: estimates of effort, a decomposition of the product function, the selection of the appropriate process model and task set, and decomposition of the tasks that are
selected.

Time-Line Charts/ Gantt chart



Work tasks	Planned start	Actual start	Planned complete	Actual complete	Assigned person	Effort allocated	Notes
I.1.1 Identify needs and benefits							Scoping will require more effort/time
Meet with customers	wk1, d1	wk1, d1	wk1, d2	wk1, d2	BLS	2 p-d	
Identify needs and project constraints	wk1, d2	wk1, d2	wk1, d2	wk1, d2	JPP	1 p-d	
Establish product statement	wk1, d3	wk1, d3	wk1, d3	wk1, d3	BLS/JPP	1 p-d	
Milestone: Product statement defined	wk1, d3	wk1, d3	wk1, d3	wk1, d3			
I.1.2 Define desired output/control/input (OCI)							
Scope keyboard functions	wk1, d4	wk1, d4	wk2, d2		BLS	1.5 p-d	
Scope voice input functions	wk1, d3	wk1, d3	wk2, d2		JPP	2 p-d	
Scope modes of interaction	wk2, d1		wk2, d3		MLL	1 p-d	
Scope document diagnostics	wk2, d1		wk2, d2		BLS	1.5 p-d	
Scope other WP functions	wk1, d4	wk1, d4	wk2, d3		JPP	2 p-d	
Document OCI	wk2, d1		wk2, d3		MLL	3 p-d	
FTR: Review OCI with customer	wk2, d3		wk2, d3		all	3 p-d	
Revise OCI as required	wk2, d4		wk2, d4		all	3 p-d	
Milestone: OCI defined	wk2, d5		wk2, d5				
I.1.3 Define the function/behavior							

Scheduling

- One of the most important things you can do is schedule.
- Also one of the first things you should do!
- Tools help
  - Microsoft Project
  - OpenProj.org
  - OpenWorkbench.org

Planning

- The bad news: time flies
- The good news: you're the pilot!

- You must begin planning immediately
  - Given limited information
  - Plan anyway and then revise

Coming up: Creating a plan: Things to know

Creating a plan: Things to know

- Scope
  - **Context**. How does the software to be built fit into a larger system, product, or business context and what constraints are imposed as a result of the context?
  - **Information objectives**. What customer-visible data objects (Chapter 8) are produced as output from the software? What data objects are required for input?
  - **Function and performance**. What function does the software perform to transform input data into output? Are any special performance characteristics to be addressed?
- Software project scope must be unambiguous and understandable at the management and technical levels.

Creating a plan: Things to do

- Problem Decomposition: Sometimes called *partitioning* or *problem elaboration*
- Once scope is defined …
  - It is decomposed into constituent functions
  - It is decomposed into user-visible data objects

  or
  - It is decomposed into a set of problem classes
- Decomposition process continues until all functions or problem classes have been defined (this won't be far at the beginning of your project)

Schedule

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
1	Don't Care	Don't Care	Don't Care	Don't Care About You	I'm in Love	Wait
Always Comes Too Late	You can fall apart	Break My Heart	Break My Heart	Doesn't Even Start	I'm in Love	Wait
Always Comes Too Late	Don't Care	Heart Attack	Heart Attack	Never Looking Back	I'm in Love	Wait
Always Comes Too Late	Hold Your Head	Stay in Bed	Stay in Bed	Watch the Walls Instead	I'm in Love	Wait
Always Comes Too Late	30	31				

- List of tasks
  - With dates
  - With assigned resources (people)
  - With durations
  - With predecessors and successors
- How do you get buy-in from the team for a schedule?
  - History
  - Increments

Coming up: Schedule Terms

Schedule Terms

- Critical path
  - Sequence of tasks that form the longest path to completion of the project. Any delay on any of these will make the overall completion date move.
- Slack
  - Amount of time a task can be delayed without affecting the overall completion date.
    - Start slack - amount before task needs to start
    - Finish slack - amount before task needs to finish
- Milestone - An import date in the schedule
- Dependencies - relationship between tasks

Schedule Dependencies

- **FS - Finish to start (most common)**
  - **A FS B. B doesn’t start until A is finished**
  - **Build wall FS Paint wall**
- FF - Finish to finish
  - A FF B. B doesn’t finish before A is finished
  - Write final chapter FF Complete Index
- SS - Start to start
  - A SS B. B doesn’t start until A has started
  - Project funded SS project management activities begin
- SF - Start to finish
  - A SF B. B doesn’t finish before A has started
  - Once A starts, B is allowed to finish
  - B=Baby sit a child, A=parent comes home

Resource Leveling

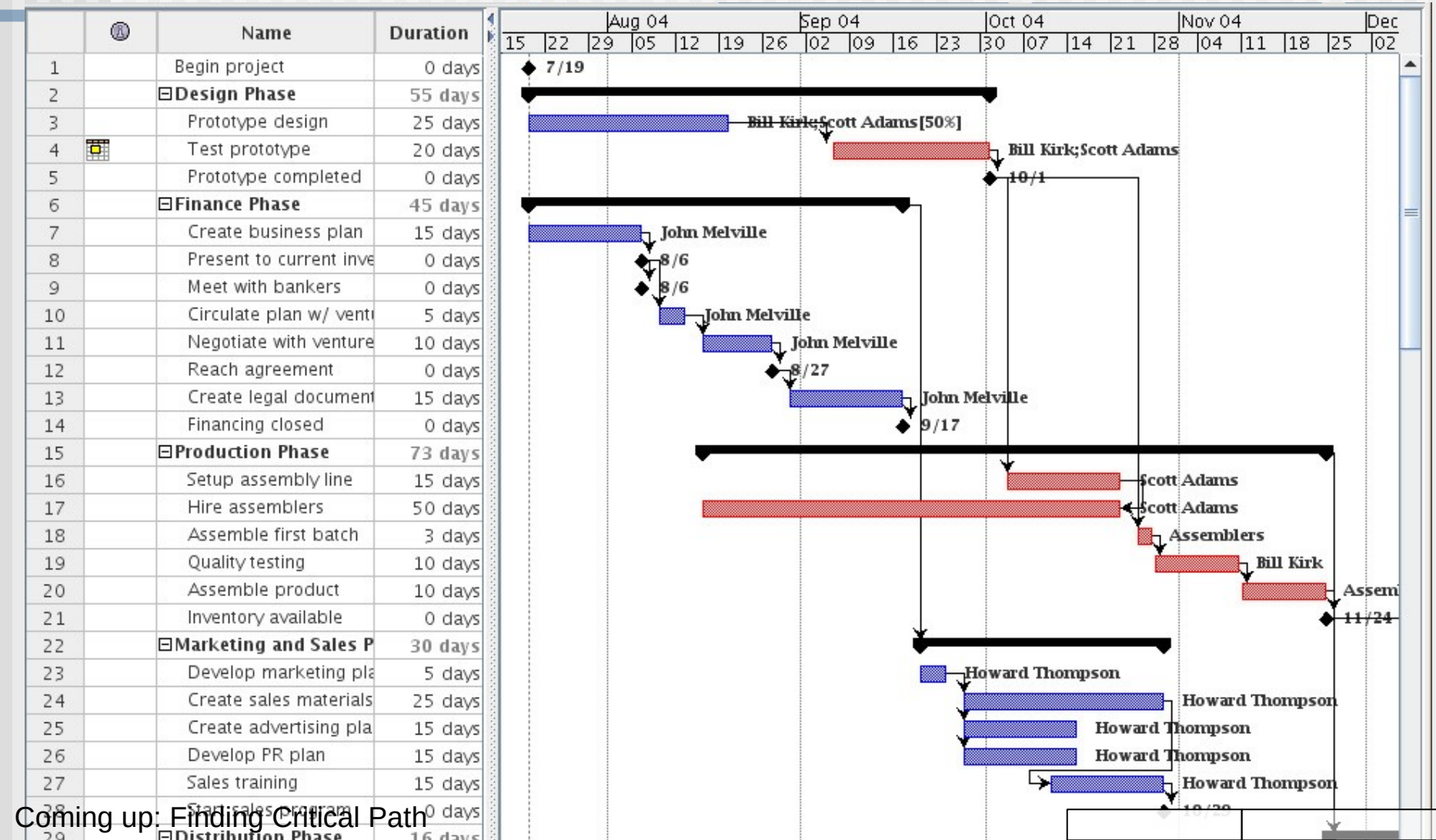
- A process to examine a project for an unbalanced use of people and to resolve over-allocations or conflicts
- Happens when multiple tasks are scheduled at the same time for the same person
- Solution:
  - Make tasks sequential by introducing “fake” dependencies
  - Split resource usage among tasks (50% on task 1, 50% on task 2)

Coming up: Auto Resource Leveling

Auto Resource Leveling

- Some tools (not Open Project) provide auto resource leveling
- Tool automatically ensures no person works over 100% of the time (automatically makes tasks sequential)
- Advantageous because this does not introduce “fake” dependencies

