UNIT-II

AGILE SCRUM FRAMEWORK

Agile

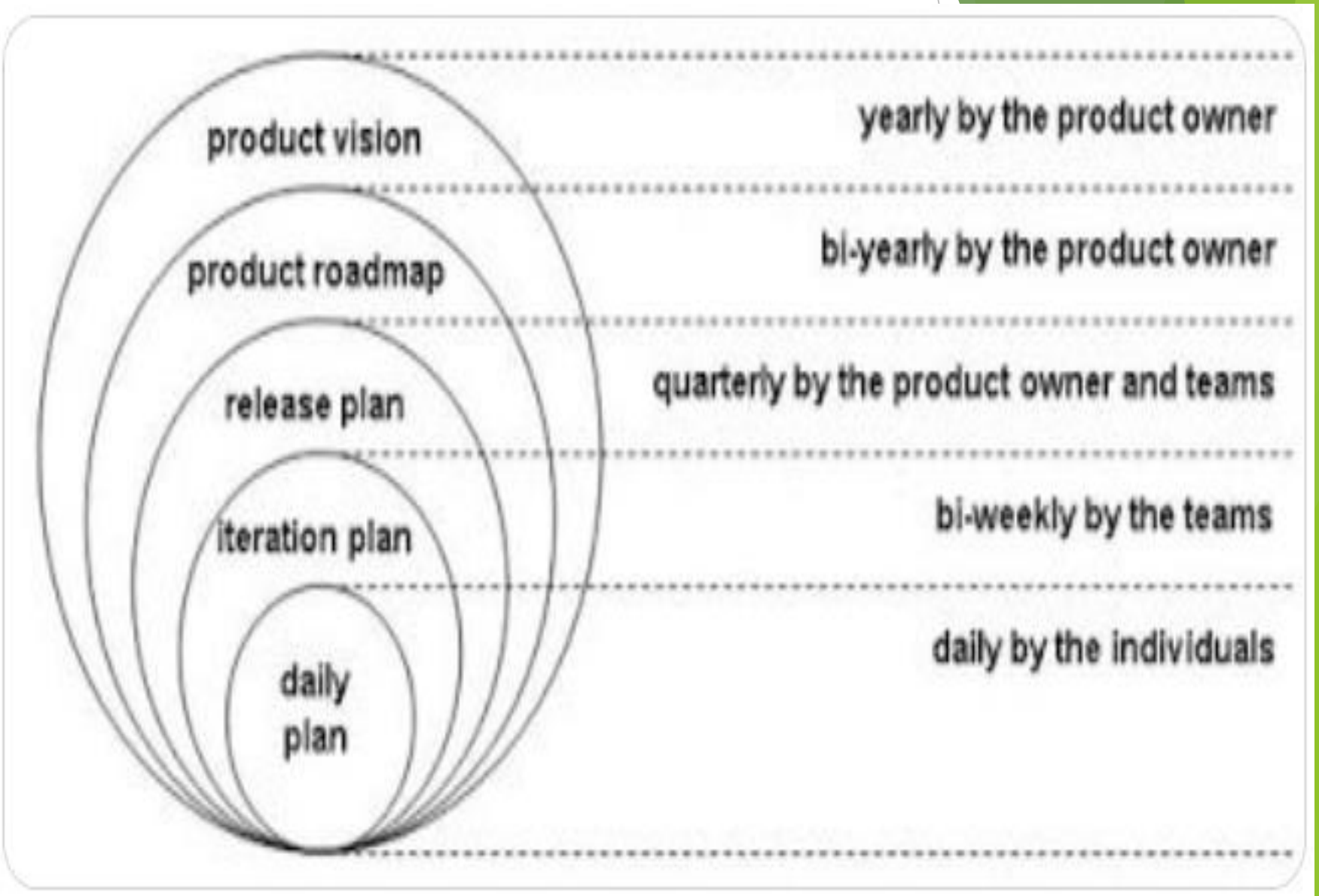
- Light weight methodology
- Small to medium size teams
- Requirements can be changed at any time
- Simple Design

Agile planning levels

- It is not time-consuming or complex; instead, these help product owners focus on the right group of professionals and the product development stage.
- The strategic Agile planning for different levels reduces considerable time, effort, and cost that is otherwise invested in repetition, correction, and last minute meetings etc.

The Five levels of Agile Planning

- Agile teams plan their projects at 5 levels as→



Level 1: Agile Planning For Product Vision

- Product vision should deliver the unique feel of ownership to keep you motivated.
- Validate your product vision with all the stakeholders, Scrum team members, users etc.
- Develop the product vision iteratively & incrementally with the scope to make it better over time.
- The product vision pitch should address all the key concerns of different stakeholder groups pertaining to quality, product goals, competition, longevity and maintenance needs etc.
- Focus your product vision on the values for users and customers not merely on the most advanced technology.

Level 2: Agile Product Roadmap Planning

Cont…

- Do all the necessary prep work including describing & validating the product strategy.
- Focus your product roadmap on goals, benefits, objectives, acquiring customers, removing technical debt and increasing engagement etc.
- Let your product roadmap tell a coherent story about the likely development of a product. To simplify the task, you can divide the product roadmap into two parts- internal product roadmap and external roadmap. **The internal product roadmap** is focused on development, service, supporting groups, marketing, sales etc; while, the **external roadmap** is focused on prospective & existing customers.
- Keep the product roadmap simple and realistic to make the features understood by everyone concerned.
- Make your product roadmap measurable; so, think twice before adding timelines and deadlines.

