SWEN90016
Software Processes & Project Management

Project Scheduling

Shanika Karunasekera
Department of Computing and Information Systems
The University of Melbourne
karus@unimelb.edu.au

2019 – Semester 1
Lecture 5

A bit about myself

- Shanika Karunasekera:
  - Professor in the Department of Computing and Information Systems – Leader of the Software Engineering Discipline

- Education:
  - B. Sc. (First Class Honours) in Electronic and Telecommunication Engineering - University of Moratuwa, Sri Lanka
  - PhD in Electrical Engineering (Specialization: Image Processing) - University of Cambridge, UK

A bit about myself

- Industry Experience:
  - – Distinguished Engineer, Software Architect (1995 – 2002) - Lucent Technologies, Bell Labs Innovation (AT&T Bell Labs), USA

- Academic Experience:
  - – Academic in the department from 2003 – to date
  - – Teaching
    - Software Engineering and Distributed Computing
  - – Research Interests
    - Bigdata analytics
    - Distributed systems
    - Data stream mining

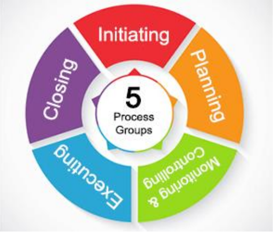
RECAP

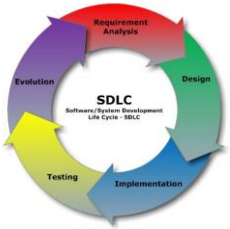
PROJECT

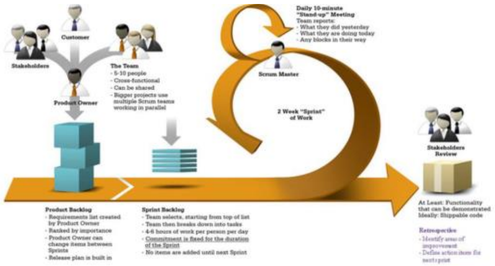
A temporary endeavour to create a unique product, service or outcome.

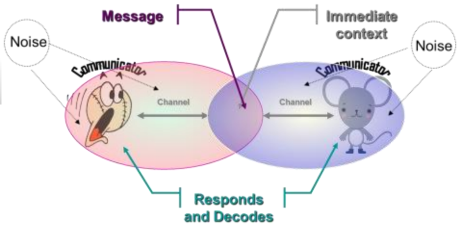
- Introduce **CHANGE** to the organization
- **TEMPORARY -** defined beginning and end
- **CROSS-FUNCTIONAL**
- Deals with the **UNKNOWN**
- **UNIQUE**
- They all vary in **SIZE**– 🧍/ 🧍 , $’s and 🕒

PROCESSES






Customer
Stakeholders
Product Owner
The Team
- 5-10 people
- Cross-functional
- Can be shared
- Bigger projects use multiple Scrum teams working in parallel
Daily 10-minute "Stand-up" Meeting
Team reports:
- What they did yesterday
- What they are doing today
- Any blocks in their way
Scrum Master
2 Week "Sprint" of Work
Stakeholders Review
Product Backlog
- Requirements list created by Product Owner
- Ranked by importance
- Product Owner can change items between Sprints
- Release plan is built in
Sprint Backlog
- Team selects, starting from top of list
- Team then breaks down into tasks
- 4-6 hours of work per person per day
- Commitment is fixed for the duration of the Sprint
- No items are added until next Sprint
At Least: Functionality that can be demonstrated
Ideally: Shippable code
Retrospective
- Identify area of improvement
- Define action items for next sprint



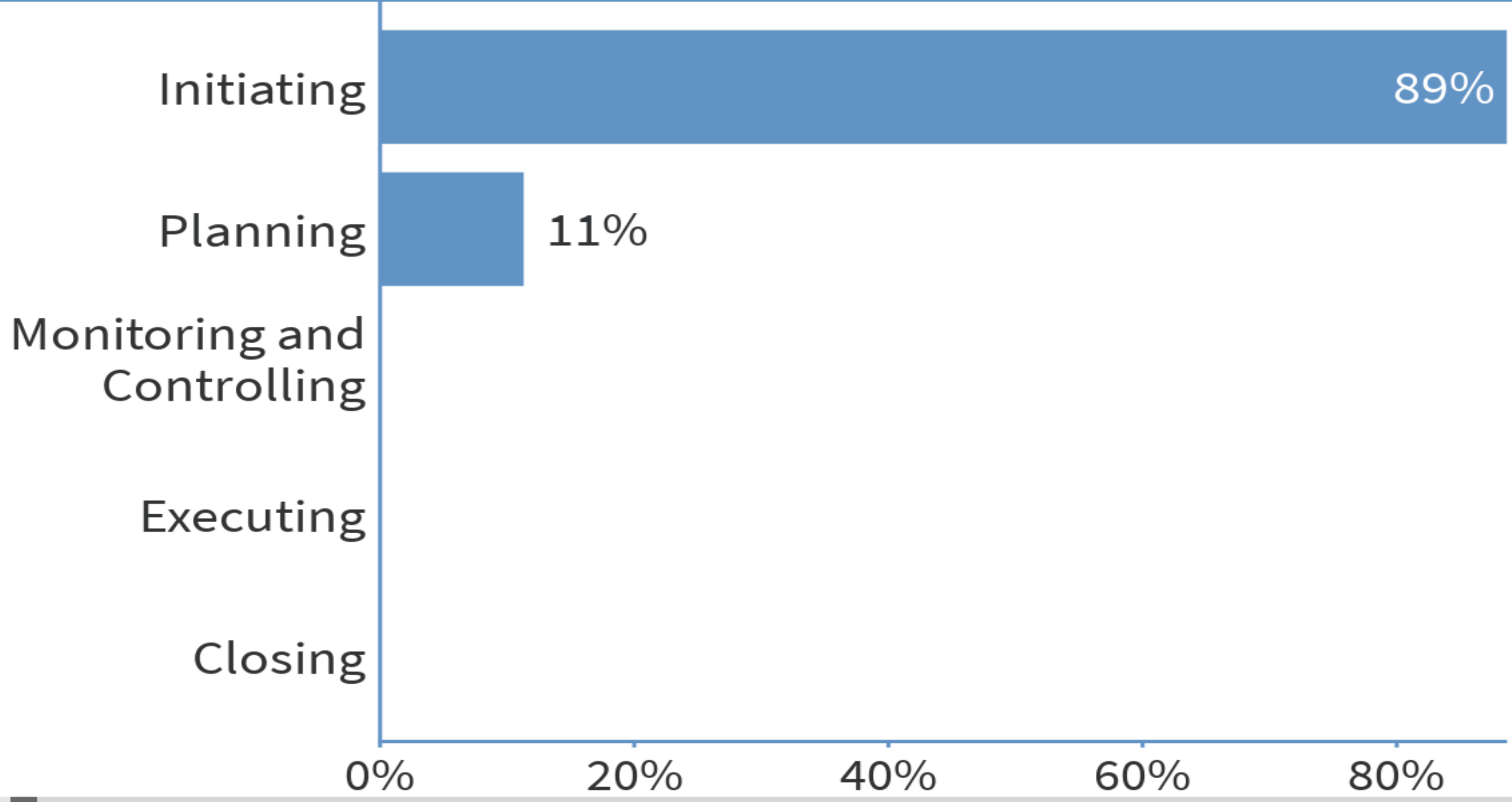
PEOPLE

- **Individuals**
- **Teams**
- **Communication**

What we are going to learn?

How to combine these ingredients: the project, the processes and the people to ***plan, execute**, **monitor** and **control** a project*.

In what stage of the project management life cycle is the Project Charter developed?



Start the presentation to see live content. Still no live content? Install the app or get help at **PollEv.com/app**

Project Statistics from Lecture 1

History tells us we have failed.

	ALL IT PROJECTS				
	2011	**2012**	**2013**	**2014**	**2015**
Successful	29%	27%	31%	28%	29%
Challenged	49%	56%	50%	55%	52%
Failed	22%	17%	19%	17%	19%

- ***Successful:*** project is completed on-time and on-budget, with all features and functions as initially specified.
- ***Challenged:*** completed and operational but over-budget, over the time estimate or offers fewer features and functions than planned.
- ***Failed:*** project is cancelled at some point during the development cycle.

Standish Group Chaos Reports: Source: Standish Group 2015 Chaos Report www.projectsmart.co.uk/white-papers/chaos-report.pdf

Challenged Projects – why?

1. Lack of a Scope Document
   - Changing scope and requirements is one of the main reasons for project failure; making a detailed scope document that highlights all the stakeholders' requirements is imperative for successful project delivery

2. Inconsistent Communication
   - 57% of projects failed due to poor communication
   - Having a good communication plan up front is critical

3. Unrealistic Expectations and Deadlines
   - 60% of failed projects have a deadline of less than a year

4. Incompetent Project Manger and Team
   - 80% of successful projects are managed by certified project managers

Challenged projects – why?

5. Lack of cohesion between team members
   - Team members should have the same goals and must move towards these goals

6. Poor Monitoring and Risk Management
   - Many projects fail due to not paying enough emphasis on risk and managing them

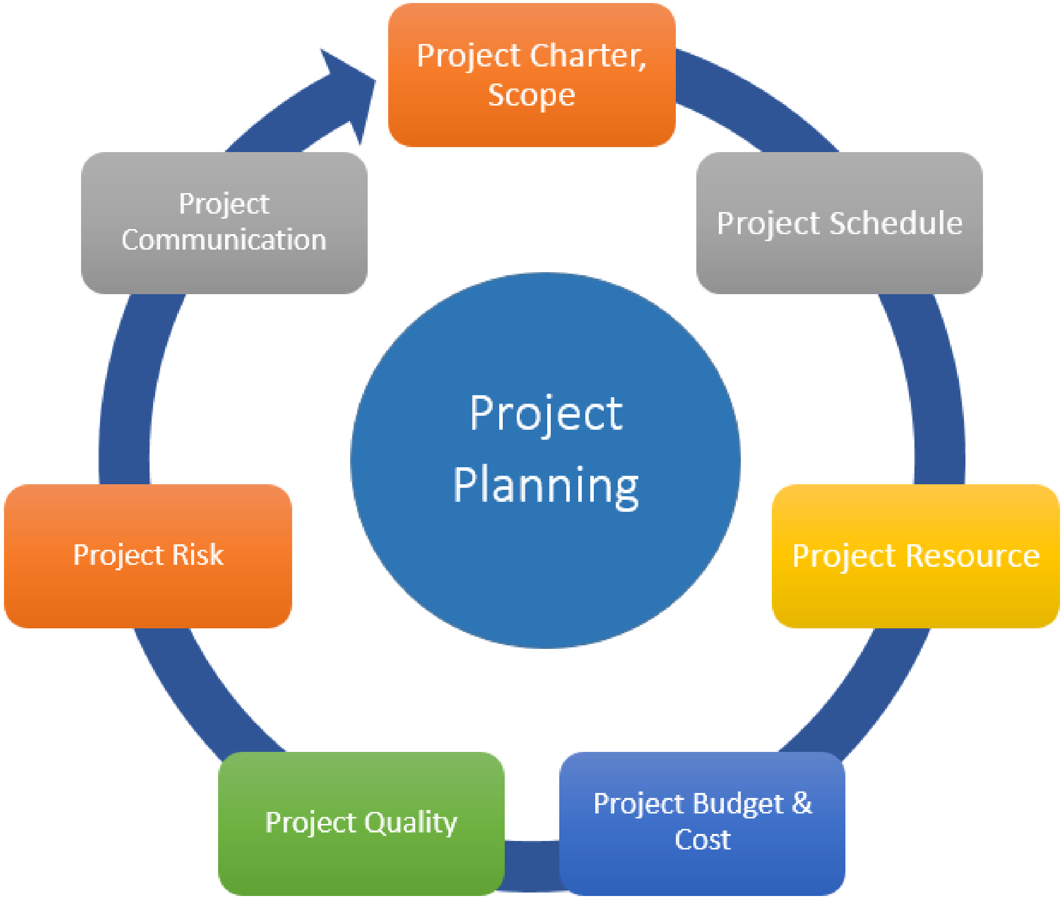
7. Poor Planning
   - 40% of projects fail due to poor planning and lack of resources

   Every minute you spend in planning saves 10 minutes in execution; this gives you a 100% return on energy!

http://www.it-cortex.com/Stat_Failure_Cause.htm

https://blog.taskque.com/causes-project-failure/

Project Planning

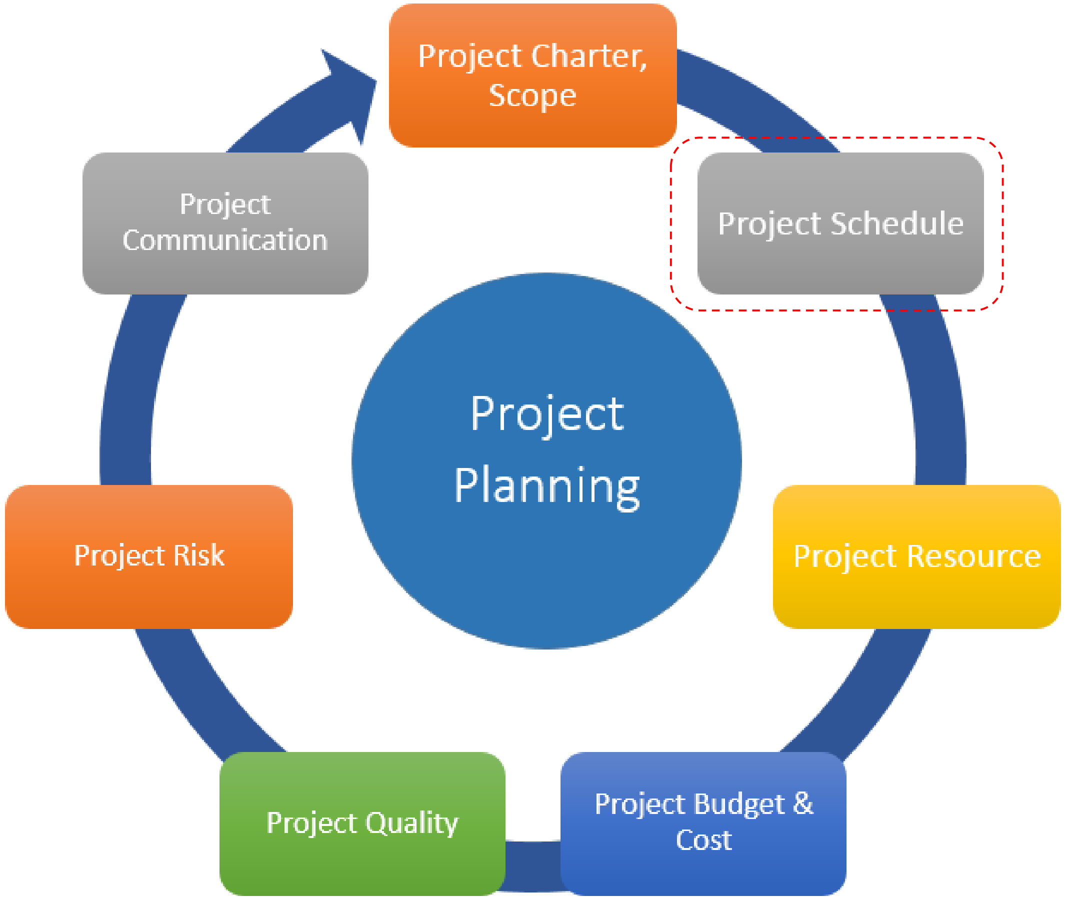


http://blog.zilicus.com/software-project-management-activities-roles/

Project Planning

- Project Management begins with a set of activities - collectively called *Project Planning*
  - Project Scheduling
  - Cost Estimation
  - Risk Management
  - Quality Management
  - Configuration Management (Change Management)
  - Resource Management
  - Communication Management