Gantt Chart

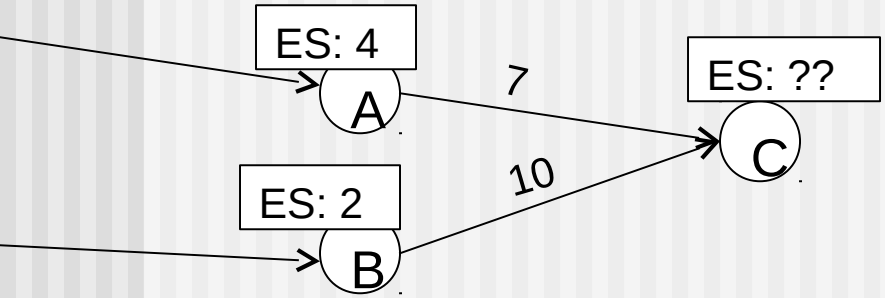


		Name	Duration
1		Begin project	0 days
2		⊟**Design Phase**	55 days
3		Prototype design	25 days
4		Test prototype	20 days
5		Prototype completed	0 days
6		⊟**Finance Phase**	45 days
7		Create business plan	15 days
8		Present to current inve	0 days
9		Meet with bankers	0 days
10		Circulate plan w/ vent	5 days
11		Negotiate with venture	10 days
12		Reach agreement	0 days
13		Create legal document	15 days
14		Financing closed	0 days
15		⊟**Production Phase**	73 days
16		Setup assembly line	15 days
17		Hire assemblers	50 days
18		Assemble first batch	3 days
19		Quality testing	10 days
20		Assemble product	10 days
21		Inventory available	0 days
22		⊟**Marketing and Sales P**	30 days
23		Develop marketing pla	5 days
24		Create sales materials	25 days
25		Create advertising pla	15 days
26		Develop PR plan	15 days
27		Sales training	15 days
28		Start sales program	0 days
29		⊟**Distribution Phase**	16 days

Coming up: Finding Critical Path

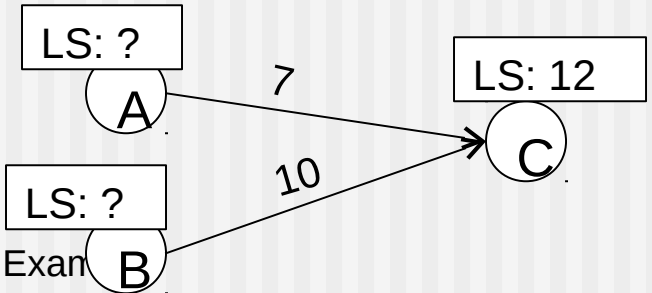
Finding Critical Path

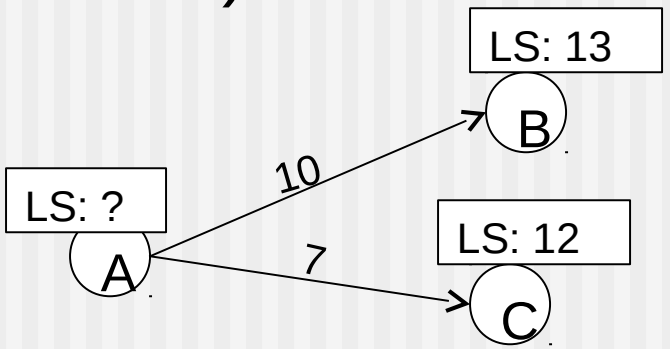
- Draw a network diagram of the activities
- Determine the Early Start (ES) of each node. Work from beginning node (ES=0) to final node
- ES - earliest time the activity can start
- ES = Max(ESprevNode + DurationPrevNode)



Finding Critical Path

- Determine the Late Start (LS) of each node. Work from the final node to the beginning node.
  - The latest time the activity can start without changing the end date of the project
  - LS = MIN(LSnext - DurationNode)
  - For the last node LS = ES





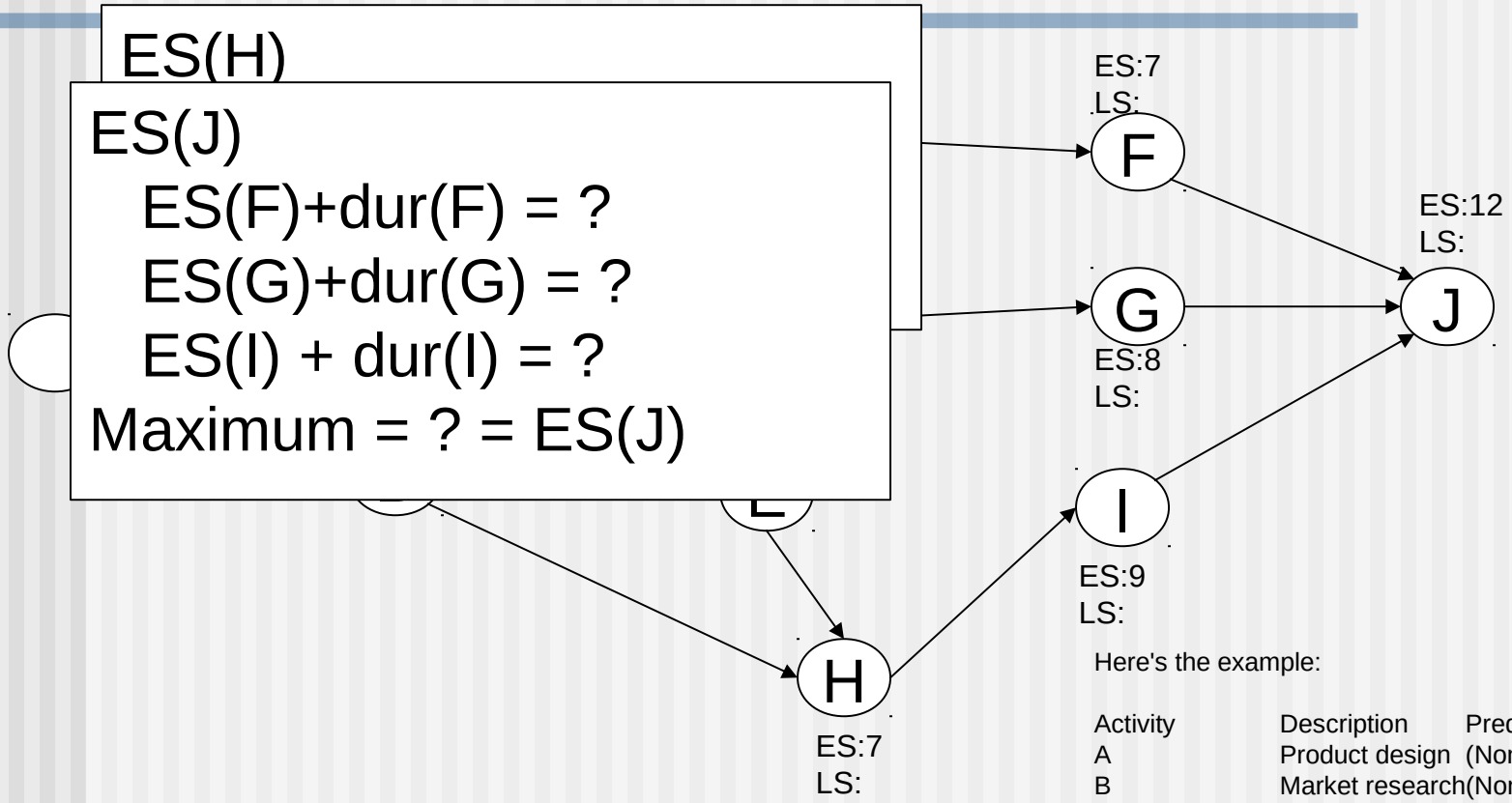
Example

Here's the example:

Activity	Description	Predecessor	Duration
A	Product design	(None)	5 months
B	Market research	(None)	1
C	Production analysis	A	2
D	Product model	A	3
E	Sales brochure	A	2
F	Cost analysis	C	3
G	Product testing	D	4
H	Sales training	B, E	2
I	Pricing	H	1
J	Project report	F, G, I	1

