

Agile Project Planning

COMP6204: Software Project Management and Secure
Development

Dr A. Rezazadeh (Reza)

Email: ra3@ecs.soton.ac.uk or ar4k06@soton.ac.uk

November 24

Overview

- Objectives
- Agile Planning vs. Traditional Planning
- Minimum Viable Product (MVP) & Minimally Marketable Product (MMP)
- Requirements Gathering
- User Stories & Prioritisation Method
- Relative Sizing & Story Points
- Agile Estimating Methods
- Ideal Time and Real Time Estimation
- Integration Planning for an Agile/Hybrid Project
- Scope Planning for an Agile/Hybrid Project

Objectives

- Summarise **planning** project **schedule** and **cost management** for agile/hybrid projects including the concepts of:
 - Timeboxing
 - The Kanban method
 - and relative sizing
- Understanding of **Minimum Viable Product** MVP and **Minimally Marketable Product** MMP concepts
- Requirements **gathering** techniques
- Requirements **prioritisation** techniques
- Techniques to estimate (do sizing) of requirements

Agile Planning vs. Traditional Planning

Traditional Plans

- Created upfront at the start
- Re-planning is lesser, only when changes happen
- Most requirements are known upfront
- Planning is mostly at the beginning of the lifecycle
- Midcourse adjustments are few

Agile Plans

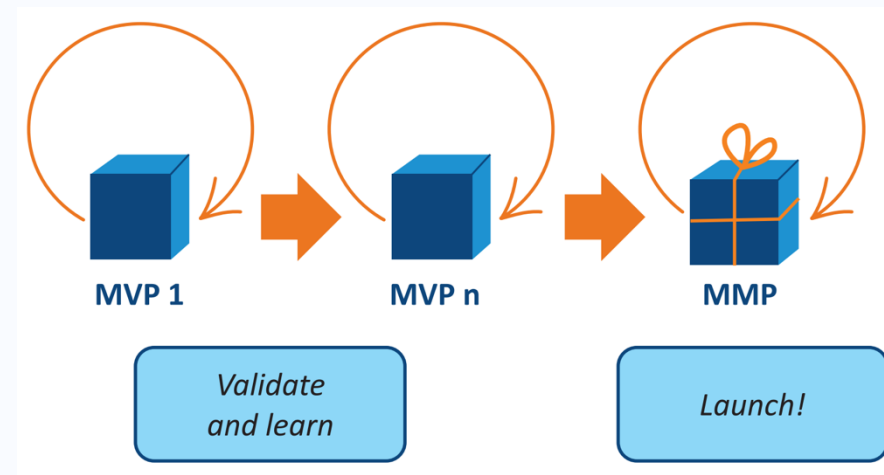
- Only high-level plan created upfront with detailed planning only for the user stories in the next release or iteration
- Re-planning is the norm as we embrace changes
- True requirements are uncovered as we proceed
- Planning is in lumps throughout the lifecycle
- Midcourse adjustments are a norm

Minimum Viable Product (MVP)

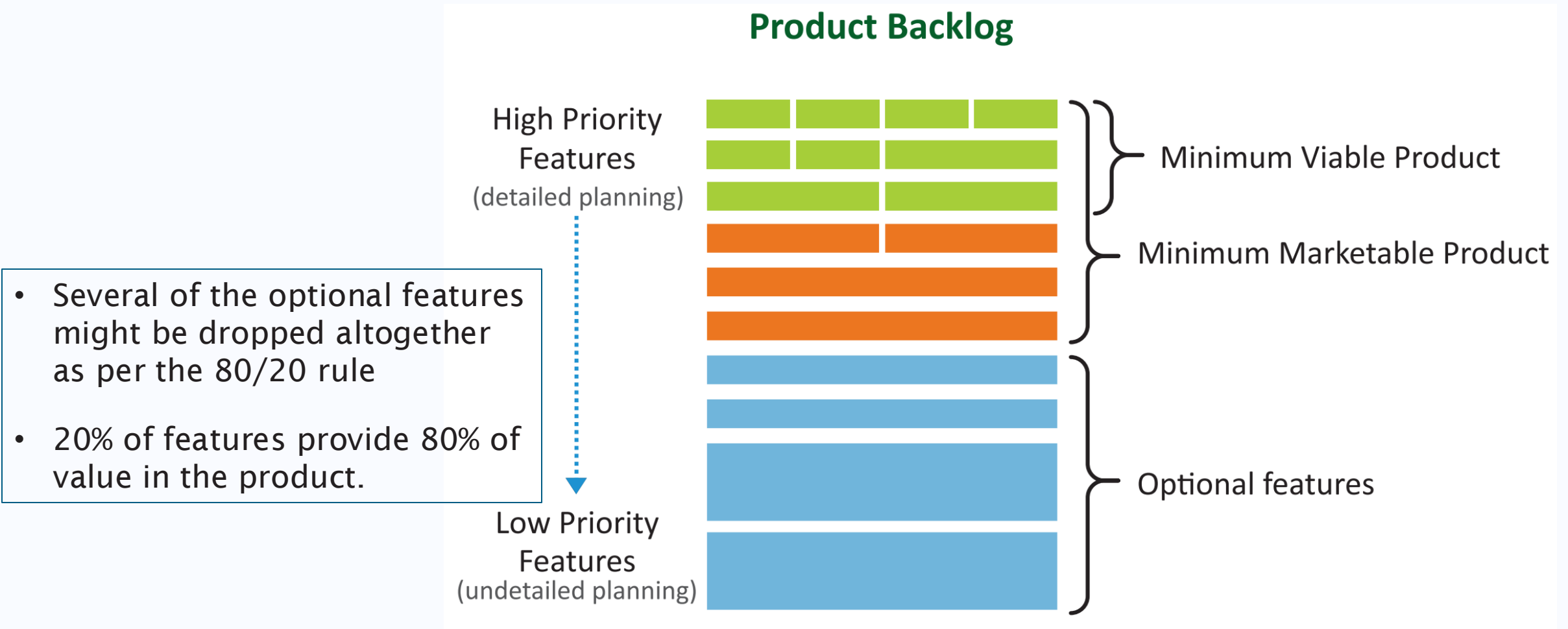
- **Minimum Viable Product (MVP)** includes those set of features in the product that help get early customer feedback.
 - All the product features are not needed to get such a feedback.
 - Hence, MVP is only a subset of the total product features that are needed for selling the product.
 - Final product release will have a greater number of features than MVP
- MVP is similar to a **mock-up** or **proof of concept**, as the purpose is to get customer feedback.
 - This ensures that the final releasable or marketable product has all the required features as expected by the customer.

Minimally Marketable Product (MMP)

- **Minimally Marketable Product (MMP)** is the list of those features that form the core functionality of the product.
- These are the mandatory features that are needed to take the product to the market.
 - For example, for an ATM machine, the ability to withdraw money from it is its MMP. The ability to allow customised shortcuts to customers to perform their transactions faster, is not an MMP.



Product Backlog and MVP



Requirements Gathering



Agile **requirements gathering** is not a **single stage**. It happens throughout the lifecycle.



In the beginning, a **few requirements that are already known** are captured and work on them is begun.



While the work is carried out, **more requirements emerge** and are included in the Product Backlog.



Requirements are generally written as **User Stories**, which are discussed below.

User Stories

User Stories are bite-sized, understandable chunks of business functionality.

- They are the preferred way of documenting requirements in Agile.