Project Planning



http://blog.zilicus.com/software-project-management-activities-roles/

THE UNIVERSITY OF MELBOURNE

Semester Structure

Week #	Lecture Week Start	Old Arts Public Lecture Theatre Friday 3.15pm to 5.15pm	Assignment
1	29/07/19	Subject Introduction, Introduction to Projects and Project Management	
2	05/08/19	Project Management Plan & SDLC's	Assignment 1 Spec available on LMS 05/08
3	12/08/19	SDLC - Agile Scrum – continued Individuals, Motivation and Teams	
4	19/08/19	Stakeholder Management Communication Management	Assignment 2 available & Groups created during the workshops / tutorials – attendance mandatory
5	26/08/19	Project Planning and Scheduling	Assignment 1 (Individual) due Fri 31/08 @ 11.59 pm
6	02/09/19	Cost Estimation	
7	09/09/19	Risk Management	
8	16/09/19	Quality Management/Configuration Management	Assignment 2 (Part 1) due Wed 18/09 @ 11.59 pm
9	***23/09/19***	***University Holiday***	
	30/09/19	***Non Teaching Week – Mid semester break***	Assignment 2 (Part 2) due Sat 28/09 @ 11.59 pm
10	07/10/19	Ethics, Outsourcing & Procurement	Assignment 2 (Part 3) due Sat 12/10 @ 11.59 pm
11	14/10/19	Guest Lecture	Assignment 2 (Final) due Sat 19/10 @ 11.59 pm
12	21/10/19	Subject Revision and Exam Prep	Assignment 2 Project Demonstration during tutorials

Intended Learning Outcomes

1. Understand the role of a project schedule

2. Understand how to develop a project schedule

3. Understand how to use a project schedule to monitor and track project progress

4. Understand agile planning principles

Intended Learning Outcomes

1. Understand the role of a project schedule
2. Understand how to develop a project schedule
3. Understand how to use a project schedule to monitor and track project progress
4. Understand agile planning principles

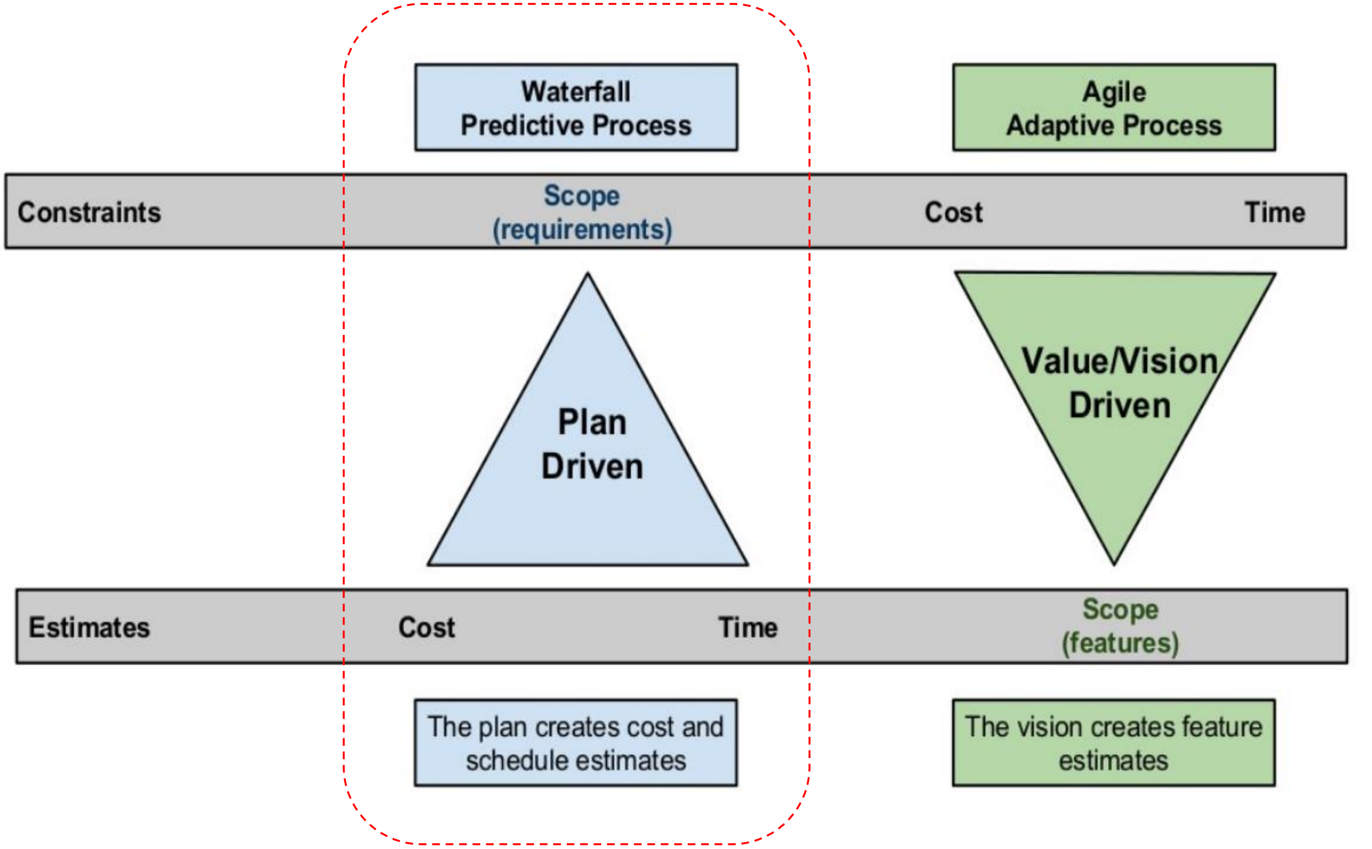
Project Schedule

- **Project Schedule:**
  - One of the important artefacts generated during the project planning phase
  - Is used and maintained throughout the project to monitor and track project progress - is a living document

- **What does the project schedule contain?**
  - Duration and dependencies for each task
  - People and physical resources required by each task
  - Milestones and deliverables
  - Project Timeline

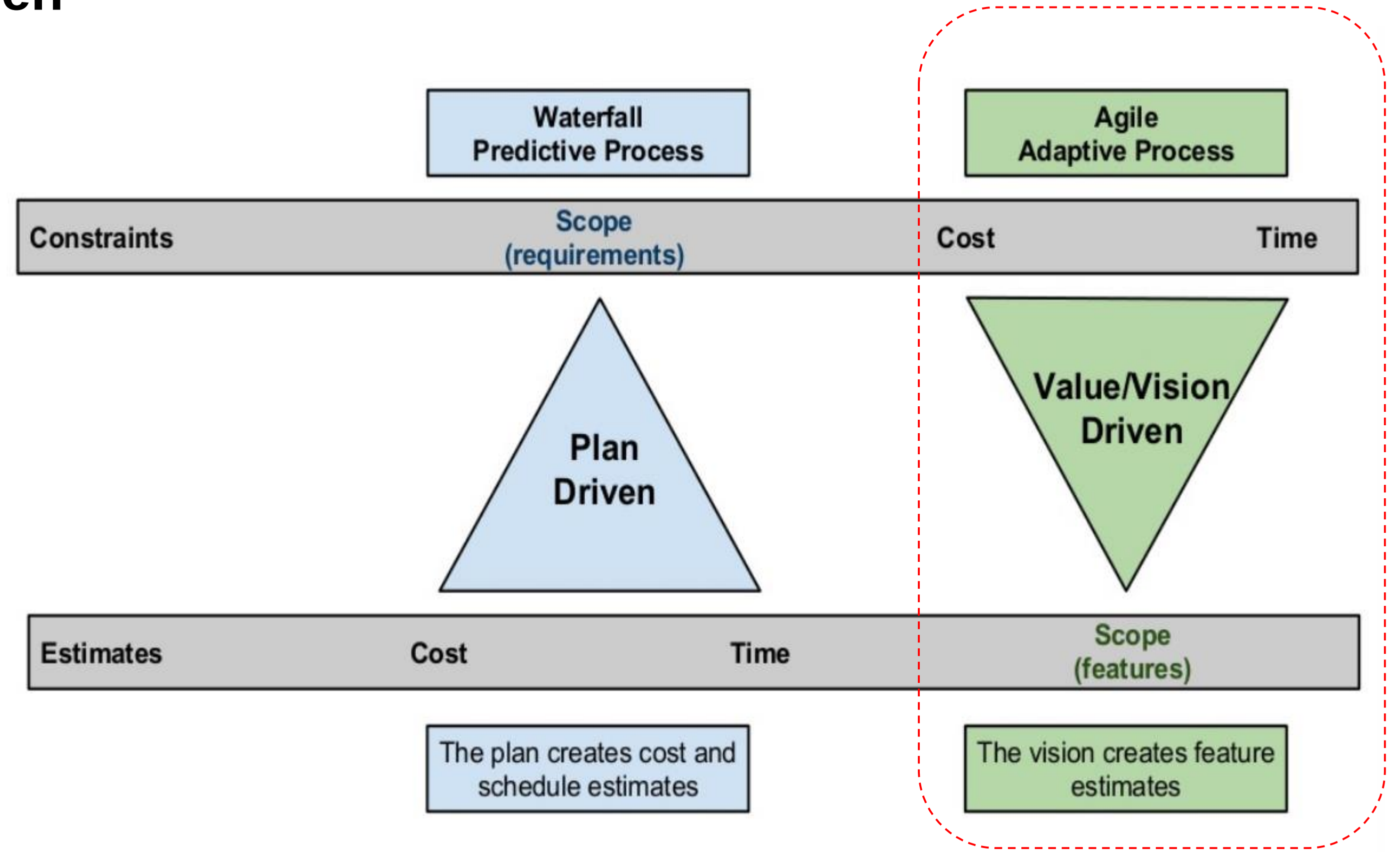
Project Schedule

Project planning and scheduling introduced in this topic apply to formal SDLC processes – Plan Driven

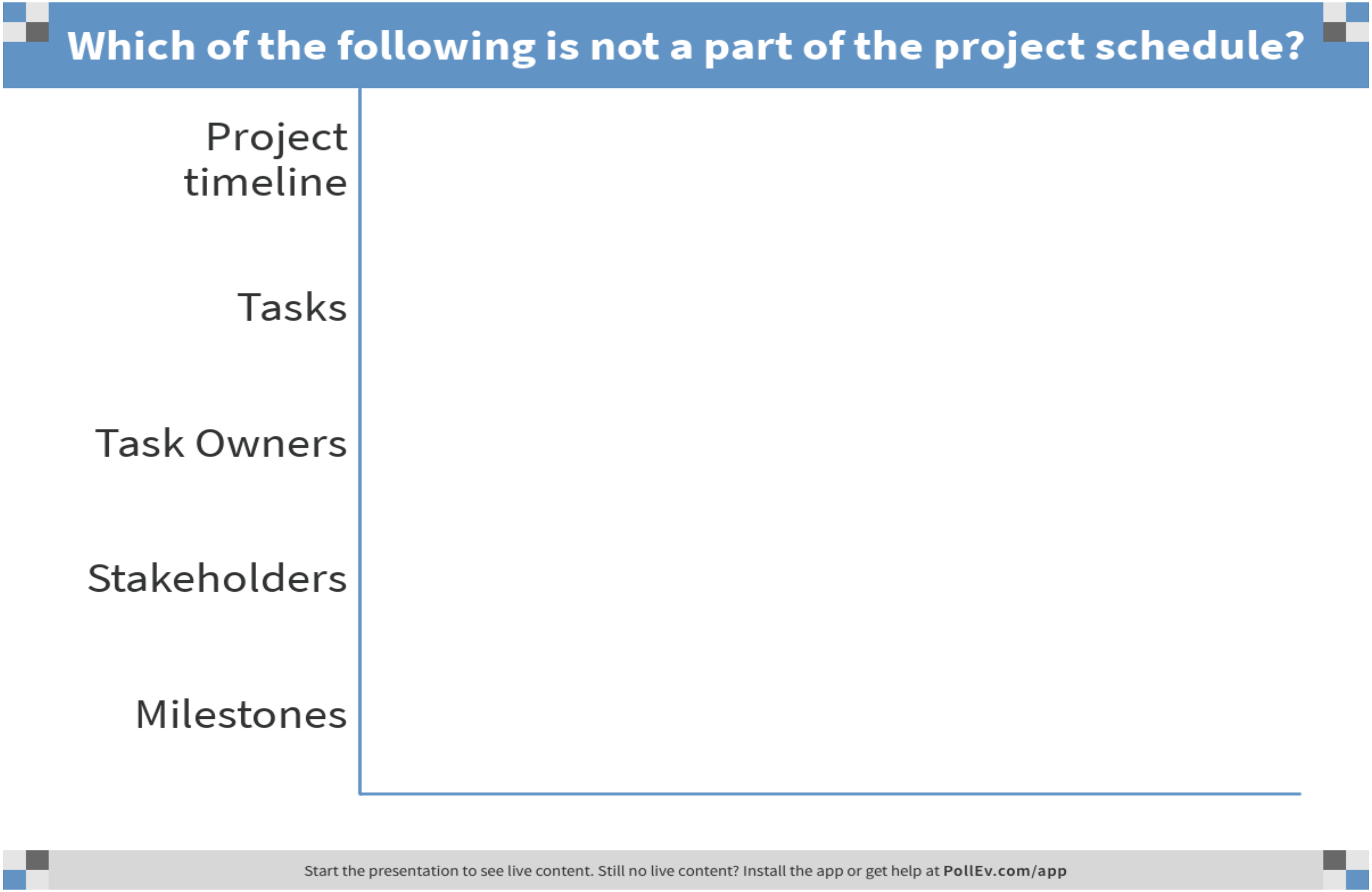


Project Schedule

Agile SDLC processes do not use a project schedule - Value/Vision Driven



Anecdotally organizations that use Agile practices also use project schedules for budgeting, contracting and reporting purposes.



Which of the following is not a part of the project schedule?