Coming up: Example Node Network

Example Node Network



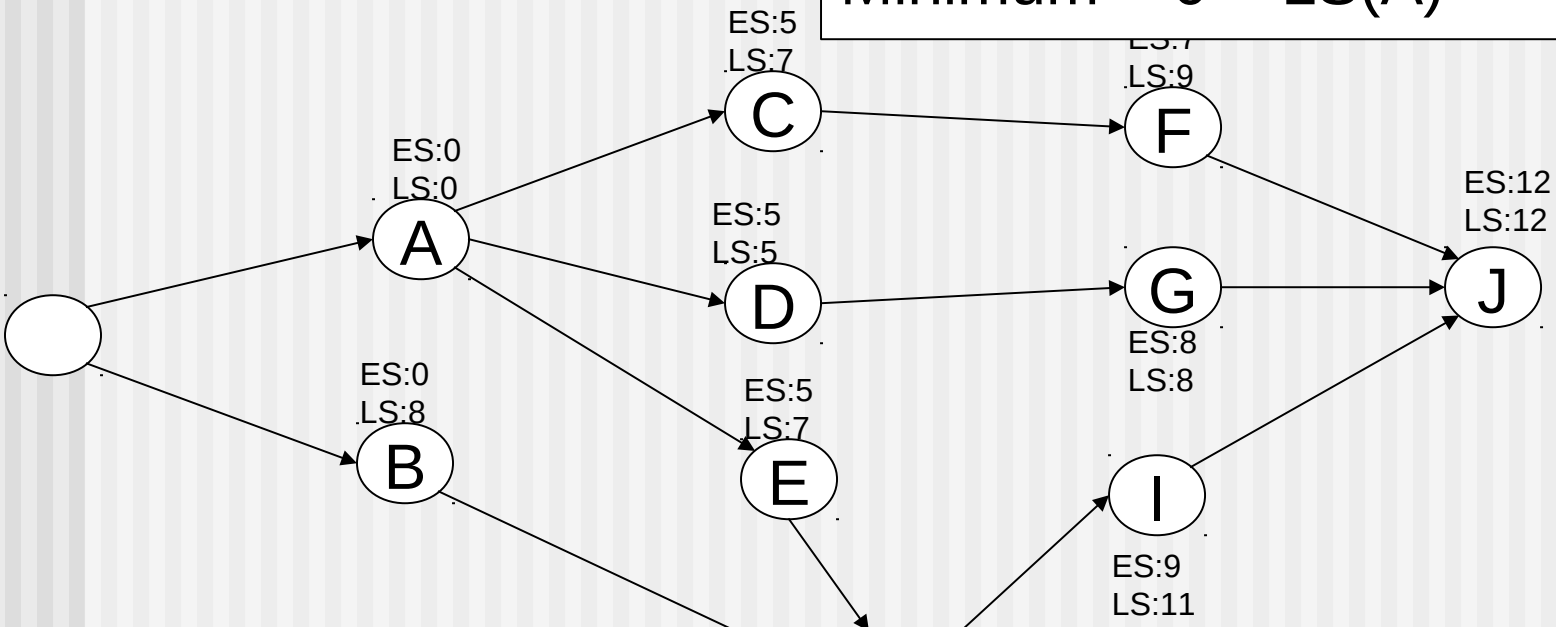
Here's the example:

Activity	Description	Predecessor	Duration
A	Product design	(None)	5 months
B	Market research	(None)	1
C	Production	A	2
D	Product model	A	3
E	Sales brochure	A	2
F	Cost analysis	C	3
G	Product testing	D	4
H	Sales training	B, E	2
I	Pricing	H	1
J	Project report	F, G, I	1

Coming up: Example Node Network

Example Noc

LS(A) =
LS(C) – dur(A) = 7 – 5 = 2
LS(D) – dur(A) = 5 – 5 = 0
LS(E) – dur (A) = 7 – 5 = 2
Minimum = 0 = LS(A)



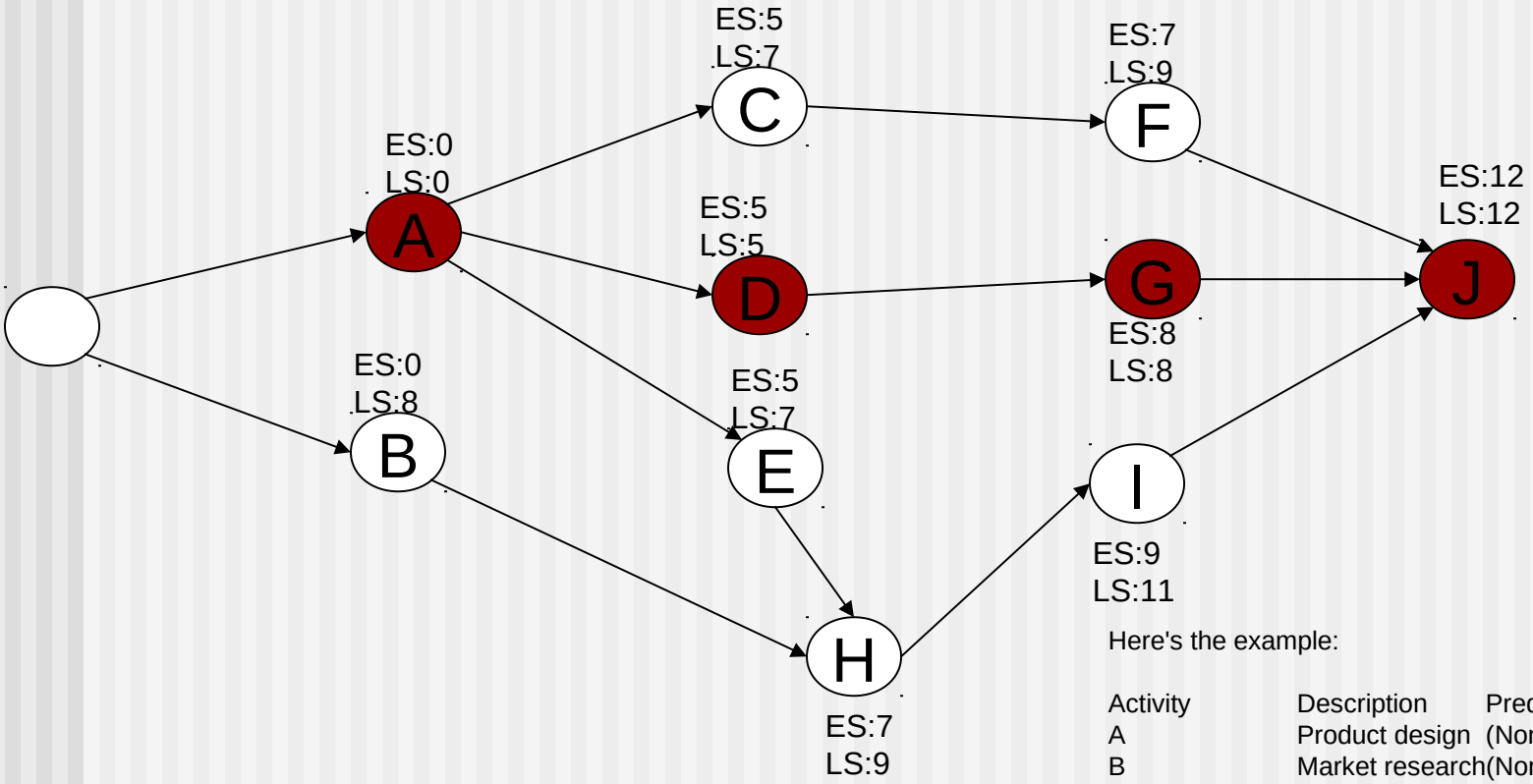
LS(F)
LS(J)-dur(F) = 12 – 3
= 9

Here's the example:

Activity	Description	Predecessor	Duration
A	Product design	(None)	5 months
B	Market research	(None)	1
C	Production	A	2
D	Product model	A	3
E	Sales brochure	A	2
F	Cost analysis	C	3
G	Product testing	D	4
H	Sales training	B, E	2
I	Pricing	H	1
J	Project report	F, G, I	1

Coming up: Example Node Network

Example Node Network



Here's the example:

Activity	Description	Predecessor	Duration
A	Product design	(None)	5 months
B	Market research	(None)	1
C	Production	A	2
D	Product model	A	3
E	Sales brochure	A	2
F	Cost analysis	C	3
G	Product testing	D	4
H	Sales training	B, E	2
I	Pricing	H	1
J	Project report	F, G, I	1

Coming up: Game Development In-Class Exercise

Game Development In-Class Exercise

Find the critical path

TASK	DURATION (days)	PREDECESSORs
A Graphics Engine	14	
B Sound Engine	5	I
C Music Engine	5	J
D Input Engine	10	A
E Gameplay/general programming	31	B, C, D
F Physics	7	E
G 2D Artwork	14	
H 3D Artwork	21	G
I Sound Effects	14	
J Music	9	
K Level Design	21	F, H

Coming up: Review Questions

CRITICAL PATH ANALYSIS :

- As soon as the network is drawn for the given project, the time analysis is required for planning of various activities of the project. The main aim of time analysis is to prepare a planning schedule for the project.
- Critical path Analysis is a special method for time analysis to determine following:
    i. Total duration for project completion
    ii. Categorize the activities of the project in two types: critical and Non-critical

Critical Activity :

- An activity in network diagram, whose delay in beginning will further delay the project completion time.

Non-critical activity :

- An activity which allows some scheduling slack, so that the start time of the activity may be delayed or advanced within some range without affecting the completion time of entire project.

Critical Path Calculation

Notations:

E_i = Earliest occurrence time of event i

L_j = Latest occurrence time of event j

(i, j) = Activity with head event j and tail event i

t_{ij} = Duration of activity (i, j)

ES_{ij} = Earliest starting time of activity (i, j)

LS_{ij} = Latest starting time of activity (i, j)

EF_{ij} = Earliest finishing time of activity (i, j)

LF_{ij} = Latest finishing time of activity (i, j)

Forward pass calculation:

- Start from the initial node (I) with starting time of the project as zero. We move along the nodes with increasing order of node number (serial no.) and end at final (terminal) node of the network.

Algorithm for forward pass:

Step 1: Initialize i=1, E_1=0 (First node)

Step 2: Calculate earliest start time for each activity that begins at node *i*; as

$$ES_{ij} = E_i \text{ for all } (i, j) \text{ with starting node } i$$

Step 3: Compute earliest finishing time for each activity that begins at node *i*; as

$$EF_{ij} = ES_{ij} + t_{ij} = E_i + t_{ij}$$

where t_{ij}=duration of (i, j).