Level 3: Release Planning

- Focus on goals, benefits, and results.
- Take dependencies and uncertainties into account.
- Release early but don't release just for the sake of scheduled releasing.
- Only release the work that is **'Done'**
- Each release process has the scope for betterment. Continuous release process improvement helps you deliver more values for the product.

Level 4: Iteration Planning

- Span the iteration planning meeting maximum up to 4 hours.
- The planning meeting is organized by the team and for the team; so, everyone should participate.
- Avoid committing anything that exceeds the historical team's velocity.
- Keep time for 'retrospectives' in the past sprints before planning for the next one.
- Follow the four principles – prepare, listen, respect & collaborate.

Level 5: Daily Commitment Planning

- Keep it around the task board.
- Start on time regardless of who is present or not.
- Let each team member go through the questions like - what he did yesterday, what is his plan for today, and, is there any impediment?.
- Use ‘Parking Lot’ for the unresolved issues. The purpose of daily Agile-Scrum planning is to let the team members know about ‘being done’, ‘needs to be done’ and ‘hindrance if any’. Anything out of this scope should be listed in ‘Parking Lot’ to be dealt later.
- Do preparation ahead of time. The team members should know ‘what they need to share’.

Rules for planning Agility

- Involve the whole team
- Plan in multiple levels
- Keep size and time estimates separate (optional)
- Consider uncertainty for features and dates
- Replan frequently
- Track and advertise progress
- Be aware of the importance of learning
- Work with features of the right size
- Prioritize features
- Base your estimates and plans in facts
- Not plan for 100% of the time
  - buffer, ideal work day
- Coordinate planning to avoid dependencies

UNIT-II

Scrum

Scrum- Introduction

- SCRUM is an agile, lightweight process for managing and controlling software and product development in rapidly changing environments.

  - Iterative, incremental process
  - Team-based approach
    - developing systems/ products with rapidly changing requirements
  - Improve communication and maximize cooperation
  - Protecting the team from disruptions
  - A way to maximize productivity

Scrum

Definition

A framework within which people can address complex adaptive problems, along with productivity and creativity by delivering products of the highest possible value.

Characteristics

Scrum is:

- Lightweight
- Simple to understand
- Difficult to master

When to use Scrum?

- Scrum is a framework that **helps teams work together**. So the scrum can be adapted when,

1. WHEN REQUIREMENTS ARE NOT CLEARLY DEFINED
2. WHEN THE PROBABILITY OF CHANGES DURING THE DEVELOPMENT IS HIGH
3. WHEN THERE IS A NEED TO TEST THE SOLUTION
4. WHEN THE PRODUCT OWNER (PO) IS FULLY AVAILABLE
5. WHEN THE TEAM HAS SELF-MANAGEMENT SKILLS
6. WHEN CONTRACTING IS TIME AND MATERIALS
7. WHEN THE CLIENT'S CULTURE IS OPEN TO INNOVATION AND ADAPTS TO CHANGE

Uses

1.Research and identify viable markets, technologies, and product capabilities;

2. Develop products and enhancements;

3. Release products and enhancements, as frequently as many times per day;

4. Develop and sustain Cloud (online, secure, on-demand) and other operational environments for product use

5. Sustain and renew products.

Scrum Theory

- Scrum is founded on empirical process control theory, or empiricism.
- **Empiricism** asserts that knowledge comes from experience *and* making decisions based on what is known. Moreover, it means working in a fact-based, experience-based, and evidence-based manner. Scrum implements an empirical process where progress is based on observations of reality, not fictitious plans. Scrum also places great emphasis on mind-set and cultural shift to achieve business and organizational Agility.
- Scrum employs an iterative, incremental approach to optimize predictability and control risk.
- 3 Pillars of empiricism are,
  - Transparency
  - Inspection
  - Adaptation

Transparency

- Significant aspects of the process must be visible to those responsible for the outcome.
- Transparency requires those aspects be defined by a common standard so observers share a common understanding of what is being seen.

For example

- A common language referring to the process must be shared by all participants; and,
- Those performing the work and those inspecting the resulting increment must share a common definition of “Done”.

Inspection

- Scrum users must frequently inspect Scrum artifacts and progress toward a Sprint Goal to detect undesirable variances.
- Their inspection should not be so frequent that inspection gets in the way of the work.
- Inspections are most beneficial when thoroughly performed by skilled inspectors at the point of work.