The most commonly used format for making a user story uses three pieces of information

- **Role**, **functionality**, and business **benefit**.

It is written

- “As a Role, I want Functionality, so that Business Benefit”

As an example, for an eCommerce portal, one of the user stories could be –

- “As an **Online Shopper**, I want to **Search for products**, so that I can make the **right selection**”

UserStories for Non-Functional Requirements

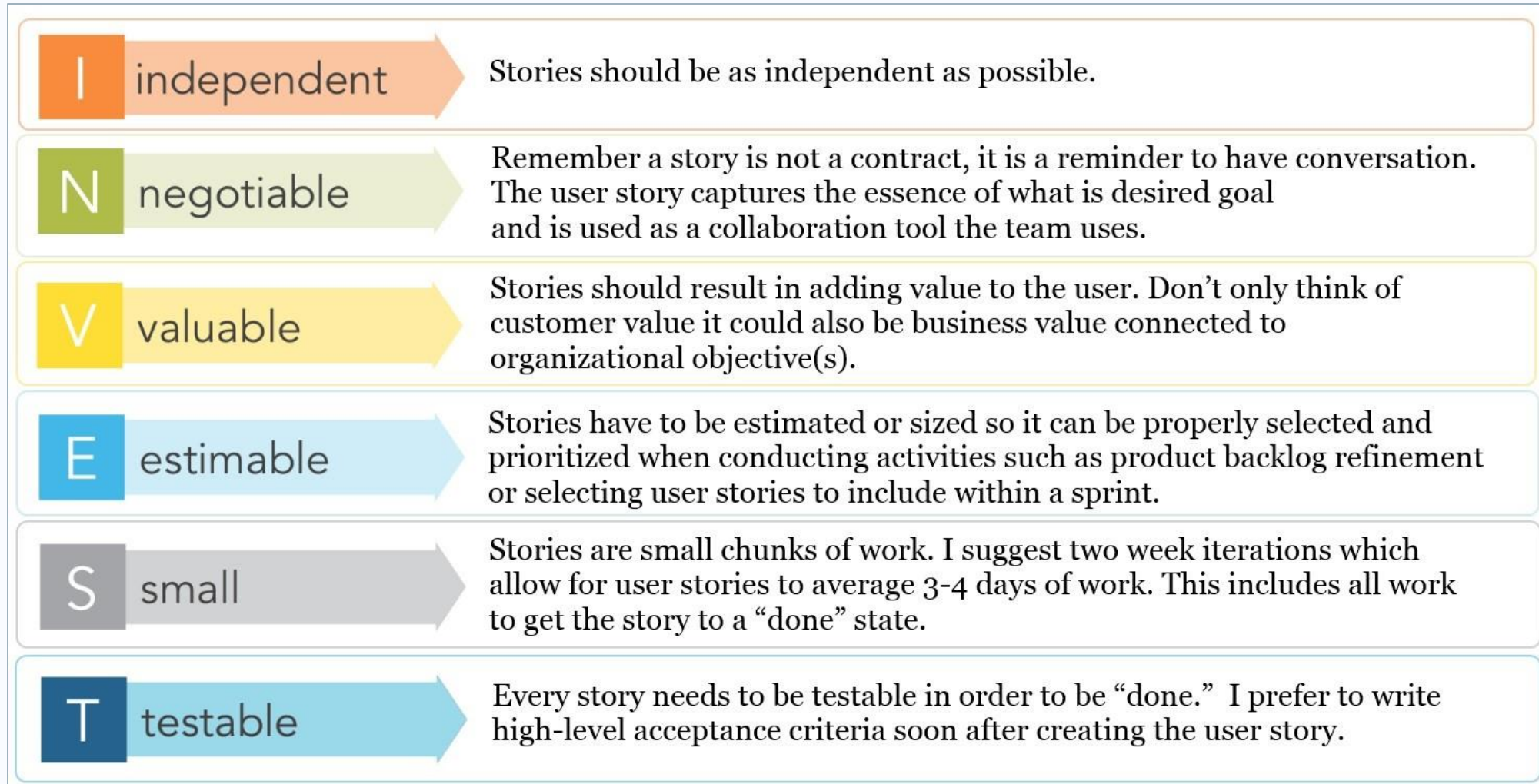
- An alternative format of documenting user stories, especially used for non-functional requirements is:
 - “Given the **pre-condition**, when an **action is performed** by the user, then what is the **action taken by the system**”
- For example:
 - “Given the user is already registered, when he enters his login- id and password, then he should be logged-in within 3 seconds”
 - This is an example of a performance requirement of the system.

User Story and Technical/Implementation Details

- A user story usually should not include **technical** or **implementation** details.
- For example, a user story like:
 - “As a User, I want My Name stored in the *database*, so that I can retrieve it later on”

is not a good user story.
- A better user story would be,
 - “As a User, I want to enter my Name in the *system*, so that it will be available whenever I login to the system”.

Effective User Stories should have 6 characteristics called INVEST



The Three C's of User Stories

Card

- Stories are traditionally concise so than can be written on note cards.

Conversation

- Details behind the story come out during conversations with product owner/customer before the implementation work starts

Confirmation

- The acceptance criteria for the user story should be included so that team now know when the story is completed

MoSCoW Prioritisation Method

In Agile, it is important to prioritise requirements as well.

MoSCoW Prioritization Method



Must have: Non-negotiable product needs that are mandatory for the team.



Should have: Important items that add significant value.



Could have: Nice to have items that will have a small impact if not provided.

