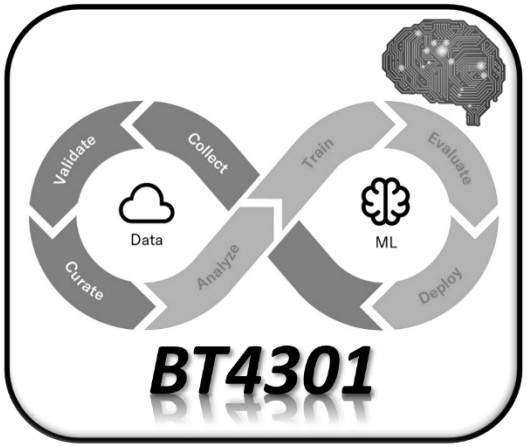


Lecture 3

Using Kanban inside Scrum.

Scrum, Kanban and Scrumban

BT4301 – Business Analytics Solutions Development and Deployment
AY 2023/24 Semester 2

Lecturer: A/P TAN Wee Kek

Email: tanwk@comp.nus.edu.sg :: **Tel**: 6516 6731 :: **Office**: COM3-02-35

Consultation: Wednesday, 9:30 pm to 10:30 pm. Additional consultations by appointment are welcome.

Learning Objectives

- At the end of this lecture, you should understand:
  - Scrum
  - Kanban
  - Scrumban

Readings

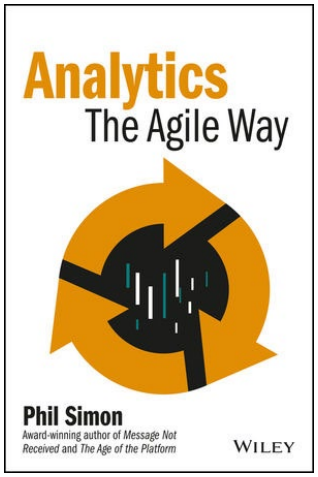
- Reference Book 2:

  Analytics: The Agile Way

  Author: Phil Simon

  1st Edition (2017), Wiley

  https://linc.nus.edu.sg/record=b3751448



Readings (cont.)

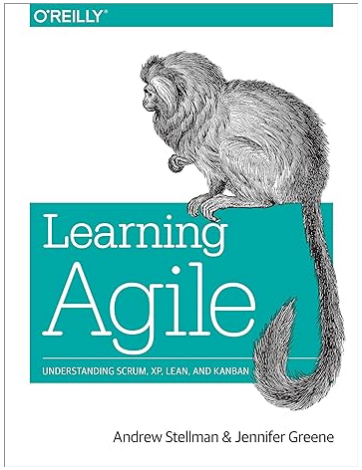
- Reference Book 3:

  Learning Agile: Understanding Scrum, XP, Lean, and Kanban

  Authors: Andrew Stellman and Jennifer Greene

  1st Edition (2014), O'Reilly Media

  NLB Link

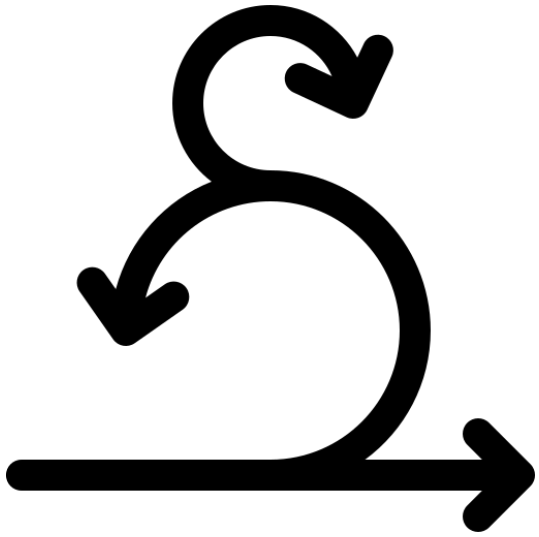


Readings (cont.)

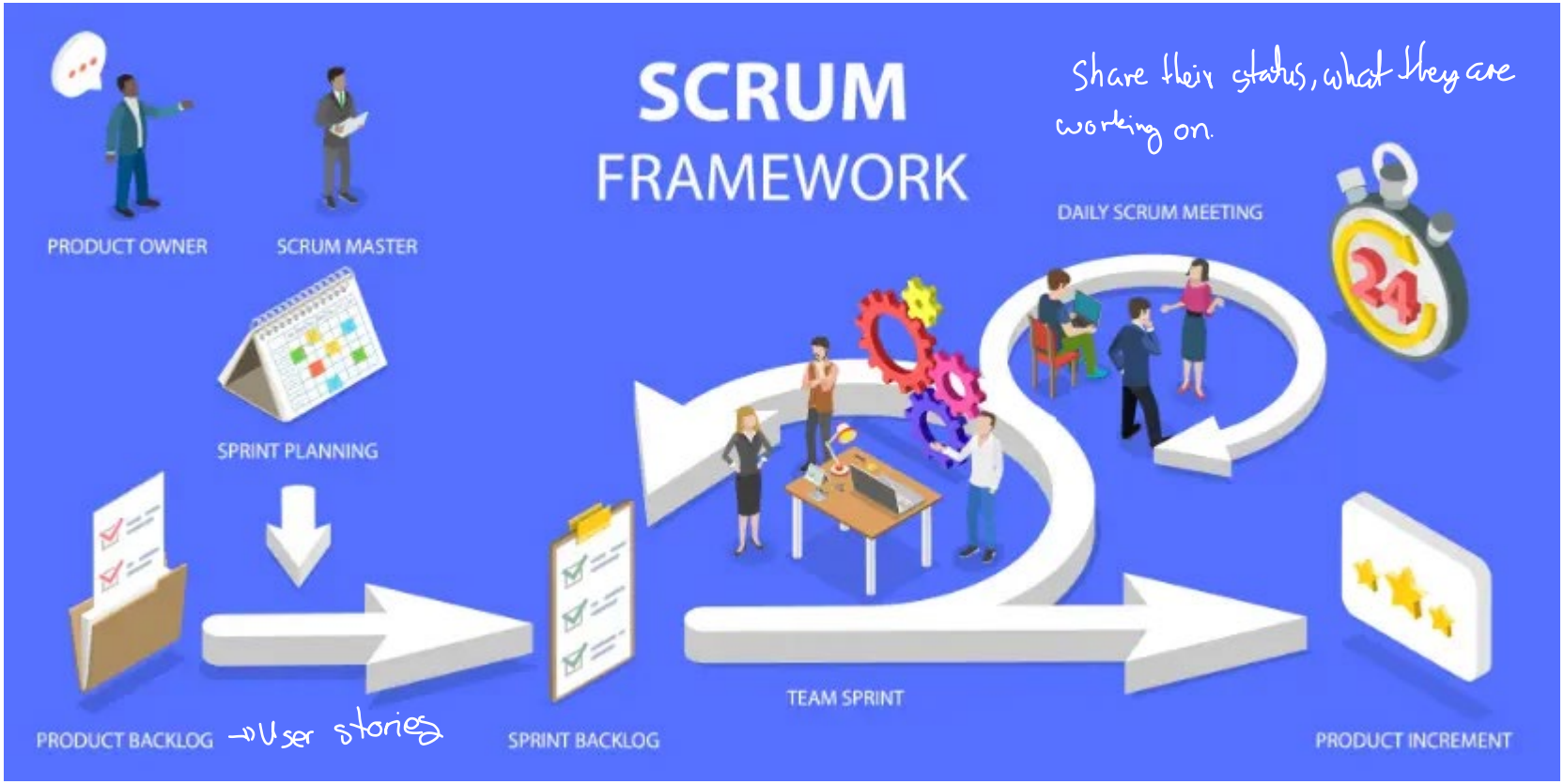
- Required readings:
  - Reference Book 2 – Chapter 5
- Suggested readings:
  - Reference Book 3 – Chapter 4 and 5

What is Scrum?