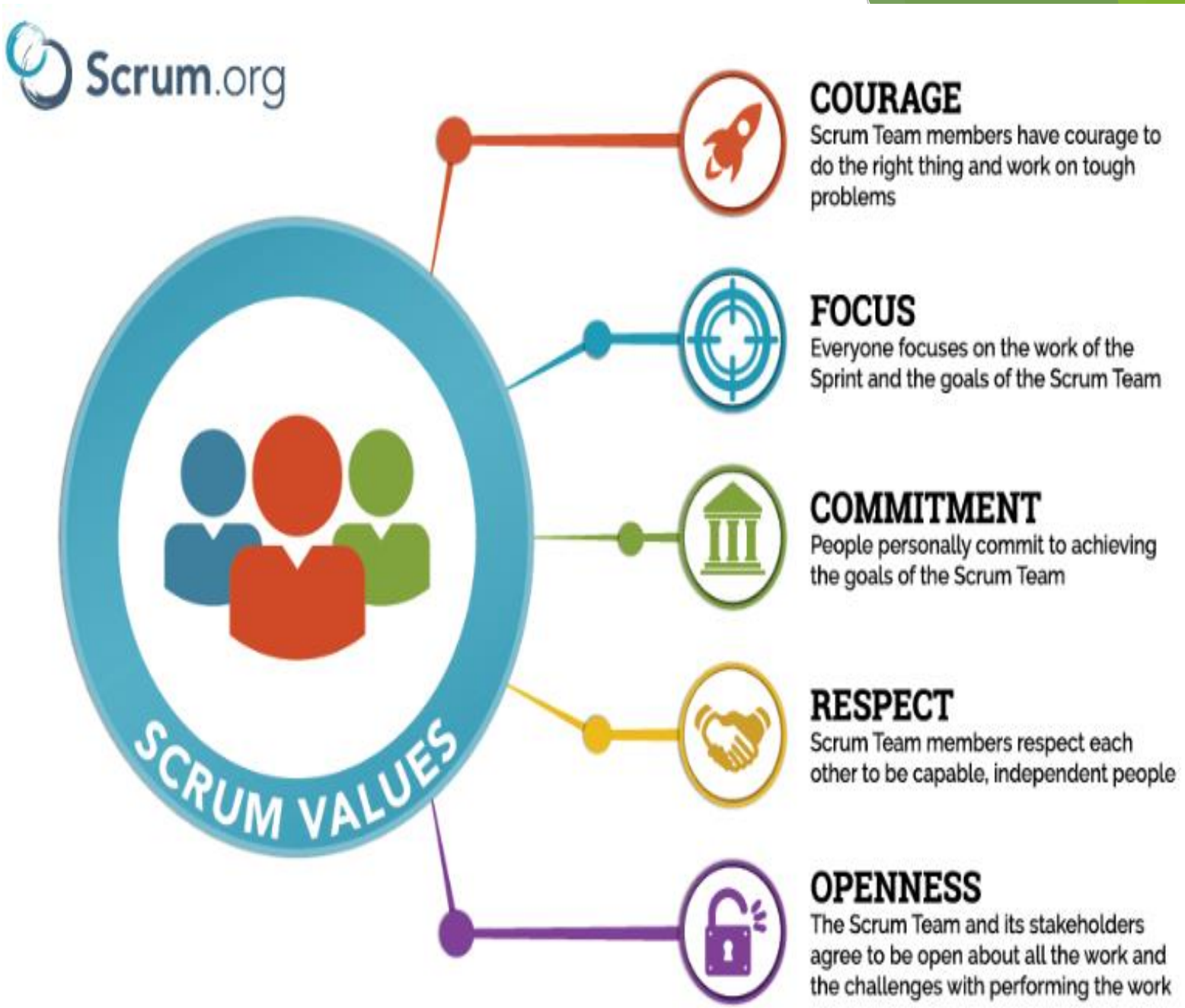
Adaptation

- If an inspector determines that one or more aspects of a process deviate outside acceptable limits, and that the resulting product will be unacceptable, the process or the material being processed must be adjusted.
- An adjustment must be made as soon as possible to minimize further deviation.

Scrum Values

- Scrum has five values
  - Commitment
  - Focus
  - Openness
  - Respect
  - Courage

WARNING: If your team lacks these values, you will most probably fail in scrum implementation



Commitment

- Scrum teams must be committed to progress and willing to have practical objectives and stick to them. This is a team activity where you are a part of a team, and you are accountable to work together and to conform to your commitments.
  - **Sprint-based commitment**
  - **Commitment as a team**
  - **Commitment as an individual**

Focus

- An iterative-incremental approach and timely delivery in Scrum helps to keep us stay focused towards the project goal.
- This motivates you for delivering faster, better and yield more
- By focusing more on a goal, you can avoid resource wastage and deliver on time.
- This Scrum value leverages lower rate of risks and provides ample time to improve and deliver the needful.

Openness

- The Scrum induction requires transparency and openness.
- We need to investigate reality with a specific end goal to make sensible adjustments.
- Team members should be open about their work, progress, what they learned and the issues they are facing

Respect

- As a part of the Scrum team, you should respect colleagues, their decisions, and their experience.
- As an efficient team member, you should also respect diversity.
- You should respect your stakeholders by not building anything in which people are not interested.