Project timeline

Tasks

Task Owners

Stakeholders

Milestones

Start the presentation to see live content. Still no live content? Install the app or get help at **PollEv.com/app**

Intended Learning Outcomes

1. Understand the role of a project schedule

2. Understand how to develop a project schedule

3. Understand how to use a project schedule to monitor and track project progress

4. Understand agile planning principles

Developing Project Schedule - Steps

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

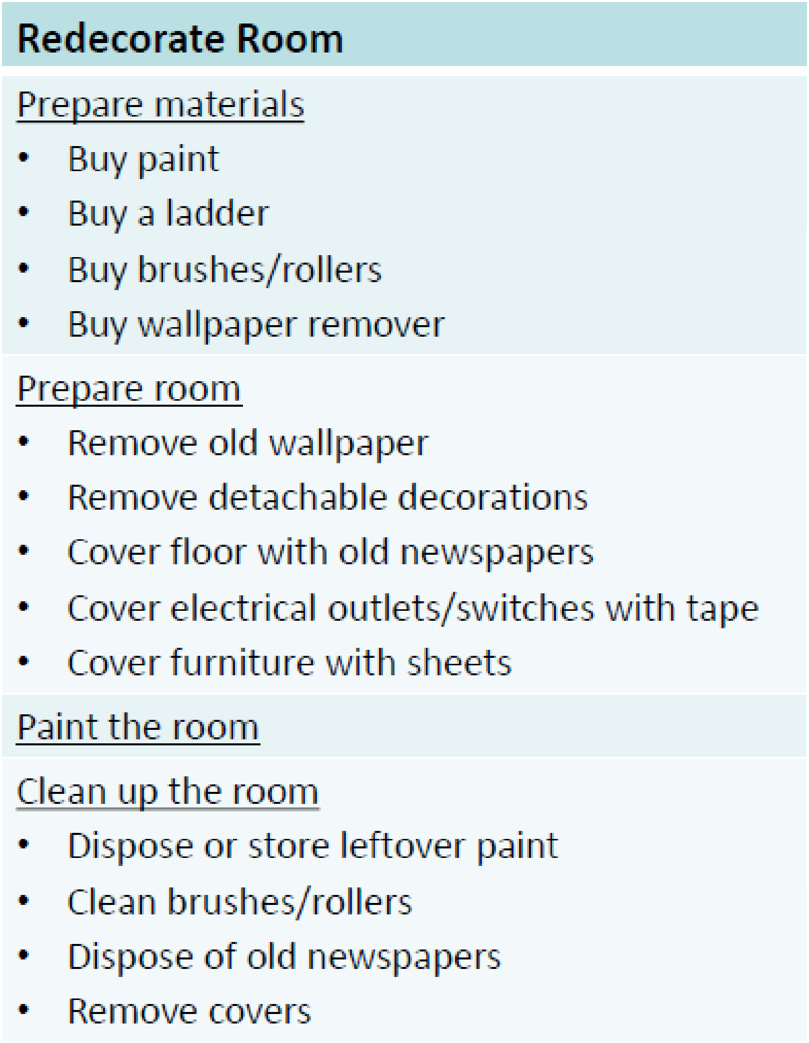
5. Develop the project schedule

Work Breakdown Structure – Step 1

- Planning and executing large tasks is challenging:
  - Estimating the time and resources
  - Identifying interim goals and deliverable
  - Progress monitoring

- Solution is to break the task down to manageable units:
  - Each task should have a specific outcome or a deliverable
  - Results in a Work Breakdown Structure (WBS)

Example - WBS



Redecorate Room

Prepare materials

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

Prepare room

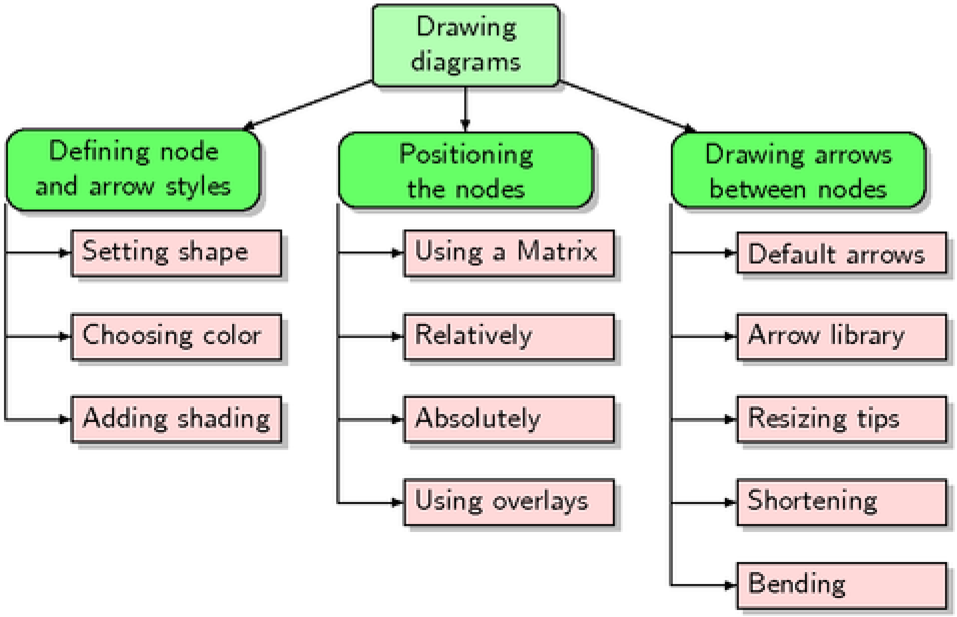
- Remove old wallpaper
- Remove detachable decorations
- Cover floor with old newspapers
- Cover electrical outlets/switches with tape
- Cover furniture with sheets

Paint the room

Clean up the room

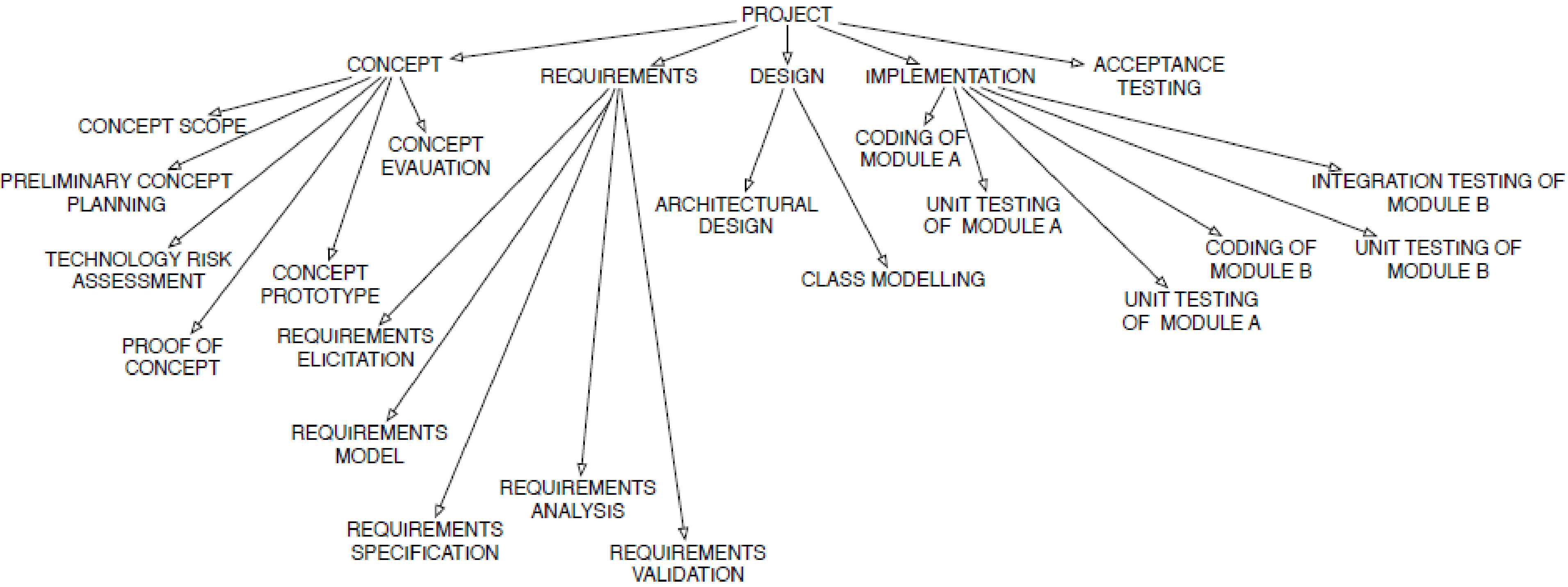
- Dispose or store leftover paint
- Clean brushes/rollers
- Dispose of old newspapers
- Remove covers

http://slideplayer.com/slide/5384158/



http://texample.net/tikz/examples/work-breakdown-structure/

Example – WBS (Software Project)



Developing a Project Plan

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 a project schedule

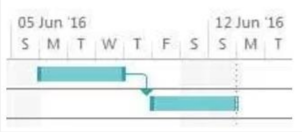
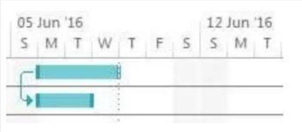
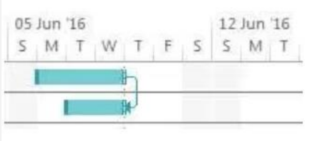
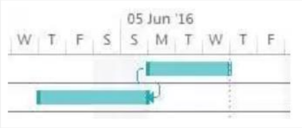
Identify Task Dependencies

- **Tasks can be:**
  - *Unconstrained:* the task can start at any time (buy paint, remove detachable decorations)
  - *Constrained:* depends on another task (cannot remove wall paper until decorations are removed)
    - If task **B** depends on task **A** (**A ->B)**
      - **B** is a Successor task (S)
      - **A** is a Predecessor task (P)
    - Remove Detachable Decorations (P) -> Remove wall paper (S)

- **Dependencies are caused by:**
  - a task needing a work product of another task
  - a task needing resources used by another task

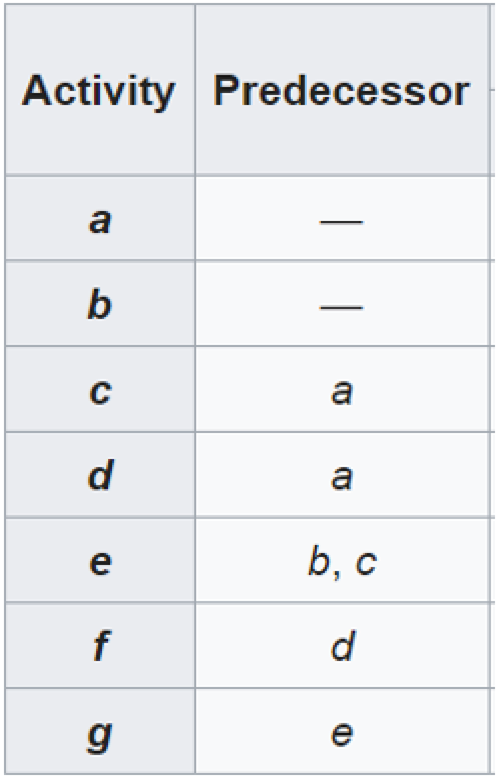
THE UNIVERSITY OF MELBOURNE

Types of Task Dependencies

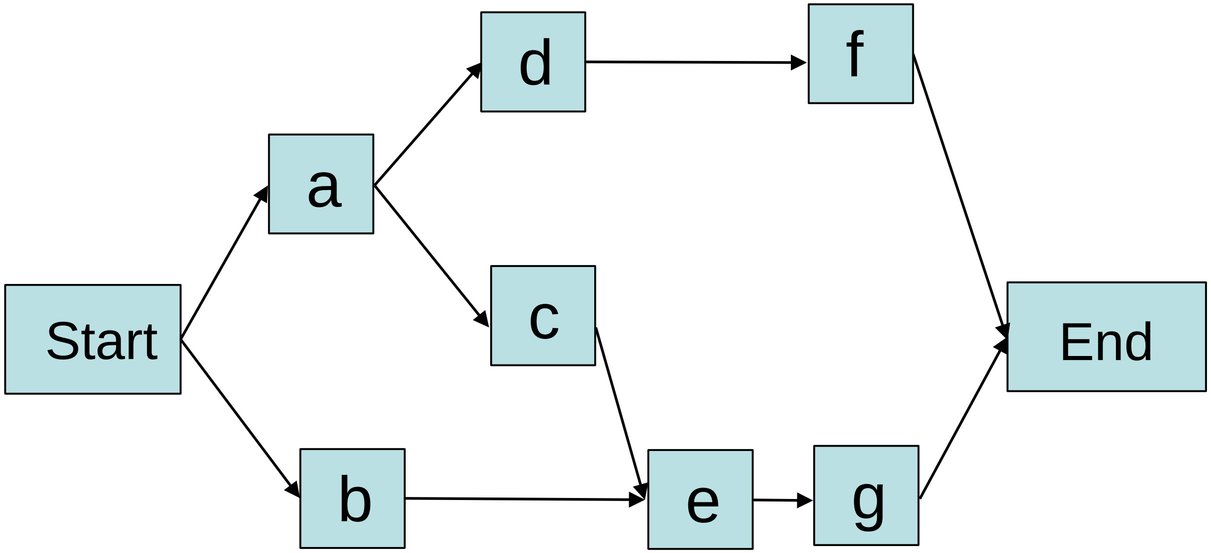
Dependency	Description	Representation
Finish-to-Start	Predecessor must finish before Successor can start	
Start-to-Start	Predecessor must start before Successor can start	
Finish-to-Finish	Predecessor must finish before the Successor can Finish	
Start-to-Finish	Predecessor must start before the Successor can finish	

The most common type of dependency is the finish-to-start dependency

Task Network



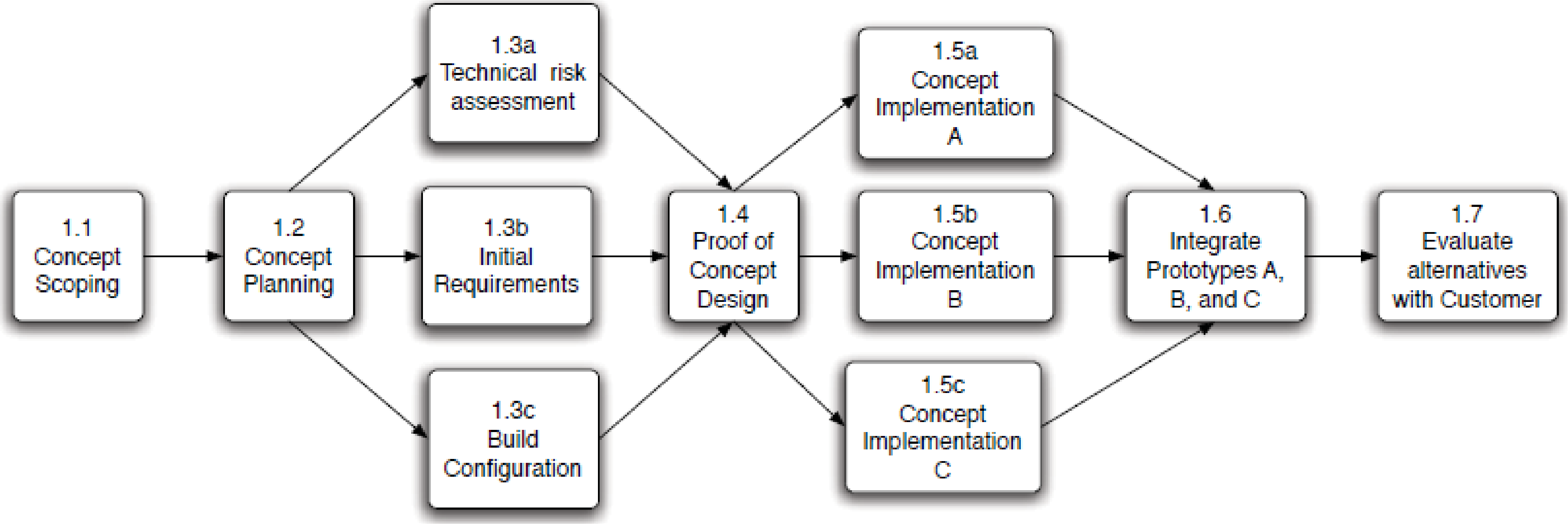
Activity	Predecessor
a	—
b	—
c	*a*
d	*a*
e	*b*, *c*
f	*d*
g	*e*