- **Scrum** is an **agile** methodology:
  - **Long definition** – A simple yet incredibly powerful set of principles and practices that help teams deliver products in short cycles, enabling fast feedback, continual improvement and rapid adaptation to change. ↳ from team & stakeholders.
  - **Short definition** – A simple, team-based framework to develop complex systems and products.



What is Scrum? (cont.)



Scrum Team

- The structure of a Scrum team is known for its simplicity with just three roles:

Very flat org. / no hierarchy

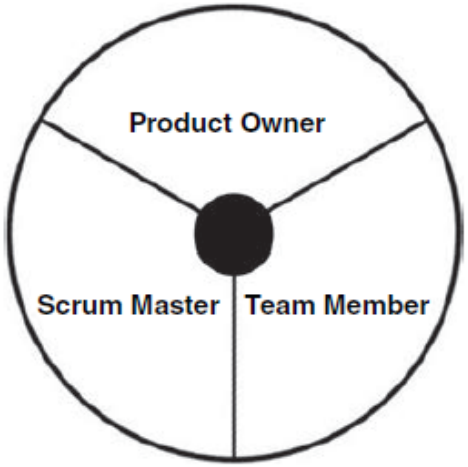


Figure 5.1 Simple Visual of Scrum Team Makeup
Source: Phil Simon.

- Each of these roles come with specific responsibilities.
- There is no hierarchy and members are treated equal regardless of job title.

Product Owner

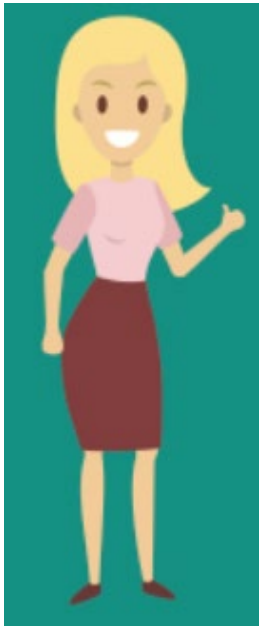
- **Product Owner** is the main business stakeholder who owns the product.
- The main person to ask the team to:
  - Do work.
  - Prioritize user stories.
  - Change the priority of product backlog items.
- Other responsibilities:
  - Ensures team members understand customer needs.
  - Maintain the product vision.
  - Maximizing return on investment.
- Demanding role although non-coding.

Product Owner (cont.)

- Must remain objective while managing any inherent tensions within the team.
- Writes acceptance criteria – Acceptance criteria will be discussed in subsequent slides.

Scrum Master

- **Scrum Master** is the main expert, advisor, coach and facilitator.
- Scrum Master may say "No" to the Product Owner:
  - Should have the maturity, experience, knowledge and humility to function effectively.
  - Certainly not a role that fresh graduate should undertake.
- Might sometime need to break impasse.
- Might gradually reduce involvement as team:
  - Matures and coheres.
  - Gradually becomes more productive.
  - Develops a better understanding of Scrum.



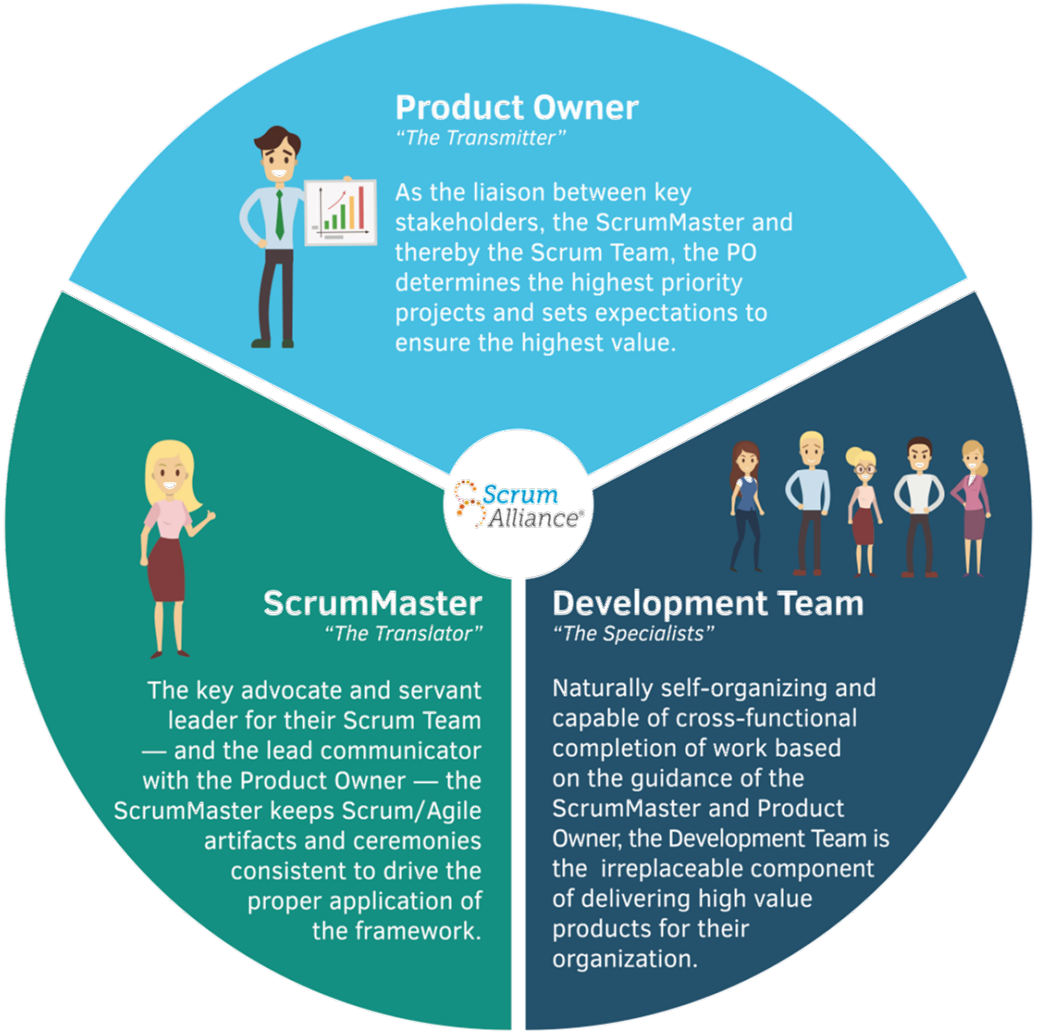
Team Members

- **Team Members** constitute the majority of Scrum team.
- Team Members:
  - Operate with high degrees of authority.
  - Determine estimate (time and effort to complete each task) on their own. → how difficult is a user story?
  - Collaborate freely, ideally, without regard to title/role.
- Committed to deliver user stories continuously.
- An effective team:
  - High degree of collaboration.
  - Individuals who are overly possessive and selfish can affect productivity.

Team Members (cont.)

- Should team be cross-functional?
  - This issue is debatable but generally should be the case.
  - Having all programmers in a team is not ideal.
  - Good to have other roles such as business analysts, quality assurance testers and UI/UX designers.
  - Analogy – Just like a soccer or basketball team, which requires players in specialised roles.

Composition of a Scrum Team



Source: https://resources.scrumalliance.org/Article/how-scrum-scrum-teams-work

User Stories

- Gathering business requirements has always been a painful process:
  - Analysts acting as third party.
- Scrum adopts **user stories** instead:
  - First party point of view.
  - Capture the descriptions of desired features from the point of view of employees, customers, clients, or other end users.
  - User stories help create simplified and understandable descriptions of business requirements.

Non scrum.

User Stories (cont.)

- All user stories should contain each of the following <u>three</u> elements:
  - Role – Type(s) of users
  - Goal – What they want
  - Benefit – Why

Who?
What?
Why?