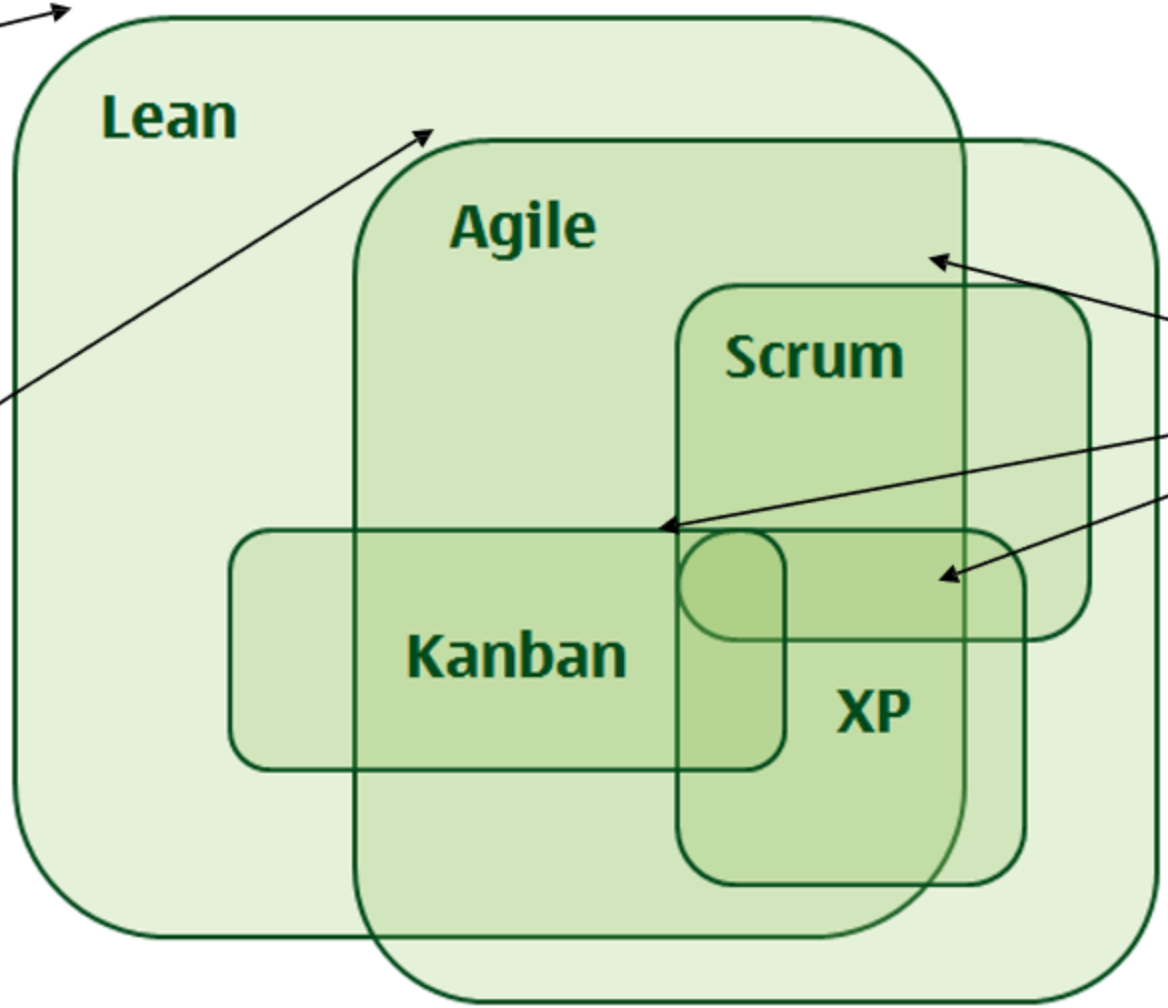
Courage

- Adaptability to change forms the base of any Scrum project and to accept a change, courage is needed.
- Scrum is all about taking risks and finding out an optimized solution.
- The Scrum team is allowed to think of different approaches to workshop the best and most appropriate solutions.
- In order to implement new things to the project, we need to explain these new ideas to the team.

Lean, Agile and Scrum

Lean – a "philosophy" to optimize everything

Agile – a project management methodology



Scrum, Kanban & XP – agile process tools with lean characteristics

UNIT-II

Project Phases

Scrum Project Phases

- The Pregame Phase
  - Planning
  - System Architecture/high Level Design
- The Game Phase
- The Post Game Phase

The pregame phase

Planning

The first step is creating the backlog - a list of necessary properties that have to be implement during the development process. The product owner is a person responsible for this.

- The delivery date
- The number and functionality of release
- The most important releases(selection)
- The list of packets(objects) for backlog items
- The project team structure
- Necessary tools
- Risk control
- Release cost(including cost of training or marketing)

The pregame phase

System Architecture/High Level Design

- After the planning there is a time for the system architecture. The team review the backlog , think what changes have to be made to implement new properties and design the implementation process.
- there is also a need to make some changes , refine the old product , learn some additional knowledge , analysis , solve problems or issues which appears during the process.
- At the end there are designed review meetings during which the teams exchange information , present progress and problems , reassigned changes as required.

The game phase

- The game phase is usually called the sprint or **development phase** and its an iterative cycle of development work.
- It may last from 1 to 4 weeks (usually its 1-2 weeks) and the duration should be constant in every cycle.
- However this requirement is not strict and sometimes the time of each sprint is different.

It consists of four steps:

- **Develop**(analysis , design, develop)-the team analyze current situation of the product , think what changes have to be made for implementation of backlogs requirements into the packets , design the process and finally proceed to development , implementation , testing and documenting the changes.
- **Wrap**-closing packets
- **Review**-meetings for presenting current work and progress , resolving problems , adding new backlog items , risk review.
- **Adjust**-the information gathered during the review is consolidated into affected packets.

The postgame phase (closure phase)

- The management ends the development process and the product is being prepared for a release.
- This includes: integration, testing, user documentation, training and marketing material preparation.