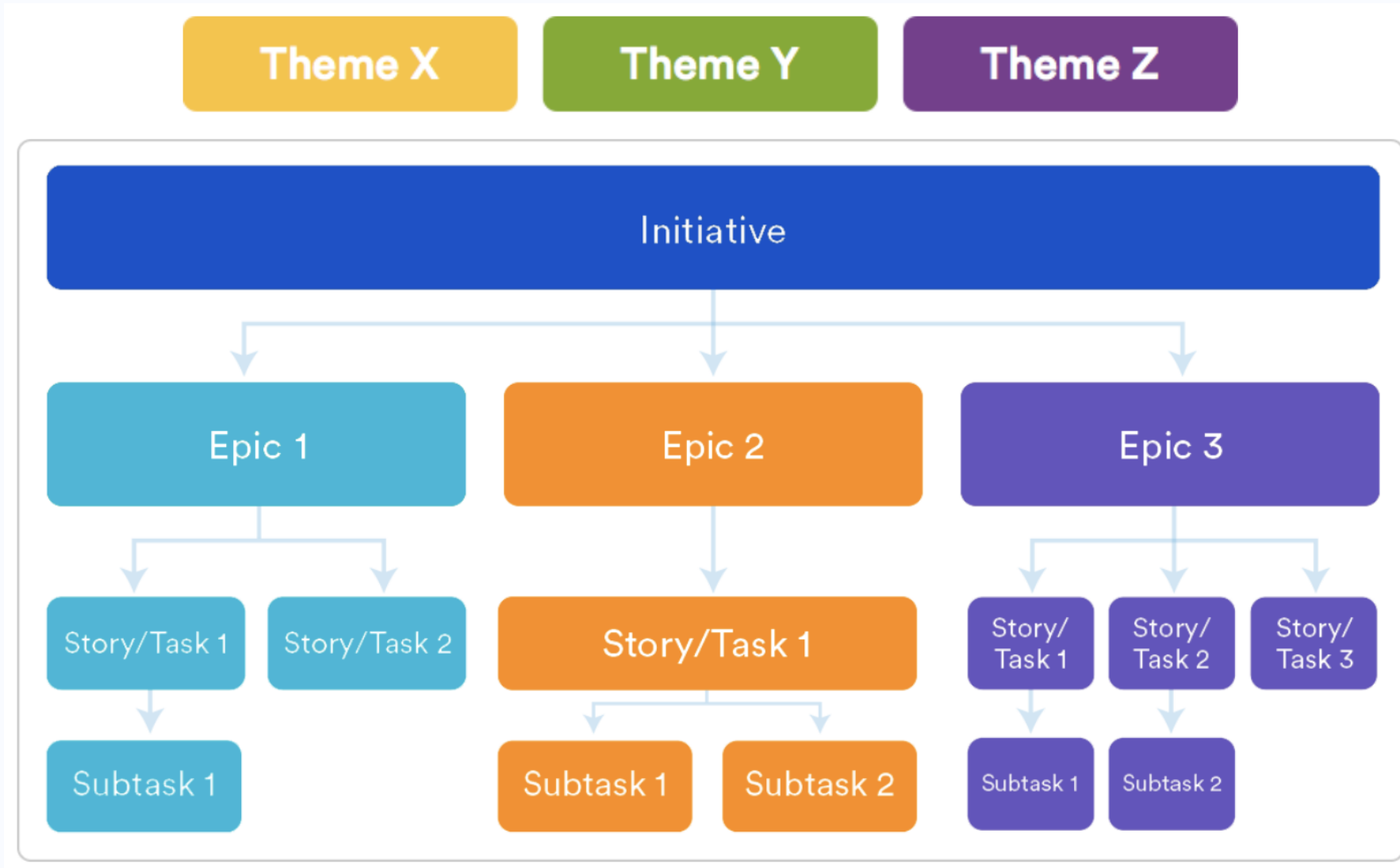


Will not have: Items that are not a priority for this specific time frame.

等级制度.

主动权.

Requirements Hierarchy – Themes, Initiatives, Epics and Story



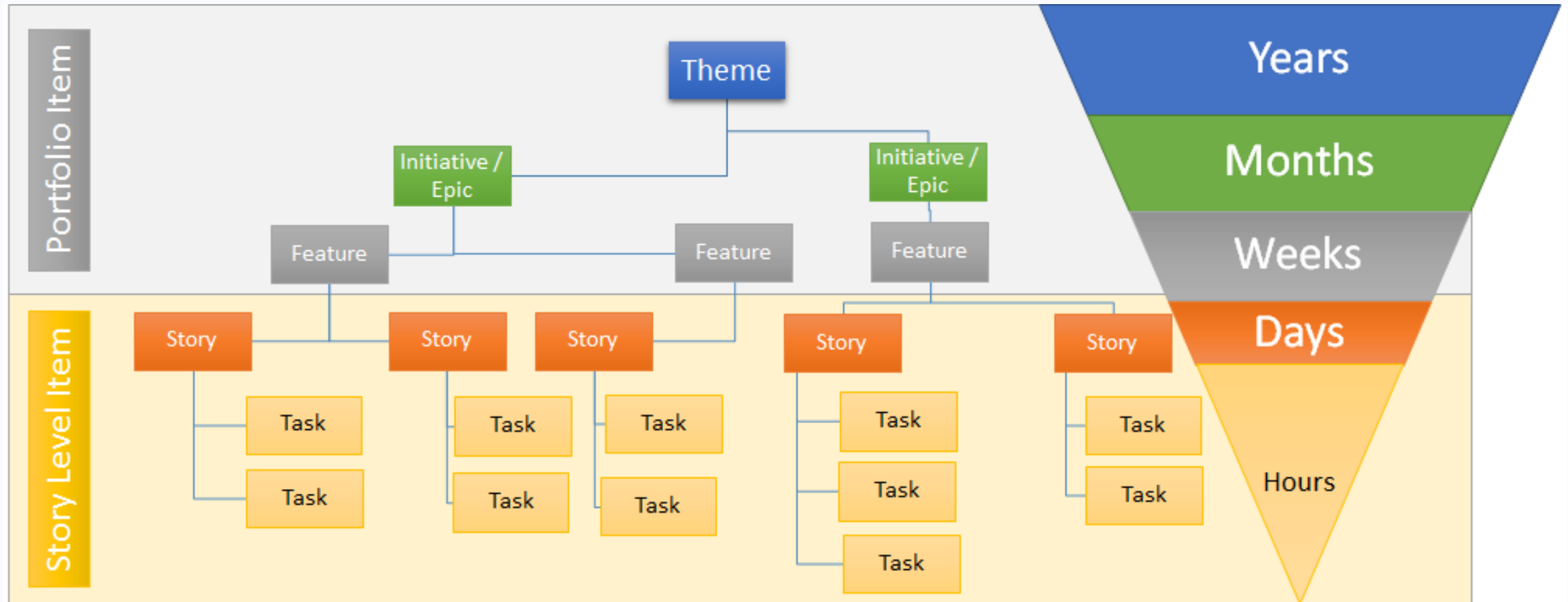
Themes, Initiatives, Epics and Story Cards

- In agile projects, we use the **Agile work structure**:
 - **Themes** are large focus areas that span the organisation.
 - **Initiatives** are collections of epics that drive toward a common goal.
 - **Epics** are large bodies of work that can be broken down into a number of smaller tasks (called stories).
 - **Stories**, also called ‘user stories,’ are short requirements or requests written from the perspective of an end user.