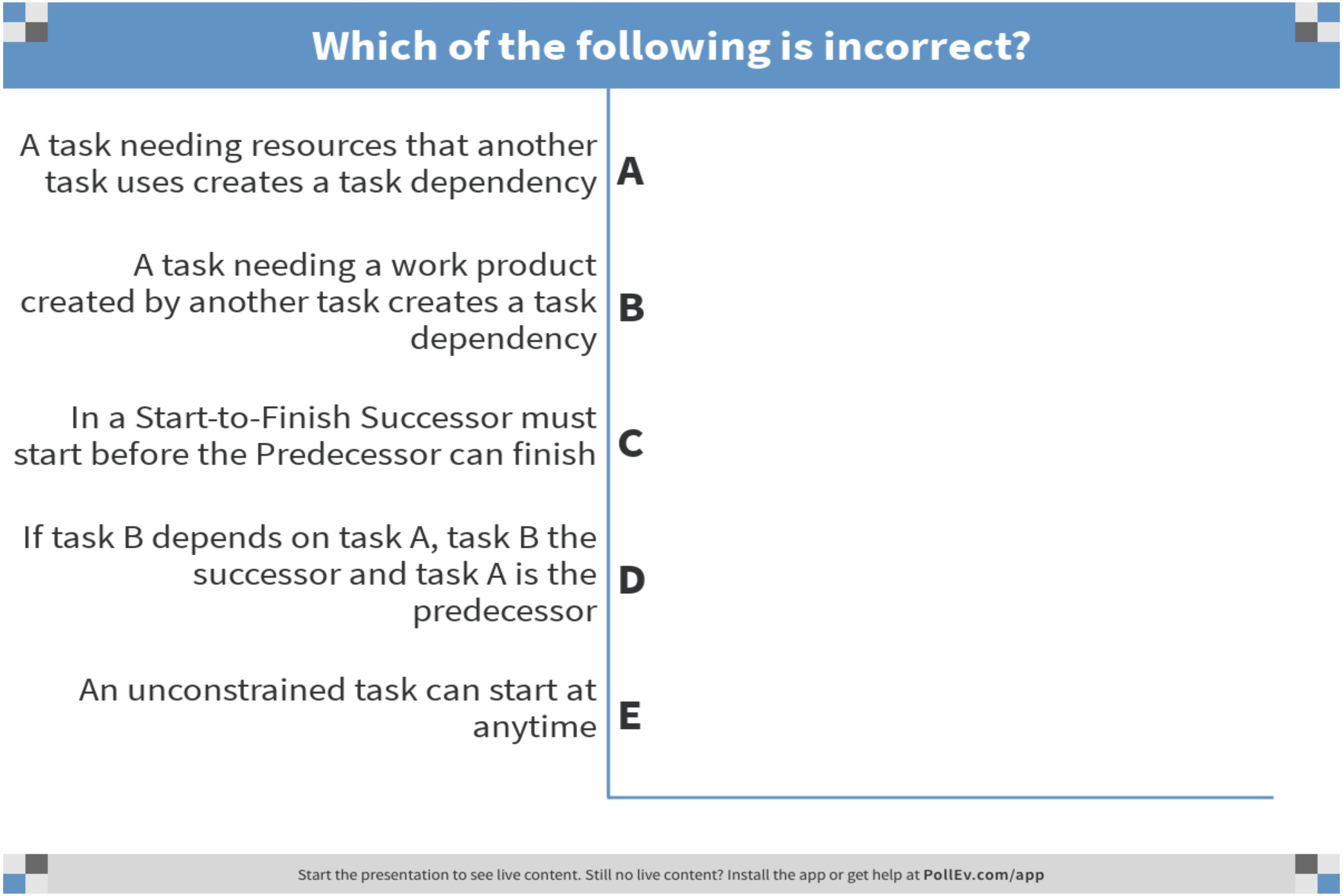


WBS – Software Project

1. Concept
    - 1.1 Concept Scope
    - 1.2 Preliminary Concept Planning
    - 1.3 Preliminary Analysis
        - 1.3a Technology Risk Assessment
        - 1.3b Initial Requirements
        - 1.3c Build Configuration
    - 1.4 Proof of Concept
    - 1.5 Concept Prototype
    - 1.6 Prototype Integration
    - 1.7 Concept Evaluation
2. Requirements
    - 2.1 Requirements Elicitation
    - 2.2 Requirements Prototype
    - 2.3 Requirements Analysis
    - 2.4 Requirements Specification
    - 2.5 Requirements Validation
3. Design
    - 3.1 Software Architecture Design
    - 3.2 Class Models
4. Implementation
    - 4.1 Coding the Client
    - 4.2 Testing the Client
    - 4.3 Coding the Server
    - 4.4 Testing the Server
    - 4.5 Integration Testing of Client with Server
5. Acceptance Testing

Task Network – Software Project





Which of the following is incorrect?

A task needing resources that another task uses creates a task dependency **A**

A task needing a work product created by another task creates a task dependency **B**

In a Start-to-Finish Successor must start before the Predecessor can finish **C**

If task B depends on task A, task B the successor and task A is the predecessor **D**

An unconstrained task can start at anytime **E**

Start the presentation to see live content. Still no live content? Install the app or get help at **PollEv.com/app**

Developing a Project Plan

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 a project schedule

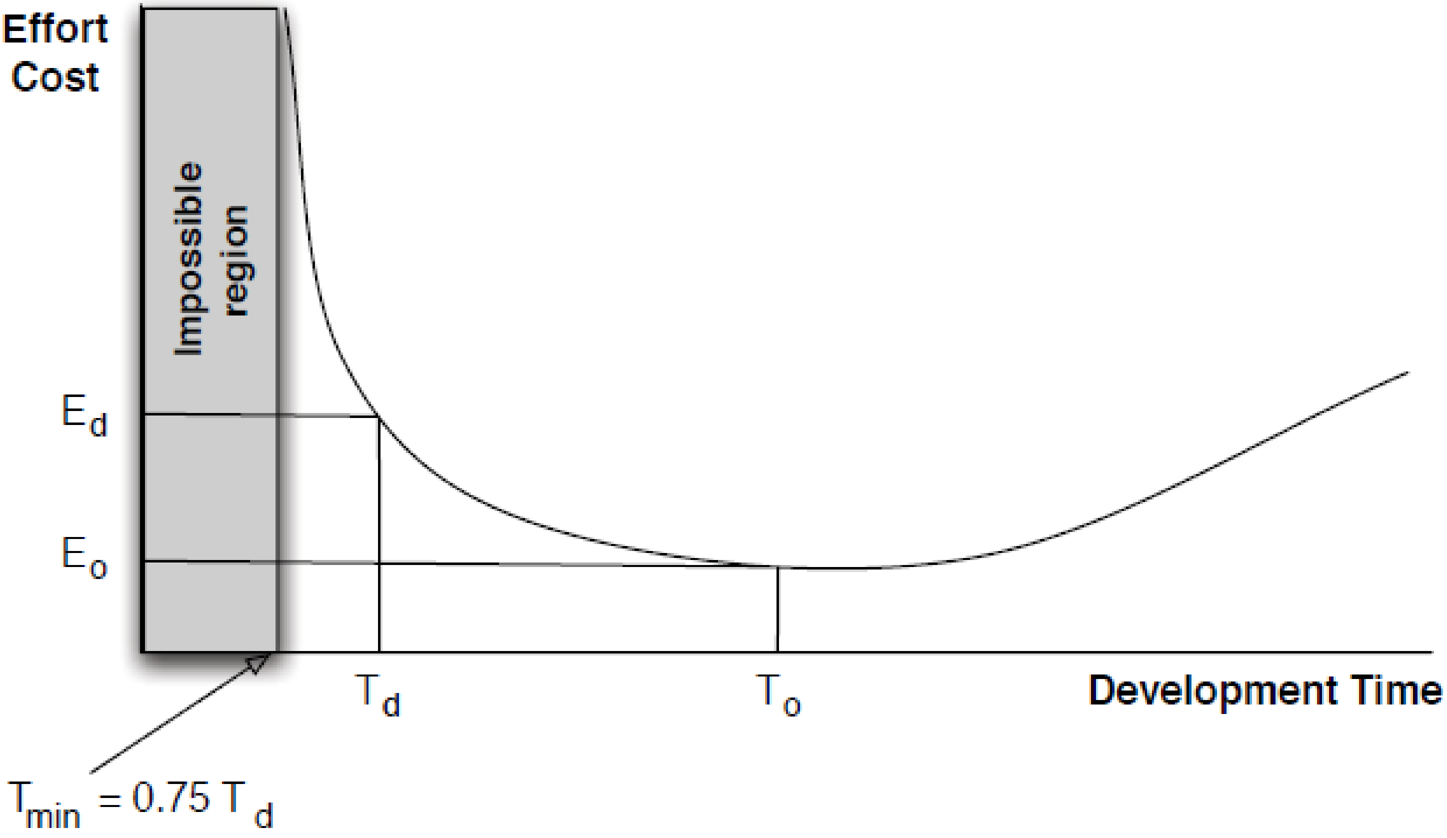
Effort-time Estimation

- **A common measure for estimating the effort for software is *man-months* (more generally *person-months*)**
  - Effort estimation will be covered in week 7

- **person-months:**
  - the time in months for a single person working full time to complete the task

- **The Mythical Man-Months [Brooks seminal paper]**
  - man-months is a misleading measure to estimate software
  - adding people to a project that is behind schedule could result in more damage than helping it

Effort vs Time



Putnam-Norden-Rayleigh curve

Time Estimation

- **Terminology**

  optimistic time - O

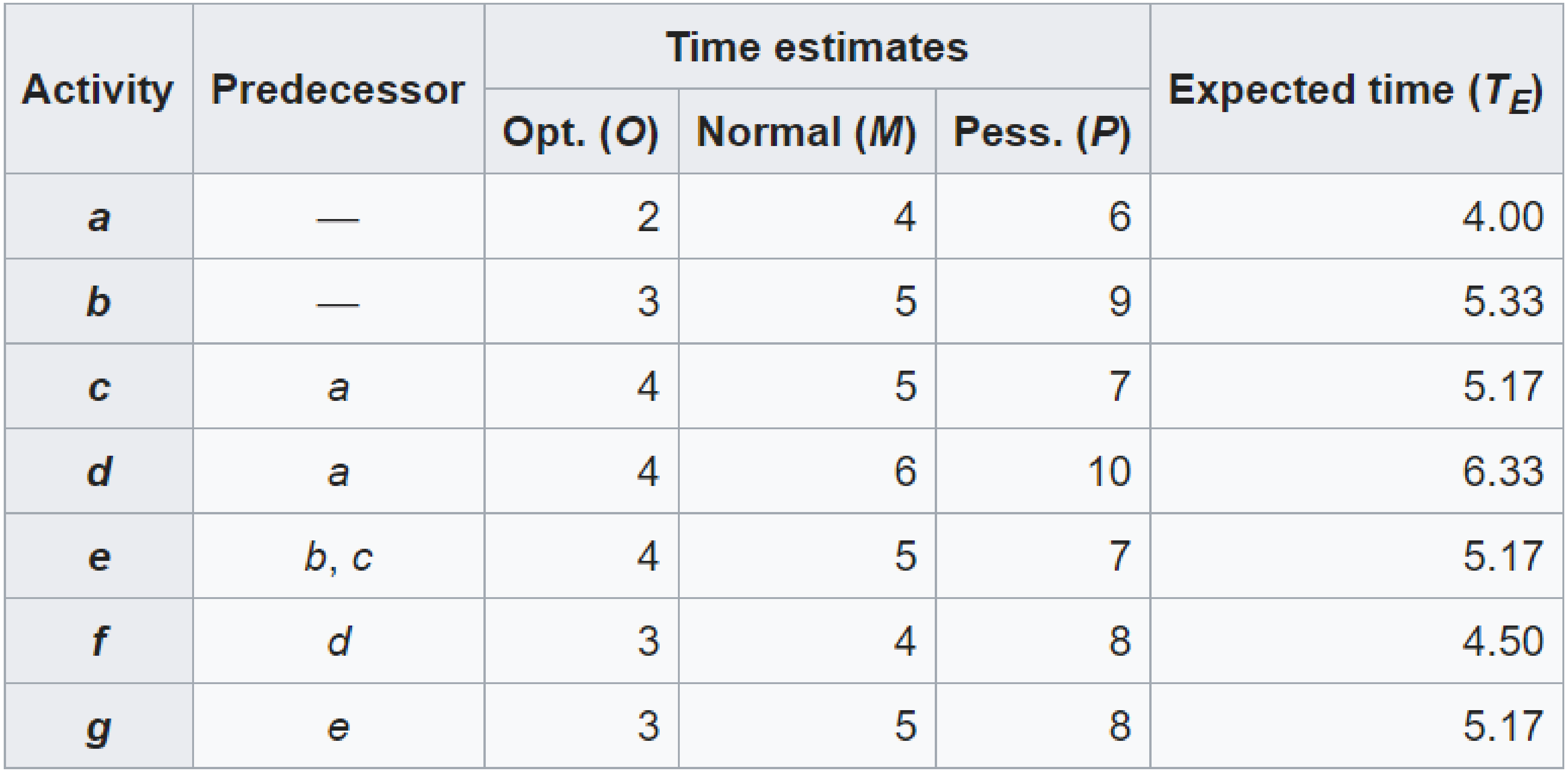
  pessimistic time - P

  most likely time - M

  expected time - T_E

$$T_E = (O + 4M + P)/6$$

Time Estimation



Activity	Predecessor	Time estimates			Expected time (T_E)
		Opt. (*O*)	Normal (*M*)	Pess. (*P*)	
a	—	2	4	6	4.00
b	—	3	5	9	5.33
c	*a*	4	5	7	5.17
d	*a*	4	6	10	6.33
e	*b*, *c*	4	5	7	5.17
f	*d*	3	4	8	4.50
g	*e*	3	5	8	5.17