- As we can see the Scrum methodology is very well organized and very effective.
- What is also worth mentioning is that the whole Scrum process is limited in time.
- If the time of the sprint has ended the process has to be stopped. It makes the process more resolute

UNIT-II

Agile Estimation

Agile Estimation

Agile estimation is the process for estimating the effort required to complete a prioritized task in the product backlog. This is an exercise that typically takes two to four weeks, depending upon the project's complexity. This effort is usually measured with respect to the time it will take to complete that task, which, in turn, leads to accurate sprint planning.

Why Run Agile Estimations?

- Agile estimations are essential for:
- Making teams accountable for deliverables
- Inducing discipline across the Agile team
- Predicting the approximate time it will take to finish a project
- Enabling better sprint management
- Improving team productivity

Why do Teams Estimate in Agile?

Overestimating and underestimating are both common for Agile development teams which leads to varying development and launch times. Though the process is complicated, considering Agile estimation in the initial stages can assist with accurate user story estimations and helps the team stick to the timely deliverables.

1. Improved Decision-Making

With accurate, agile estimation, the development team will be able to conduct effective backlog grooming sessions, which will further help in precise sprint planning. When they make informed decisions and plan well, their user story delivery time will improve.

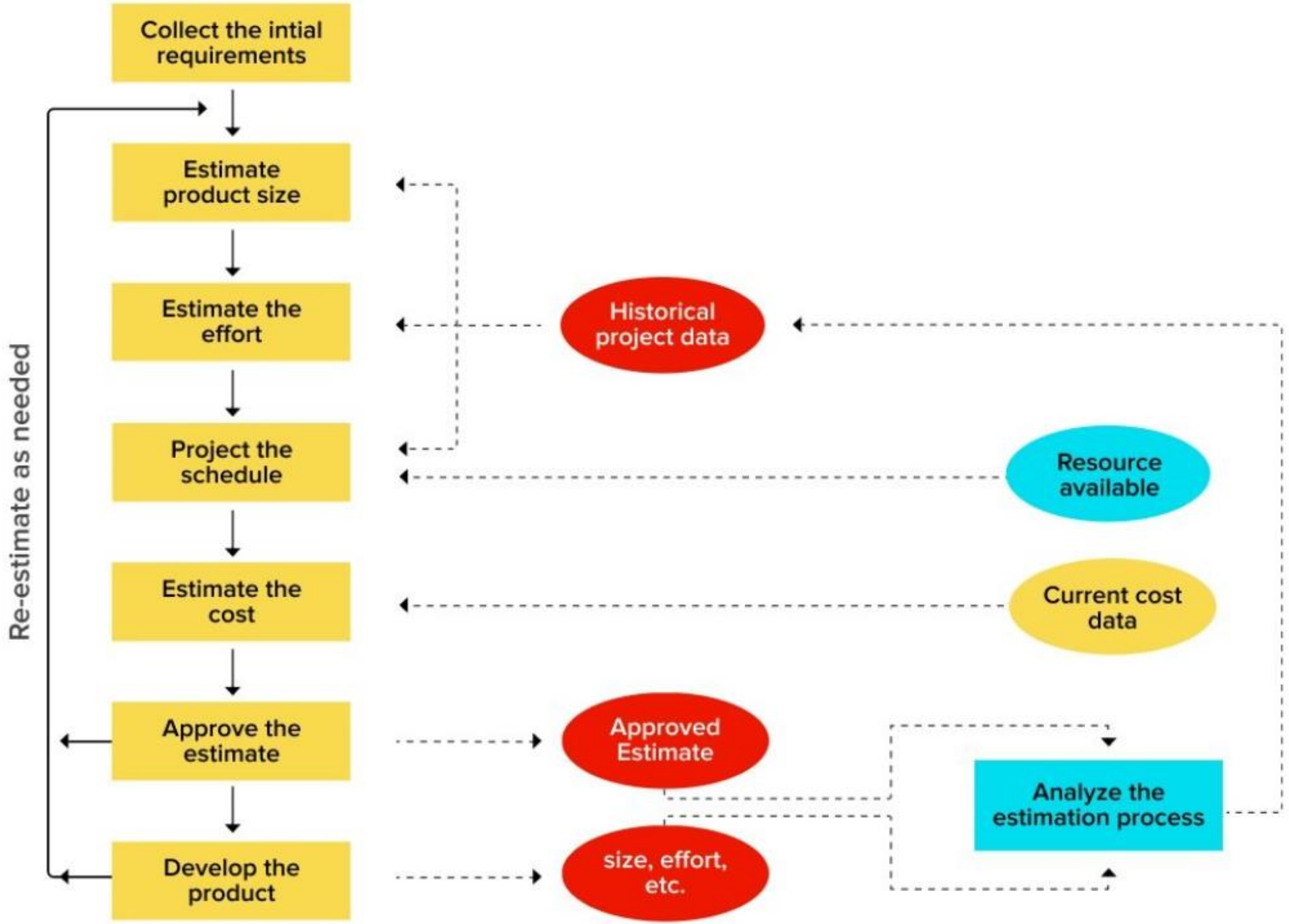
2. Better Coordination

Let's say that the estimated effort for user story A is two weeks. On the other hand, the estimation effort for user story B is four weeks. Now, suppose both the user stories depend on each other and are connected. In that case, the team needs to prioritize work so that both user stories get completed simultaneously, thus leading to better coordination among teams.

