SWEN90016

Software Processes & Project Management

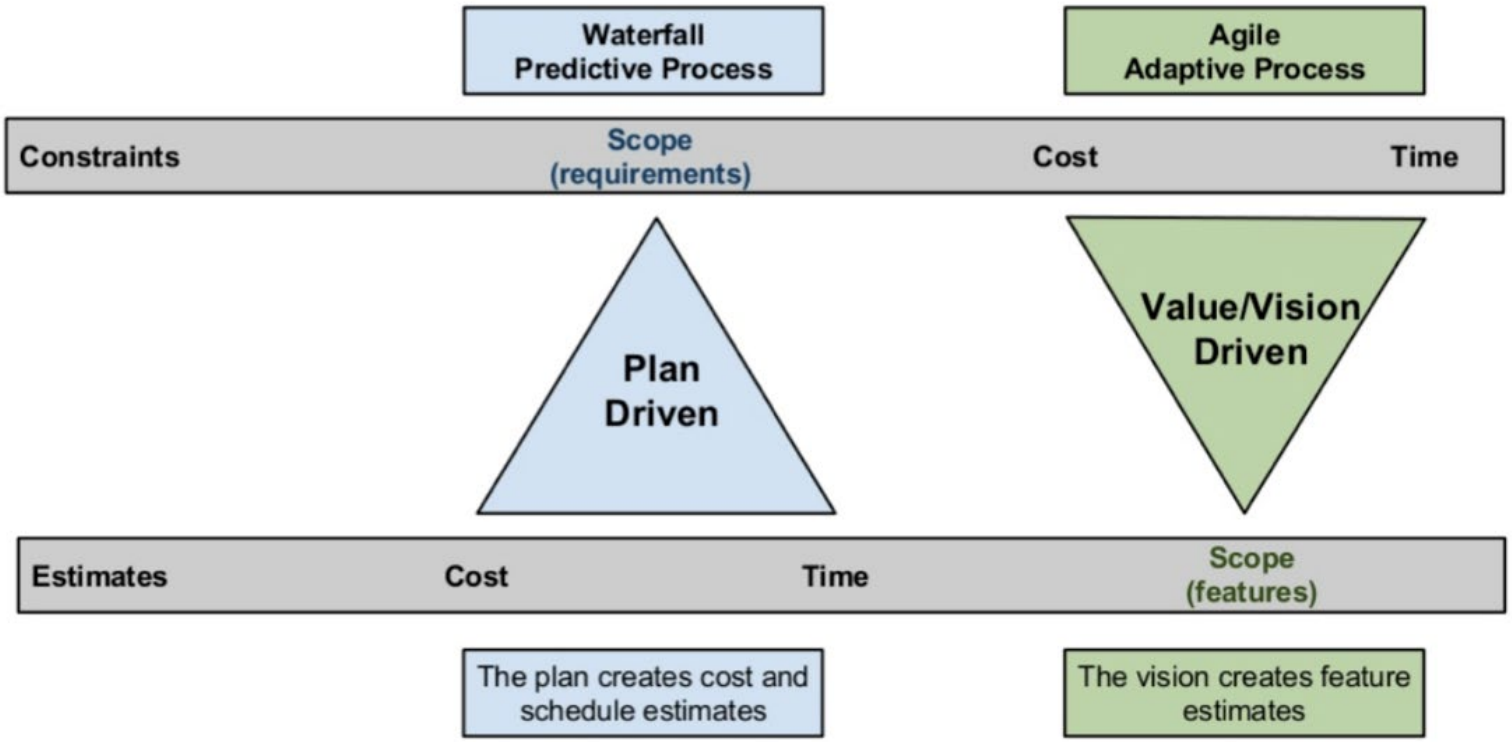
Project Planning and Scheduling

2020 – Semester 1

Tutorial 4

Today's aim

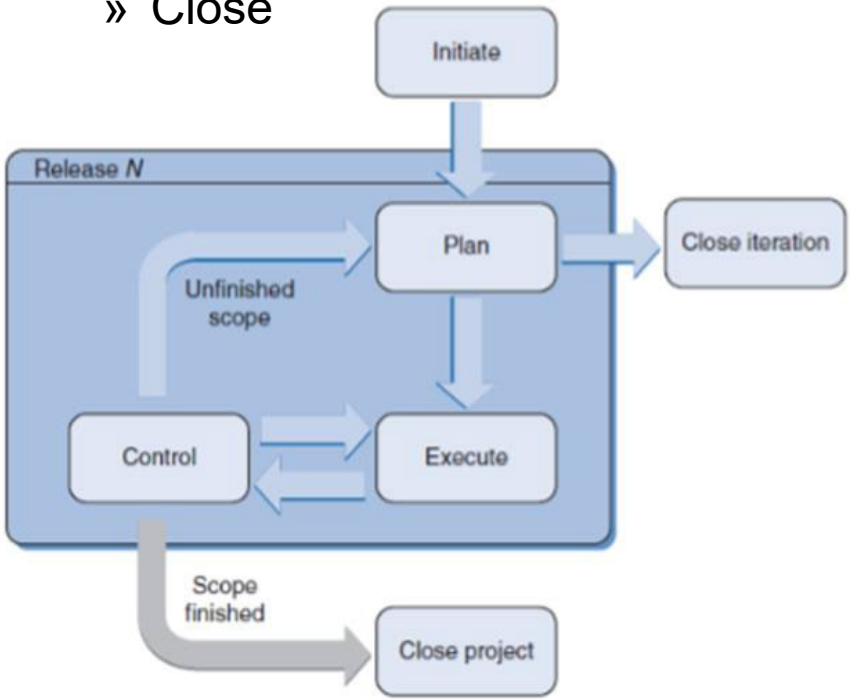
How to plan and control the schedule of software projects.



Software Projects

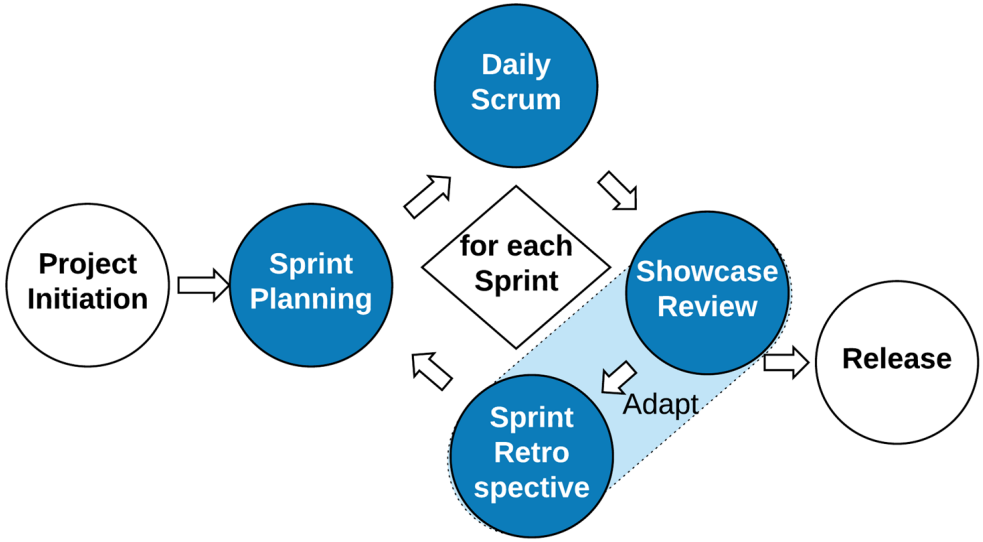
Formal PM Stages:

- Initiate
- Plan
- Execute
- Monitor & Control
- Close



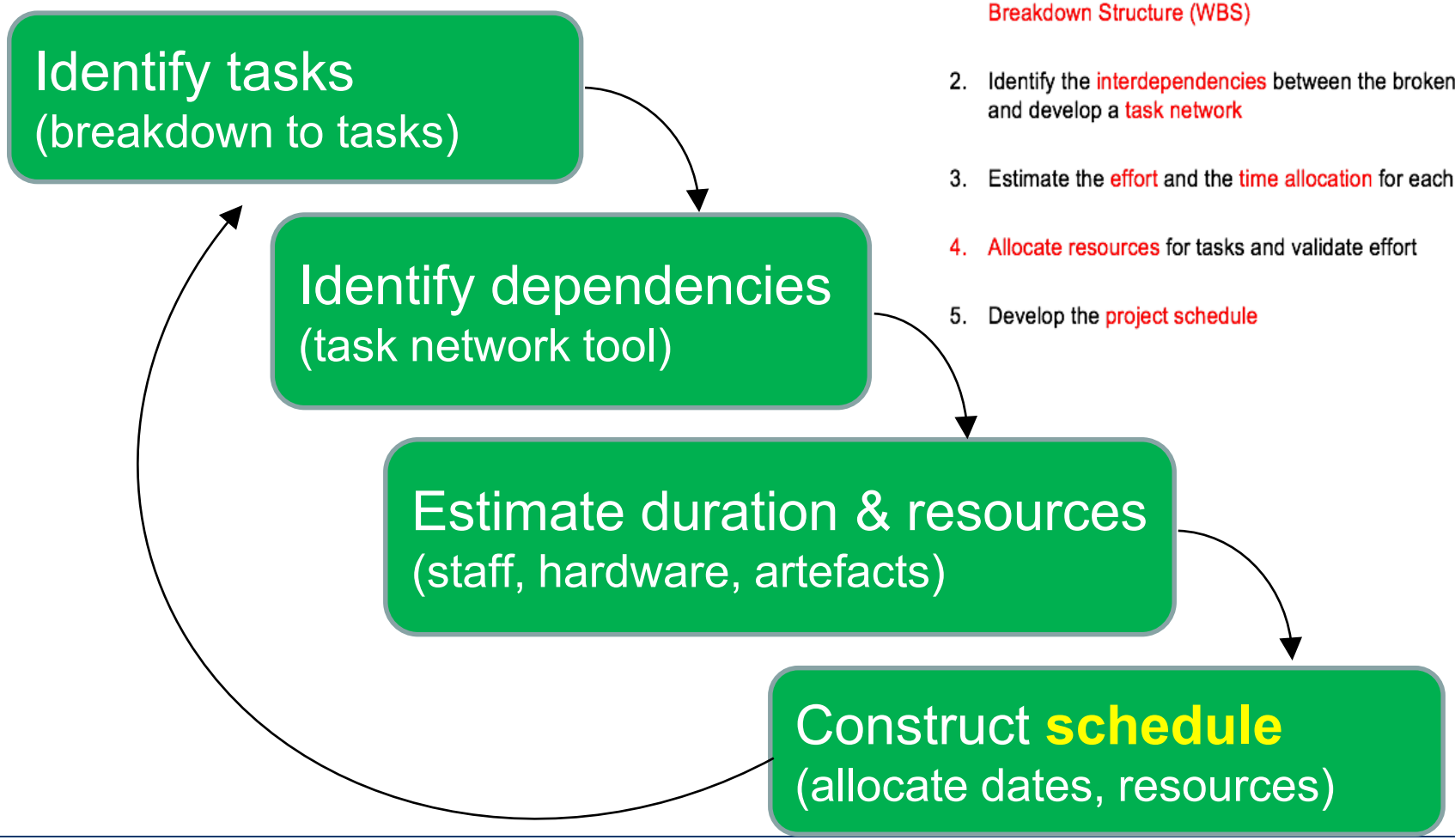
Agile PM Stages:

- Initiate
- Sprint Plan
- Scrum (or Sprint)
- Review & Retrospective (or Adapt)
- Release



Formal Project Schedule

What steps are involved in developing a project schedule?



1. Breakdown the task into small chunks you can deal with – Work Breakdown Structure (WBS)
2. Identify the interdependencies between the broken down tasks and develop a task network
3. Estimate the effort and the time allocation for each task
4. Allocate resources for tasks and validate effort
5. Develop the project schedule

Formal Project Scheduling

1. Work Breakdown Structure

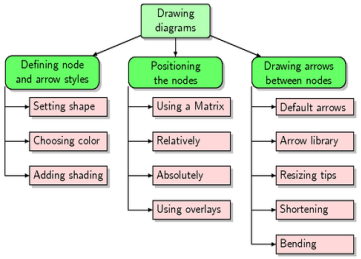
Redecorate Room

Prepare materials
- Buy paint
- Buy a ladder
- Buy brushes/rollers
- Buy wallpaper remover

Prepare room
- Remove old wallpaper
- Remove detachable decorations
- Cover floor with old newspapers
- Cover electrical outlets/switches with tape
- Cover furniture with sheets

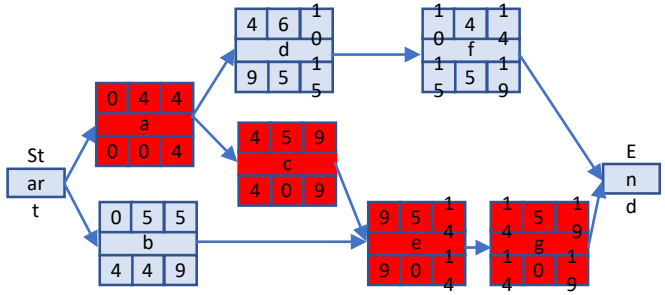
Paint the room

Clean up the room
- Dispose or store leftover paint
- Clean brushes/rollers
- Dispose of old newspapers
- Remove covers



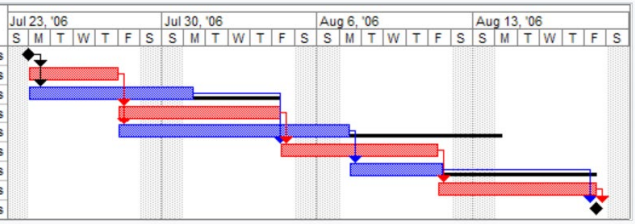
how to plan the schedule

2. PERT Chart



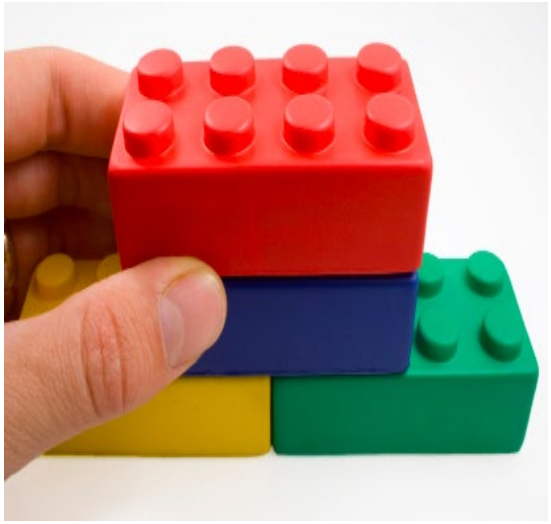
3. Gantt Chart

ID	Task Name	Predecessors	Duration
1	Start		0 days
2	a	1	4 days
3	b	1	5.33 days
4	c	2	5.17 days
5	d	2	6.33 days
6	e	3,4	5.17 days
7	f	5	4.5 days
8	g	6	5.17 days
9	Finish	7,8	0 days



A Gantt chart created using Microsoft Project (MSP). Note (1) the critical path is in red, (2) the slack is the black lines connected to non-critical activities, (3) since Saturday and Sunday are not work days and are thus excluded from the schedule, some bars on the Gantt chart are longer if they cut through a weekend.