Step 4: Proceed to next node, say node *j* ($j>i$) and compute the earliest occurrence foe node *j* using

$$E_j = max_i \{EF_{ij}\} = max_i \{E_i + t_{ij}\}$$

for all immediate predecessor *j*.

Step 5: If $j=n$ (final node), then the earliest finish time for the project is given by: $E_n = max \{EF_{ij}\} = max\{E_{n-1} + t_{ij}\}$

Backward pass calculation:

- Here, we begin from terminal (last) node of the network proceed through the network visiting nodes in the decreasing order of node numbers and end at the initial node. At each node we calculate the least finish time for each activity.

Step 1: Initialize $L_n=E_n$ for $j=n$.

Step2: Set the latest finishing time for each activity (i, j) that end at node j.

$$LF_{ij}=L_j$$

Step 3: Calculate latest starting time for each (i, j)

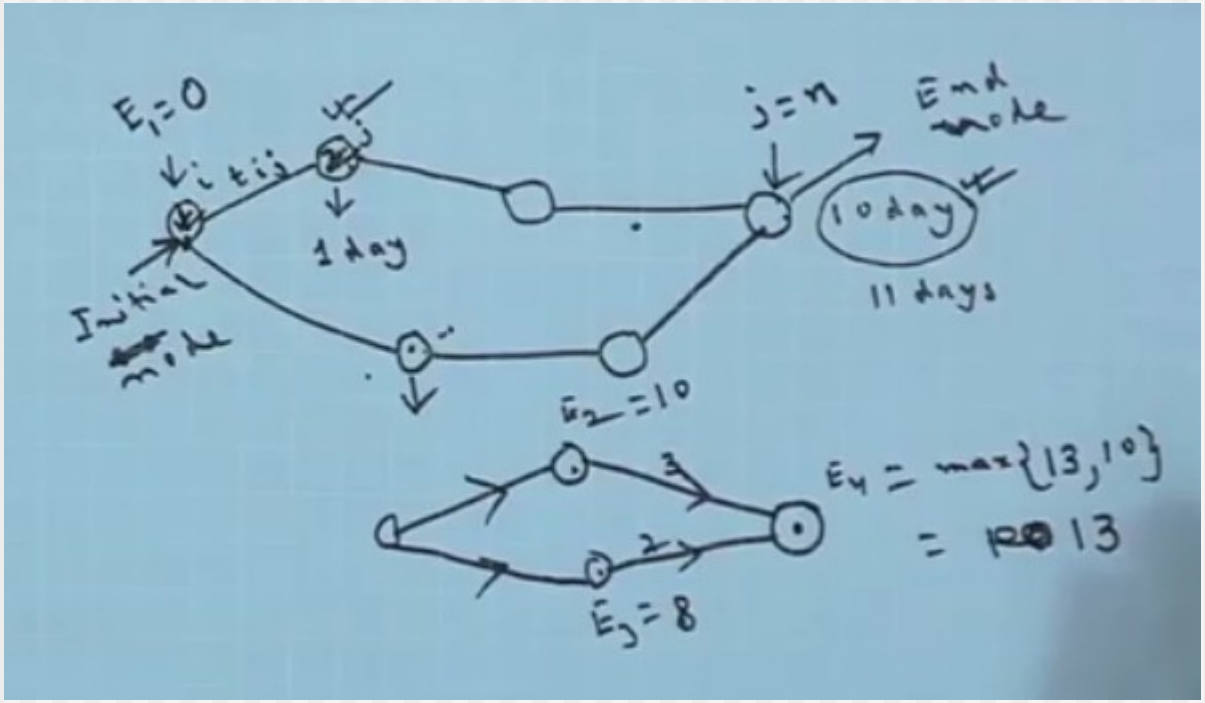
$$LS_{ij}= LF_{ij}-t_{ij}$$

Step 4: Proceed backward to the node in the sequence that decrease j by 1.

Latest occurrence time of node i $(i<j)$

$$L_i=min_j \{LS_{ij}\} = min_j \{L_j-t_{ij}\}$$

Step5: If $j=1$ (beginning node) $L_1=min \{LS_{ij}\}=min \{L_2-t_{ij}\}$

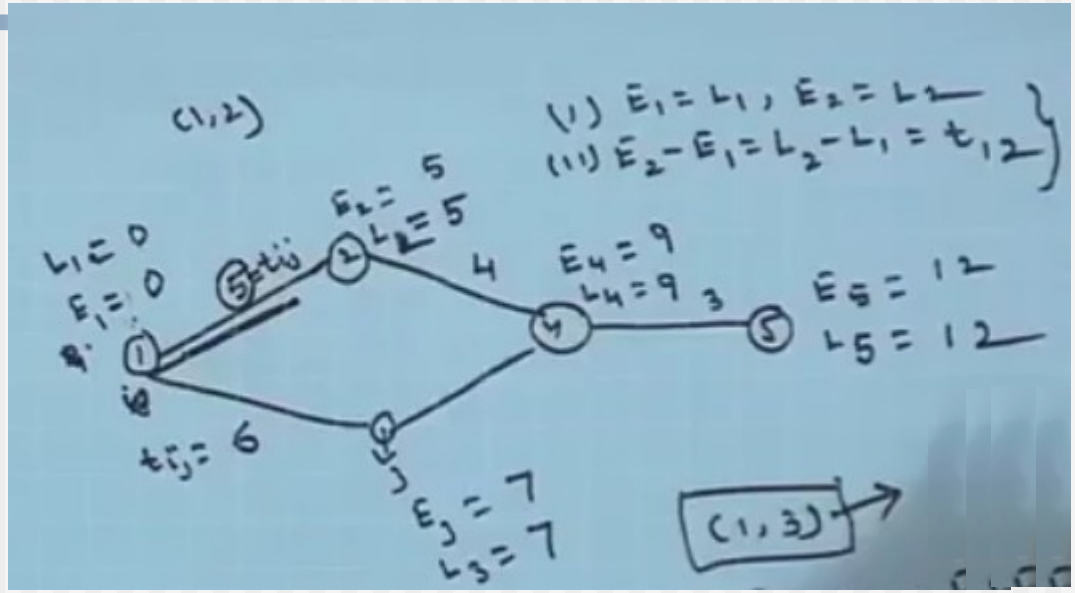

$E_1 = 0$
$j = n$
End node
1 day
10 day
11 days
Initial node
$E_2 = 10$
3
2
$E_4 = \max\{13, 10\}$
$= 13$
$E_3 = 8$

Mathematical Expression for Critical Activity:

- Based on forward pass and backward pass calculations, an activity (i, j) is critical if it satisfy following conditions:
  - I. $E_i = L_i$ and $E_j = L_j$
  - II. $E_j - E_i = L_j - L_i = t_{ij}$
- An activity (i, j) that does not satisfy the above conditions is termed as non-critical.

Critical path:

- The sequence of critical activities in a network is called critical path. It is the longest path in the network from starting event to ending event and determines the minimum time for project completion.
- If an activity on a critical path is delayed by a day, the project would also be delayed by a day unless the duration of future (subsequent) activities are shortened by some other means.
- To specify critical path on the network, double lines are used to distinguish from non-critical path.


(1,2)
(i) $E_1 = L_1$, $E_2 = L_2$
(ii) $E_2 - E_1 = L_2 - L_1 = t_{12}$
$L_1 = 0$
$E_1 = 0$
$E_2 = 5$
$L_2 = 5$
$E_4 = 9$
$L_4 = 9$
$E_5 = 12$
$L_5 = 12$
$t_{ij} = 6$
$E_3 = 7$
$L_3 = 7$
(1,3)

Float and Slack times:

- After labeling the network diagram and performing the forward and backward pass computations one has to determine float and slack times.
  Float → used for activities
  Slack → used for events.

Slack of an event: (Also known as Event float)

- The slack of an event is difference between its latest time (L_i) and its earliest time (E_i). That is

$$Event\ Slack = L_i - E_i.$$

- An event with zero slack time is called critical event.

Float of an activity: (Activity Float)

- There are four types of Activity floats, which determine activity's time-estimates.

i. Total float
ii. Free float
iii. Independent float
iv. Inference float

(i) Total float:

- Amount of time by which an activity can be delayed without delay in project completion date. It implies the free time associated with the activity which can be used before, during or after completion of this activity.

Mathematically,

Total Float= Latest Finish time-Earliest finish time

Or *= Latest start time – Earliest start time*

i.e. $TF_{ij}=LS_{ij}-ES_{ij}$

- An activity with zero total float is called critical activity.

(ii) Free Float:

- A portion of the total float within which an activity can be manipulated without affecting the float of the subsequent activities.