- Example from Netflix:
  - Subscribers could download movies to their devices for future viewing.
  - As a Netflix subscriber, I would like the ability to download shows and movies so I can watch them in areas with no or poor network connectivity.

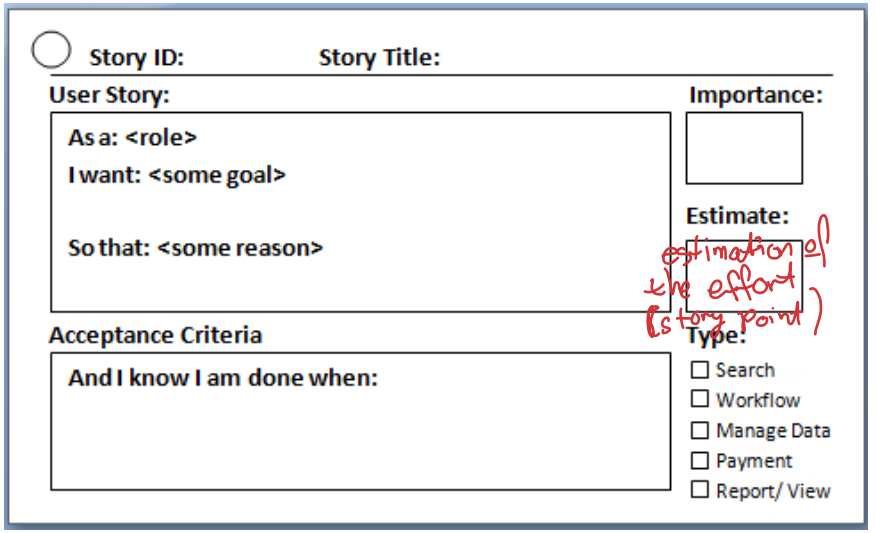
Developers/ understand the painpoint.
UX

User Stories (cont.)

- User stories intentionally omit the "how":
  - Should specify neither the technologies nor the data required to solve the business problem.
  - Scrum Team Members normally perform technologies-related tasks.
  - Team Members should also maintain a high degree of autonomy in tackling their user stories.
- Clear, concise user stories are critical successful factors of all agile projects:
  - Including analytics projects.

User Stories (cont.)

- The 3 Cs of user stories
  - **Card**:
    - Written on card, having just enough description of the requirement.
  - **Conversation**:
    - A collaborative conversation facilitated by the Product Owner with the team.
    - This is an in-person conversation.
  - **Confirmation**:
    - Acceptance criteria.
    - Determine when the user story has met the goal of the user.



Epics

- **Epics** are user stories that a Scrum team cannot complete during a single sprint: they are bigger scale.
  - Usually combine many smaller user stories into one.
- Why is this a problem?
  - If a user story qualifies as an epic, it could suffer from problems associated with waterfall projects.
- Two examples of epics:
  - As a supply-chain manager, I need to see all inventory data throughout the company, receive alerts when inventory levels for key products fall below certain thresholds, understand my delivery network via geoanalytics, and view information regarding potential substitutes if weather delays or union issues prevent normal raw-material production.

multiple user stories

Epics (cont.)

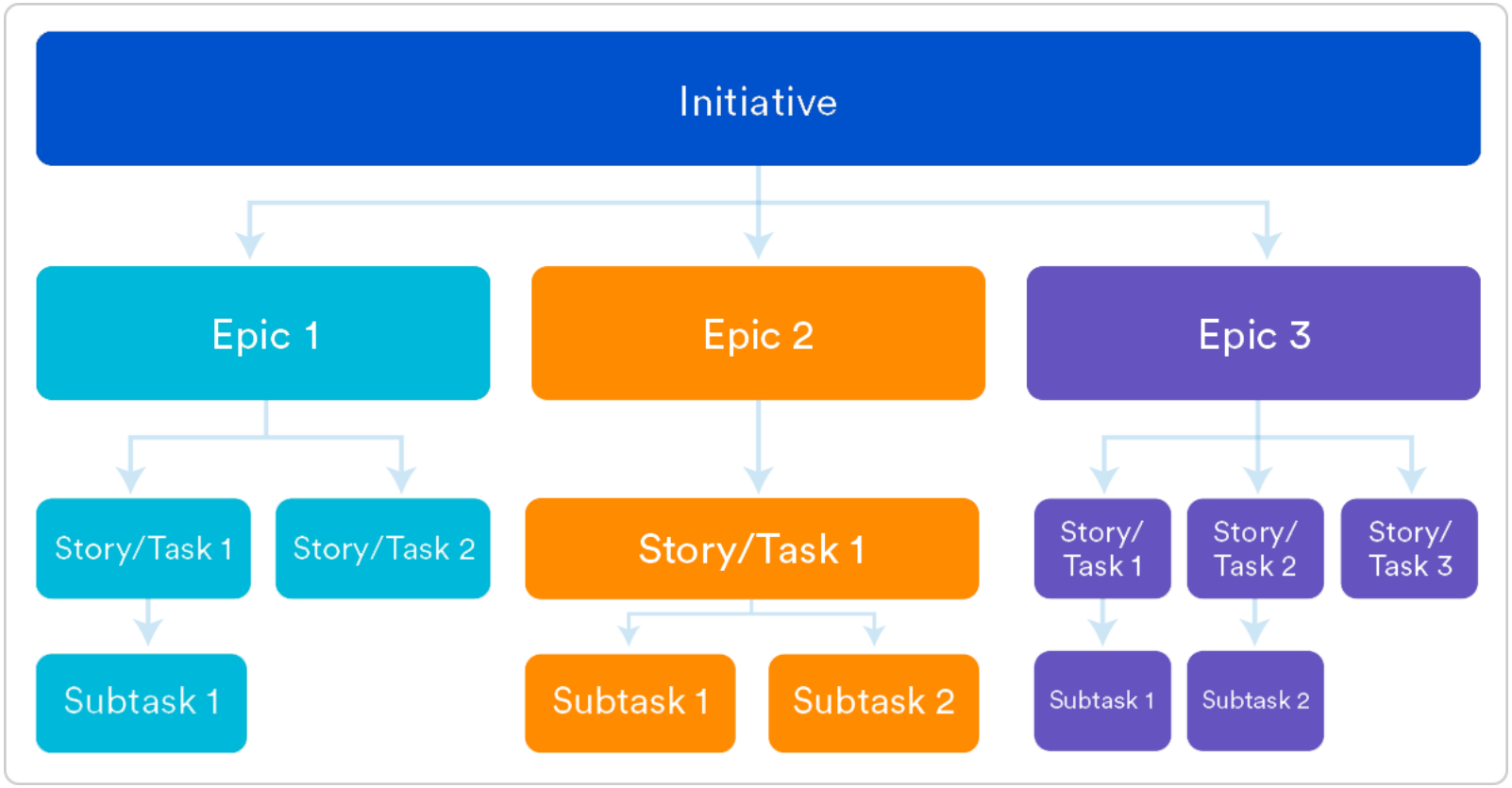
- As the head of training and development, I need to view employee survey results in real time to immediately send struggling managers emails equipped with links to resources and online training programs. I also need to see which managers have taken which courses and how they performed.

- Reflections:
  - There is no doubt that each of these tasks is important to the respective users.
  - But it is better to separate each task into several user stories that Scrum team can complete over the course of one or two sprints.

If you look at Epochs as a way of classifying user stories, it is OK
If not, it is better to breakk it up.

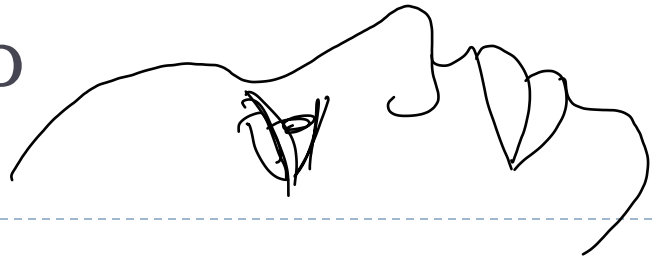
Another Point of View on Epic

- Initiative, epic and story are just agile terminologies for organizing use cases.