3. Better Risk Management

Software projects often suffer from exceeding budgets and timelines. To lessen this risk, Agile project estimation is an ideal solution. Agile product estimation helps estimate story points and stick to budgets, estimates, and the project's scope. The more accurate the estimates, the better the chances of on-time, quality delivery.

AGILE ESTIMATION AND PLANNING



1. Conduct Stakeholder Interviews

- The Business Analyst (BA) assigned to the discovery team revisits any existing documentation shared initially and extracts the gaps and queries. The BA then conducts regular workshops with the stakeholders to discuss the gaps and clarify doubts in the system workflow.
- Based on these workshops, the BA comes up with the business and functional requirements:
  - Business Requirements Document (BRD): defines the end-goal of the project
  - Functional Requirements Document (FRD): defines the features required to achieve the end-goal
- These workshops can be conducted over a call with the client or when they visit the premises to have one-on-one sessions.

2. Define High-Level Product Backlog

- The next step of Agile Estimation involves the BA and the Technical Architect. They frame an initial outcome that the stakeholders are looking for with a feasible solution or product.
- A high-level product backlog is defined in terms of epics and story titles, which describe the bare bones of the application. They then validate if the backlog addresses the scope of the project for the client.

3. Understand the Client and its Potential Customers

- Depending upon the complexity of the problem that the application is intended to solve, a UX design anchor is taken on board along with the BA for the discovery phase. The UX analyst's prime deliverable is to understand not just the client but also their potential customers.
- The UX analyst works on personas of the possible user group who might use the application, the ecosystem in which the personas will be using it, and the touchpoints of the user personas within the system. The deliverables here would be ecosystem maps, personas, user journeys, and storyboards.

4. Prioritize Requirements

- The discovery team becomes involved in the agile cost estimation project and works on the high-level backlog after it has been validated by the stakeholder.
- The analysis is employed with the prioritization method to decide which requirements to complete first, which ones can come later, and which ones to exclude. The backlog items are divided on the basis of the MoSCoW method, which segments features based on must-haves, should-haves, could-haves & won't-haves.

5. Prepare the Minimum Viable Product (MVP) Backlog

- Based on the prioritization activity, the BA assembles the requirements as ‘must-haves’ to the backlog and sections them as the requirements for the MVP Development.
- The MVP backlog might also contain a few items from the ‘should haves’ list, ensuring that the product is sufficiently competitive in the market.
- In some instances, depending on the budget and time to market, this step is skipped, and the agile teams jump directly to developing the fully-fledged product.

6. Estimate the Project Cost and Timeline

- The discovery team estimates the MVP backlog to define the estimated cost and timeline for the first release.
- This is followed by build, rinse, and repeat until they arrive at an estimate that fits the business needs. This also allows flexibility to load and off-load features as product development starts.

Story Point Estimation in Agile

- The story points approach in the Agile estimation technique uses historical data to compare features of previous, similar projects to generate a precise estimate.
- The steps involved in the estimation method with story points are as follows:
  - Identify user stories
  - Discuss the requirements of the user story. It is the responsibility of the Product Owner or business analyst to answer the questions and explain what precisely the referred story entails
  - Create a matrix for estimation: The estimation matrix is a numeric scale that is used to evaluate the selected pieces of work. This scale can be the Fibonacci sequence (…5, 8, 13, 21, 34 …) or the straightforward linear scale (… 3, 4, 5, 6, 7 …). Many teams prefer the Fibonacci scale because the gaps between numbers in this series are more extensive than in the linear scale. This simplifies visualizing the difference between the values assigned to the separate tasks.
  - Choose an Agile estimation technique
  - Conduct sprint planning
  - Validate that the estimates are internally consistent and align with the stories as the process goes along