Mathematically,

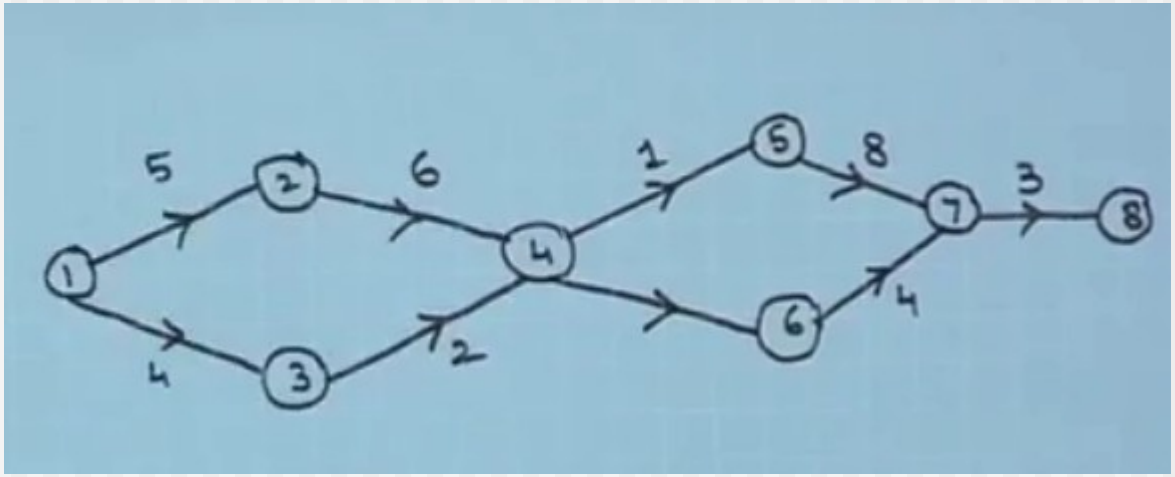
Free Float of activity (i, j) $FF_{ij}=(E_j-E_i)-t_{ij}$

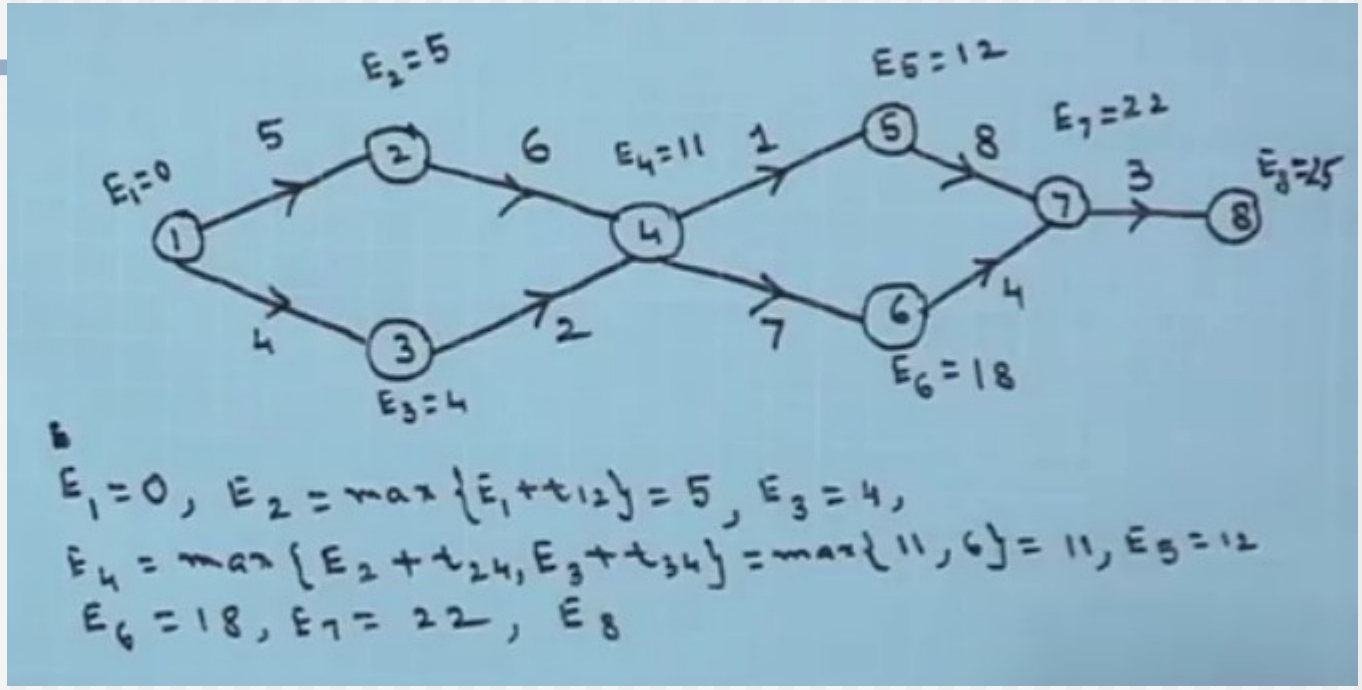
Note that, $0 \leq FF_{ij} \leq TF_{ij}$

Free float is used to reschedule activities with minimum disruption of earlier plans.

Draw the network for the following project. Compute the latest and earliest time for each node and also find critical path.

Activity	Immediate Predecessor	Time
1-2	-	5
1-3	-	4
2-4	1-2	6
3-4	1-3	2
4-5	2-4, 3-4	1
4-6	2-4, 3-4	7
5-7	4-5	8
6-7	4-6	4
7-8	6-7, 5-7	3