Developing a Project Plan

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 a project schedule

Resource Allocation

- If the effort (person-months) and the time are known, the number of personnel can be computed as:

$$N = \frac{Effort}{T}$$

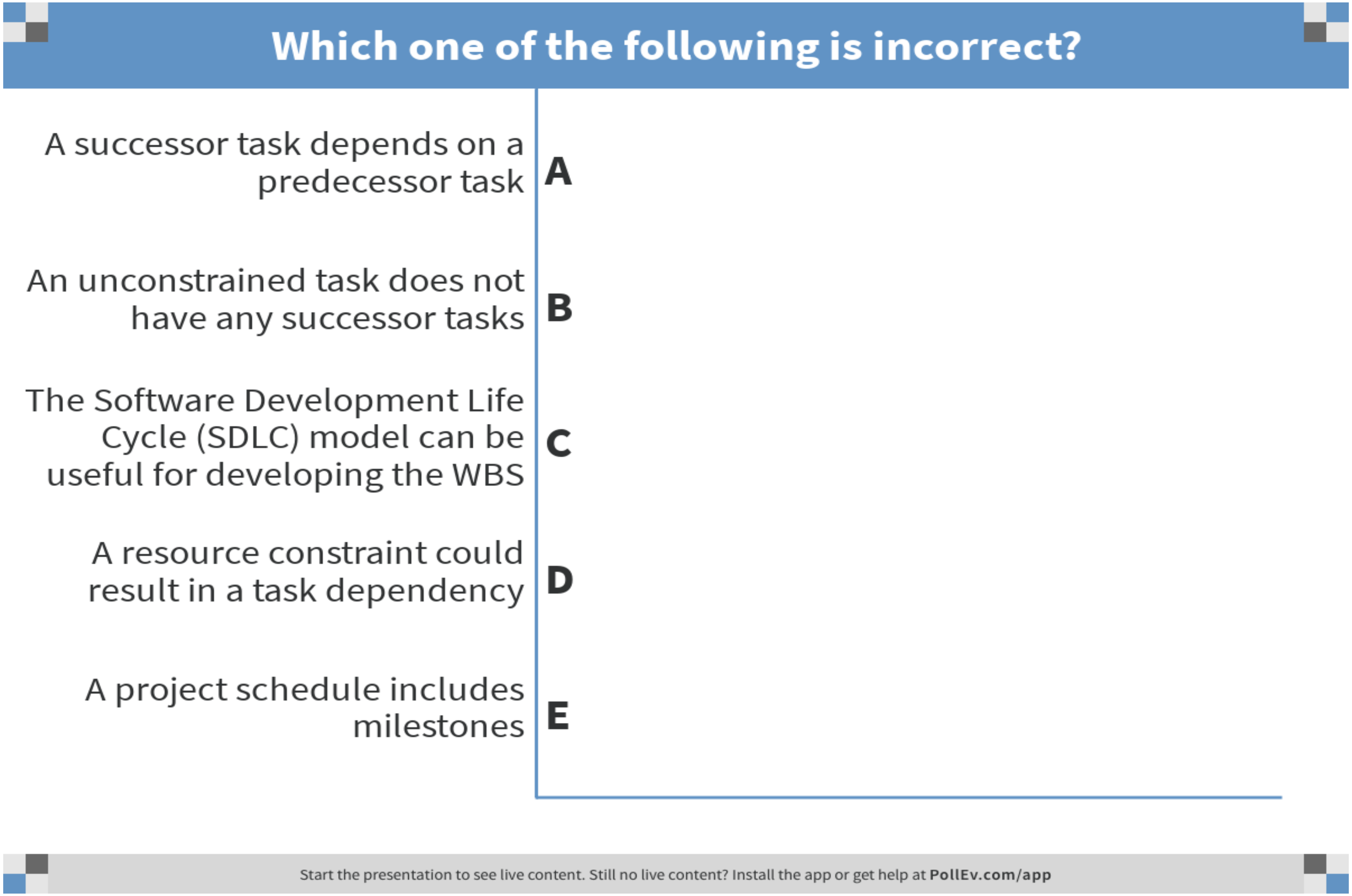
- Assigning people to tasks
  - Although computing the number of personnel required for each task appears simple, resource allocation is complicated task
  - The project manager has to carefully consider the expertise of the people, and the availability of them for tasks, which might require validation and adjustment of the schedule

Lecture Break

BREAK

Please return promptly as the

Lecture will re-start in *5 mins*



Which one of the following is incorrect?

A successor task depends on a predecessor task **A**

An unconstrained task does not have any successor tasks **B**

The Software Development Life Cycle (SDLC) model can be useful for developing the WBS **C**

A resource constraint could result in a task dependency **D**

A project schedule includes milestones **E**

Start the presentation to see live content. Still no live content? Install the app or get help at **PollEv.com/app**

Developing a Project Plan

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 a project schedule

Project Schedule

- **Project Schedule will answer two important questions not answered so far:**
  - How long will the system take to develop?
  - How much will it cost?

- **Two widely used graphical notations to represent the Project Schedule**
  - Gantt charts
    - A bar chart that shows the schedule against a calendar
  - PERT (Program Evaluation and Review Technique) charts
    - An activity network that shows the dependencies among tasks and the *critical path*

Project Scheduling - Definitions

Term	Description
Activity (Task)	Is part of a project that requires resources and time
Milestone	Is the completion of an activity that provides evidence of a deliverable completion or end of a phase – is an event that takes zero time
Free float (free slack)	Is the amount of time that a task can be delayed without causing a delay to subsequent tasks
Total float (total slack)	Is the amount of time that a task can be delayed without delaying project completion
Critical path	Is the longest possible continuous path taken from the initial event to the terminal event
Critical activity	Is an activity that has total float equal to zero

Milestones vs Deliverables

- **Milestones**
  - Mark specific points along a project timeline
  - These points may signal anchors such as:
    - a project start and end date
    - a need for external review
    - start and end of a phase
    - a completion of a deliverable
- **Deliverable**
  - Specific artefacts that are of interest
  - Examples of deliverables include:
    - Project documents such as the Project Management Plan, Requirements Specification, Design Document, Test Plan etc.
    - Prototypes
    - Final application

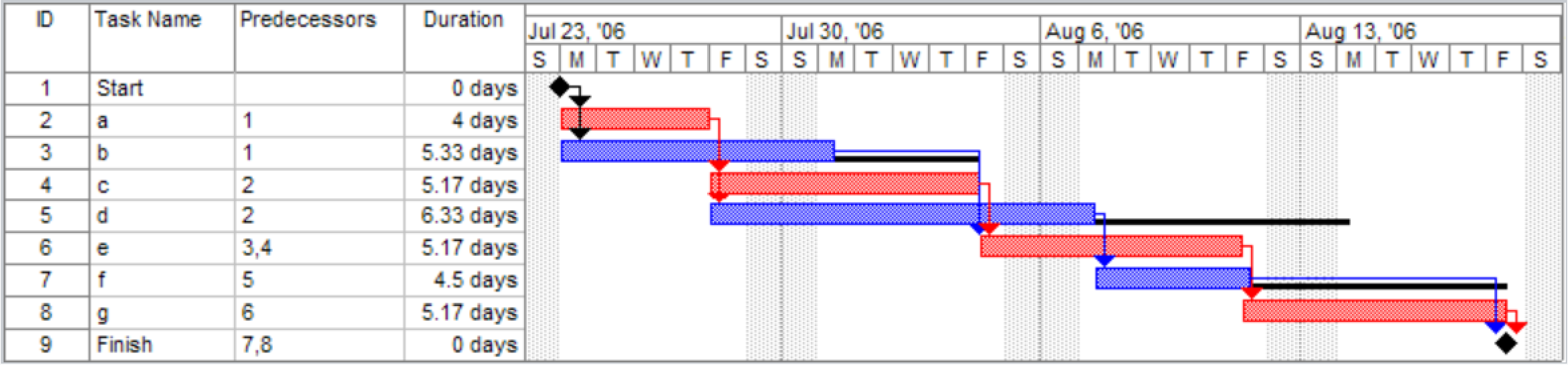
Gantt Chart

- Was introduced by Henry Gantt in 1910

- Gantt chart is a horizontal bar chart which shows tasks against a timeline – project schedule

- Can be used to view planned activities vs progress and therefore is a useful tool for monitoring project progress

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

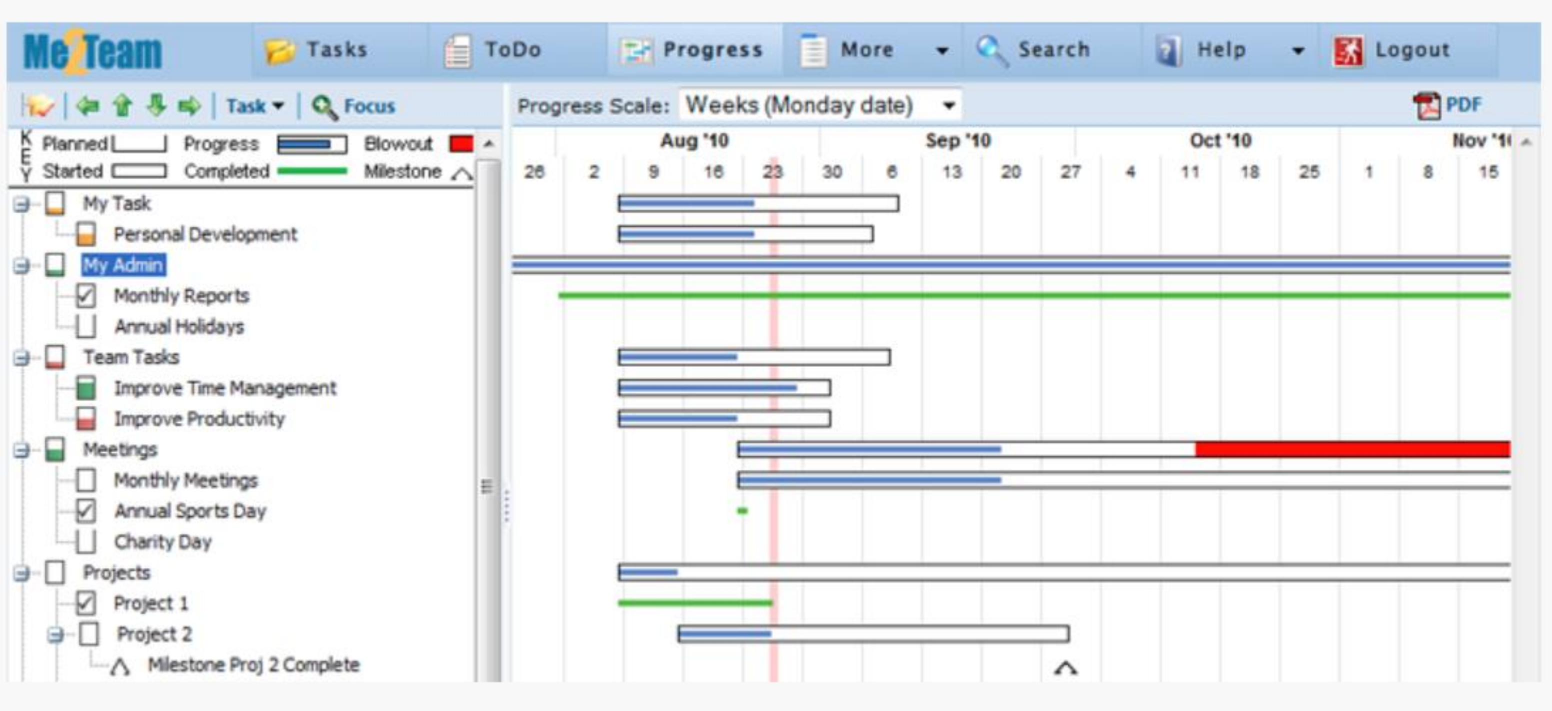
A Gantt chart created using Microsoft Project (MSP). Note (1) the critical path is in red, (2) the slack is the black lines connected to non-critical activities, (3) since Saturday and Sunday are not work days and are thus excluded from the schedule, some bars on the Gantt chart are longer if they cut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

Linked Gantt charts

- contain lines indicating the dependencies between tasks

THE UNIVERSITY OF MELBOURNE

Gantt Chart



Progress Scale: Weeks (Monday date)

Progress Gantt charts

- tasks are shaded in proportion to the degree of their completion
- used for progress tracking – gives a visual representation of the progress

PERT Chart

- PERT (Program Evaluation and Review Technique) chart:
  - A task network which shows the dependencies along with time related information and the critical path

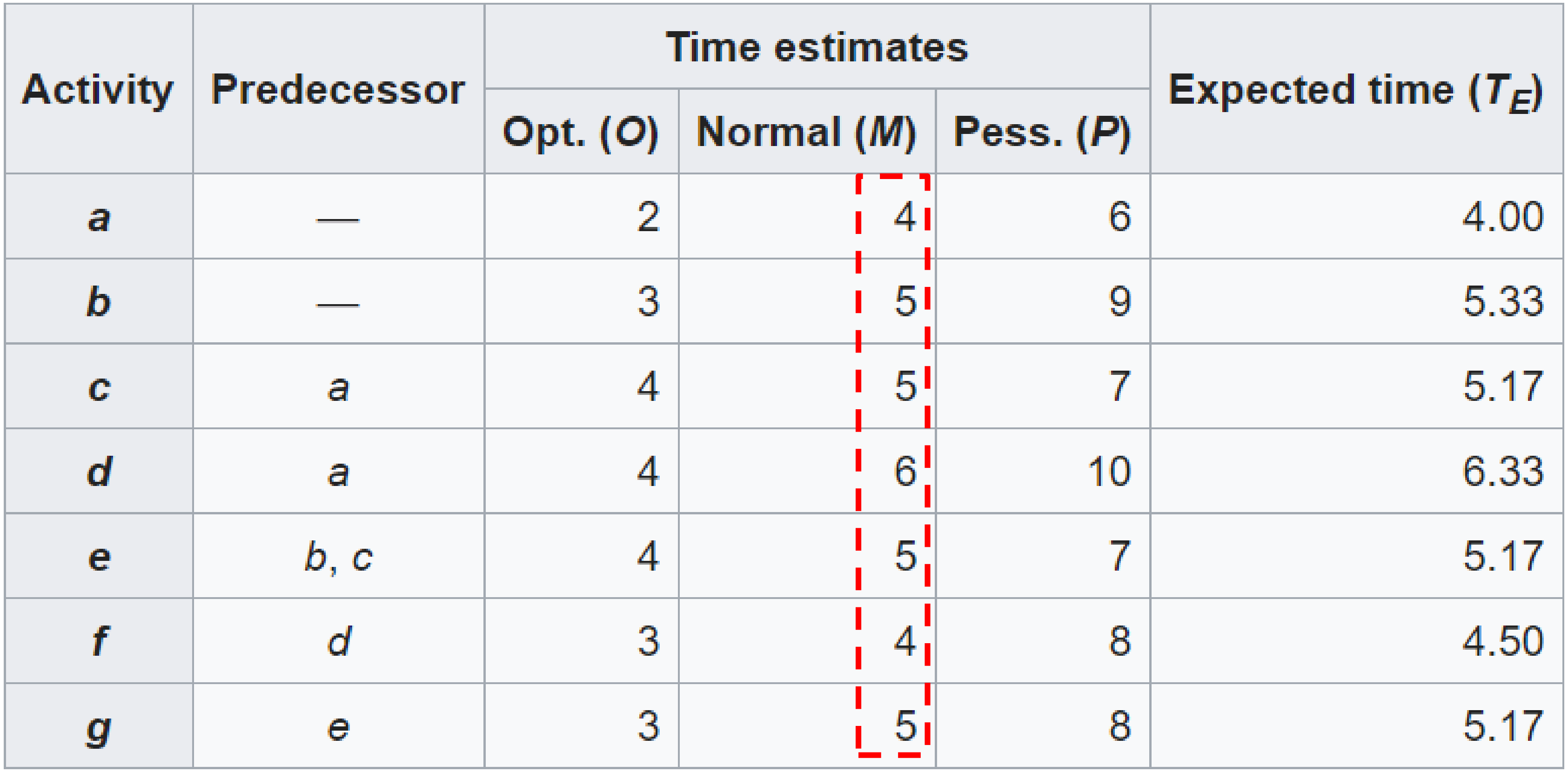
- PERT analysis helps:
  - understand the characteristics of the project that will let project managers do scheduling trade-offs
  - perform critical path analysis
  - monitor project progress and re-plan

PERT Chart

- Involves calculating the following estimates:
  - Earliest start time (ES)
  - Latest start time (LS)
  - Earliest finish time (EF)
  - Latest finish time (LF)
  - Slack time