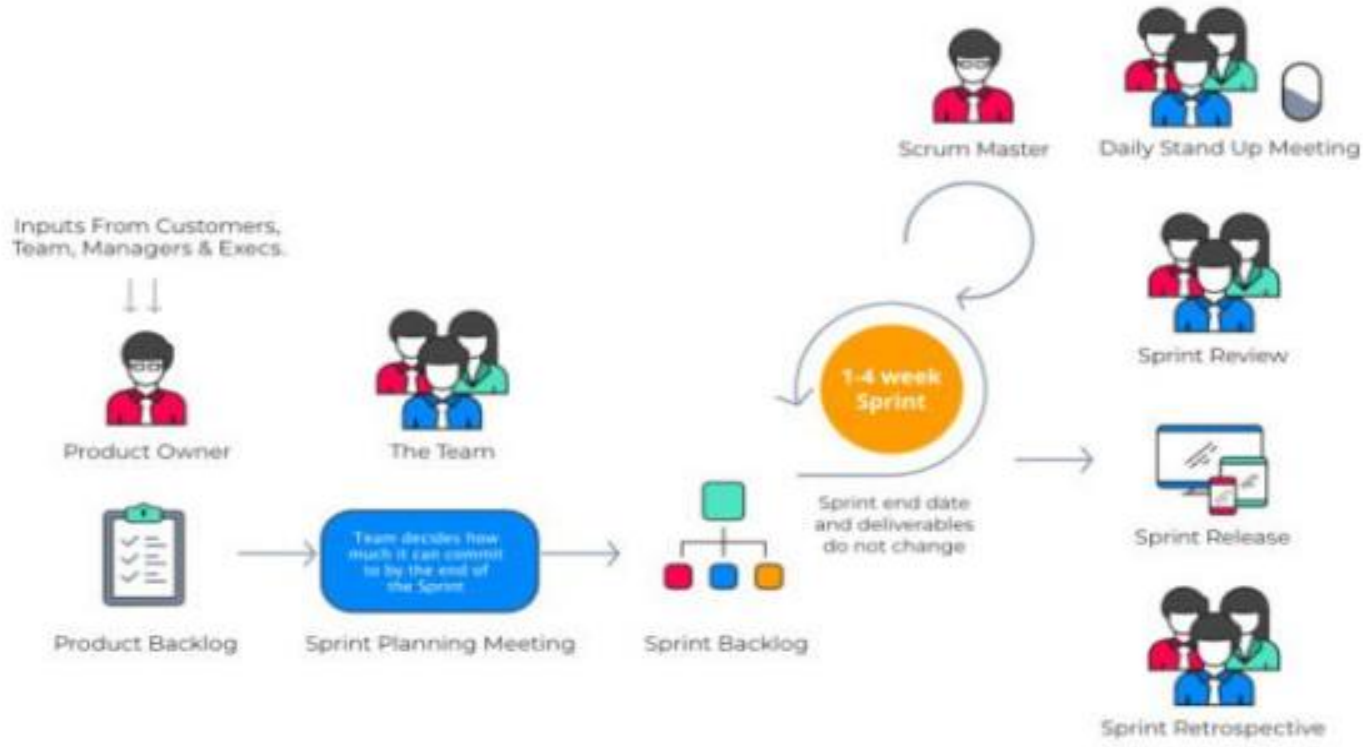
Step-I


Identify Base Stories
Story Points in agile are a complex unit that includes three elements: risk, complexity, and repetition.
STORY POINT
Risks
Complexity
Repetition
Unclear demand
Dependence on 3 Party
Uncertainty in the future
Efforts needed to develop a particular feature
Monotonous tasks without any risks and complexity

Step-II

Discuss the Requirements of the Story

Product Owner or Proxy PO, will answer questions and provide an explanation about what exactly this story entails.



Step-III

Create a Matrix for Estimation

There are two types of scales used for creating an estimation matrix: the linear scale (1,2,3,4,5,6,7…) and Fibonacci sequence numbers (0.5, 1, 2, 3, 5, 8, 13 …).

Step-IV

Choose an Agile Estimation Technique

A meeting is held to decide upon the best agile estimation technique based upon the user story count and development complexity.

Step-V

Plan the Sprint

A sprint planning meeting is conducted to decide what stories will be estimated and worked upon based on priority.

Agile Estimation

Each team member brings a different perspective on the product and the work required to deliver a user story.

- Major techniques:

1) Expert opinion

- In an Expert Opinion based Estimation approach, an expert is asked how long something will take or how big it will be.
- The expert provides an estimate relying on his or her experience or intuition or gut feel.
- Expert Opinion Estimation usually doesn't take much time and is more accurate compared to some of the analytical methods.

2)Analogy

- Analogy Estimation uses comparison of User Stories. The User Story under Estimation is compared with similar User Stories implemented earlier. This results in accurate results as the estimation is based on proven data.

3)Divide and conquer

- Major problems:
  - Availability of expert
  - Estimator is not developer
  - Estimate by feature and not by task

UNIT-II

Planning Game

Planning Poker

The Planning Game

- A game with a set of rules that ensures that Customer and Developers don't become mortal enemies
- **Goal:**
  - Maximize the value of the software produced by Developers.
- **Overview:**
  1. **Release Planning: Customer** selects the scope of the next release
  2. **Iteration Planning: Developers** decide on what to do and in which order