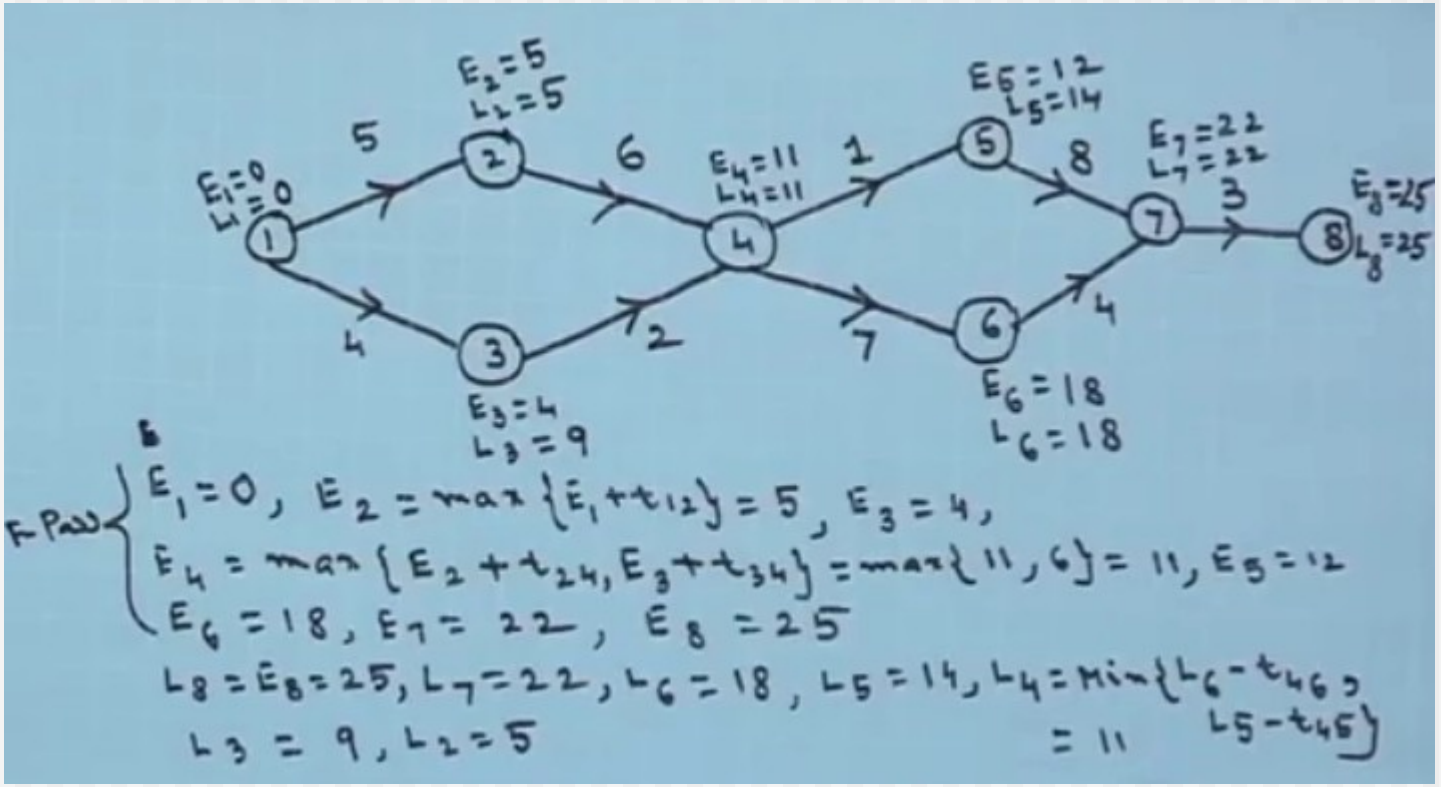

1
2
3
4
5
6
7
8
5
6
4
2
1
8
4
3



$E_1 = 0$, $E_2 = \max\{E_1 + t_{12}\} = 5$, $E_3 = 4$,

$E_4 = \max\{E_2 + t_{24}, E_3 + t_{34}\} = \max\{11, 6\} = 11$, $E_5 = 12$

$E_6 = 18$, $E_7 = 22$, E_8



F Pass

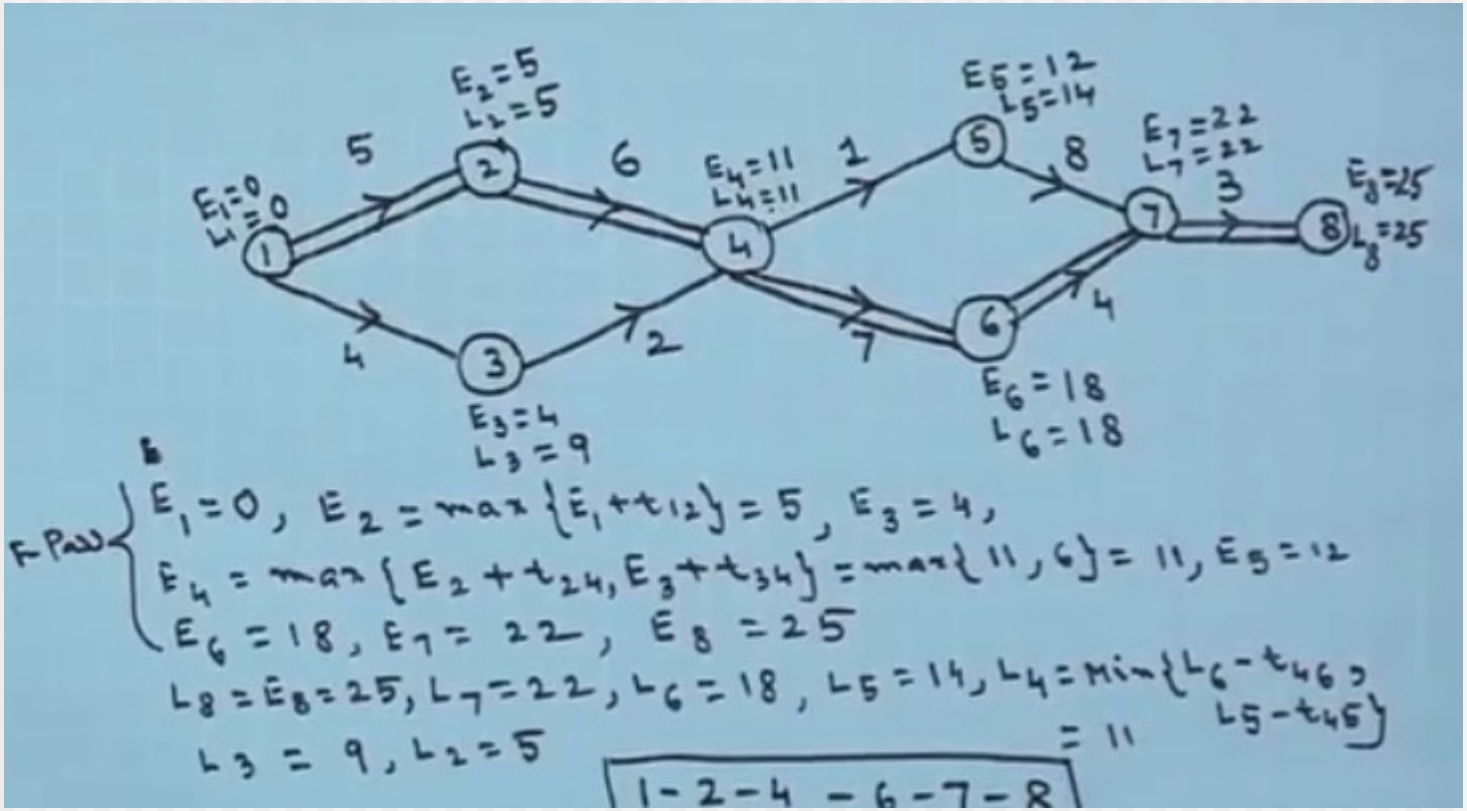
$E_1 = 0$, $E_2 = \max\{E_1 + t_{12}\} = 5$, $E_3 = 4$,

$E_4 = \max\{E_2 + t_{24}, E_3 + t_{34}\} = \max\{11, 6\} = 11$, $E_5 = 12$

$E_6 = 18$, $E_7 = 22$, $E_8 = 25$

$L_8 = E_8 = 25$, $L_7 = 22$, $L_6 = 18$, $L_5 = 14$, $L_4 = \min\{L_6 - t_{46}, L_5 - t_{45}\} = 11$

$L_3 = 9$, $L_2 = 5$



For Path

$E_1 = 0,\ E_2 = \max\{E_1 + t_{12}\} = 5,\ E_3 = 4,$

$E_4 = \max\{E_2 + t_{24}, E_3 + t_{34}\} = \max\{11, 6\} = 11,\ E_5 = 12$

$E_6 = 18,\ E_7 = 22,\ E_8 = 25$

$L_8 = E_8 = 25,\ L_7 = 22,\ L_6 = 18,\ L_5 = 14,\ L_4 = \min\{L_6 - t_{46},\ L_5 - t_{45}\}$

$= 11$

$L_3 = 9,\ L_2 = 5$

1-2-4-6-7-8

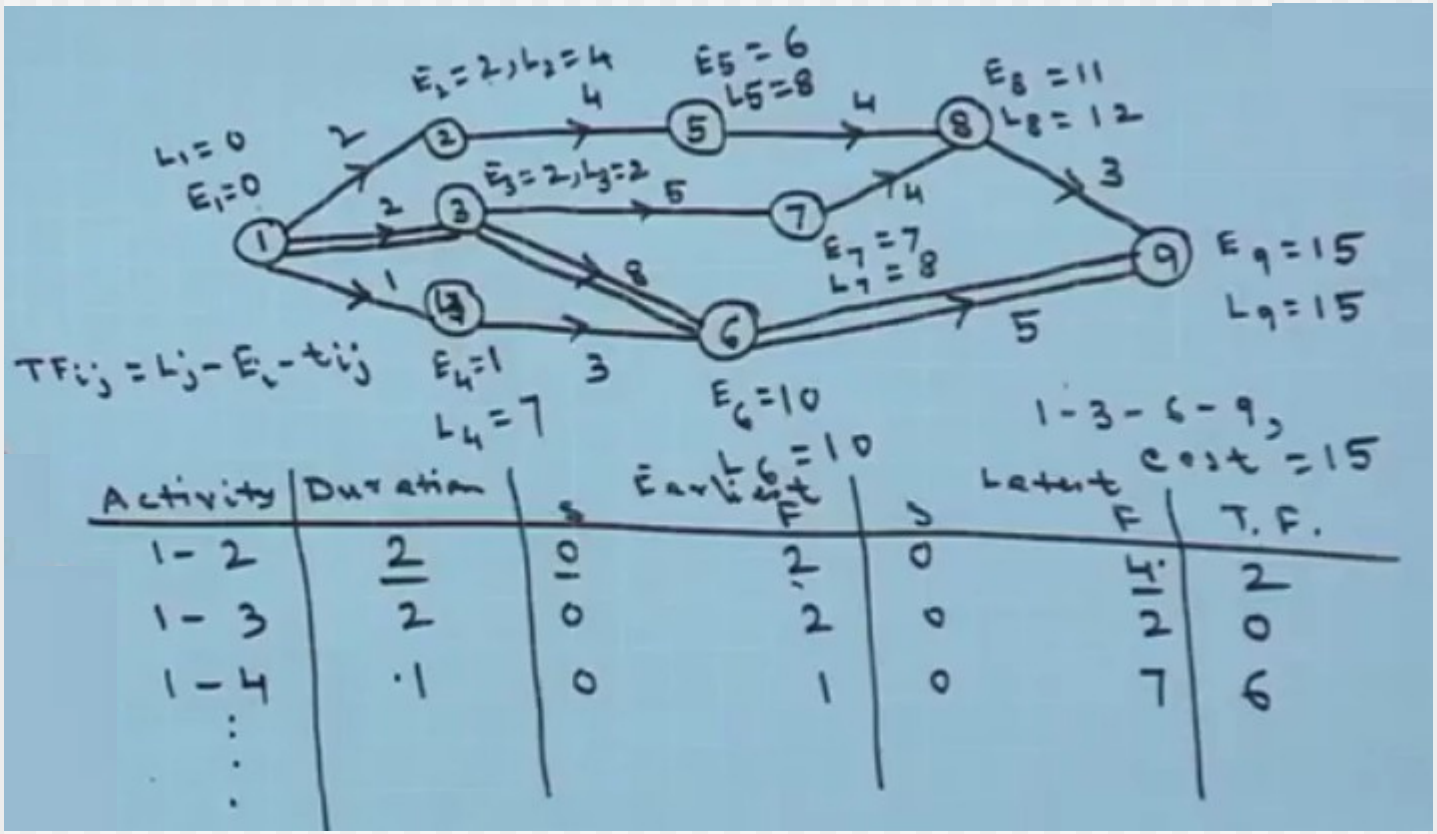
A building construction project has following time schedule:

Activity	Times (in month)
1-2	2
1-3	2
1-4	1
2-5	4
3-6	8
3-7	5
4-6	3
5-8	1
6-9	5
7-8	4
8-9	3

a) Construct the network diagram and
Construct the network diagram and

Compute total float for each activity.

Find the critical path and its duration.