Source: https://www.atlassian.com/agile/project-management/epics-stories-themes

User Stories that are Too Narrow/Detailed



- Some user stories are too granular.
- Two examples:
  - As the head of sales, I need to view each prospect's name, lead origin, lead date, address, and zip code in an Excel spreadsheet sorted by lead origin so I can run mail merge in Microsoft Word. → not good, b.c. this technology might not be the correct one.
  - As the head of analytics, I need the ability to run multivariate regression equations and post the results on my company website, allowing nontechnical users to import the results in Tableau for future visualization and analysis. → too specific!

Just Right User Stories

- Just right user stories:
  - User stories should pass the **Goldilocks Test**, i.e., just the right amount of information.
  - They are neither too broad nor too narrow.
- Two examples:
  - As the head of talent management, I need to view attrition rates of high-potential employees to determine whether they are leaving at an accelerated rate.
  - As a bookstore owner, I need to see real-time inventory levels and trends to determine which books to order.

The Spike – A Special User Story

- All user stories inhere some degree of risk regardless of their required effort or anticipated complexity:
  - Agile methods explicitly reject the conceit of waterfall methods.
  - In a waterfall project, the project manager knows precisely how long each item will take.
- But risk is not binary but rather on spectrum of degrees:
  - Decisions about development framework, programming languages and statistical techniques should not be taken lightly.
  - May not be permanent but not easy to reversed.

The Spike – A Special User Story (cont.)

- For big, risky decisions, spikes are very useful:
  - Specific types of user stories that seek "to gain the knowledge necessary to reduce the risk of a technical approach, better understand a requirement, or increase the reliability of a user story estimate." It is "time to explore, think, decide."
- Example:
  - Suppose a team is deciding between competing statistical software packages.
  - Available options are R, SPSS, and SAS Enterprise Miner.
  - This is a big decision, and the wrong one will set the organisation and the team back months.
  - To avoid wrong decision, the Product Owner invests X hours of the team's time in this user story.

Backlogs

- **Product** and **sprint backlogs** are the cornerstone of Scrum.
- A product backlog lists all desired product features in the form of user stories:
  - Also known as backlog items or just plain stories.
- Product Owner is responsible for the product backlog.
- **Product backlog** is constantly evolving:
  - Items at the top tend to be smaller and better defined than those at the bottom. and more essential
  - Those items at the bottom qualify as nice to have.

Backlogs (cont.)

- The **sprint backlog** serves as the team's to-do list for the sprint:
  - It is a subset of the product backlog.
  - Unlike the product backlog, the sprint backlog is finite.
  - The team generates this backlog during sprint planning.
  - Remains the team's focus throughout the sprint duration.
  - Team Members may change the tasks required to complete these user stories.
  - However, the user stories themselves should remain constant.

Figure 5.2 Relationship between Product and Sprint Backlogs
Source: Phil Simon.

Sprints and Meetings

- In Scrum, a **sprint** represents the period in which a team completes a predetermined number of user stories. (sprint backlog)
- A sprint typically lasts one or two weeks.
  - The general structure of a one week sprint is shown on the right.
- Scrum is designed to maximise speed, user acceptance and the odds of successful outcome:
  - Each meeting is intended to serve a specific and valuable purpose.
  - No time wastage on meetings.

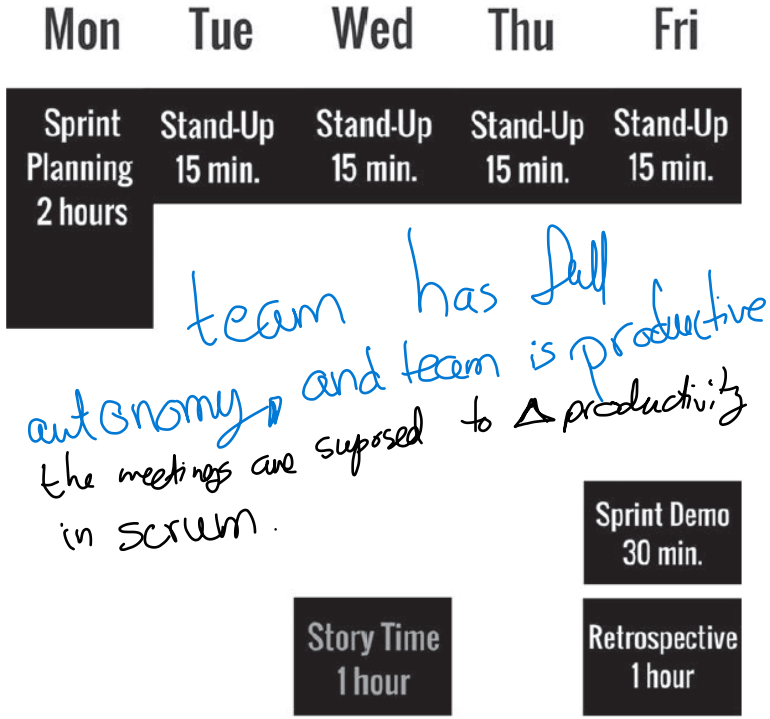


Figure 5.3 Schedule for a One-Week Sprint
Source: Phil Simon.

Sprint Planning

- **Sprint planning** usually takes about two hours.
- The objectives are two-fold:
  - A sprint goal.
  - A sprint backlog
- Product Owner and Team Members should avoid verbose and complicated goals: Keep it simple!
  - Goals should be short, one- or two-sentence descriptions of what the team plans to achieve during the sprint.
- **Sprint backlog** reflects the specific user stories that the team commits to completing during the sprint:
  - Team Members should begin by thinking about the specific tasks they need to complete to deliver the user stories.

Sprint Planning (cont.)

- Product Owner does not assign tasks akin to what a traditional project manager would:
  - Scrum is supposed to be collaborative.
  - Conversation among members should be the norm for determining the sprint backlog.

"Influence without power"

Daily Stand-Up

- **Daily stand-up** is also known as **daily scrum**.
- These are mandatory 15-minute daily meetings in the morning:
  - Should be brief and focused.
  - Most meetings tend to become unproductive when people are comfortable (i.e., sitting down).
- Each Team Member should quickly answer the following three questions:
  - What did you do yesterday?
  - What will you do today?
  - Are there any impediments in your way?

Daily Stand-Up (cont.)

- Goals of daily stand-up include transparency and problem identification:
  - Stand-up is neither a traditional status meeting nor a technical discussion. → gives visibility to everyone & understand blockers.
  - Should team members start pontificating or losing focus, the Scrum Master should intervene.

Story Time → optional meeting, but good practice.

- **Story time** is also known as **backlog grooming**.
- This optional meeting typically takes place when the sprint is a slightly more than halfway complete:
  - Goal is to maintain small and well-understood user stories.
  - Without such stories, the team will likely lose momentum when the next sprint begins.
- The focus of this meeting (in mid-week) should be upcoming user stories:
  - Not the current sprint.

Prepare for the start of next sprint.

Demo

- **Demo** is meant for Team Members to showcase its work to stakeholders who are not in the team.
- Goals of demo include:
  - Maximising transparency and gaining stakeholder trust.
  - By showing each iteration of the product development, stakeholders are less likely to be surprised at the final release compared to waterfall projects.

    ↳ miss understanding are detected early

Sprint Retrospective

- Scrum may be more agile than waterfall model, but it is not perfect.
- Scrum is designed to:
  - Launch product more efficiently.
  - Improve the process and efficiency of the team. IMPORTANT?
- **Sprint retrospective** is crucial for the second reason in answering the following three questions:
  - What did we learn during the sprint?
  - What went wrong during the sprint?
  - How can we improve next time?