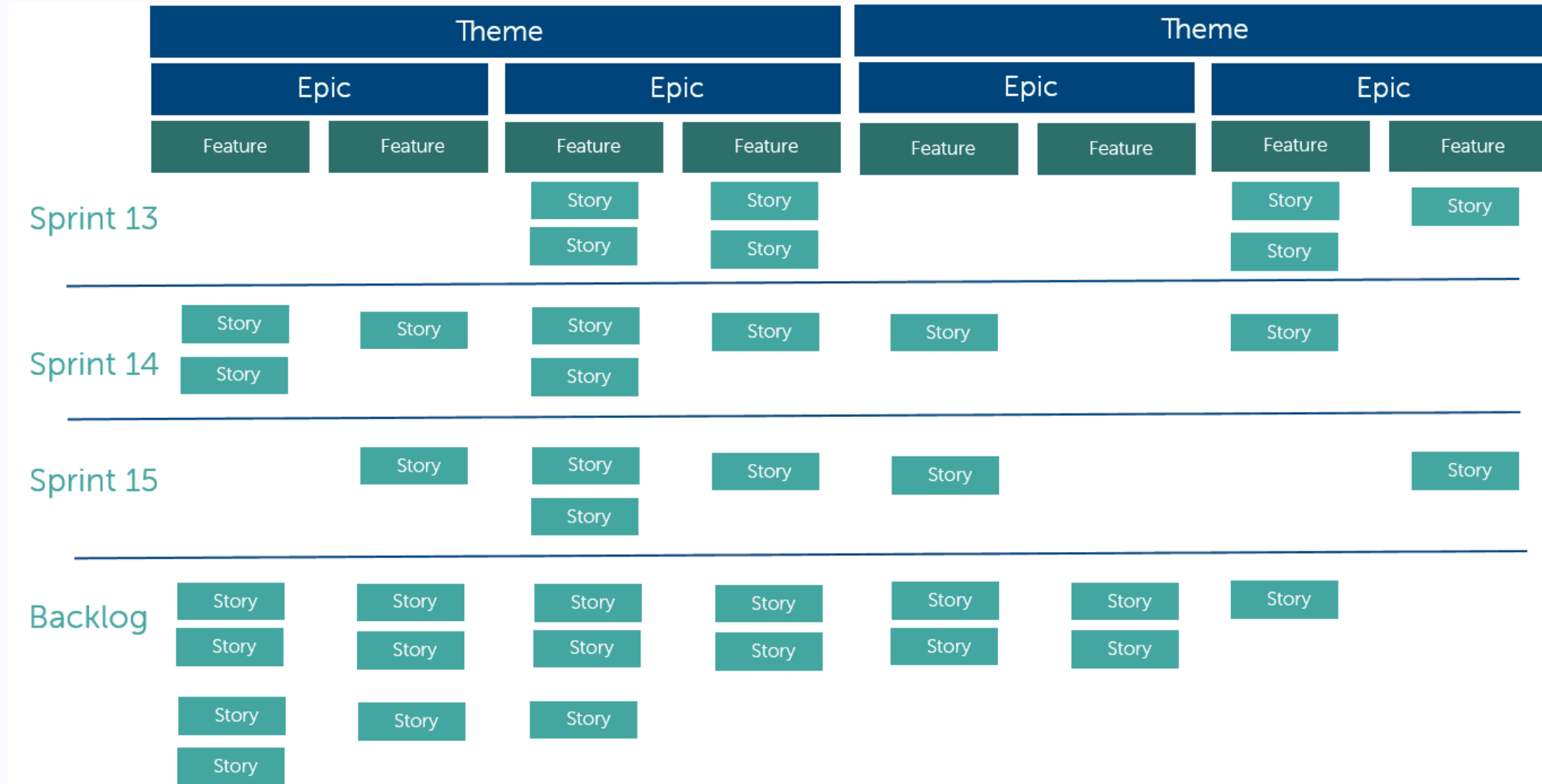
An Example

Themes	Initiative	Epic	Tasks
To increase revenue	Penetrating the PM software market	Develop a new module for workflow performance reports	PM tools research New features design Features development

Requirements Hierarchy



Work Structure vs Sprints



Relative Sizing & Story Points

- It is usually considered difficult to estimate the amount of work on a standalone basis. But it is comparatively easier to do relative estimation.
 - For example, if you are asked to say how much time a particular user story will take to build, it might be difficult to give an answer.
 - However, if you are asked to compare the **size of two user stories**, it is somewhat easier to come up with a more confident answer.
- **Story Point Estimation** is a relative estimation technique where the team estimates a story relative to another story.
- For this, the team selects a **baseline user story**, which is generally a small one, and its size is called 1 Story Point.
- Story Point is a unit that is **unrelated to time**. It is about the **size of the work**.

Agile Estimating Methods

- **T-shirt sizing**: S, M, L, XL, XXL
- **Planning poker**: Each team member estimates user stories with numbered cards (1, 2, 3, 5, 8, 13, 21)
- **The bucket system**: Extension of planning poker when there are many user stories, and the team is large
- **Affinity estimates**: Three steps:
吸引力
 1. Silent relative estimates (Sticky note “small” on the left and “large” on the right)
 2. Editing the wall (discussion done by group)
 3. Placing items into more specific sizes (S, M, L, XL, XXL)

Sample Relative Sizing Estimates

Team	User Story Name	Relative Size
A. Incentives	Determine new hire assessment content	M
	Develop hiring days event plan	L
	Develop hiring and retention survey	M
	Administer hiring and retention survey	S
	Analyze hiring and retention survey	S
	Research hiring and retention strategies	L
	Summarize hiring and retention research and survey results	M
	Draft new hiring policies	M
	Draft retention policies	M
B. Education	Determine content for Course 1 for new hires	L
	Develop content for Course 1 for new hires	XL
	Deliver Course 1 for new hires	XL
	Determine content for Course 2 for new hires	L
	Develop content for Course 2 for new hires	XL
	Deliver Course 2 for new hires	XL
	Research potential education partners	L
	Summarize education partner research	S
C. Adoption	Create website for hiring information	M
	Create assessment quiz for new hires	S
	Create website for Course 1	XXL
	Create website for Course 2	XXL
	Advertise for hiring days event on website and social media	M
	Advertise for hiring days event via radio, TV, signage, etc.	L

Ideal Time and Real Time Estimation

- When a user story is picked up for delivery in an iteration, it is usually broken down into **tasks** to make it easy to track.
- Each task may have a **time-based estimate** associated to it.
- When estimating time for each task, we may use **Ideal Time Estimation** or **Real Time Estimation**.
- In **Ideal Time Estimation** the entire day (8 hours) is available to **perform productive work** without factoring in any **interruptions**.
 - When we received the estimation from the team members, we calculate duration by factoring in **unproductive time**.
- In **Real Time Estimation**, we inform the team to **consider interruptions in their work**.
 - Hence, we do not need to do any further calculations once the estimate is received from the team.

Integration Planning for an Agile/Hybrid Project

- Planning for **agile** projects remains at a **high-level** for the **long-term**, but more detailed plans are created for the short-term.
 - Why? Because **change is expected**, and requirements can change after every iteration.
- Instead of putting detailed plans in writing, agile teams write down only what is **necessary** and have **discussions** to make sure everyone understands what is happening.
 - The **daily Scrum meetings**, the **sprint planning** meeting, the **sprint review**, and sprint **retrospectives** allow for necessary discussions and interactions.

回流的

Scope Planning for an Agile/Hybrid Project

- For **predictive** projects, the scope is defined at the **beginning** of the project.
- For **agile** projects, the scope is **not completely known** until the **end** of the project because the **customer** can **add** and **remove features** from the overall scope **at the start of every iteration**.
- During **backlog refinement** teams ***progressively*** elaborate and **reprioritise** the work to determine what can be accomplished during that iteration.
- **New features** can be **added** at any time to ensure that projects deliver the most value.