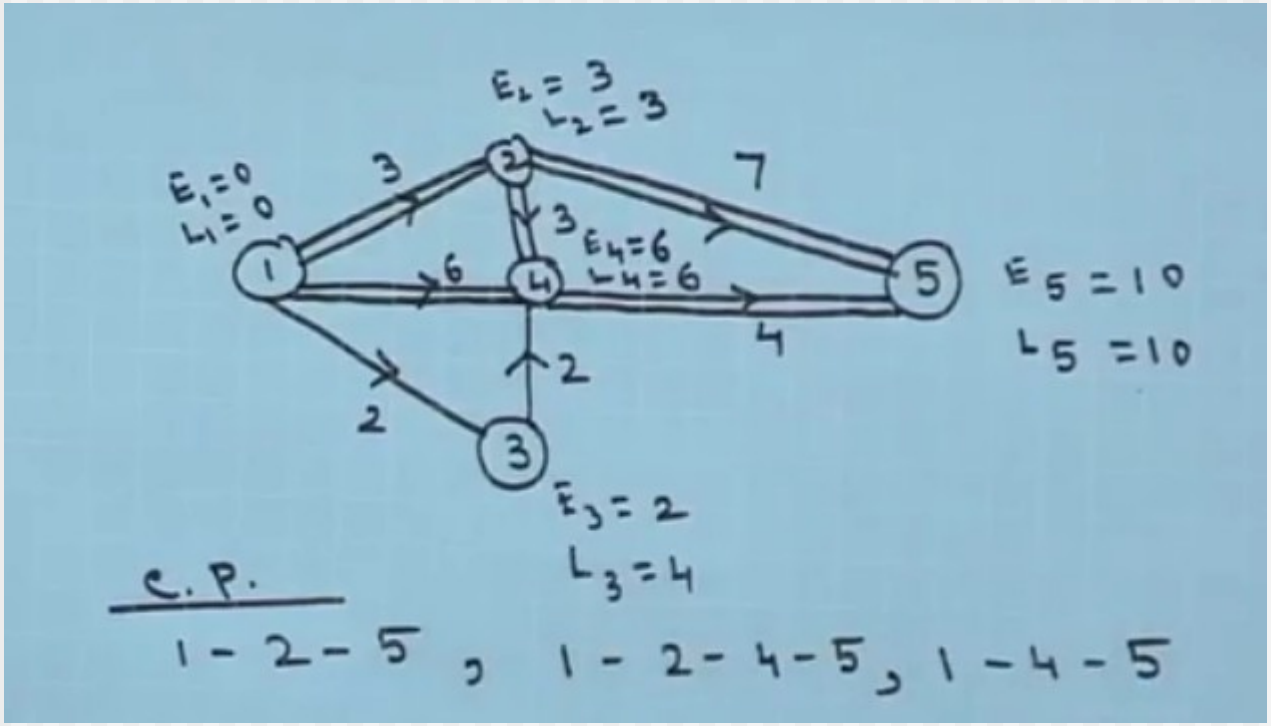


Activity	Duration	Earliest		Latest		
		S	F	S	F	T.F.
1-2	2	0	2	0	4	2
1-3	2	0	2	0	2	0
1-4	1	0	1	0	7	6
⋮						

Example:

Consider a problem with the following data.

Activity	Times (in month)
1-2	3
1-3	2
1-4	6
2-4	3
2-5	7
3-4	2
4-5	4


E1 = 3
L2 = 3
E1 = 0
L1 = 0
3
7
3
E4 = 6
L4 = 6
6
E5 = 10
L5 = 10
4
2
2
E3 = 2
L3 = 4
C.P.
1 - 2 - 5 , 1 - 2 - 4 - 5 , 1 - 4 - 5

(iii) Independent Float:

- A portion of total float within which an activity can be delayed for start without affecting floats of preceding activities.

Mathematically,

$$IF_{ij}=(E_j-L_i)-t_{ij}$$

- Negative independent float is assumed zero.

(iv) Inference Float:

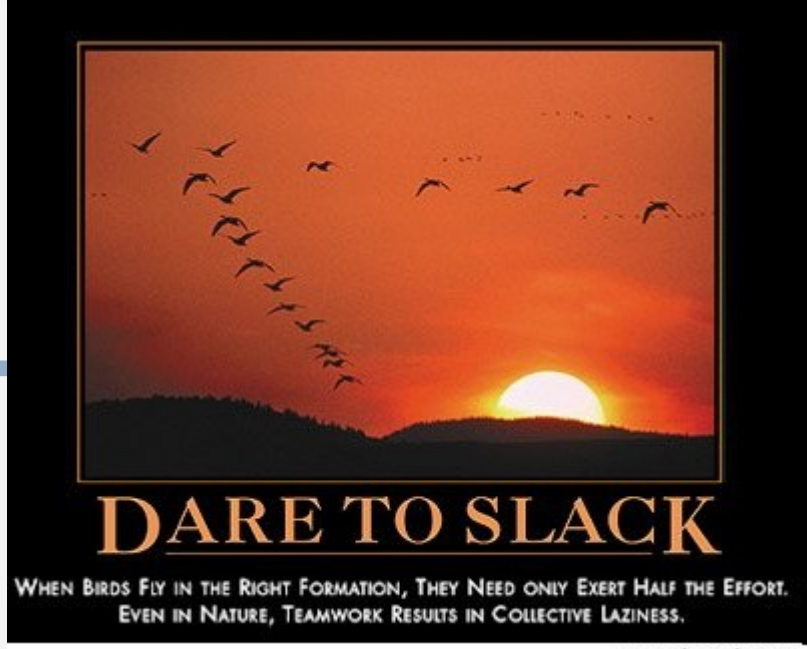
- A portion of total float which causes reduction in the float of the successor activities.

$$INF_{ij}=TF_{ij}-FF_{ij}$$

- Note the relation:

$$Independent\ Float \leq free\ Float \leq Total\ Float$$

Review Questions



- What is the critical path?
- Do all nodes on the critical path have to be connected to each other? (directly)
- What is slack?
- When should you write your schedule for the work?
- What is resource leveling?

Coming up: What about Agile?

Scheduling Rules of Thumb

- One person should always edit the schedule (you!)
  - If you have two people that need to, create two files and link them together
- Keep it simple and useful
- Level your resources
- Share the schedule with your team
- 40-20-40; coding is 20% of the effort

Schedule Example

- Lets try to schedule this work among our three developers “John, Mary, Carl”

TASK	DURATION (days)	PREDECESSORs
A Graphics Engine	14	
B Sound Engine	5	I
C Music Engine	5	J
D Input Engine	10	A
E Gameplay/general programming	31	B, C, D
F Physics	7	E
G 2D Artwork	14	
H 3D Artwork	21	G
I Sound Effects	14	
J Music	9	
K Level Design	21	F, H

Coming up: Scheduling Steps

Scheduling Steps

- Add in all the tasks (preferably in a hierarchy)
- Add in all the dependencies
- Break down large tasks into smaller tasks. Optimally (in Dan Fleck's opinion) you want to schedule so the duration of each smallest task is at most 3-5 days
- Assign people (resources) to tasks
- Level your resources

Classic Mistakes

- Overly optimistic schedule
- Failing to monitor schedule
- Failing to update schedule
- Adding people to a late project
- Failure to manage expectations of others
- Leaving out a task

Coming up: Scope Creep

Tracking Progress

It is reasonable to ask whether there is a quantitative technique for assessing progress as the software team progresses through the work tasks allocated to the project schedule.

The earned value system provides a common value scale for every [software project] task, regardless of the type of work being performed. The total hours to do the whole project are estimated, and every task is given an earned value based on its estimated percentage of the total.

Earned Value Analysis

Stated even more simply, earned value is a measure of progress. It enables you to assess the “percent of completeness” of a project using quantitative analysis rather than rely on a gut feeling.

Earned Value Analysis

1. The *budgeted cost of work scheduled* (BCWS) is determined for each work task represented in the schedule. During estimation, the work (in person-hours or person-days) of each software engineering task is planned. Hence, BCWS*i* is the effort planned for work task *i.* To determine progress at a given pointalong the project schedule, the value of BCWS is the sum of the BCWS*i* valuesfor all work tasks that should have been completed by that point in time on the project schedule.

2. The BCWS values for all work tasks are summed to derive the *budget atcompletion* (BAC). Hence,BAC (BCWS*k*) for all tasks *k*.

3. Next, the value for *budgeted cost of work performed* (BCWP) is computed. Theva lue for BCWP is the sum of the BCWS values for all work tasks that have actually been completed by a point in time on the project schedule.

Earned Value Analysis

Wilkens [Wil99] notes that “the distinction between the BCWS and the BCWP is that the former represents the budget of the activities that were planned to be completed and the latter represents the budget of the activities that actually were completed.”

Earned Value Analysis

Given values for BCWS, BAC, and BCWP, important progress indicators can be computed:

$$\text{Schedule performance index, SPI} = \frac{\text{BCWP}}{\text{BCWS}}$$

$$\text{Schedule variance, SV} = \text{BCWP} - \text{BCWS}$$

SPI is an indication of the efficiency with which the project is utilizing scheduled resources. An SPI value close to 1.0 indicates efficient execution of the project schedule.

SV is simply an absolute indication of variance from the planned schedule.