Sprint Retrospective (cont.)

- Sprint retrospective is not meant to be a gripe session:
  - Scrum Master should ensure that all issues are addressed professionally.

Releases

- Each product or model **release** consists of several sprints: DevOps automatizes
  - After two to three weeks of completing user stories, the team has developed sufficient features to justify a new version.
  - Entails releasing the new iteration into production.
- Other than regular or scheduled releases, other factors may affect releases.
- Date: Important criteria for WHEN to release.
  - Especially for ecommerce websites and apps.
  - Special attention may be given to key dates such as Black Friday and Cyber Monday.

Releases (cont.)

- Functionality:
  - In response to competitor launching a killer feature.
  - Product Owner may prioritise new user stories
- Exigent circumstances:
  - Users discover a key problem or bug that requires immediate attention.*
  - May need to push back previously scheduled user stories.
- Scrum is designed to handle unforeseen changes and crises unlike waterfall.

* If you find a bug, you need to freeze the current branch, duplicate your main branch, fix it there, and then go on coding with your version → DevOps?

Estimation Techniques

- In waterfall model:
  - Projects suffer from the difficulty of estimating how long individual tasks will take.
  - Further complication when estimating an entire phrase.
  - One of the main causes of project failure.
  - Example:
    - Will it take three or four months to collect, clean, deduplicate and load enterprise data into a new business-intelligence tool or data warehouse?
    - Even a modest 20% delay in one step of a waterfall project will affect all other phases, immediately.

Estimation Techniques (cont.)

- Agile methods such as Scrum avoid this problem in several ways:
  - Scrum team works to complete tasks in small batches through sprints. → Never if you are 30% wrong, it is not a lot of delay.
  - By breaking work into more manageable parts, Scrum tends to be more successful than waterfall.
  - Example:
    - The team first completes user stories around collecting data, and even then, only a certain type of data.
    - Perhaps it first tackles customer data and then proceeds to product data or employee data.

Estimation Techniques (cont.)

- Scrum also uses a better way of estimation: We estimate if one story is bigger than another.
  - Scrum uses relative estimation instead of absolute estimation.
  - E.g., time taken to mow the author's new lawn:
    - The lawn in his new house in Nevada is about twice the size of the one in his old house in Arizona.
    - If he needs half-a-day in the past, he will probably need one whole day now.

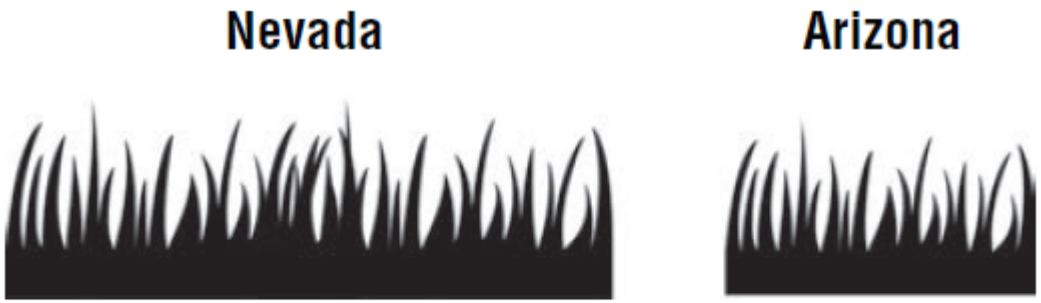


Figure 5.4 Simple Representation of Grass at Author's Former and Current Home
Source: Phil Simon.

Estimation Techniques (cont.)

- Humans are generally better at making relative estimates than absolute estimates:
  - But the two points of comparison must be sufficiently different.
- **Fibonacci numbers**:
  - Fibonacci numbers get big quickly.
  - The numbers in the sequence are meant to denote differences in a way that humans can easily discern.
  - Examples:

$$0 + 1 = 1$$
$$1 + 1 = 2$$
$$1 + 2 = 3$$
$$2 + 3 = 5$$
$$3 + 5 = 8$$
$$5 + 8 = 13$$
$$8 + 13 = 21$$

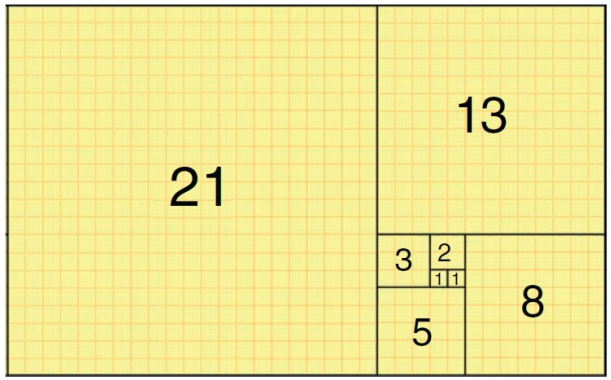
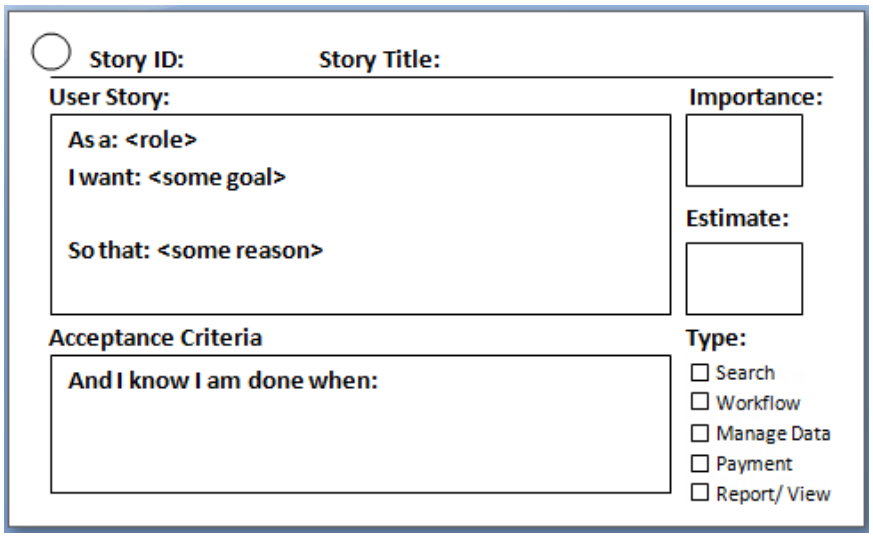


Figure 5.7 Fibonacci Sequence
*Source: By 克勞棣—Own work, CC BY-SA 4.0.**

Estimation Techniques (cont.)

- In Scrum, **story points** refer to the Fibonacci numbers.



Story Points Estimation Cheat Sheet

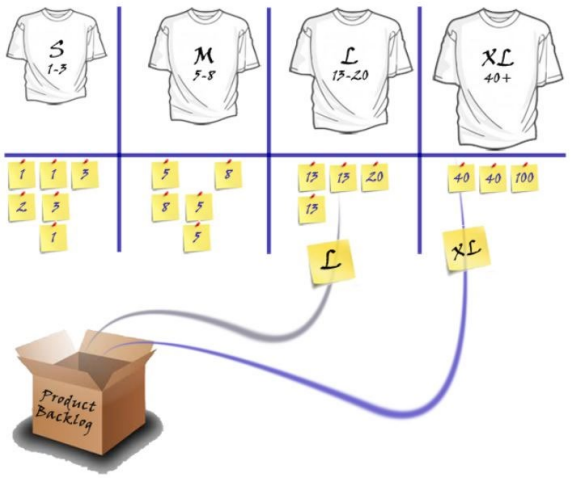
How much is known about the task	Everything	Almost everything	Something	Almost nothing	Nothing	Nothing
Dependencies	None	Almost none	Some	Few	More than few	Unknown
How much work effort	Less than 2 hours	Half a day	Up to two days	Few days	Around a week	More than one week
Story Points	**1**	**2**	**3**	**5**	**8** Should be split into smaller items	**13** Must be split into smaller items

To get story points - pick the column which represents your task the best. If it fits more than one column, pick higher one.

estimate on a relative basis is easier.