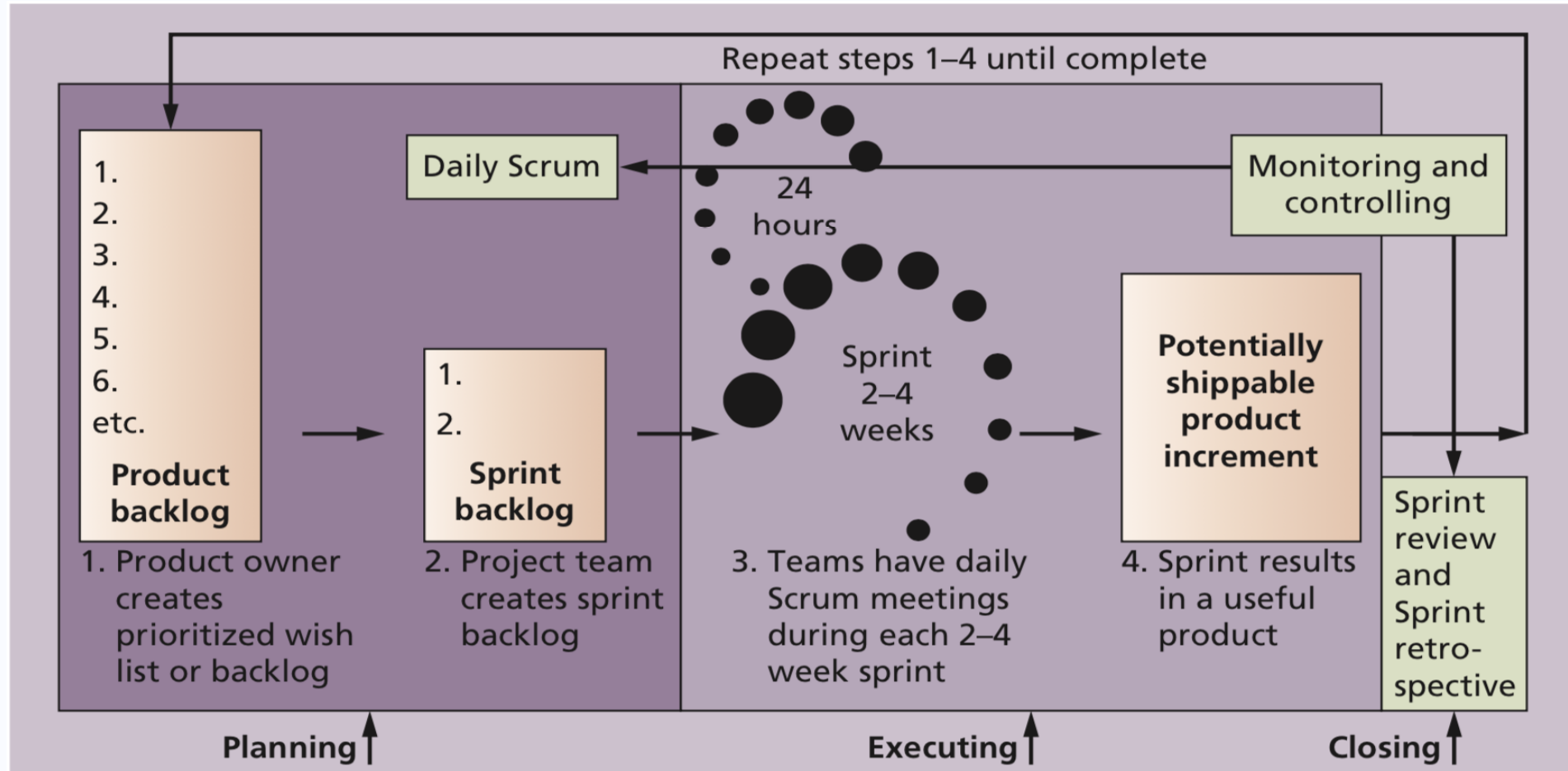
Schedule Planning for an Agile/Hybrid Project

- Instead of creating a **detailed schedule** for **all of the activities required** to complete an entire project, agile teams focus on the *most valuable work* they can complete within each iteration.
- This approach is often called *time-boxing*.
 - A *timebox* is a previously agreed upon time period during which a team works towards completion of a goal.
 - A **sprint**, for example is a timebox of 30 days or less.

exam

Scrum Framework and Process Groups

purpose



What About Dependencies?

- Ideally, one Scrum team can perform all the work in their *scrum backlog*.
 - If there are dependencies within backlog items, the team should identify those and work on them accordingly.
- When there are multiple Scrum teams, you can hold a **Scrum of Scrums**, where representatives from each team meet to *coordinate efforts* and dependencies.

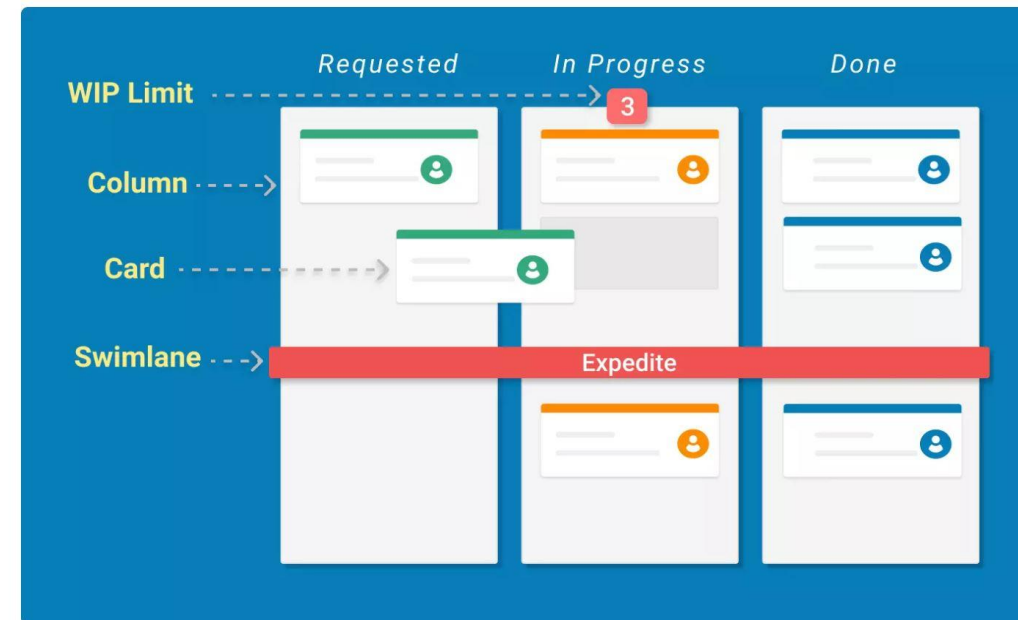
Kanban Method

- The term *kanban* is a Japanese term meaning *visual board*.
- The **Kanban method** is a means to **design**, **manage**, and **improve** flow systems for knowledge work.
- It also allows organisations to start with their existing workflow and drive evolutionary change by visualising their flow of work, limit work in progress (WIP), and **stop starting** and **start finishing**.
- The Kanban method does not **use time-boxing**, does not define any specific roles for the team, and focuses on cycle time.

Sample Kanban Board

Main Components of the Kanban board

Kanban boards use Card, Column, Swimlanes, and WIP Limits to enable teams to visualize and manage their workflows effectively. Let us introduce you to the main components more closely:



Source: <https://kanbanize.com/kanban-resources/getting-started/what-is-kanban-board> (2021)

Kanban Board – Elements

- **Kanban Cards** – This is the visual representation of tasks. Each card contains information about the task and its status, such as deadline, assignee, description, etc.
- **Kanban Columns** – Each column on the board represents a **different stage** of your **workflow**. The cards go through the **workflow** until their full completion.
- **Work-in-Progress Limits** – They **restrict** the **maximum** number of tasks in the different stages of the workflow. Limiting **WIP** allows you to finish work items faster by helping your team focus only on current tasks.
- **Kanban Swimlanes** – These are horizontal lanes you can use to separate different **activities**, teams, **classes** of service, and more.

Cost Planning for an Agile/Hybrid Project

- Unlike predictive projects, there is no *total project budget* or *detailed cost estimate* for the entire project for agile project.
- There is some estimating involved when using an agile approach, but instead of using *hours* or *Pounds*, most estimates are done in a *relative* fashion.
- *Relative estimates* are created by *comparing* work or grouping it by equivalent difficulty based on factors like *risk*, *complexity*, and *required labour*.