Earned Value Analysis

$$\text{Percent scheduled for completion} = \frac{\text{BCWS}}{\text{BAC}}$$

provides an indication of the percentage of work that should have been completed by time *t.*

$$\text{Percent complete} = \frac{\text{BCWP}}{\text{BAC}}$$

provides a quantitative indication of the percent of completeness of the project at a given point in time *t.*

Earned Value Analysis

It is also possible to compute the *actual cost of work performed* (ACWP). The value for ACWP is the sum of the effort actually expended on work tasks that have been completed by a point in time on the project schedule. It is then possible to compute

$$\text{Cost performance index, CPI} = \frac{\text{BCWP}}{\text{ACWP}}$$

$$\text{Cost variance, CV} = \text{BCWP} - \text{ACWP}$$

A CPI value close to 1.0 provides a strong indication that the project is within its defined budget. CV is an absolute indication of cost savings (against planned costs) or shortfall at a particular stage of a project.

Earned Value Management

- How much work you planned to have accomplished by now (in dollars or hours) called the **Planned Value**
- How much you have actually spent by now (in dollars or hours), called **Actual Cost**
- The value, in terms of your baseline budget, of the work accomplished by now (in dollars or hours), called the **Earned Value**!

Idea is to link schedule and cost together to monitor both in the same “units” of value

Earned Value Management

- Planned value (PV) - the value of all resources needed to meet the project's objectives
  - Each objective of a project has an associated planned value
- Budgeted (cost) at completion (BAC) - The sum of all the PVs
- Earned value (EV) - the amount of value completed at any point during the project
- Actual Cost (AC) - actual amount of money you have spent so far. In a perfect project AC and EV are the same.

Coming up: Earned Value Management Example

Earned Value Management Example

- We’ve budgeted $200 to buy, setup, network and test a new system
  - Our planned values (PVs) of each task are:
    - Buy - $50, Setup - $75, network - $50, test - $25
    - Our BAC is therefore $200
  - We’ve now completed phase one, and thus our earned value (EV) is now $50.
  - To do this we spent $60 (our actual cost (AC))

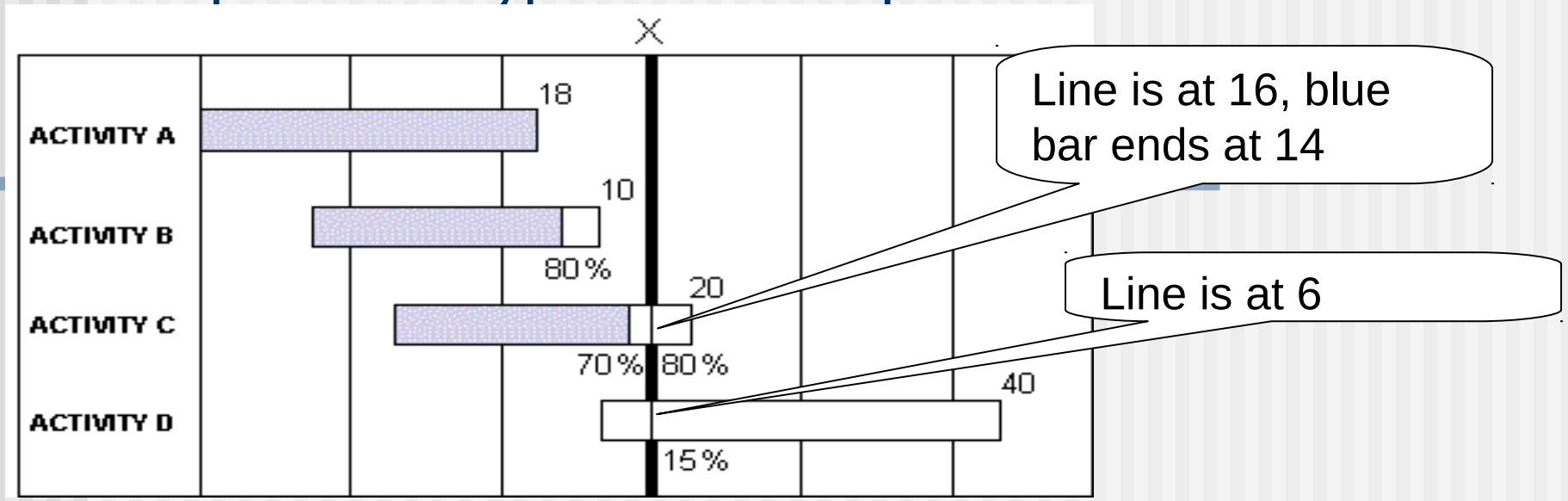
Earned Value Management Example

- Schedule performance index (SPI)
  - EV / PV --> 50/50 = 1 (perfect).
  - Our group is on schedule
- Cost performance index (CPI)
  - EV / AC --> 50/60 = 0.83
  - For every dollar spent, I get 83 cents worth of work.
- Estimated cost at completion (EAC)
  - BAC / CPI = 200 / 0.83 = $240.96
- Schedule Variance (SV) : EV - PV
- Cost Variance (CV) : EV - AC

Memorization Hint: Most equations begin with earned value

Coming up: EVM Example 2 from:
http://www.hyperthot.com/pm_cscs.htm

EVM Example 2 from: http://www.hyperthot.com/pm_cscs.htm



- **PLANNED VALUE (Budgeted cost of the work scheduled)**
  = What is planned value at time X?
- **EARNED VALUE (Budgeted cost of the work performed)**
  = What is earned value at time X?
- **ACTUAL COST (of the work performed) = $45 (Data from Acct. System)**
- Therefore:
  - **Schedule Variance =** Earned – Planned. Perfect is?
  - **Schedule Performance Index = 40 / 50 = 0.8**

Coming up: What is earned value?

What is earned value?

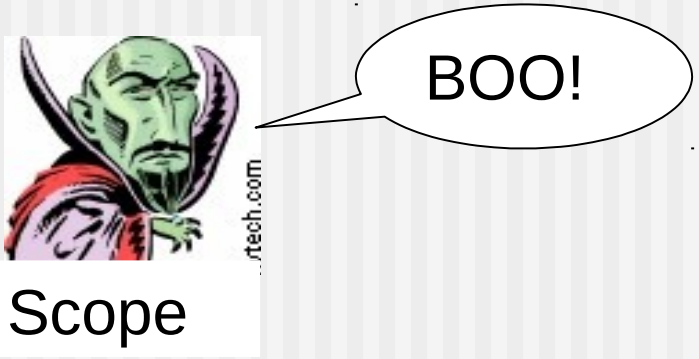
- A. The amount of money you get upon completion of a task
- B. The value of an activity
- C. The value of the work completed by now in the schedule
- D. The value of all activities planned to be completed by now in the schedule

Coming up: Why do you use earned value management?

Why do you use earned value management?

- A.It is required by my contract
- B. Measuring value give you more information than measuring cost or time alone
- C. I don’t use it
- D. It guarantees my project will be done on time

Scope Creep



- The scope of your project is all the work you initially planned to do.
- Scope creep is when your project gets new tasks throughout it's lifetime without adding more resources to handle new tasks. The scope is "creeping" up…
- Scope changes are OK, and really unavoidable… that's fine. However you must update the resources (time, features or people accordingly)

Coming up: Why would scope changes occur?

Why would scope changes occur?

- A. You get more money to do more things
- B. The customer asks you to do something extra because “it is critical for success”
- C. A competing product has a feature that you must have to be competitive
- D. All of these

Coming up: Which are causes of scope creep?

Scope Change versus Creep

Your company has a $1million dollar contract with a defined scope.

Scope change:

Customer: please add all these requirements, and I'll increase the contract to $2million dollars

Manager: Certainly! ☺

Scope creep:

Customer: please add all these requirements, and I'll be really happy.

Manager: Certainly! ☹

Which are causes of scope creep?

- A. poor change control
- B. lack of proper initial identification of what is required to satisfy project objectives
- C. a weak project manager
- D. all of these

Source: Wikipedia: Scope Creep

Managing Scope


Scope

- How to deal with the inevitable “Scope creep”?
- Joint Application Development and prototyping
- Formal change approval
- Defer additional requirements as future system enhancements

Managing Risk

- Document your risks in a risk management plan
  1. Description of risk
  2. Likelihood of occurrence (0-100%)
  3. Impact - 1(low) → 5 (high), or cost $20,000
  4. Exposure = Impact * Likelihood
  5. Mitigation strategy
     - How to lessen the impact of the risk
     - How to lessen the likelihood
     - An action plan if risk occurs
- Update and track your risks
- Communicate your risks to upper management

Projects get into trouble when…

- Software people don’t understand their customer’s needs.
- The product scope is poorly defined.
- Changes are managed poorly.
- The chosen technology changes.
- Business needs change [or are ill-defined].
- Deadlines are unrealistic.
- Users are resistant.
- Sponsorship is lost [or was never properly obtained].
- The project team lacks people with appropriate skills.
- Managers [and practitioners] avoid best practices and lessons learned.

Common-Sense Approach to Projects

- *Start on the right foot.* This is accomplished by working hard (very hard) to **understand the problem** that is to be solved and then setting realistic objectives and expectations.

- *Maintain momentum. The* project manager must **provide incentives** to keep turnover of personnel to an absolute minimum, the team should **emphasize quality** in every task it performs, and senior management should do everything possible to stay out of the team's way.

- *Track progress.* For a software project, progress is tracked as work products (e.g., models, source code, sets of test cases) are produced and approved (using formal technical reviews) as part of a quality assurance activity.

- *Make smart decisions.* In essence, the decisions of the project manager and the software team should be to "keep it simple."

- *Conduct a postmortem analysis.* Establish a consistent mechanism for **extracting lessons learned** for each project.