Estimation Techniques (cont.)

- **T-Shirt sizes**:
  - Some Scrum teams prefer to use T-shirt sizes in lieu of Fibonacci numbers for relative estimates.
  - In this case, small, medium, large, and extra large replace 8, 13, 21, and so on.



Task: ABC Coding			
Size	Calendar	People	Cost($)
XS	1-3 weeks	1	5-15
S	1-2 months	1-5	20-200
M	3-6 months	5-10	300-1,200
L	7-18 months	10-20	1,400-7200
XL	18+ months	20+	7,200+

When Teams Disagree

- Scrum teams tend to be inherently more collaborative than waterfall teams:
  - Waterfall teams are more egalitarian.
  - Scrum teams are generally smaller, consisting of five to nine members.
  - Scrum team members are supposed to work with a great deal of autonomy:
    - User stories describe what employees want but they do not mandate how to deliver specific features.
- But disagreement can still occur especially among newer teams.

When Teams Disagree (cont.)

- Team conflicts are resolved by playing games :)
- **Planning Poker**:
  - A few team members disagree on the number of points to assign to a user story.
  - Product Owner moderates a discussion among other team members who then indicate their own estimate.
  - The process repeats until there is a consensus.

Table 5.1 Estimates for User Story Points

Team Member	Estimate
Dinesh	8
Gilfoyle	5
Nelson	21

Source: Phil Simon.

Table 5.2 Planning Poker, Round 1

Team Member	Estimate
Richard	5
Monica	8
Russ	21

Source: Phil Simon.

Table 5.3 Planning Poker, Round 2

Team Member	Estimate
Richard	5
Monica	5
Russ	21

Source: Phil Simon.

When Teams Disagree (cont.)

- **Team Estimation Game**:
  - This technique involves placing user stories in order of perceived difficulty or amount of work.
  - Product Owners select certain number of user stories from backlog.
  - The user stories are distributed evenly to all team members.
  - Team members then take turn to put their stories on a whiteboard arranged by perceived story size.
  - Discussion and reshuffling takes place until a consensus on the ranking is reached.
  - Fibonacci number is then assigned to each column or group of stories.

When Teams Disagree (cont.)

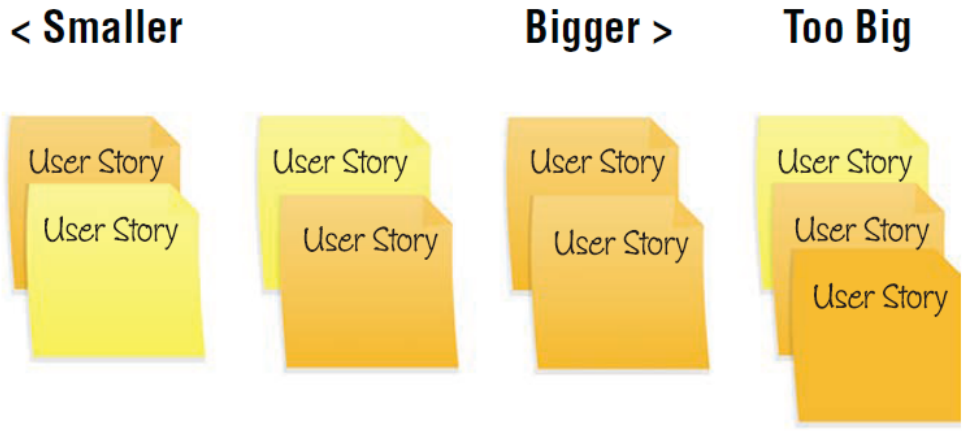


Figure 5.8 Team Estimation Game, Round 1
Source: Figure from Phil Simon.

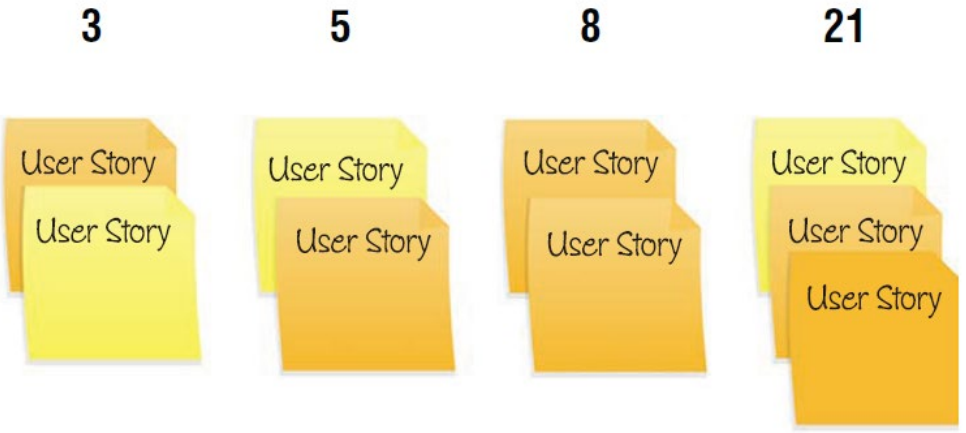


Figure 5.9 Team Estimation Game, Round 2
Source: Figure from Phil Simon.

When Teams Disagree (cont.)

- It is not uncommon for a Scrum team to breeze through 20 or more user stories in an hour.
- As an added benefit, the game often creates a positive team dynamic among members.

Other Scrum Artifacts, Tools and Concepts

- Scrum teams not only benefit from employing superior estimation techniques.
- Other advantages of adopting Scrum as opposed to waterfall include:
  - Velocities.
  - Burn-down charts.
  - Kanban boards.
  - Etc.

Velocities

- A sprint's **velocity** simply represents the total number of story points from completed user stories.
- Example:
  - Suppose a team completes four user stories during its first sprint.
  - The table below shows the team's velocity for sprint number 1.

Table 5.4 Sample Points for User Story and Sprint Velocity

User Story	Points
1	5
2	8
3	5
4	3
Sprint Velocity	**23**

Source: Phil Simon.

Velocities (cont.)

- How many points' worth of user stories should a team attempt to complete during a sprint?
  - Always wise to start conservatively.
  - Avoid overcommitting the team.
  - Even after completing the first sprint, the team may still not know how much it can realistically accomplish next sprint.
  - Helpful to use prior velocities as a guide, a technique called yesterday's weather.
- As teams cohere and members learn each others' strengths and weaknesses, velocities should increase over time.

Burn-Down Charts

- A **burn-down chart** graphically reflects each sprint's velocities.
- The figure below shows a sample burn-down chart:

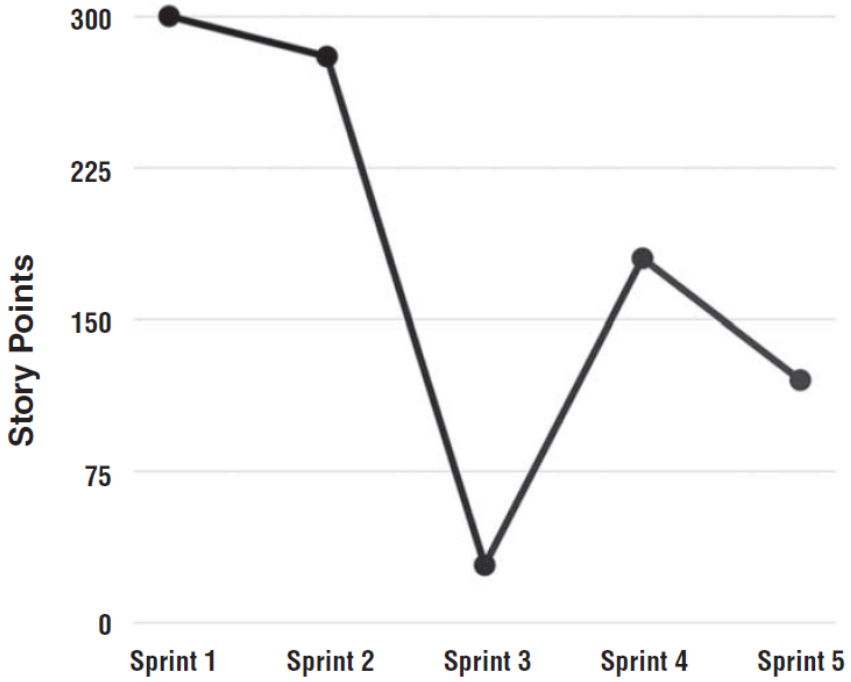


Figure 5.10 Generic Burn-Down Chart
Source: Figure from Phil Simon.