Identify Tasks - Work Breakdown



	Activity	Work Breakdown
1.		**Concept Phase**
	1.1	Concept Planning
	1.2	Initial Research
	1.3	Problem definition with client
	1.4	Initial Project Plan
2.		**Requirements**
	2.1	Requirements Iteration 1
		2.1.1 Requirement Elicitation
		2.1.2 Requirements Analysis
		2.1.3 Requirement Model
	2.2	Requirements Iteration 2
		2.2.1 Requirement Elicitation
		2.2.2 Requirements Analysis
		2.2.3 Requirement Model
	2.3	Requirements Specification
	2.4	Requirements Validation
	2.5	Requirements Sign-off
3.		**Project Planning**
	3.1	Technological Risk Assessment

Identify Dependencies

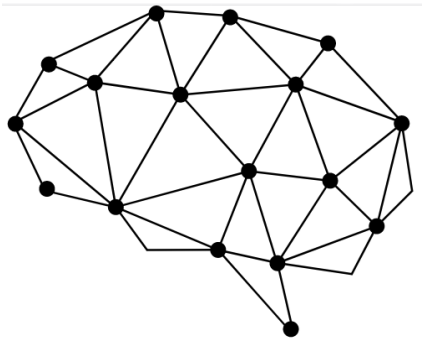
	Activity	Work Breakdown	Dependencies predecessor	Duration
1.		**Concept Phase**		
	1.1	Concept Planning		1
	1.2	Initial Research		4
	1.3	Problem definition with client		2
	1.4	Initial Project Plan	1.1, 1.2, 1.3	1
2.		**Requirements**		
	2.1	Requirements Iteration 1		
		2.1.1 Requirement Elicitation	1.4	2
		2.1.2 Requirements Analysis	2.1.1	3
		2.1.3 Requirement Model	2.1.2	3
	2.2	Requirements Iteration 2		
		2.2.1 Requirement Elicitation	2.1.2	3
		2.2.2 Requirements Analysis	2.2.1	3
		2.2.3 Requirement Model	2.2.2	4
	2.3	Requirements Specification	2.2.3	5
	2.4	Requirements Validation	2.3	4
	2.5	Requirements Sign-off	3.1, 2.4	4
3.		**Project Planning**		
	3.1	Technological Risk Assessment	2.1.2	4

Identify Dependencies

Develop a task network

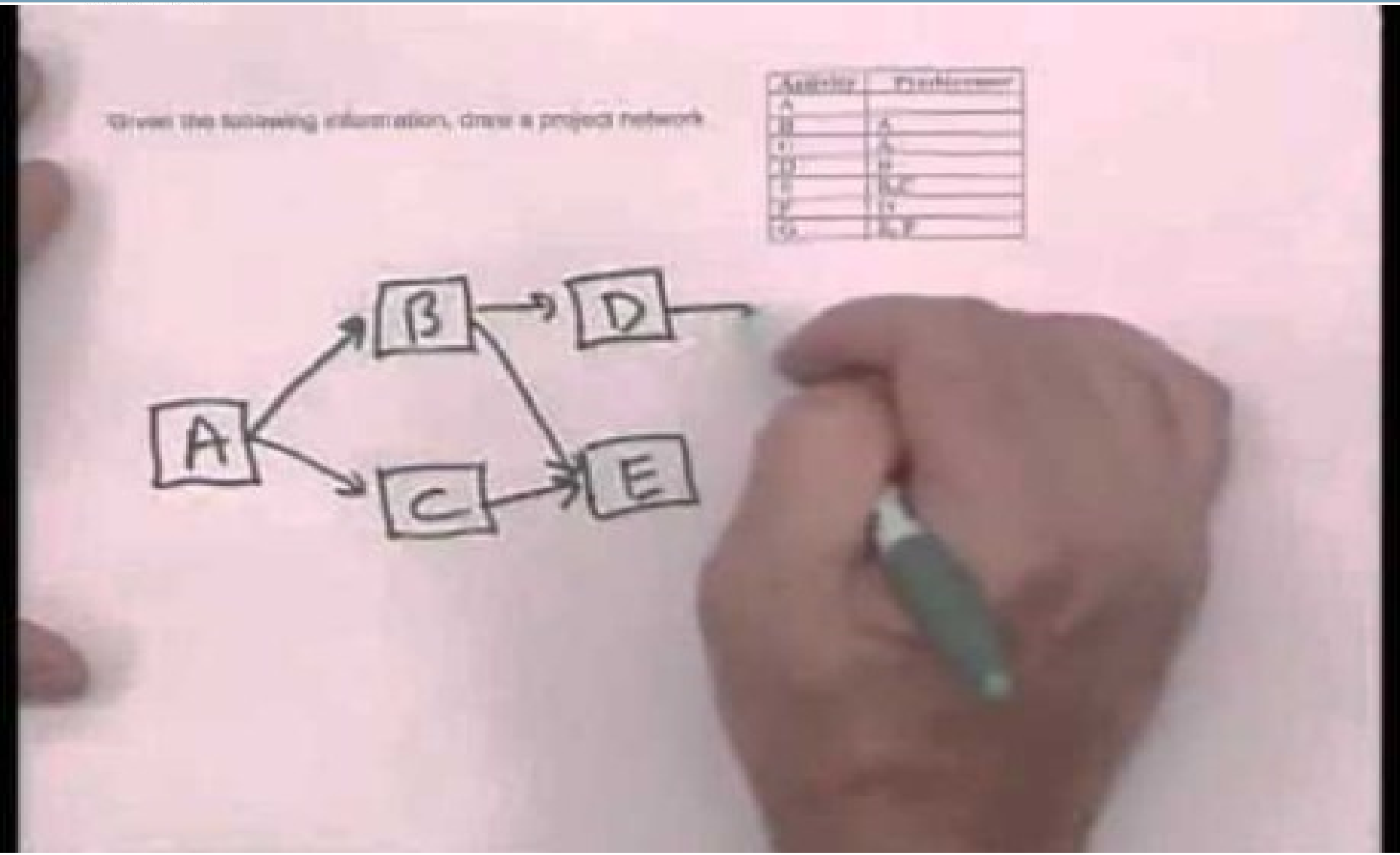
(activity on node)

given dependencies



	activity	predecessor	duration
1	1.1		1
2	1.2		4
3	1.3		2
4	1.4	1.1 1.2 1.3	1
5	2.1.1	1.4	2
6	2.1.2	2.1.1	3
7	2.1.3	2.1.2	3
8	2.2.1	2.1.2	3
9	2.2.2	2.2.1	3
10	2.2.3	2.2.2	4
11	2.3	2.2.3	5
12	2.4	2.3	4
13	2.5	2.4 3.1	4
14	3.1	2.1.2	4