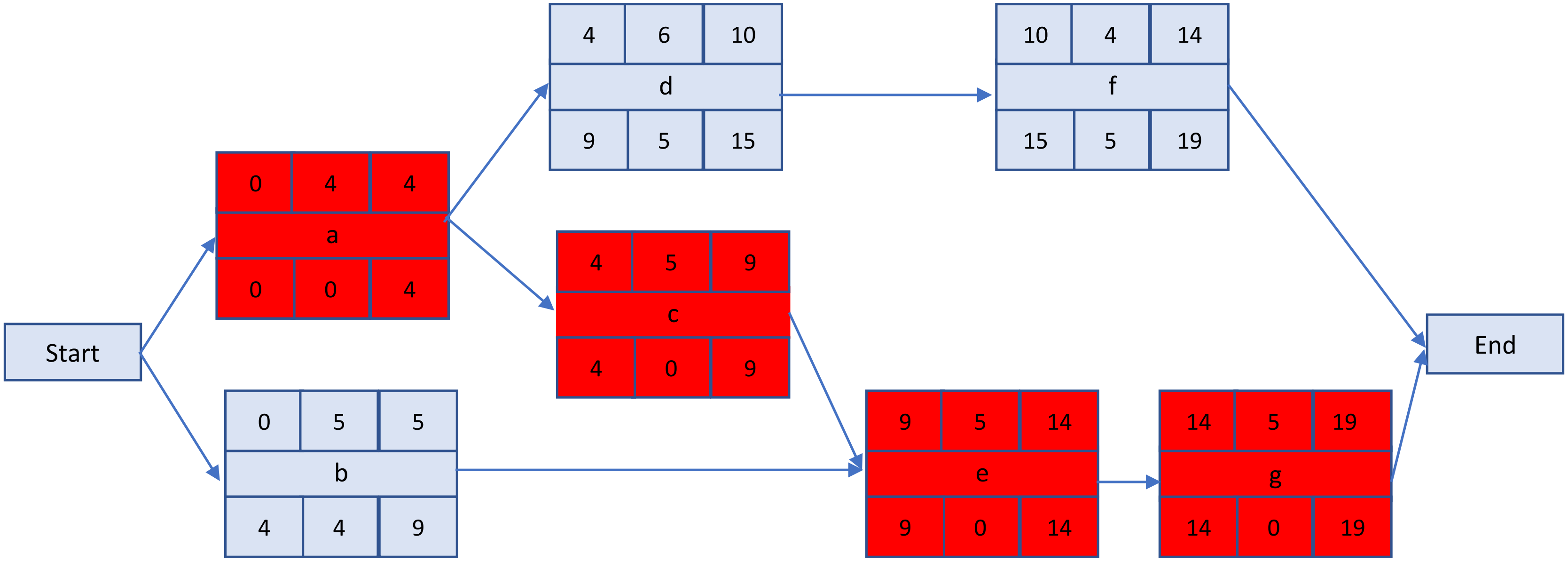
<table>
<tr><td>ES</td><td>Duration</td><td>EF</td></tr>
<tr><td colspan="3">Task Name</td></tr>
<tr><td>LS</td><td>Slack</td><td>LF</td></tr>
</table>

PERT Chart – Example



Activity	Predecessor	Time estimates			Expected time (T_E)
		Opt. (O)	Normal (M)	Pess. (P)	
a	—	2	4	6	4.00
b	—	3	5	9	5.33
c	*a*	4	5	7	5.17
d	*a*	4	6	10	6.33
e	*b*, *c*	4	5	7	5.17
f	*d*	3	4	8	4.50
g	*e*	3	5	8	5.17

PERT Chart - Example



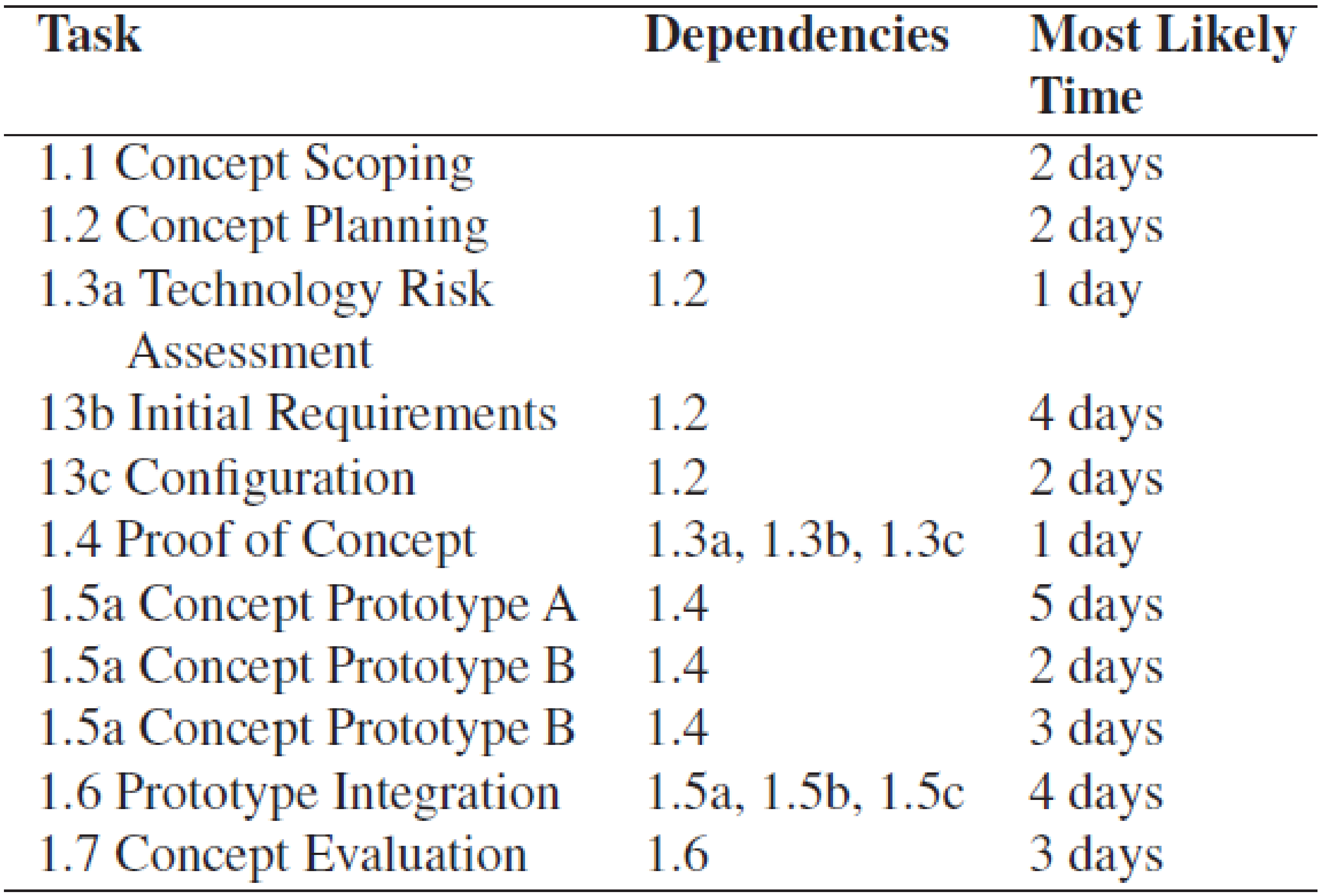
Critical Path: a, c, e, g

Duration: 19 days

Notes:

- Critical path activities have a total free slack of 0
- Two parallel paths could be critical paths

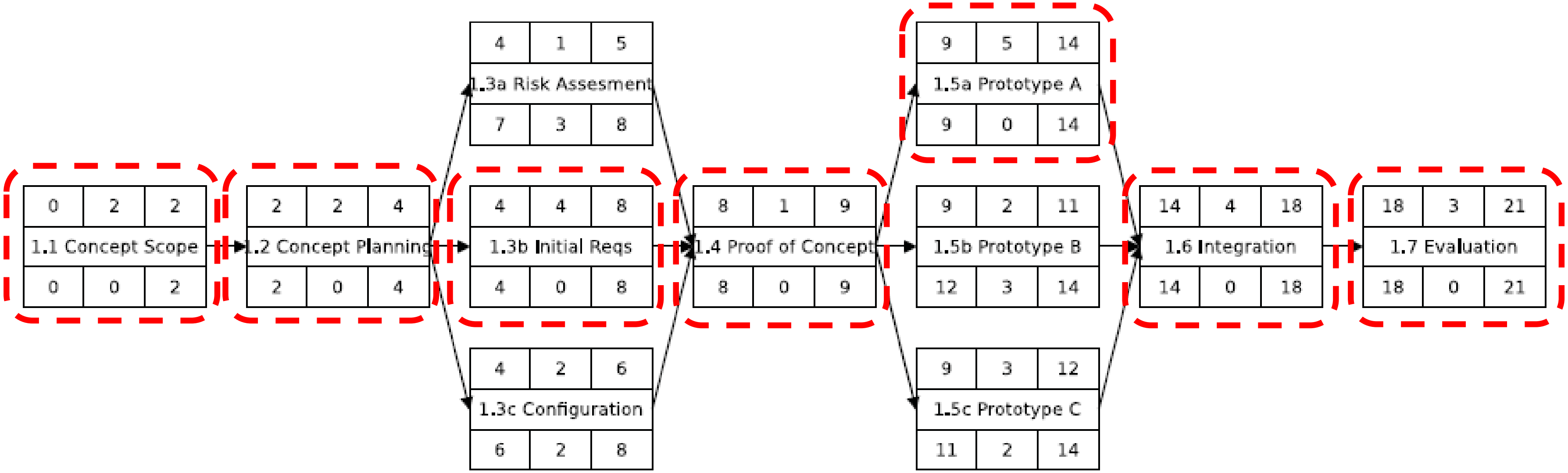
PERT Chart Example



Task	Dependencies	Most Likely Time
1.1 Concept Scoping		2 days
1.2 Concept Planning	1.1	2 days
1.3a Technology Risk Assessment	1.2	1 day
13b Initial Requirements	1.2	4 days
13c Configuration	1.2	2 days
1.4 Proof of Concept	1.3a, 1.3b, 1.3c	1 day
1.5a Concept Prototype A	1.4	5 days
1.5a Concept Prototype B	1.4	2 days
1.5a Concept Prototype B	1.4	3 days
1.6 Prototype Integration	1.5a, 1.5b, 1.5c	4 days
1.7 Concept Evaluation	1.6	3 days

THE UNIVERSITY OF MELBOURNE

PERT Chart Example



Critical Path: 1.1, 1.2, 1.3b, 1.4, 1.5a, 1.6, 1.7

Duration: 21 days

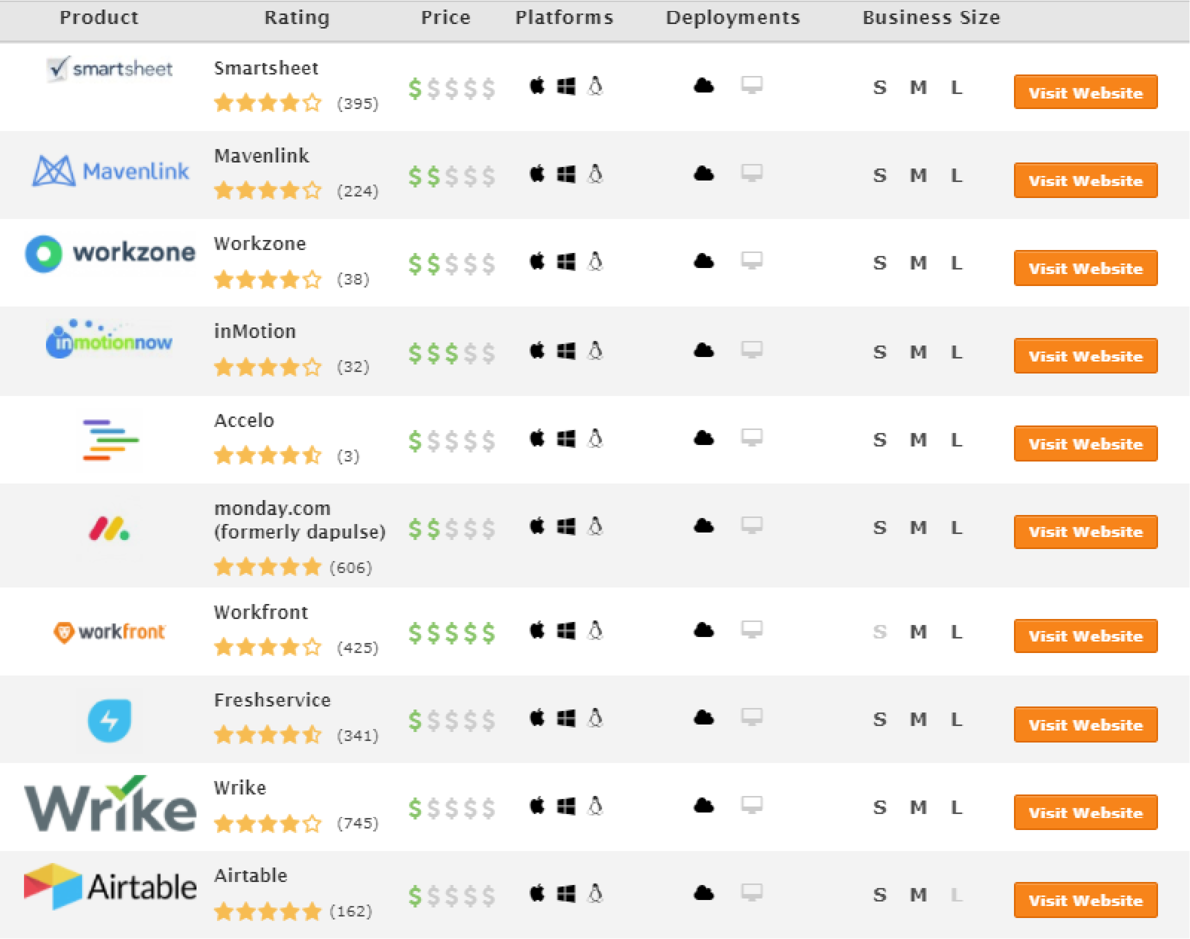
Note: Critical path activities have a total free slack of 0

Critical Path Methods

- **Critical Path**
  - path with the longest duration
  - activities on the critical path have a total free slack of 0
  - a delay in any of the activities in the critical path will cause the project to delay

- **Crashing the project schedule**
  - shortening the total duration of the project by shortening the critical path
    - By removing the dependencies between activities in the critical path; or
    - Shortening the duration of activities in the critical path