Burn-Down Charts (cont.)

- The previous figure does not show any increases in total story points associated with the project:
  - This chart is a simple one by design.
  - But in reality, this happens frequently as a Product Owner adds more user stories to the product backlog.
  - Recall that Scrum does not require teams to gather all user stories prior to commencing.
- In general:
  - The slope of the line increases over time.
  - This happens because team velocities tend to increase over time.

Definition of Done and Acceptance Criteria

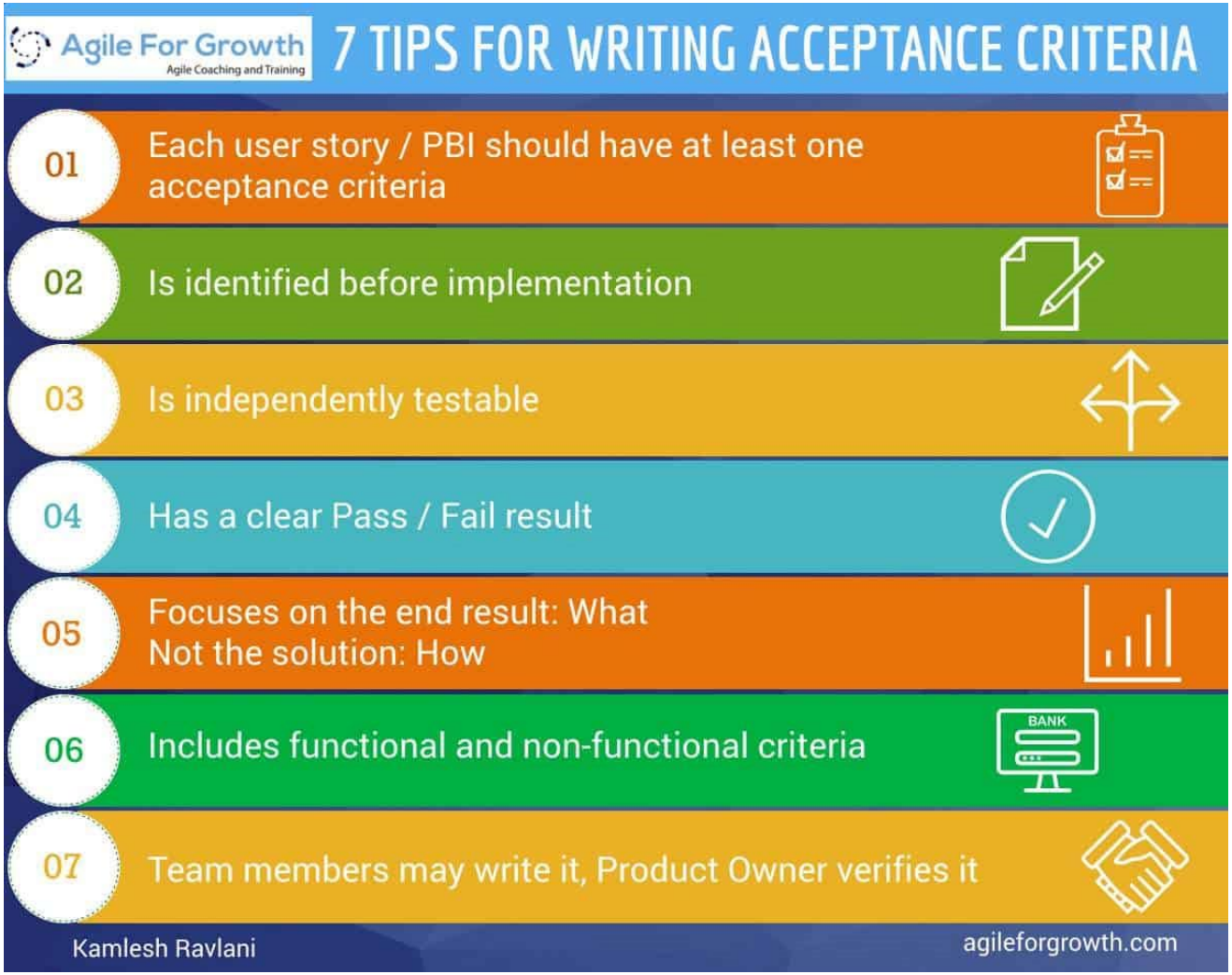
- A Scrum team has completed a task, but how do team members know that it is really done?
  - More important, does everyone agree that that task is indeed finished?
  - A shared definition of done is essential.
- In IT software development, done usually means that something is ready to ship:
  - This is an end. → the feature is ready to be shipped.
  - Likely includes both code and design reviews as well as different types of testing.

Definition of Done and Acceptance Criteria (cont.)

- Sometime there might be complication:
  - Suppose a team member completes a user story by adding some new code.
  - However, that addition breaks other key features of a product.
  - So, is that new user story really done after all? No! It needs to pass the automated test.
- **Acceptance criteria** are more specific rules on done:
  - Conditions that a software product must satisfy to be accepted by a user, customer or a receiving system.
  - There is no partial acceptance.
  - A criterion is either met or it is not.

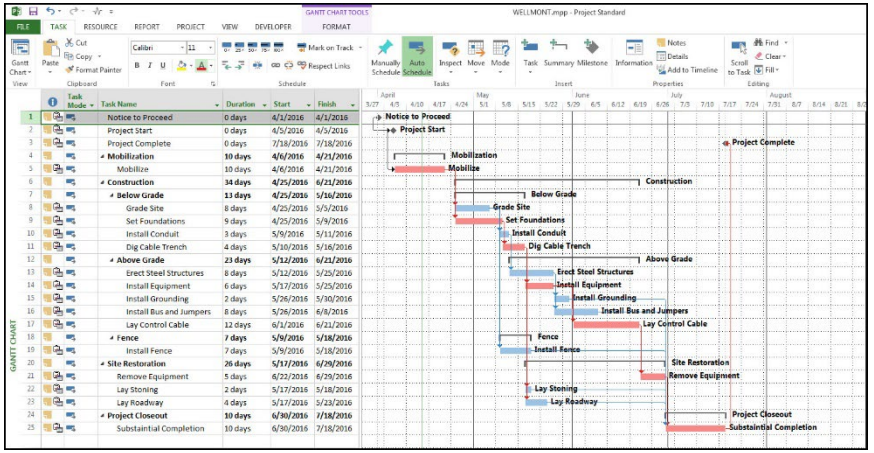
↳ if any criteria is not met, the feature is not ready for production.

Definition of Done and Acceptance Criteria (cont.)



Kanban Boards

- Waterfall projects are typically of longer duration:
  - Project managers rely on Microsoft Project and very detailed Gantt charts.
  - These tools appear to offered the illusion of predictability and control. → But this is not true!
  - They are not inherently unhelpful or confusing.
  - But they do not allow team members to easily understand what is going on.



Kanban Boards (cont.)