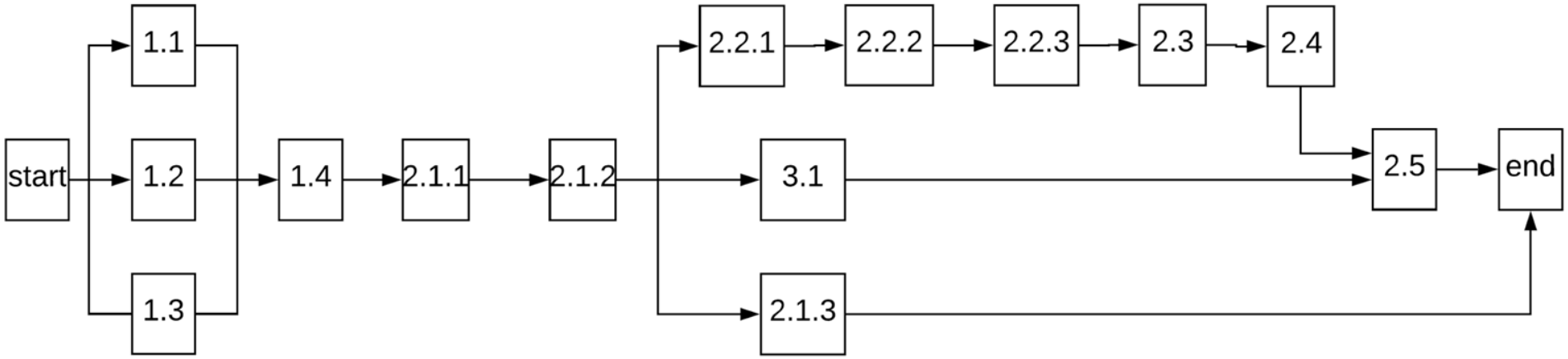
How to draw a simple network diagram



Task Network

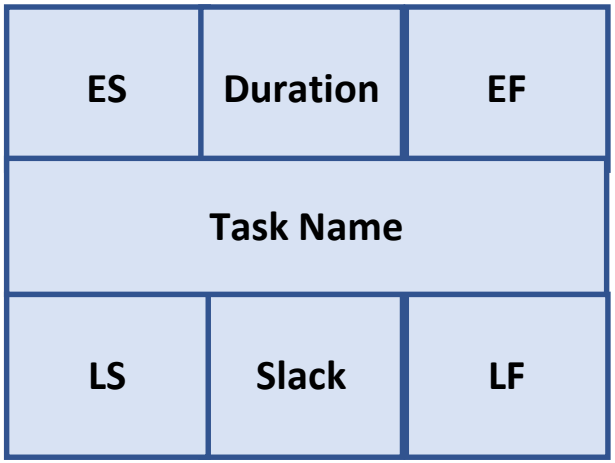
Network Diagram

- Sequential nodes
- Few details



Pert Chart

PERT: Program Evaluation & Review Technique



The activity node

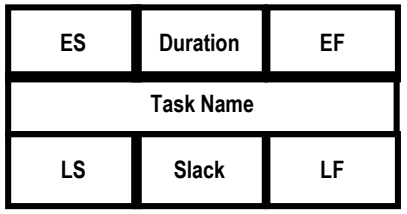
Earliest start time (ES)
Duration in people days
Earliest finish time (EF)

Latest start time (LS)
Slack time
Latest finish time (LF)