Tools



Product	Rating	Price	Platforms	Deployments	Business Size	
smartsheet	Smartsheet ★★★★☆ (395)	**$**$$$$	🍎 ⊞ 🐧	☁ 🖥	**S M L**	Visit Website
Mavenlink	Mavenlink ★★★★☆ (224)	**$$**$$$	🍎 ⊞ 🐧	☁ 🖥	**S M L**	Visit Website
workzone	Workzone ★★★★☆ (38)	**$$**$$$	🍎 ⊞ 🐧	☁ 🖥	**S M L**	Visit Website
inmotionnow	inMotion ★★★★☆ (32)	**$$$**$$	🍎 ⊞ 🐧	☁ 🖥	**S M L**	Visit Website
	Accelo ★★★★½ (3)	**$**$$$$	🍎 ⊞ 🐧	☁ 🖥	**S M L**	Visit Website
	monday.com (formerly dapulse) ★★★★★ (606)	**$$**$$$	🍎 ⊞ 🐧	☁ 🖥	**S M L**	Visit Website
workfront	Workfront ★★★★☆ (425)	**$$$$$**	🍎 ⊞ 🐧	☁ 🖥	S **M L**	Visit Website
	Freshservice ★★★★½ (341)	**$**$$$$	🍎 ⊞ 🐧	☁ 🖥	**S M L**	Visit Website
Wrike	Wrike ★★★★☆ (745)	**$**$$$$	🍎 ⊞ 🐧	☁ 🖥	**S M L**	Visit Website
Airtable	Airtable ★★★★★ (162)	**$**$$$$	🍎 ⊞ 🐧	☁ 🖥	**S M** L	Visit Website

https://www.workzone.com/blog/gantt-chart-software/

Intended Learning Outcomes

- Understand the role the project schedule
- Understand how to develop a project schedule
- Understand how to use a project schedule to monitor and track project progress
- Understand agile planning principles

Project Tracking and Control

- **How do software projects fall behind schedule?**

  One day at a time

  – Fred Brooks, the well-known author of the seminal article Mythical Man-Months

- **Project scheduling is important, but *tracking and controlling* are even more important!**

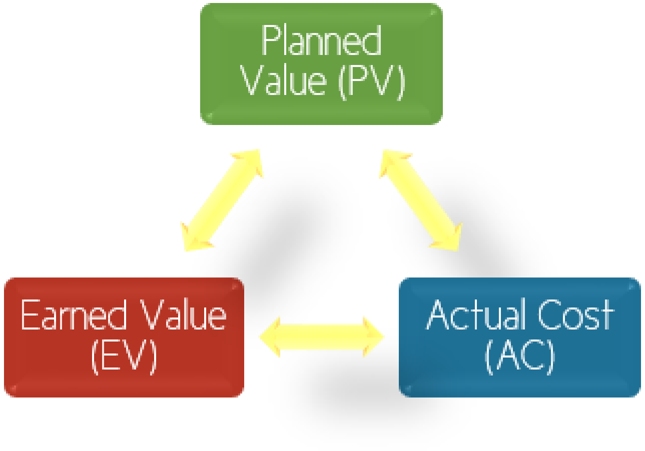
Project Tracking and Control

- How to track and control project progress?
  - Periodic meetings where team members report progress
  - Evaluating the results of reviews and audits conducted as part of the software engineering process
  - Tracking formal project milestones
  - Comparing actual start dates with scheduled start dates
  - Meeting engineers and having informal discussions
  - Using a formal method like *earned value analysis*

Earned Value Analysis (EVA)

- EVA can be used to:
  - report current/past project performance
  - predict future project performance based on current/past performance

- Results can be expressed in dollars and/or percentage



Computing EVA

- Planned Value (PV)
  - that portion of the approved cost estimate planned to be spent on the given activity during a given period
- The Earned Value (EV)
  - the value of the work actually completed
- Actual Cost (AC)
  - the total of the costs incurred in accomplishing work on the activity in a given period

EVA - Example

- Consider the following scenario:

You are assigned to manage a project that is planned to finish in 12 months, estimated to cost $100,000. At the end of the third month, based on the project Gantt chart, 20% of the work had been reported as completed.   The finance department has reported the cost of the project to date as $35,000.

What is the PV?

What is the EV?

What is the AC?

EVA - Example

- Consider the following scenario:

You are assigned to manage a project that is planned to finish in 12 months, estimated to cost $100,000. At the end of the third month, based on the project Gantt chart, 20% of the work had been reported as completed. The finance department has reported the cost of the project to date as $35,000.

PV = $100,000*3/12 = $25,000 (assuming equal work distribution over the period, which may not be the case always)

EV = $100,000*20/100 = $20,000

AC = $35,000

EVA – Schedule Variance

- **Schedule Variance Analysis**
  - Uses EV and PV to calculate a variance to the project schedule

- **Schedule Variance: expressed in dollars**

$$SV = EV - PV$$
$$= 20{,}000 - 25{,}000$$
$$= (5000)$$

- **Schedule Performance Index: expressed as a fraction**

$$SPI = EV/PV$$
$$= 20{,}000/25{,}000$$
$$= 0.8$$

EVA – Cost Variance

- **Cost Variance Analysis**
  - Uses EV and AC to calculate a variance to the project schedule

- **Cost Variance: expressed in dollars**

$$CV = EV - AC$$

$$= 20{,}000 - 35{,}000$$

$$= (15{,}000)$$

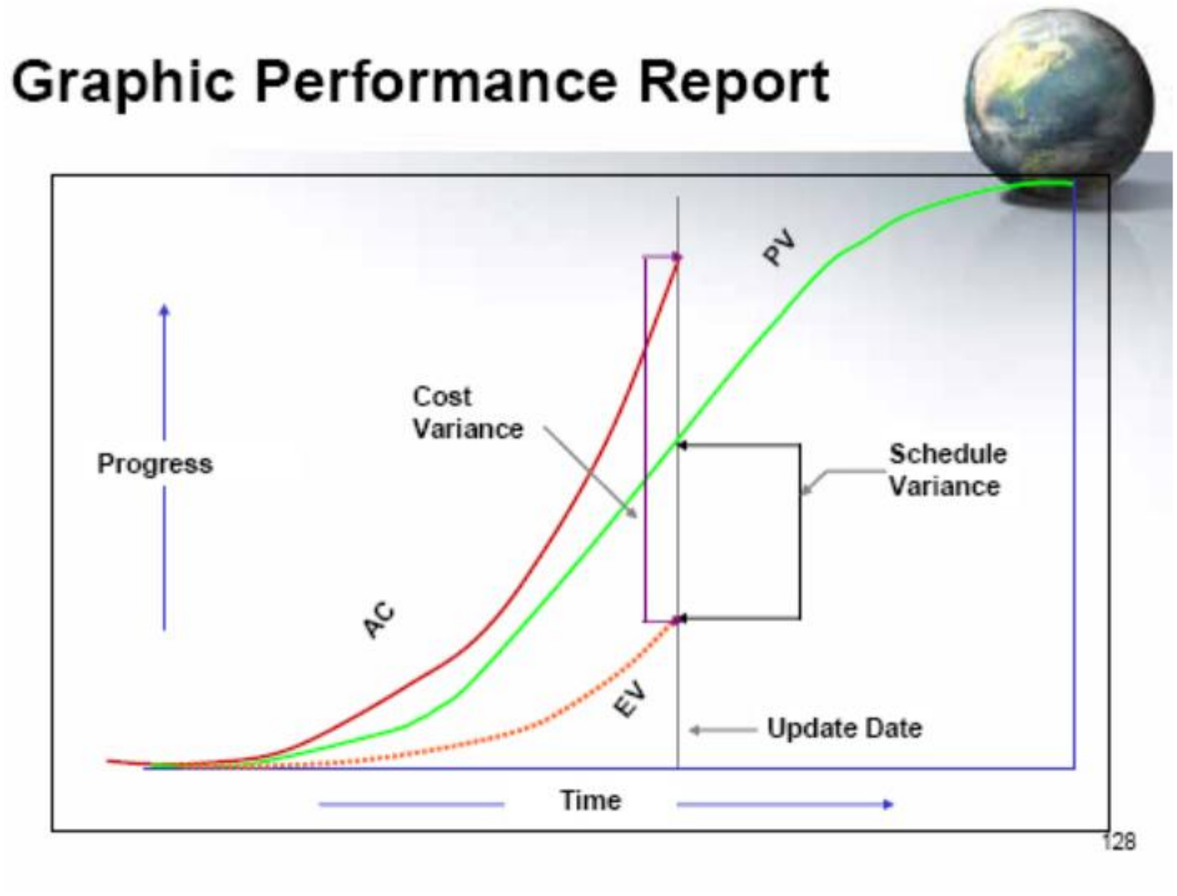
- **Cost Performance Index: expressed as a fraction**

$$CPI = EV/AC$$

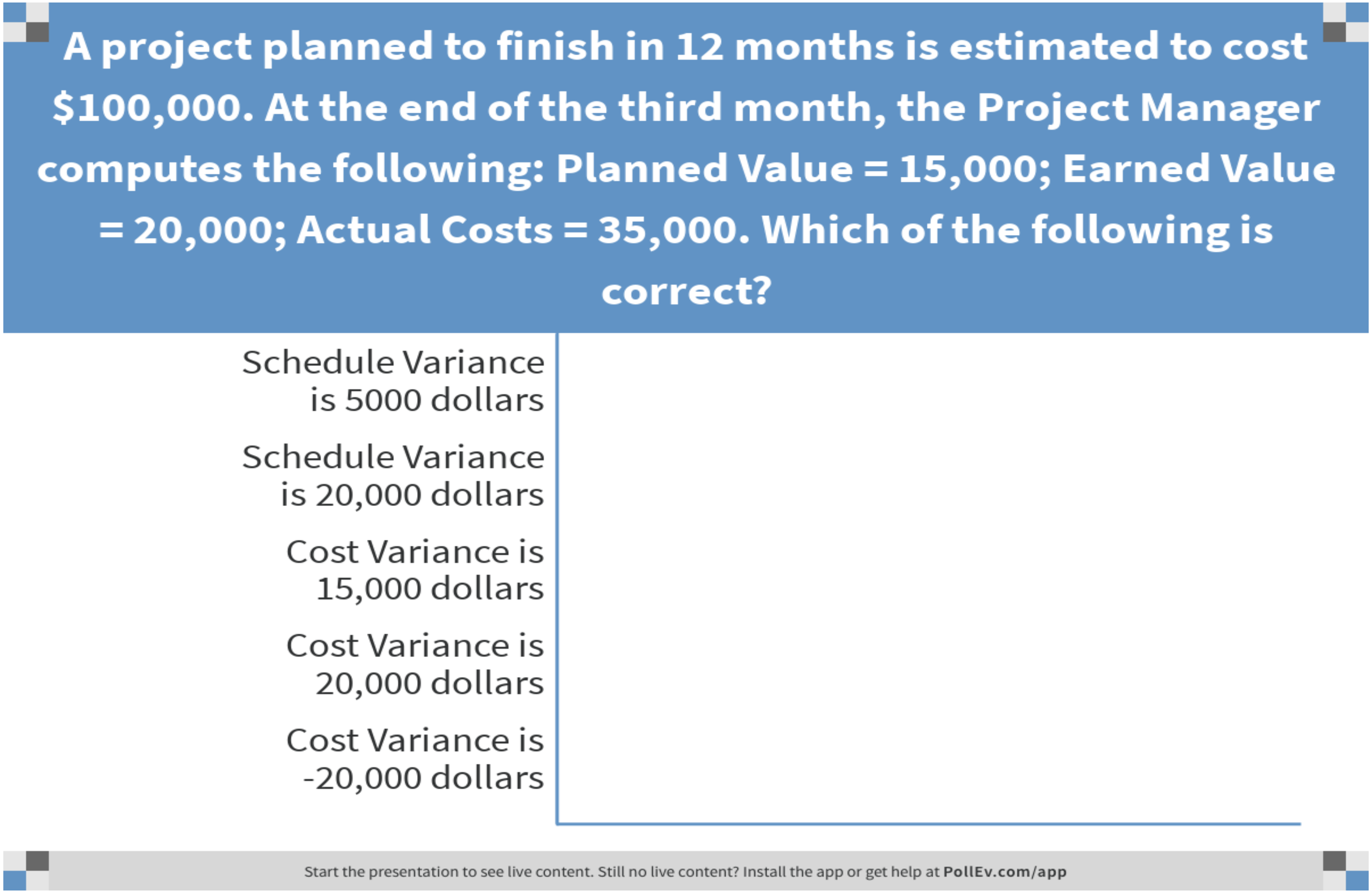
$$= 20{,}000/35{,}000$$

$$= 0.57$$

EVA – Graphical Representation



https://www.pmi.org/learning/library/earned-value-management-systems-analysis-8026



A project planned to finish in 12 months is estimated to cost $100,000. At the end of the third month, the Project Manager computes the following: Planned Value = 15,000; Earned Value = 20,000; Actual Costs = 35,000. Which of the following is correct?

- Schedule Variance is 5000 dollars
- Schedule Variance is 20,000 dollars
- Cost Variance is 15,000 dollars
- Cost Variance is 20,000 dollars
- Cost Variance is -20,000 dollars

Start the presentation to see live content. Still no live content? Install the app or get help at **PollEv.com/app**

Intended Learning Outcomes

1. Understand the role of a project schedule

2. Understand how to develop a project schedule

3. Understand how to use a project schedule to monitor and track project progress

4. Understand agile planning principles

Planning in Agile Development

- Takes a significantly different flavour from traditional approaches

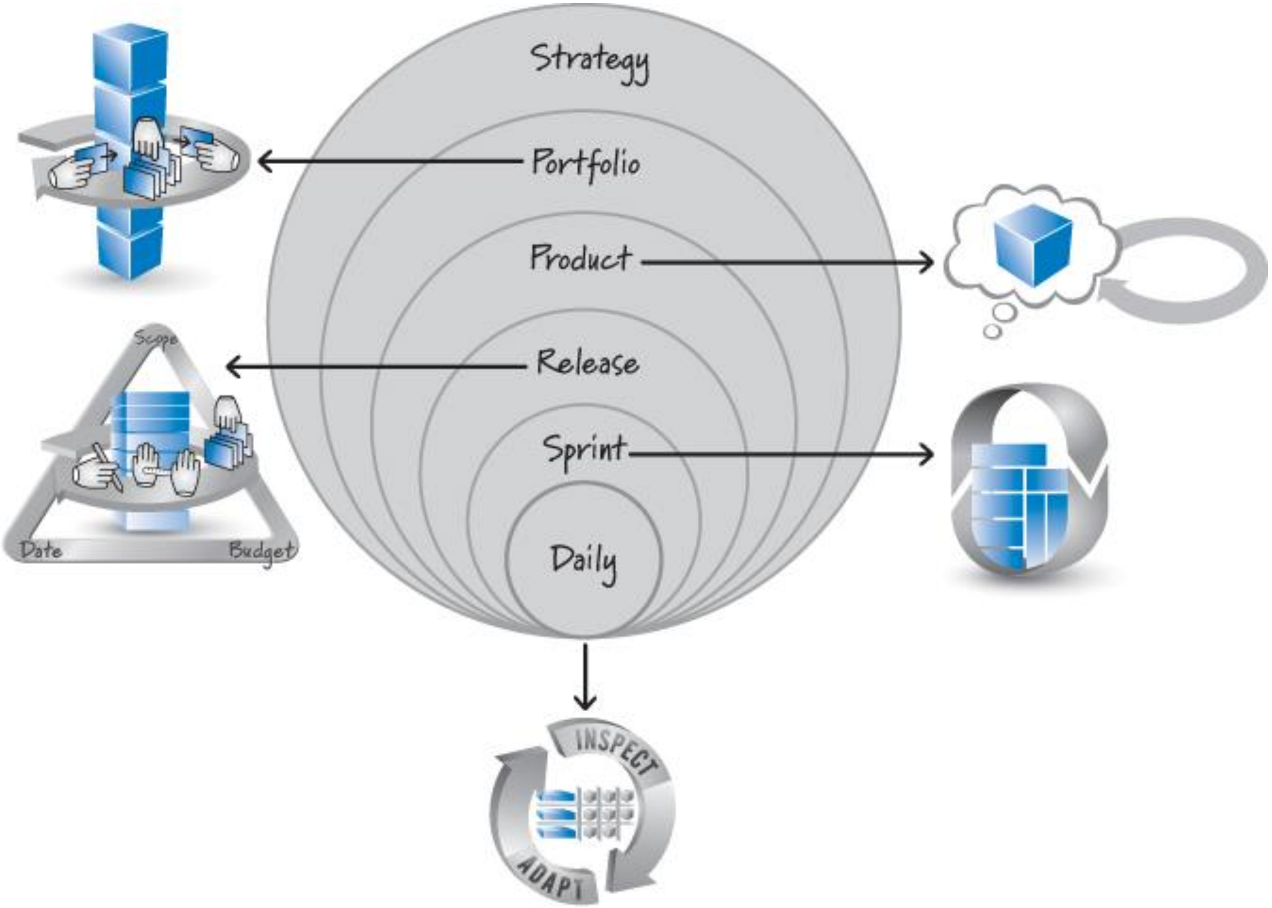
- Detailed planning is deferred until the start of the iteration
  - Designed to handle change
  - An iteration includes all phases (requirements, design and test)