Monitoring And Controlling Agile/Hybrid Projects

- In agile, work **progress** and results are monitored in each **Daily Scrum** and in each **Sprint Review**.
- In case of **deviations** from the plan, one can quickly initiate actions , i.e. controls, such as rescheduling, possibly adapt the procedure and learn from it (inspect, adapt and learn).
- **Burn charts** show project team velocity.
- **Velocity** measures the **productivity** rate at which the deliverables are produced, validated, and accepted within a predefined interval.

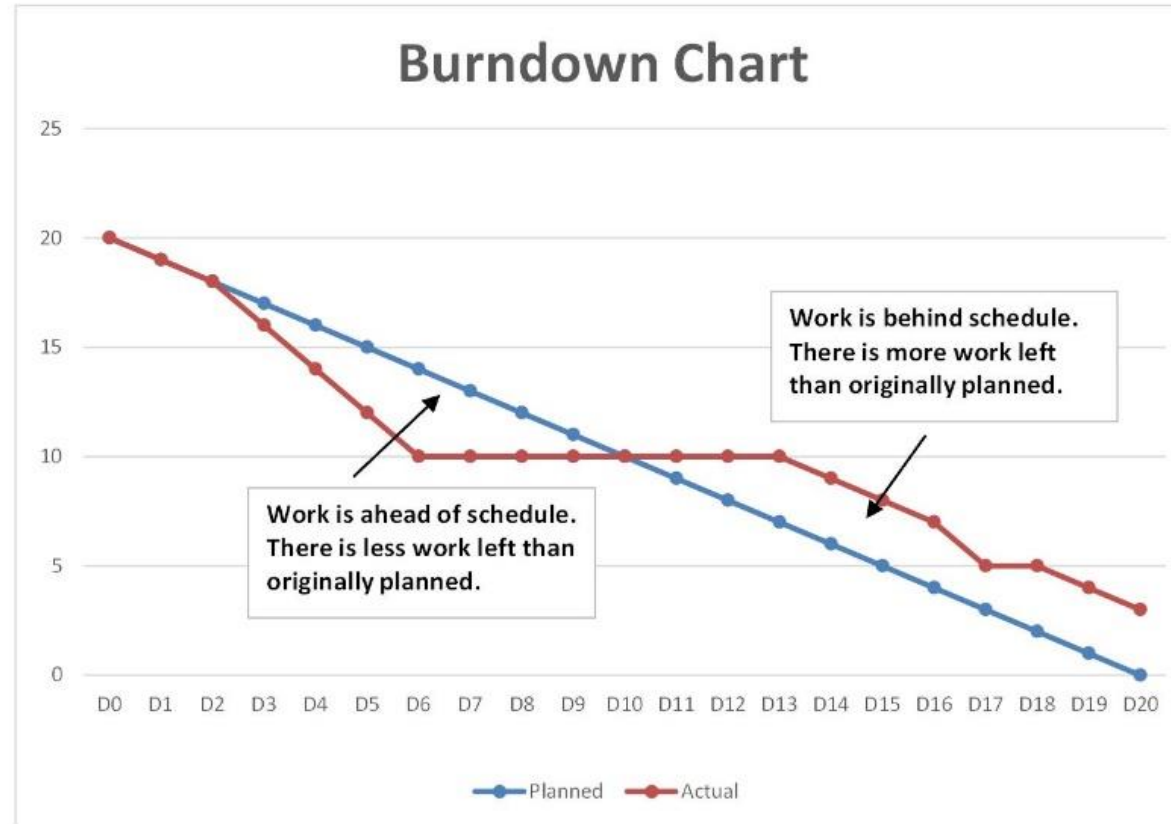
Monitoring And Controlling Agile/Hybrid Projects

- You can create the following types of burn charts:
 - **Burndown charts** show the amount of work (number of tasks) remaining compared to the plan. They are often used for each sprint and discussed during sprint retrospectives.
 - **Burnup charts** show the amount of work (tasks) completed compared to the plan. They can be used during each sprint, and they can also show progress for several sprints.
 - **Combined burn charts** show how much work has been completed and how much remains.

Sprint Burndown Chart

Number of Tasks

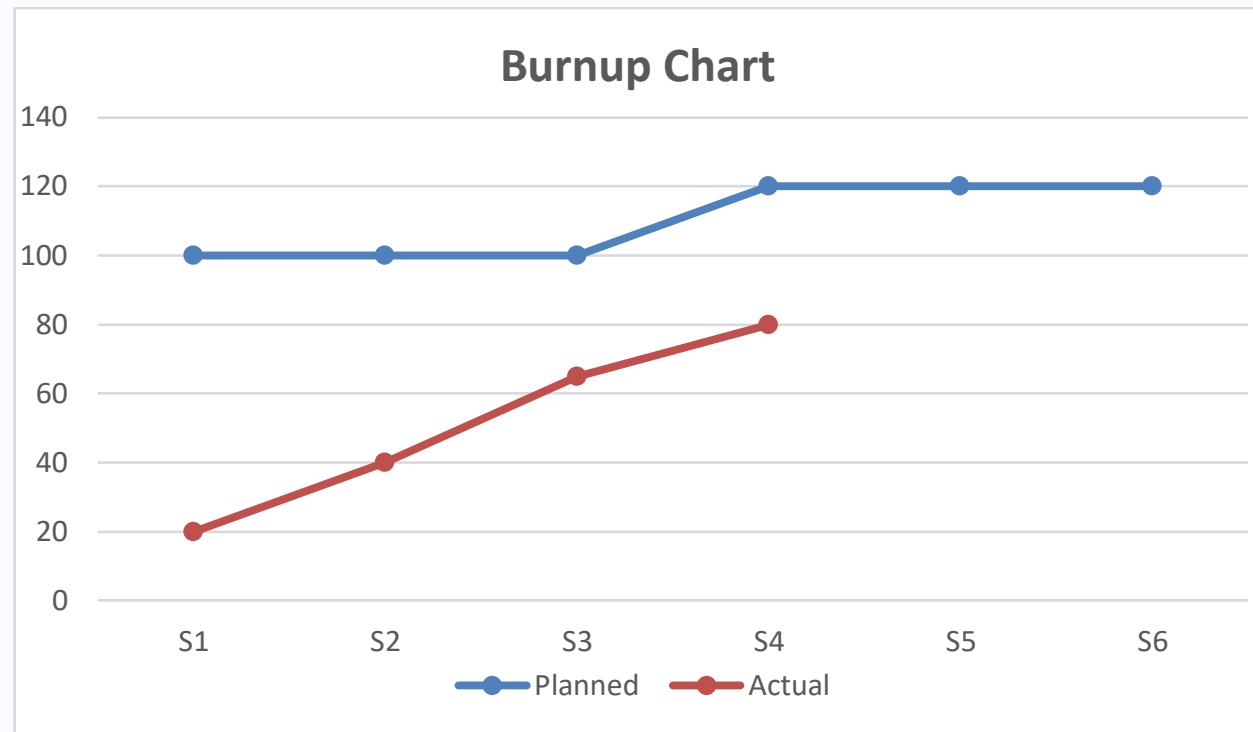
Days	Planned	Actual
D0	20	20
D1	19	19
D2	18	18
D3	17	16
D4	16	14
D5	15	12
D6	14	10
D7	13	10
D8	12	10
D9	11	10
D10	10	10
D11	9	10
D12	8	10
D13	7	10
D14	6	9
D15	5	8
D16	4	7
D17	3	5
D18	2	5
D19	1	4
D20	0	3