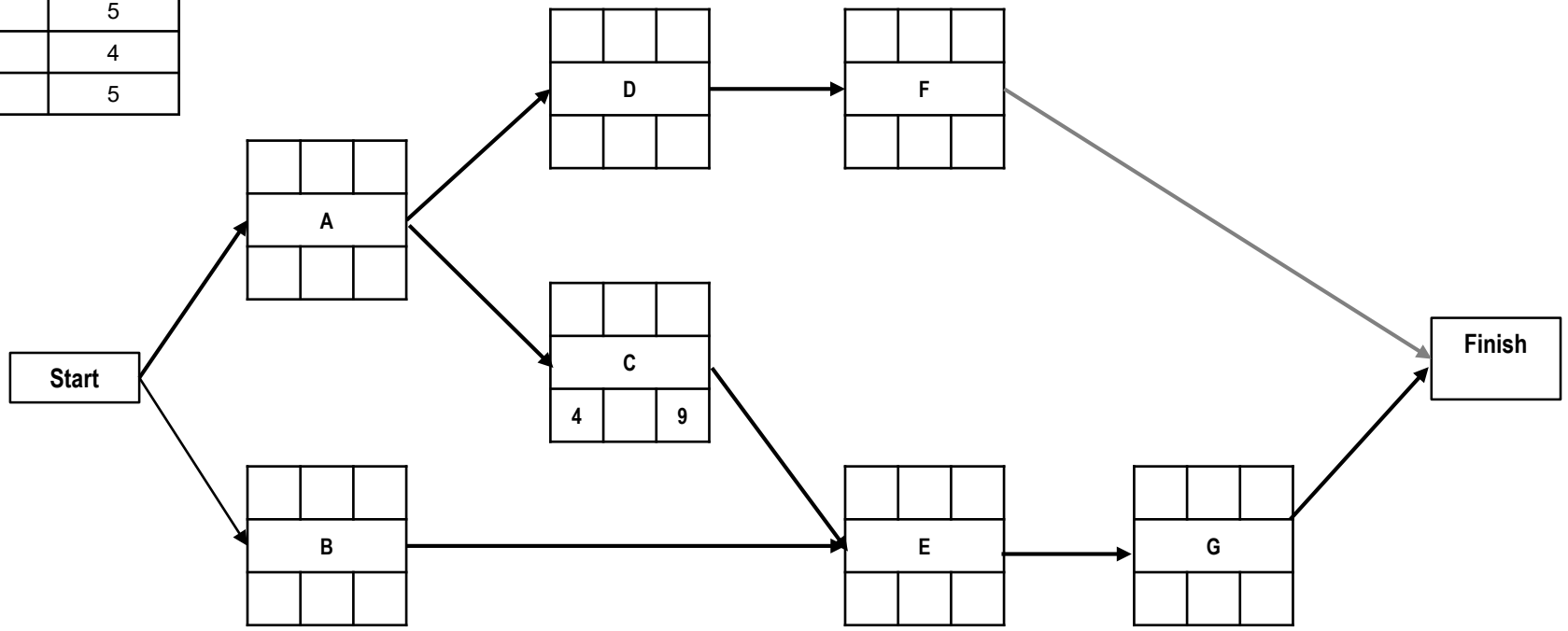
Pert Chart: example

Show a PERT chart: use task durations & task network diagram

Activity	Duration
A	4
B	5
C	5
D	6
E	5
F	4
G	5

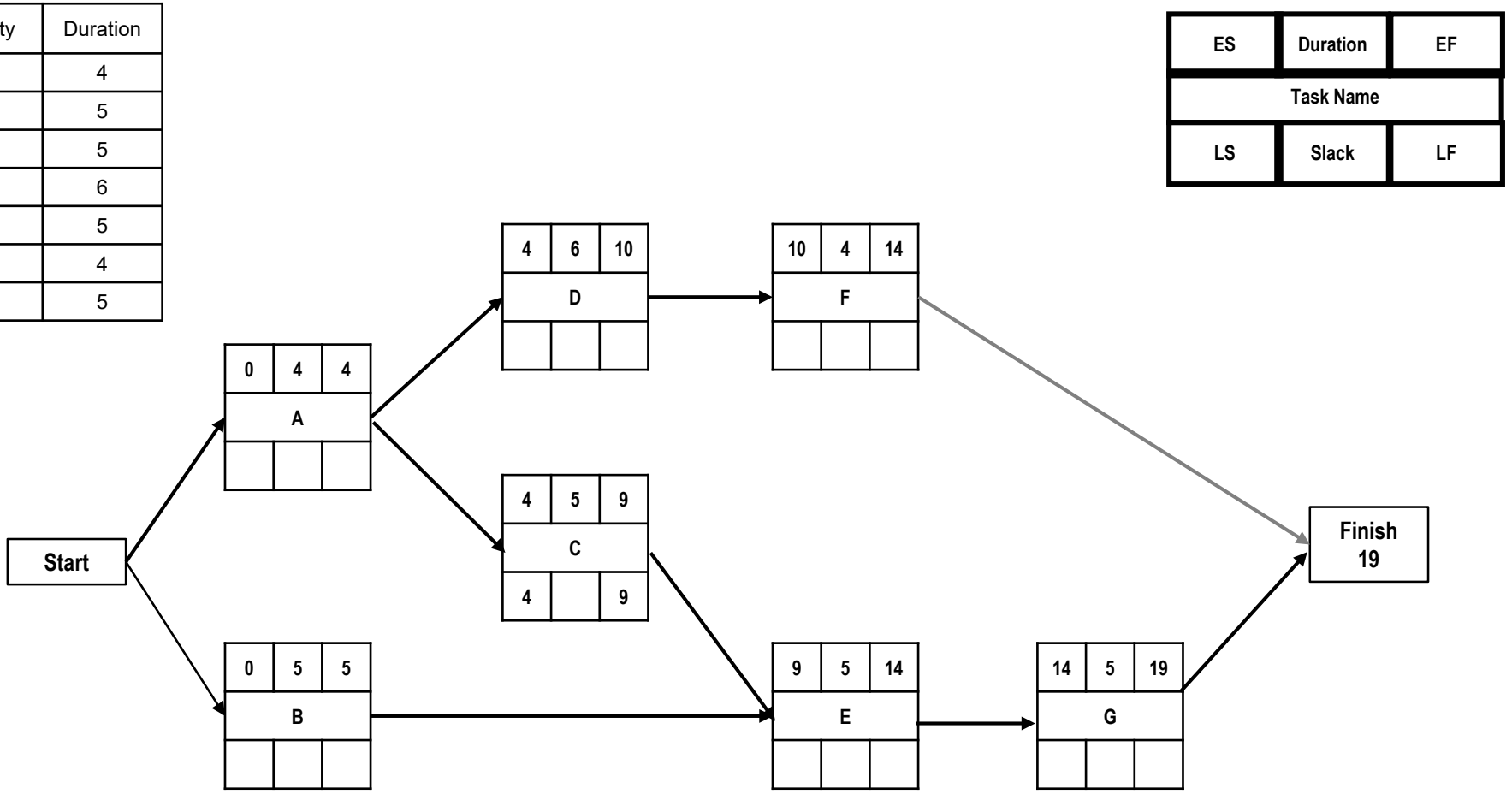


Task Network Diagram



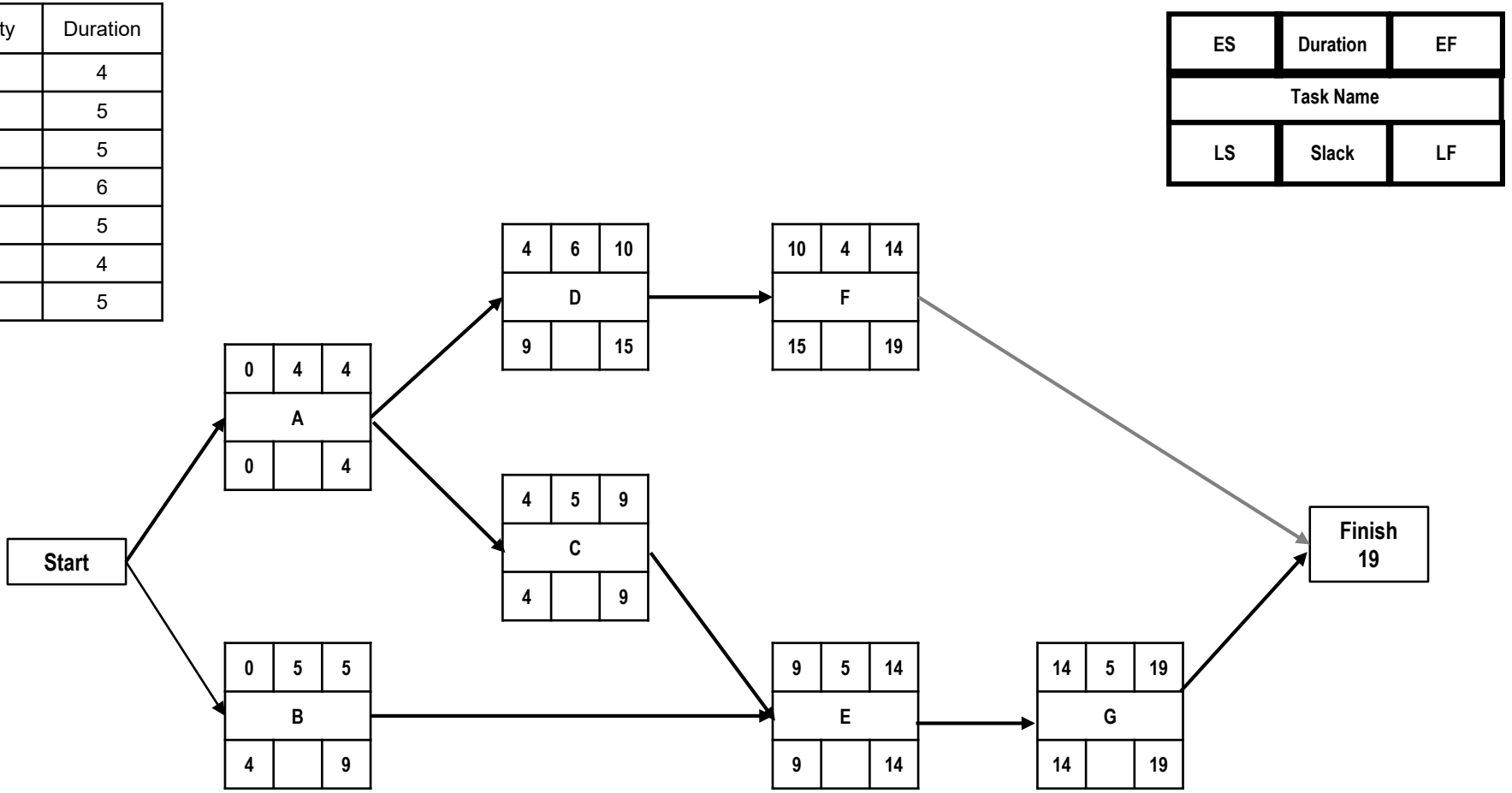
Forward Pass

Activity	Duration
A	4
B	5
C	5
D	6
E	5
F	4
G	5



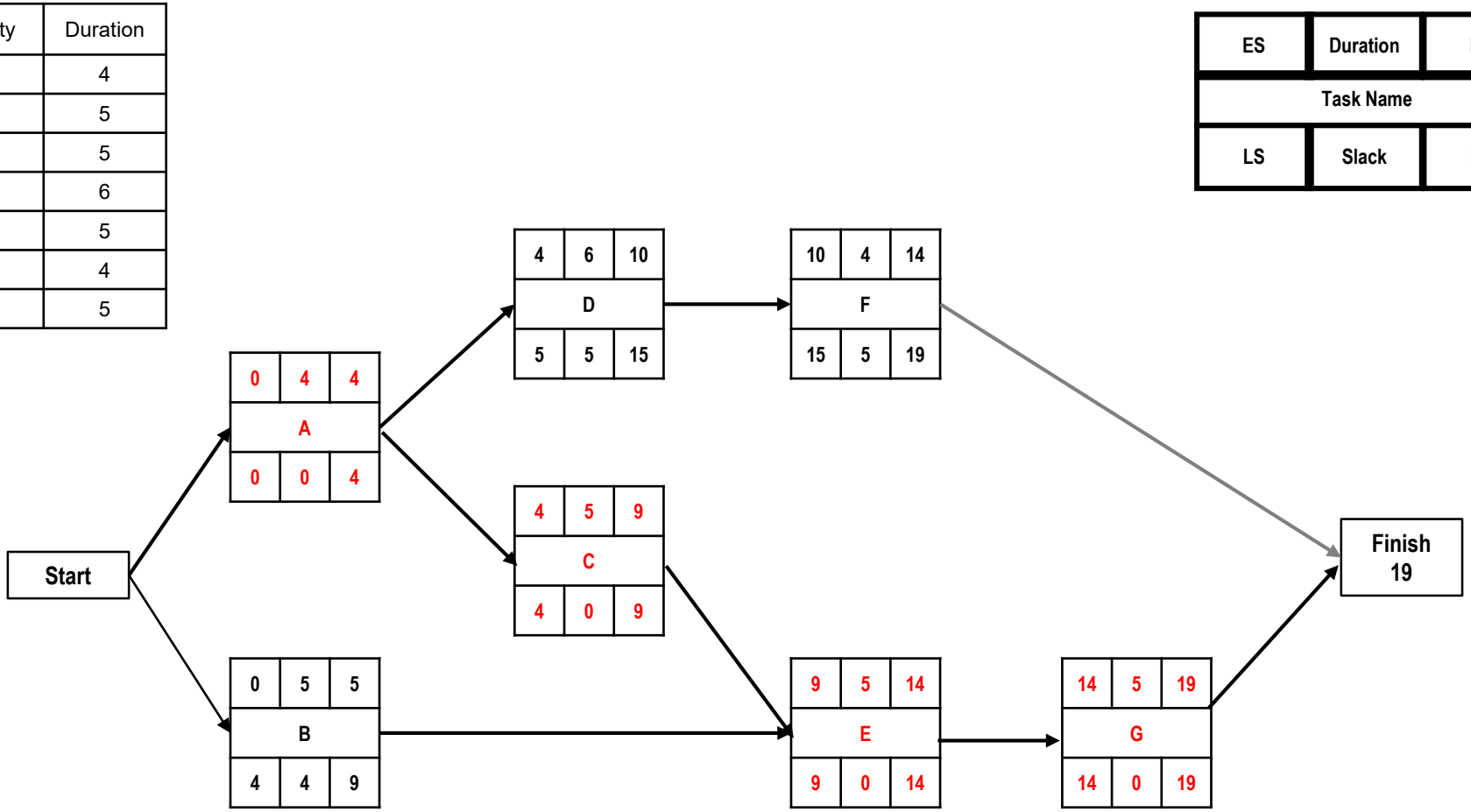
Backward Pass

Activity	Duration