- In contrast, a simple Kanban board can effectively convey the status of user stories and tasks on a sprint:
  - The figure below shows a sample Kanban board.

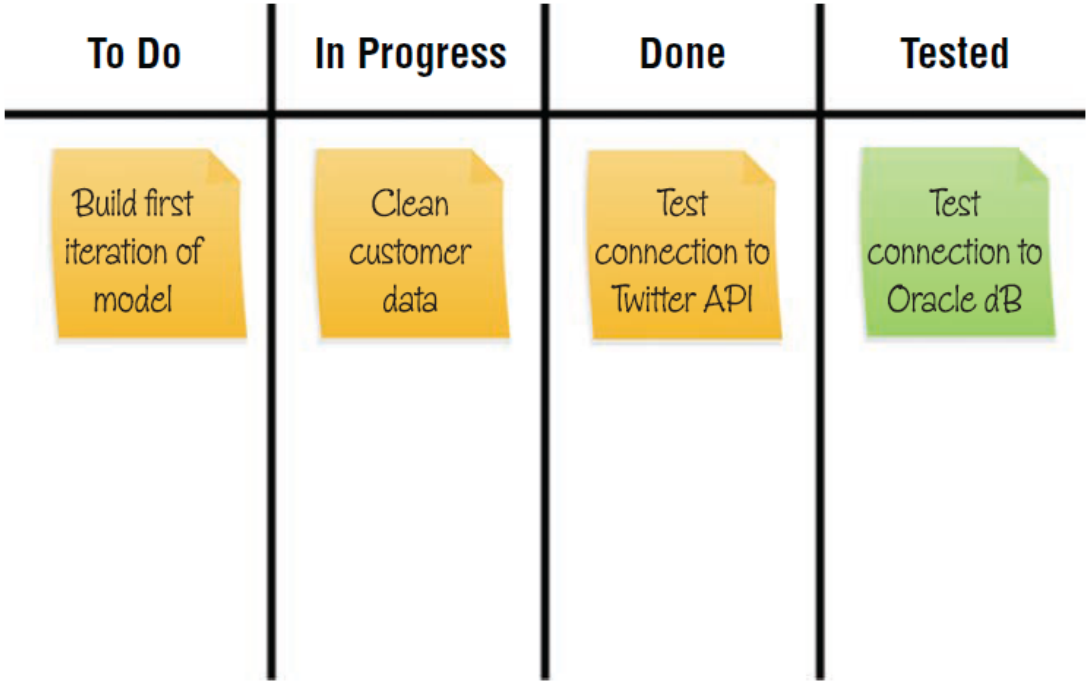
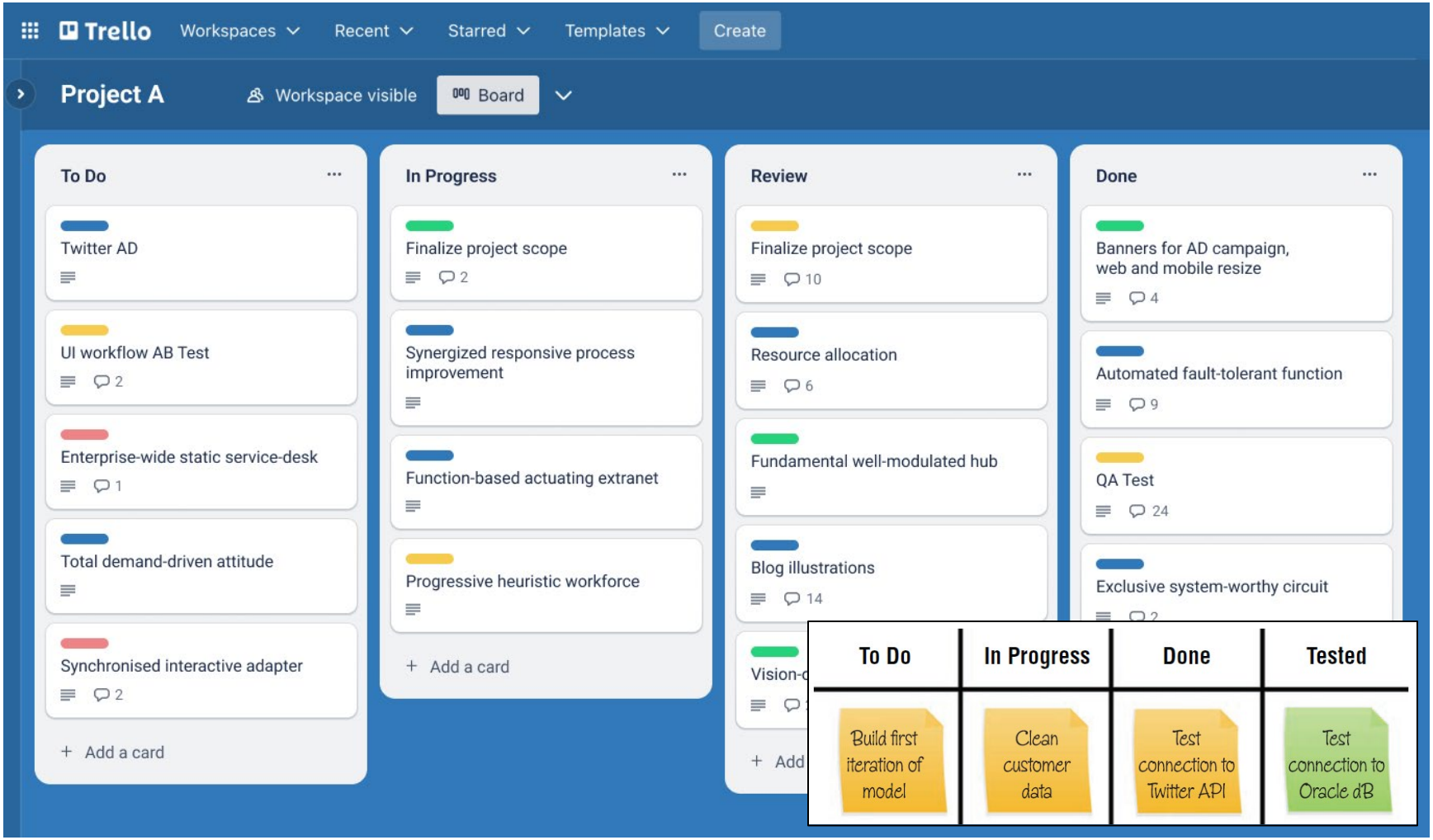


Figure 5.11 Sample Kanban Board for Analytics Project
Source: Phil Simon.

Kanban Boards (cont.)

- Many Scrum teams use physical Kanban boards:
  - Made up of different colors of index cards.
  - Common when team members are colocated.
  - Low tech but individuals often enjoy the process of physically moving a task or user story from the "in process" column to the "done" column.
- For distributed or virtual teams:
  - Many project-management applications allow the same movement among columns on digital Kanban boards.
  - Example – Trello

Kanban Boards (cont.)



Summary

- Scrum is an agile methodology that is simple, team-based meant for developing complex systems and products.
- A Scrum team consists of three roles and less than nine members.
- User stories represent first-party description of business requirements and is the central artifact guiding the Scrum development process.
- A Scrum sprint is about one to two weeks and consists of purposeful meetings such as planning, daily stand-up, story times, demo and retrospective at strategic juncture.

Summary (cont.)

- Scrum relies on relative estimation, which is more accurate, to determine story points or efforts to implement a user story.
- Team conflicts are resolved through playing games such as planning poker and team estimation game.
- Velocities, burn-down charts and Kanban boards are useful tools for managing a Scrum project.
- Acceptance criteria must be carefully defined to help determine completion of user stories.

Q&A

Next Lecture…

- Learn about:
  - Overview of software engineering process.
  - DevOps
  - Implementation and toolchain for DevOps
  - Overview of analytics process.
  - DataOps
  - MLOps