Sample Burnup Chart

Number of Tasks

Sprints	Planned	Actual
S1	100	20
S2	100	40
S3	100	65
S4	120	80
S5	120	
S6	120	



Velocity Charts



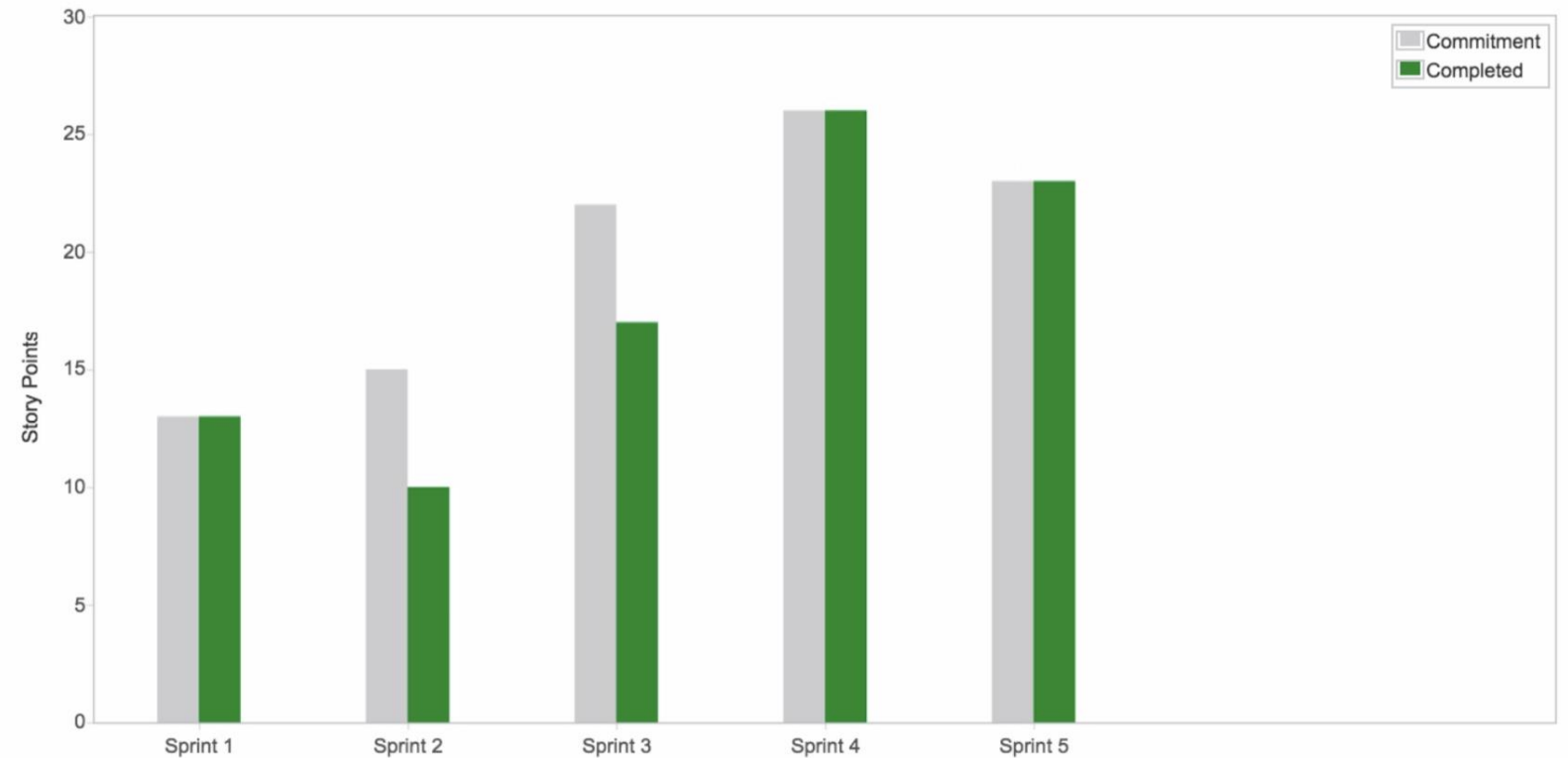
- Sprint teams use **velocity** to measure how much work they can complete in each **iteration**.
- It is widely used to help teams create accurate and efficient timelines.
- Sprint team velocity is not constant; rather, it varies.
 - Note that velocity charts are not intended to be a tool for monitoring the team.
 - They are most useful as a tool for release planning.
- The velocity chart is created after the first sprint and updated after each completed sprint.

Velocity Charts – Cont.

- The velocity of the sprints helps managers to calibrate the release plan.
- The Velocity Chart shows the amount of value delivered in each sprint, enabling you to predict the amount of work the team can get done in future sprints.
- It is useful during your sprint planning meetings, to help you decide how much work you can feasibly commit to.

Sample Velocity Chart

Velocity Chart



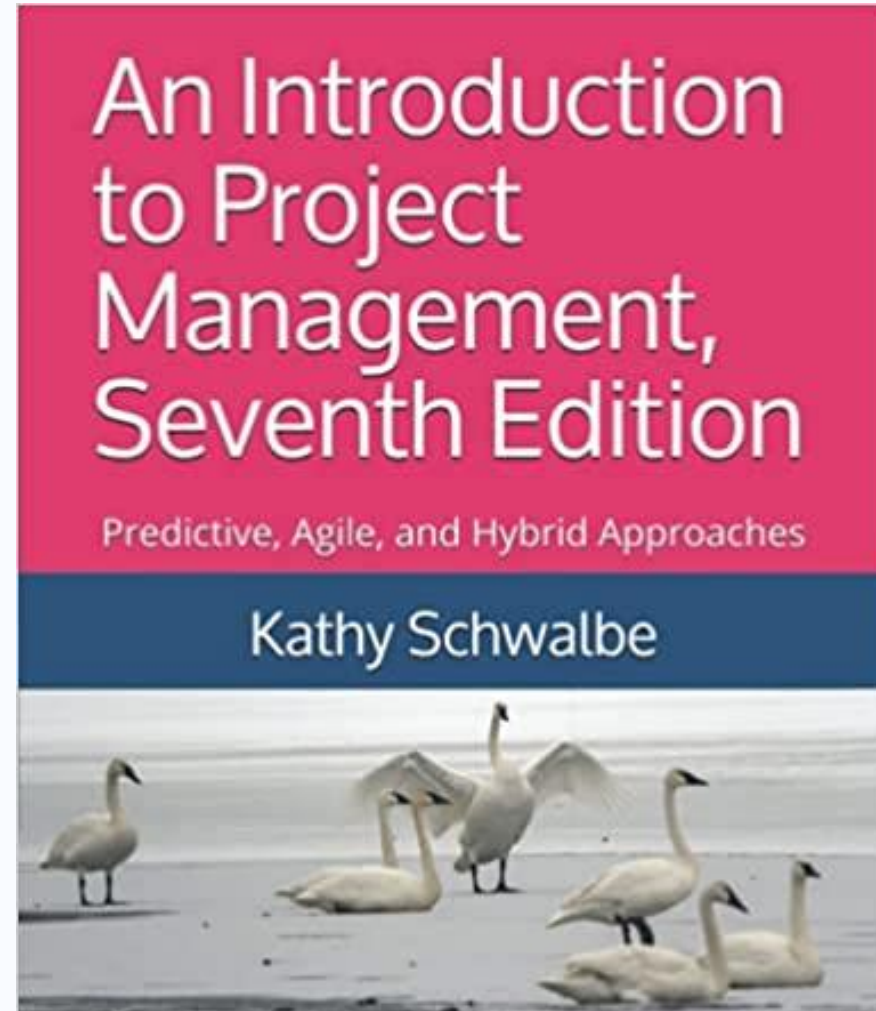
Sprint	Commitment	Completed
Sprint 1	13	13
Sprint 2	15	10
Sprint 3	22	17
Sprint 4	26	26