A	4
B	5
C	5
D	6
E	5
F	4
G	5



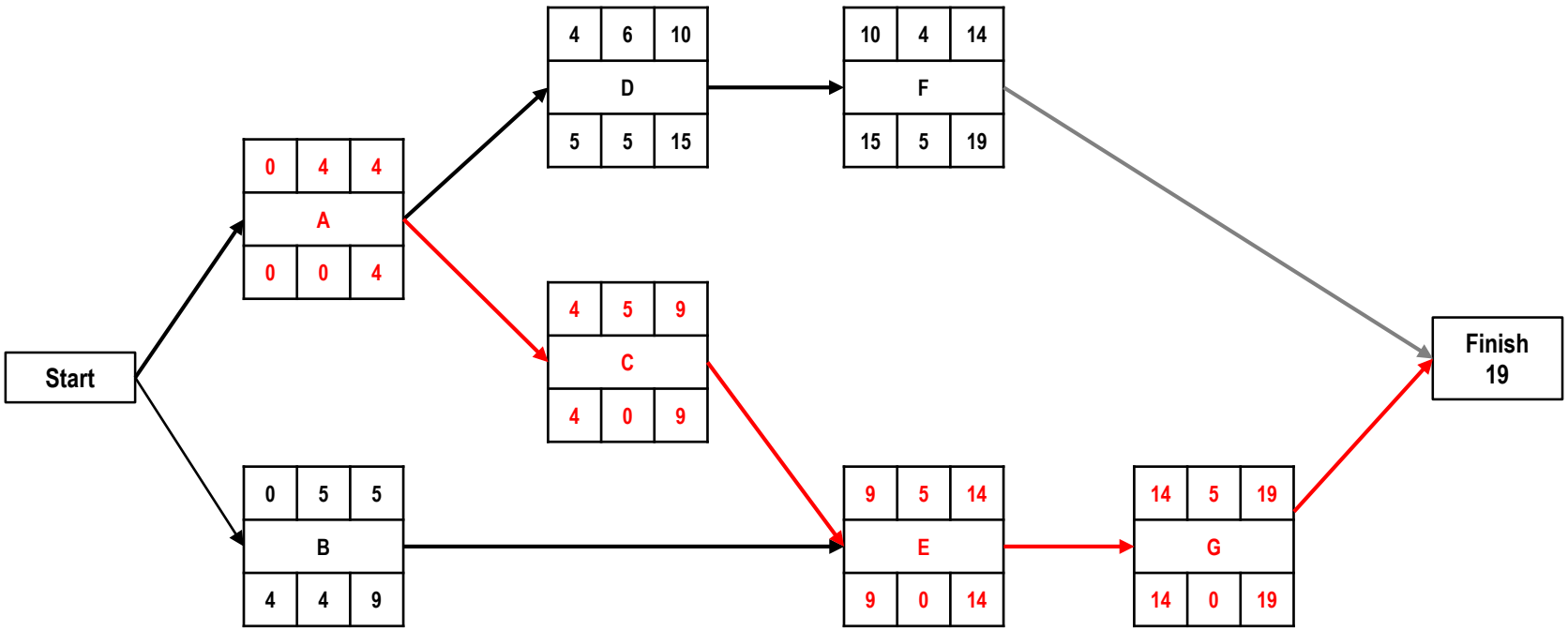
Slack

Activity	Duration
A	4
B	5
C	5
D	6
E	5
F	4
G	5



Critical Path

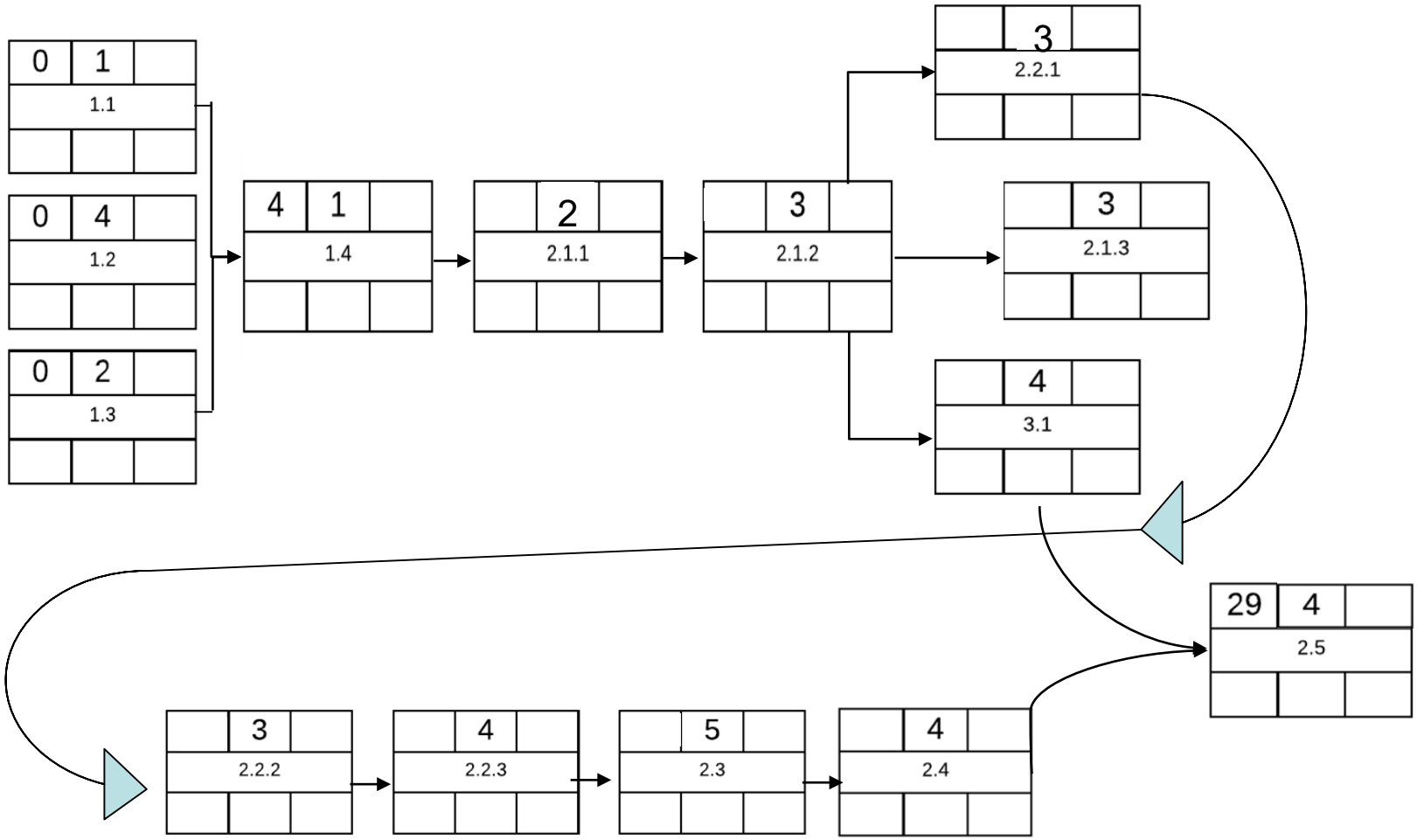
Critical Path = A + C + E + G



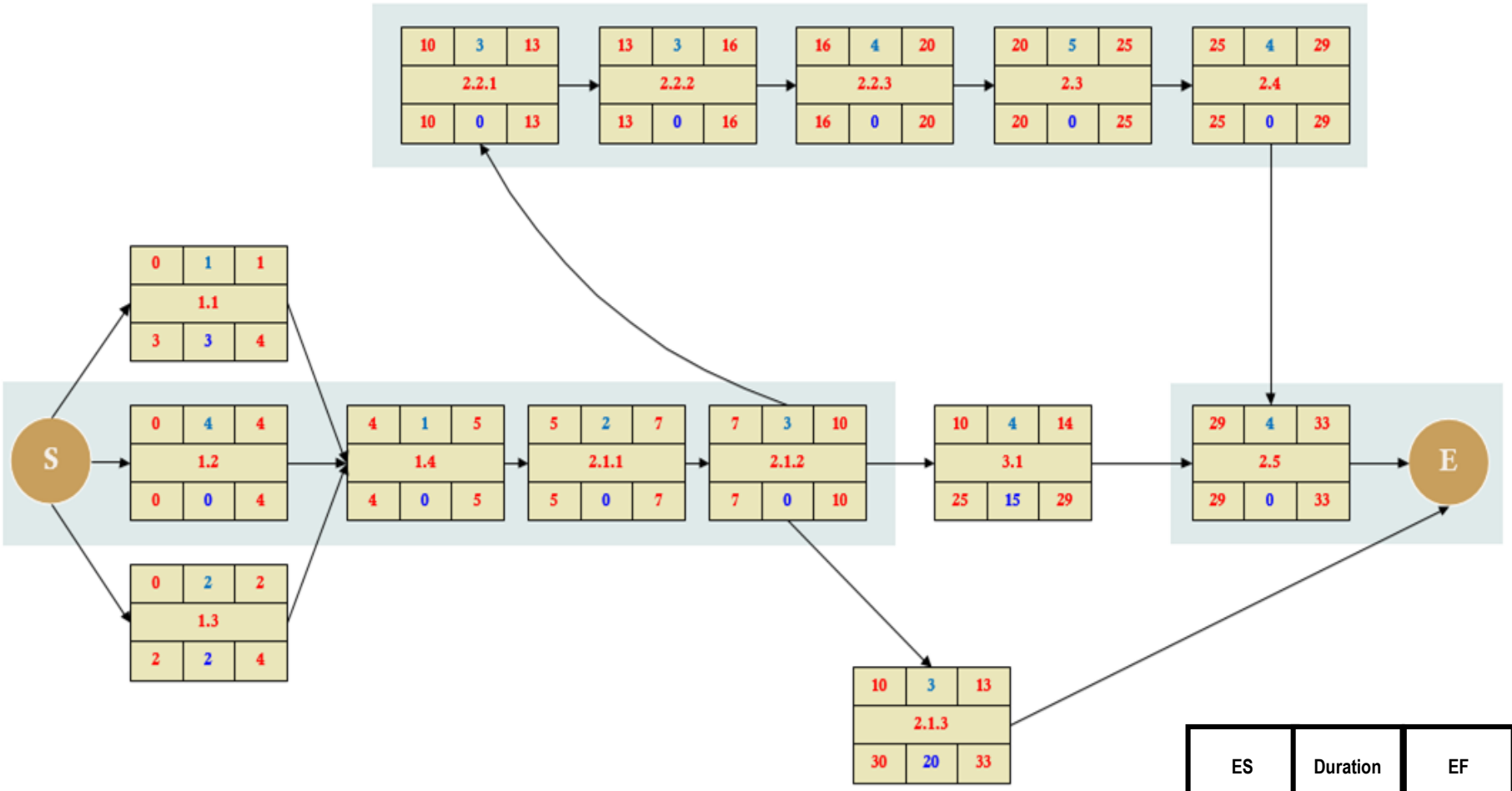
PERT Chart: activity

Use duration estimates & task network to construct PERT chart

activ	durn
1.1	1
1.2	4
1.3	2
1.4	1
2.1.1	2
2.1.2	3
2.1.3	3
2.2.1	3