- Planning is based on light weight lists
  - Gantt and PERT charts are considered less useful

Planning in Agile Development

- Plan short iterations
- Deliver working software
- Use “Just in time (JIT) planning” – next iteration
- Use the team

Planning in Scrum



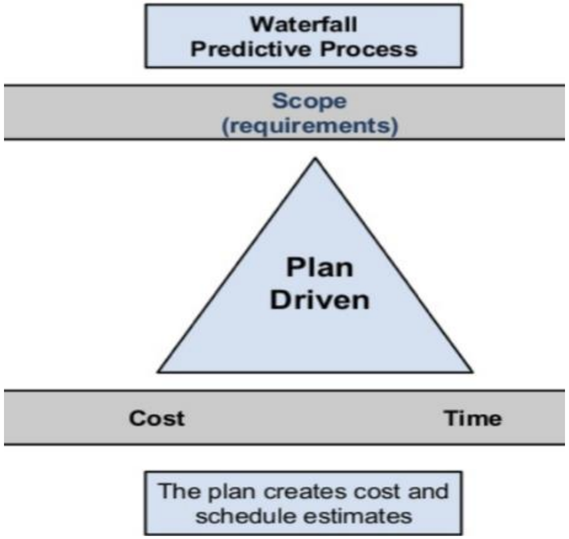
Different levels of planning in Scrum

Planning in Scrum

Level	Horizon	Who	Focus	Deliverables
Portfolio	Possibly a year of more	Stakeholders and product owners	Managing a portfolio of products	Portfolio backlog and collection of in-process products
Product (envisioning)	Up to many months or longer	Product owner, stakeholders	Visions and product evolution over time	Product vision, roadmap, and high-level features
Release	Three (or fewer) to nine months	Entire Scrum Team, Stakeholders	Continuously balance customer value and overall quality against the constraints of scope, schedule and budget	Release Plan
Sprint	Every iteration (one week to one month)	Entire Scrum Team	What features to deliver in the next Sprint	Sprint goals and sprint backlog
Daily	Every day	Scrum Master, development team	How to complete committed features	Inspection of current progress and adaptation

Release Planning

- Assumptions in Formal Planning:
  - Scope fixed – requirements are stable
  - Budget fixed – cost estimations are accurate
  - Schedule fixed - derived based on scope and budget



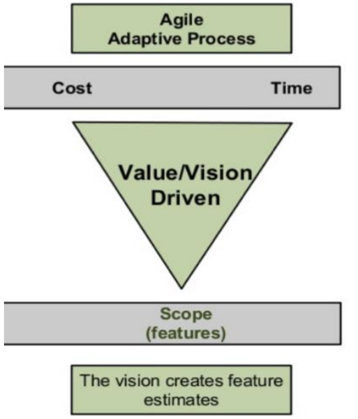
THE UNIVERSITY OF MELBOURNE

Release Planning

- Agile Planning
  - Recognizes that all three factors: scope, budget and time cannot be fixed in reality - not recommended
  - Can we fix scope and date and make the budget flexible?
    - Not really because increasing the budget, hence the resources will not always help to improve speed – not recommended
  - So what are our options?
    - Fix date and budget

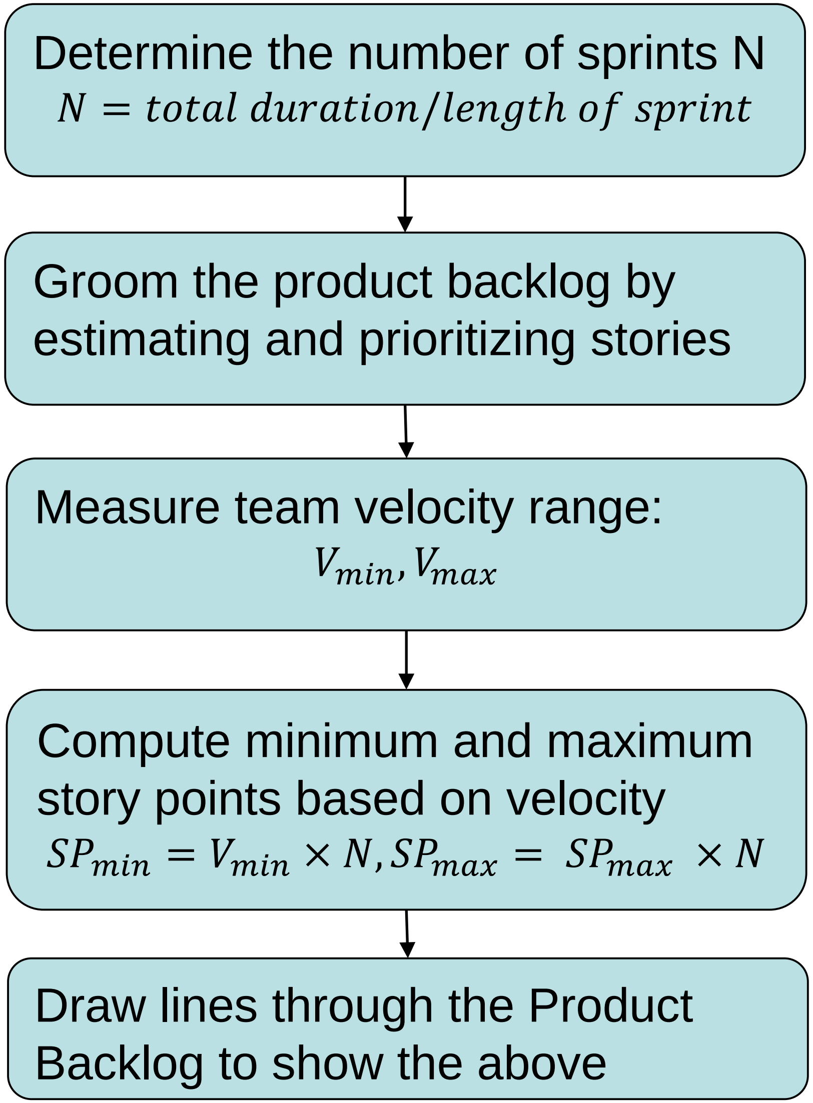
      and have the scope flexible

      Fixed-Date release planning



    - Fix scope and have the date and budget flexible – *Fixed-Scope release planning*

Fixed-Date Release Planning



Determine the number of sprints N

$N = total\ duration/length\ of\ sprint$

Groom the product backlog by estimating and prioritizing stories

Measure team velocity range:

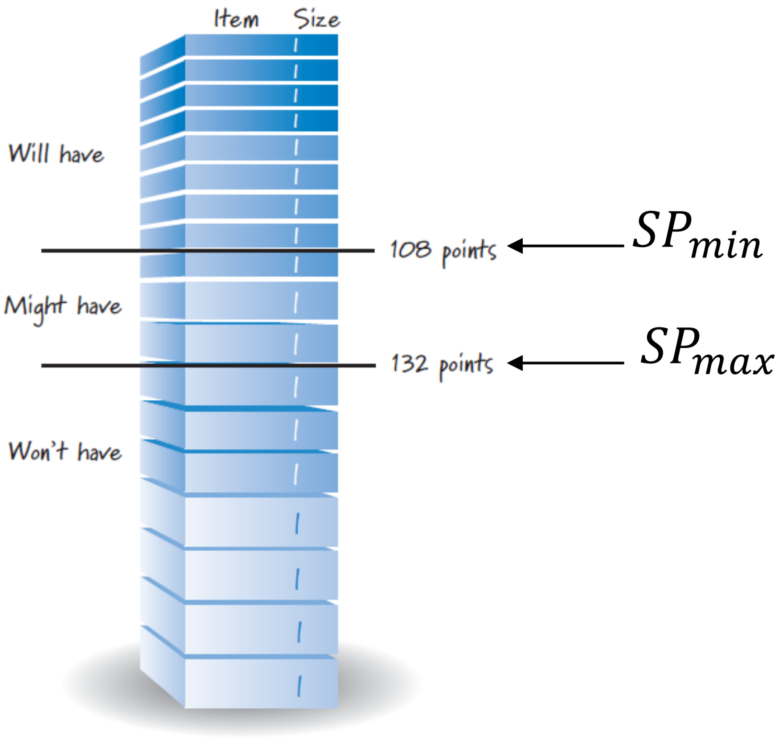
V_{min}, V_{max}

Compute minimum and maximum story points based on velocity

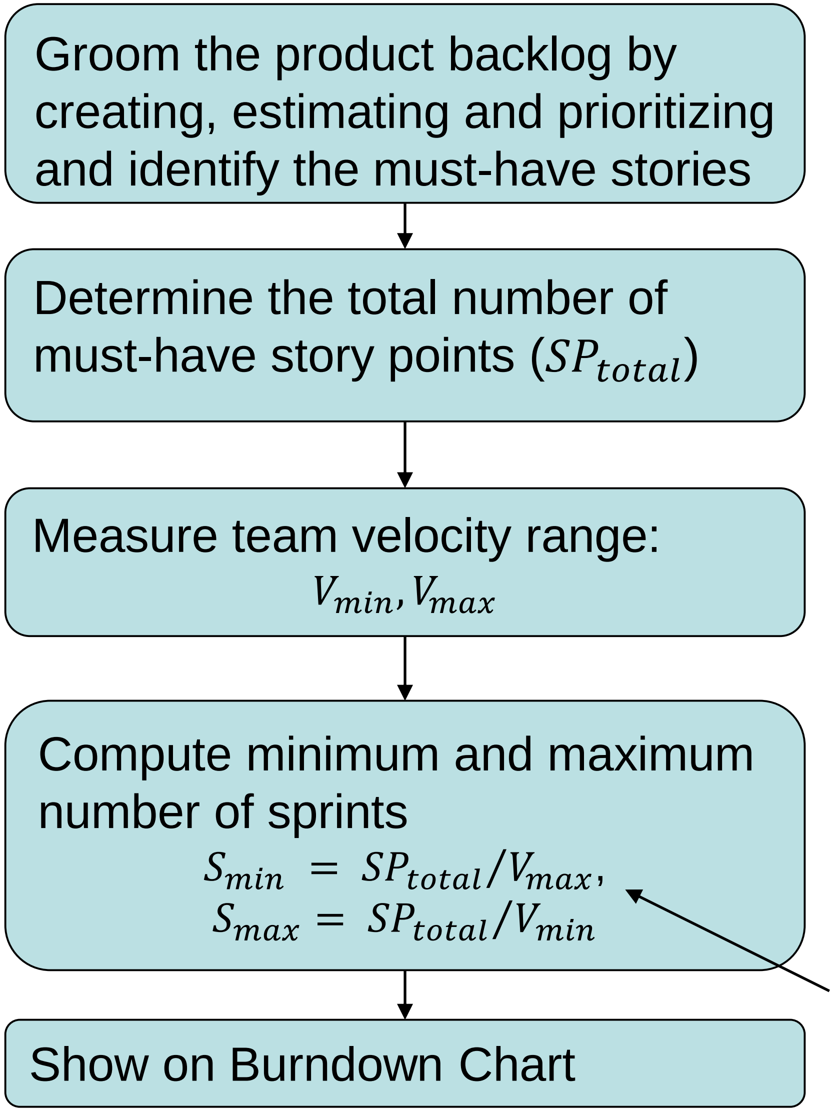
$SP_{min} = V_{min} \times N, SP_{max} = SP_{max} \times N$

Draw lines through the Product Backlog to show the above

Fixed-Date: used when date is more important



Fixed-Scope Release Planning



Groom the product backlog by creating, estimating and prioritizing and identify the must-have stories

Determine the total number of must-have story points (SP_{total})

Measure team velocity range:

$$V_{min}, V_{max}$$

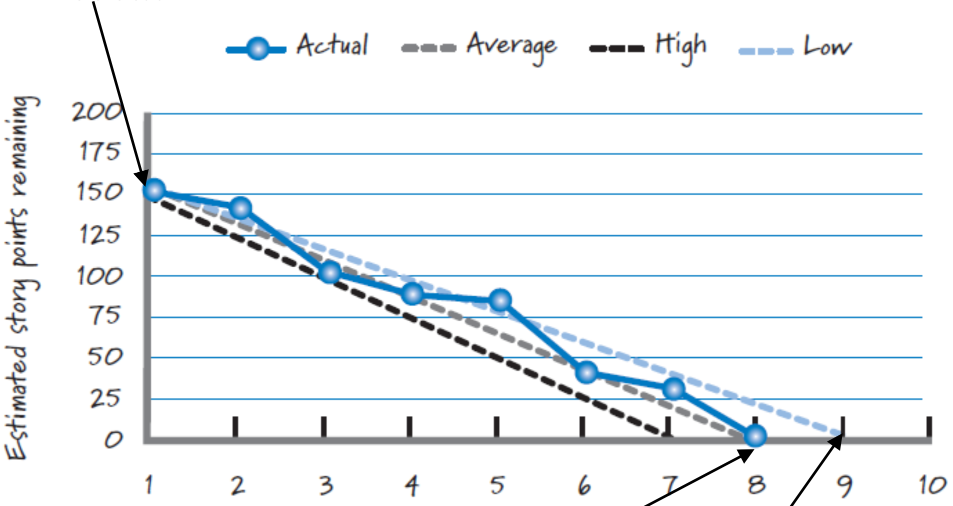
Compute minimum and maximum number of sprints

$$S_{min} = SP_{total}/V_{max},$$
$$S_{max} = SP_{total}/V_{min}$$

Show on Burndown Chart

Fixed-Scope: used when scope is more important

SP_{total}



Burndown chart

S_{min} S_{max}

May require rounding up to be an integer

Intended Learning Outcomes

1. Understand the role of a project schedule
2. Understand how to develop a project schedule
3. Understand how to use a project schedule to monitor and track project progress
4. Understand agile planning principles

References

1. F. P. Brooks. The mythical man-month. In Essays on software engineering. Addison-Wesley, 1995.

2. R. S. Pressman. Software Engineering: A Practitioner’s Approach. McGraw Hill, seventh edition, 2009.

3. Kenneth S. Rubin. Essential Scrum – A Practical Guide to the Most Popular Agile Process. Addison-Wesley, 2013.