Closing Agile/Hybrid Projects

- Just like **predictive** projects, **agile and hybrid** projects should be closed.
 - **Hybrid** projects can use any of the project closing processes listed earlier.
- A strength of the Scrum events, if used on agile projects, is an intentional moment of closure, as follows:
 - **Daily Scrum**: You can think of these meetings as providing **closure** for the day before.
 - **Sprint reviews**: Sprint **reviews** provide **closure** for sprints, and sometimes entire projects.
 - **Sprint retrospectives**: This event is similar to a **lessons-learned**, but it only focuses on a particular sprint. When a project ends, teams should hold a **final retrospective** to focus on **all lessons learned**.
- Teams should still hold a **close-out** meeting and celebrate!

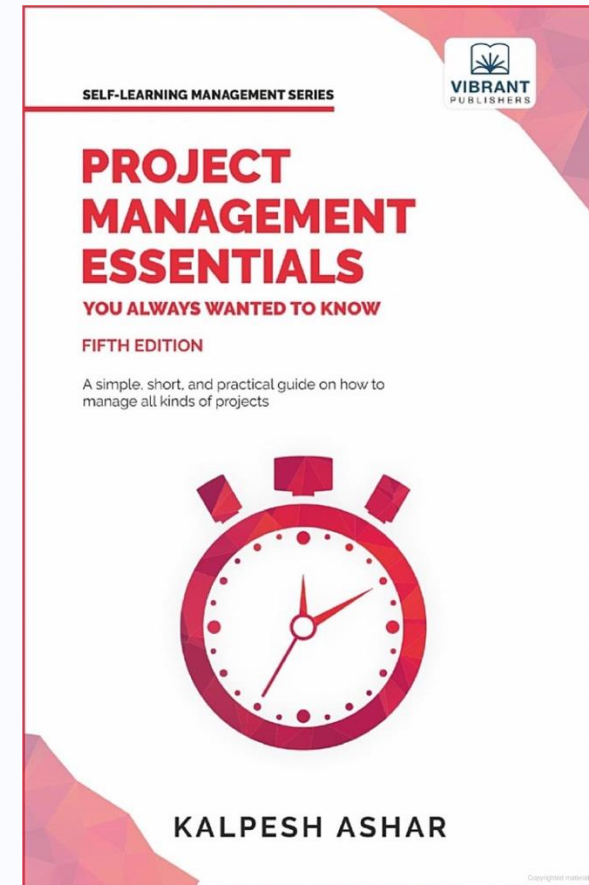
Reference

- Chapter 5:
Planning Projects,
Part 2 (Schedule and
Cost Management)



Reference

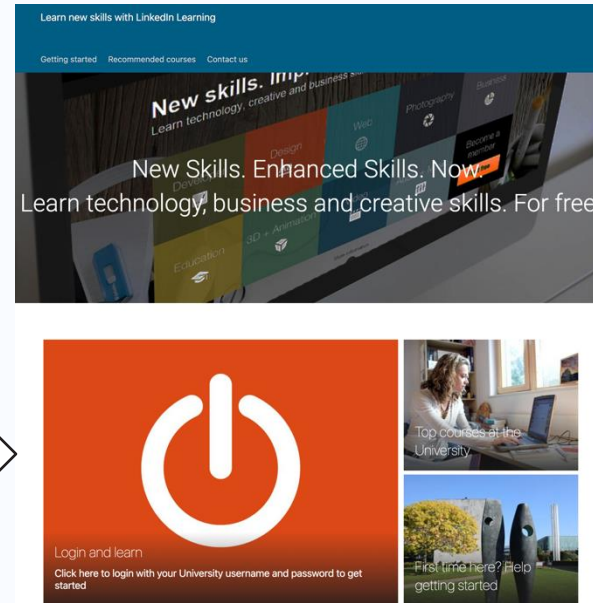
- Chapter 4 of:
Project Management Essentials You Always Wanted To Know, 5ed



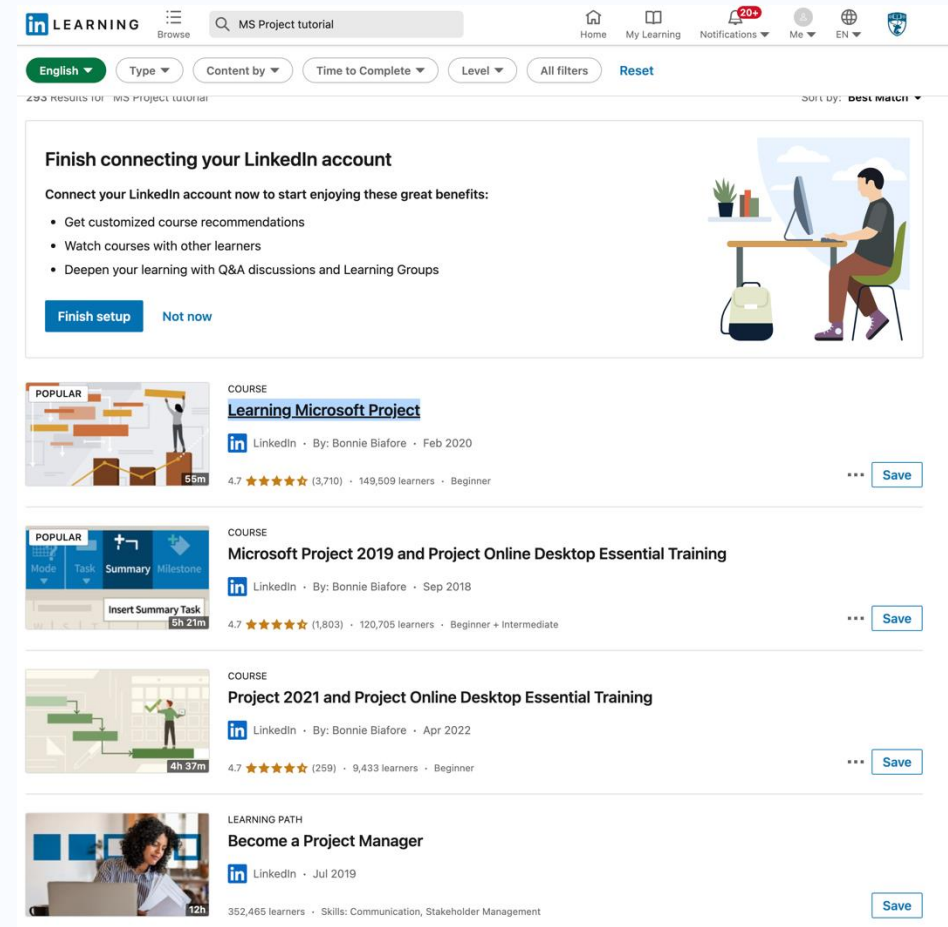
Resources

- Soton LinkedIn Learning - [Learn new skills with LinkedIn Learning](#)

Login using
your
University
Credentials



Search For Project Managment



YOUR QUESTIONS