2.2.2	3
2.2.3	4
2.3	5
2.4	4
2.5	4
3.1	4



PERT Chart

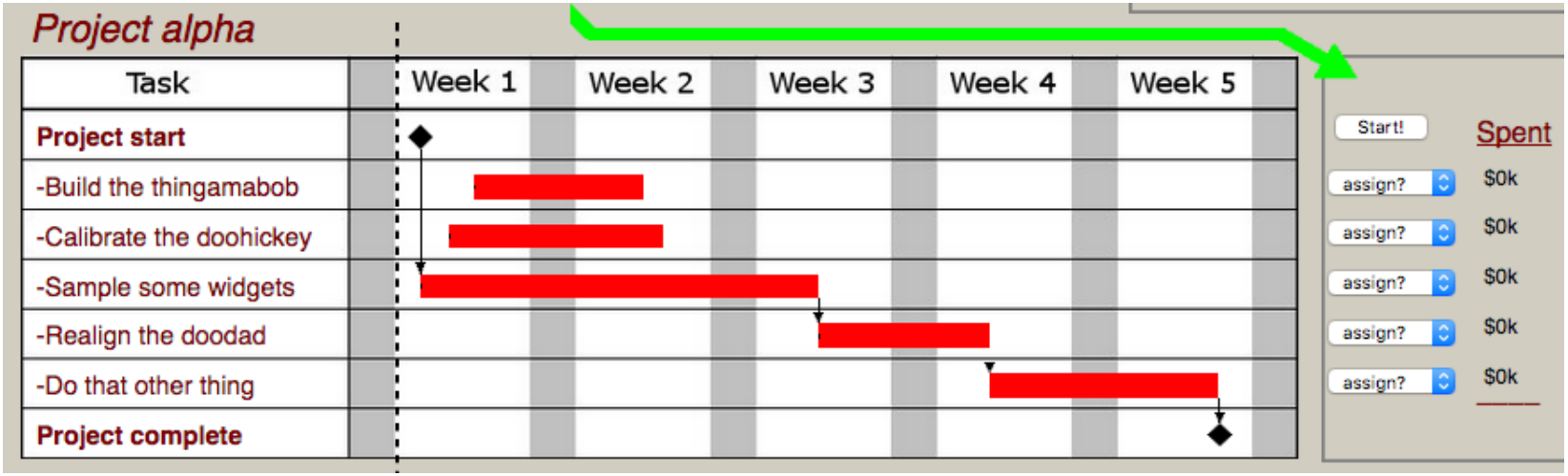


Critical Path = 1.2 + 1.4 + 2.1.1 + 2.1.2 + 2.2.1 + 2.2.2 + 2.2.3 + 2.3 + 2.4 + 2.5

Construct Resource Schedule

Play the Project Management Game:

http://thatpmgame.com/

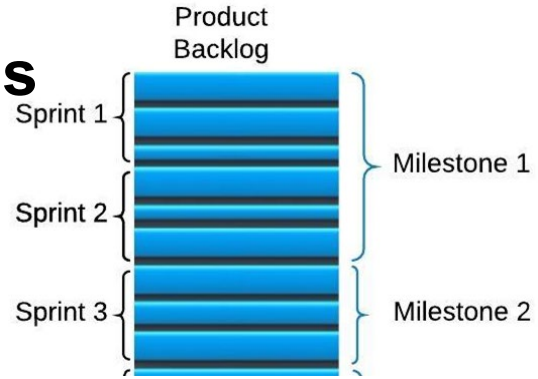


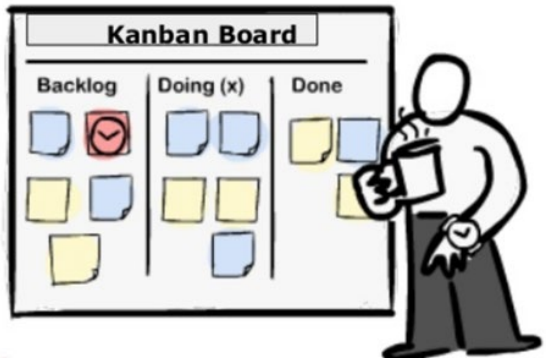
Use a Gantt chart to assign staff to various tasks.

Is the project completed on time and on budget?

Scrum Project Scheduling

1. Product Backlog with milestones





2. Sprint Backlog on Kanban board

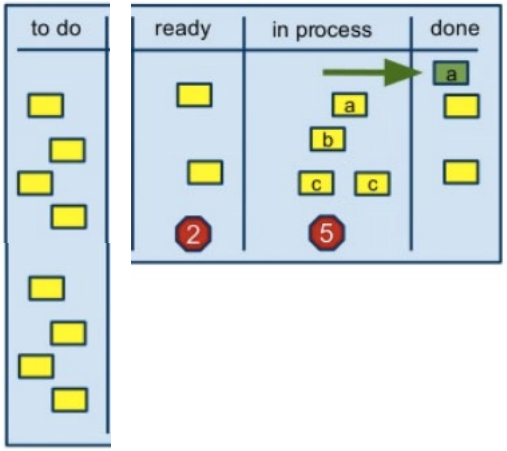
3. Burndown Charts

how to plan the schedule

Velocity and Visual Board

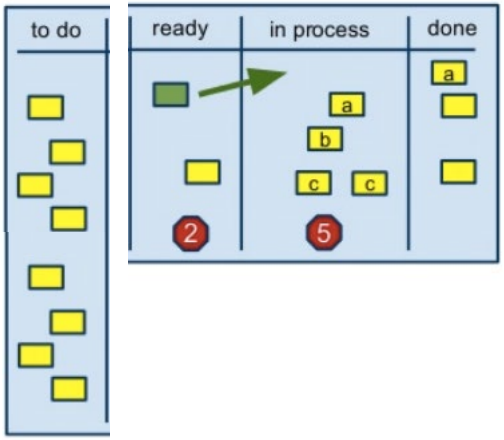
How many User Stories are ***"done"*** over the time-boxed Sprint?

- Only count 100% complete stories
- Predict when the release milestones will be reached

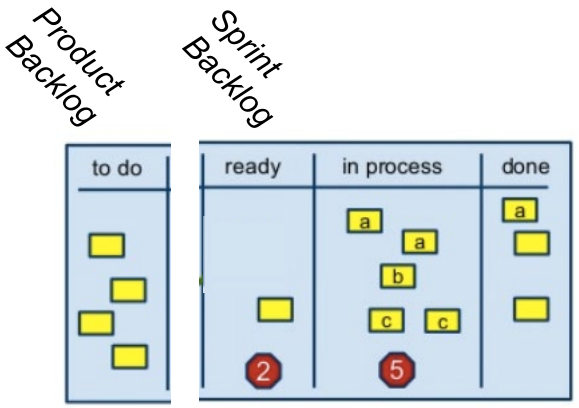


Team member A completes code for a card and moves it to "done"

Swimlane Board



Team member A "pulls" a new card from "ready" and moves it to "doing"

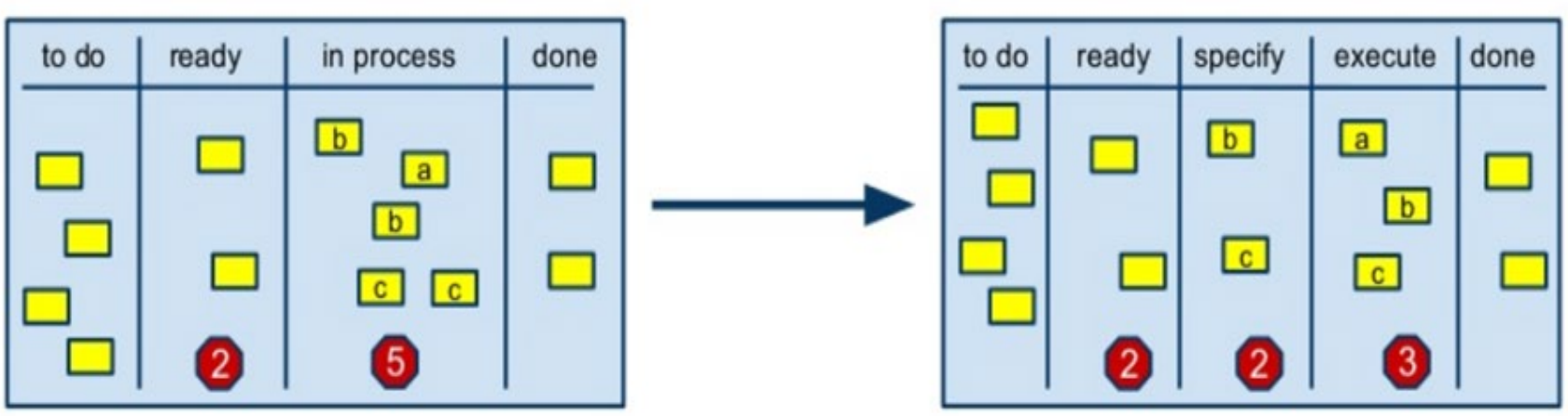


The Product Owner selects the next priority set of cards (Sprint Backlog) and moves it to "ready"

Agile Scrum Velocity

Velocity determines when dev team can deliver

- Dev team velocity emerges over a number of Sprints
- Predict when the release milestones will be reached

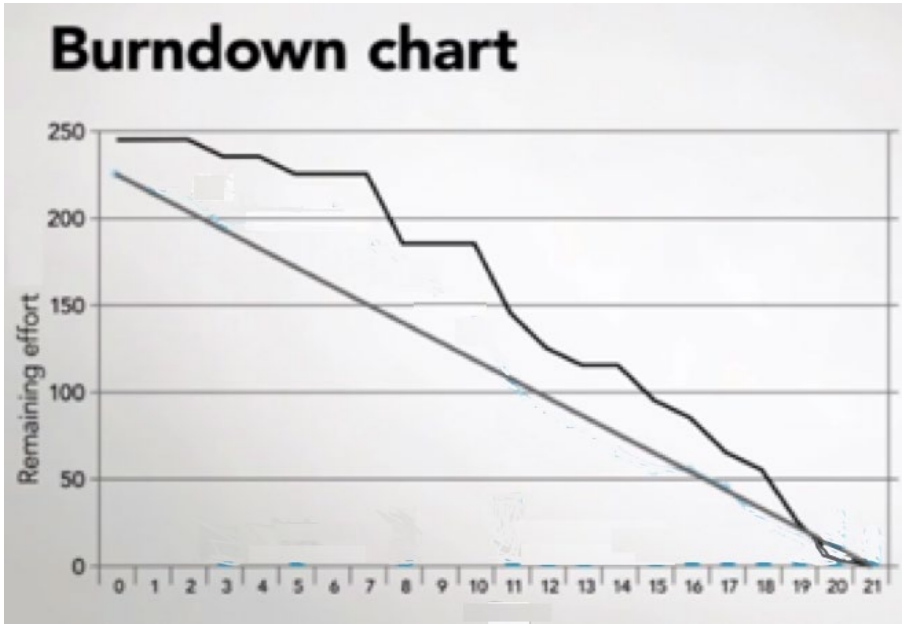


Agile Scrum Velocity

Velocity determines the slope of the BurnDown charts

- The Scrum master can track remaining effort
- Predict when the release milestones will be reached

Y-axis: effort



X-axis: time

Thank You!