The Release Planning Game

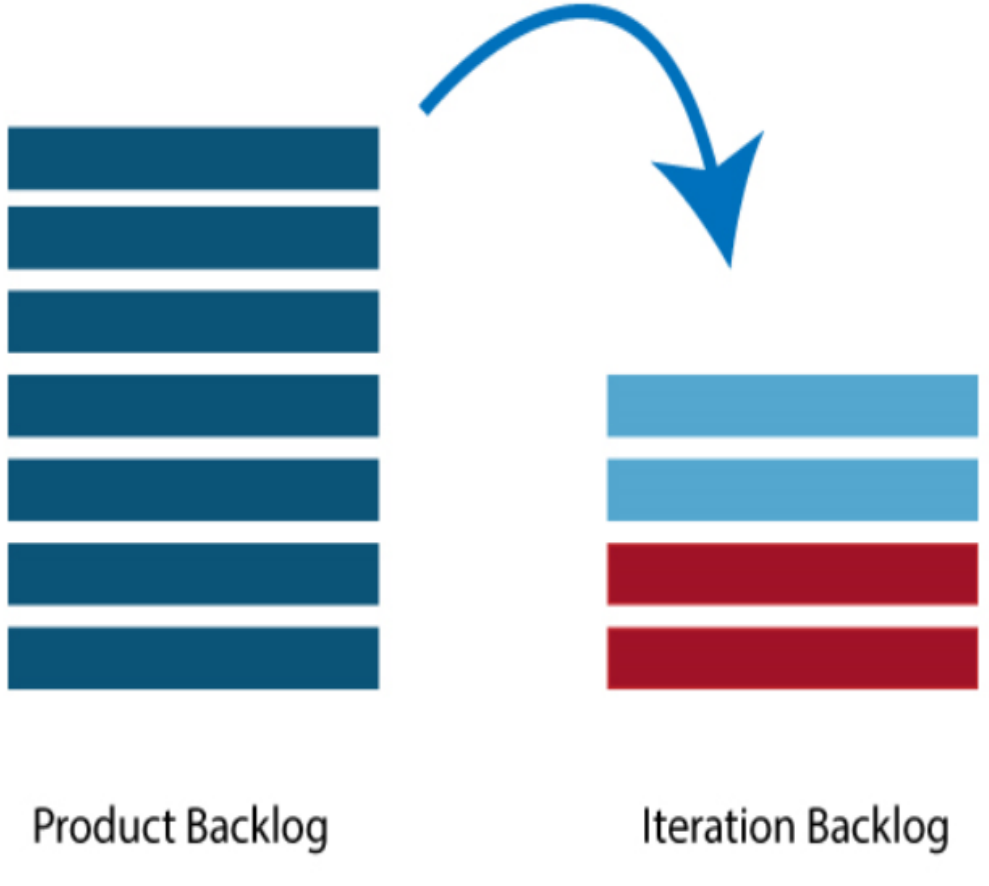
	Customer	***Developers***
Exploration Phase	Write Story	
		Estimate Story
	Split Story	
Commitment Phase	Sort Stories by Value	
		Sort Stories by Risk
		Set Velocity
	Choose Scope	
Steering Phase	Iteration	
		Recovery
	New Story	Reestimate

Iteration Planning

- **Customer** selects stories to be implemented in this iteration.
- **Customer** explains the stories in detail to the Developers
  - Resolve ambiguities and unclear parts in discussion
- **Developers** brainstorm engineering tasks
  - A task is small enough that everybody fully understands it and can estimate it.
  - Use short UML sessions to determine how a story is accomplished.
  - Observing the design process builds common knowledge and confidence throughout the team
- **Developers** /pairs sign up for work and estimates
  - Assignments are not forced upon anybody (Principle of Accepted Responsibility)
  - The person responsible for a task gets to do the estimate

Iteration Planning

- The purpose of the iteration planning meeting is for the team to commit to the completion of a set of the highest-ranked product backlog items.
- This commitment defines the iteration backlog and is based on the team's velocity or capacity and the length of the iteration timebox



Who’s Involved in Iteration Planning?

- **Scrum Master** – The scrum master acts as a facilitator for the agile delivery team.
- **Product Owner** – The product owner deals with the detailed view of the product backlog and their acceptance criteria.
- **Agile Team** – Agile delivery defines their tasks and sets the effort estimates required to fulfil the commitment.

Planning Process

Iteration planning involved the following steps:

- Determines how many requirements (stories) are fit in an iteration.
- Break this requirement into tasks. Assign each task to their owners.
- Each task is set to some estimated time.
- These estimates help the team members to check how many hours for each member will be required to iterate.
- Team members are assigned tasks by seeing their velocity or capacity. Due to this, the team member is not overburdened.

Velocity Calculation

- The agile team calculates the velocity based on the previous iterations.
- A velocity is an average number of units that required finishing user stories in the iteration.
- Assume that, a team took 10,12,8 story points in each iteration for the previous three iterations, this shows that the team can take 10 as velocity for the next iteration.
- Planned velocity tells the team how many user requirements can be completed in the current iteration.
- If the team instantly finishes the work assigned, then more user requirements can be pulled in.
- Otherwise, the requirement can be moved out too to the next iteration.

Task Capacity

Three factors determine the capacity of the team:

- Total number of ideal working hours in a day
- A person gives total days in each iteration
- Percentage of time a member is entirely available for the team.
- Considered a team has 6 members,
  - committed to work full time of 8 hours a day on a project
  - no member is on leave during iteration,

  then the task capacity for two-week iteration will be,

  6(members) x 8(hours) x 10(days) = 480 hours

Iteration Planning Steps

- Product Owner describes the highest ranked item of the product backlog.
- Team member describes the tasks required to complete the item.
- Team members own the tasks.
- The team member estimates their own time to finish each task.
- The above steps are repeated for all the items in the iteration.
- If any member is overloaded with work, then his/her tasks distributed among other team members.

Prerequisites of Planning

Before getting started ensure,

- The items in product backlog have been sized by the team and assigned a relative story point value
- The product backlog is stack ranked to reflect the priorities of the product owner
- There is a general understanding of the acceptance criteria for these ranked backlog items

Agenda

1. **Opening**
Welcome meeting participants, review purpose, agenda, and organizing tools.
2. **Product vision and roadmap**
Remind the team of the larger picture.
3. **Development status, state of our architecture, results of previous iterations**
Discuss any new information that may impact the plan.
4. **Iteration name and theme**
Make a collaborative decision on name and theme.
5. **Velocity in previous iterations**
Present the velocity to be used for this iteration.
6. **Iteration timebox (dates, working days)**
Determine the timebox and total working days. Subtract days for holidays or other whole team-impacting events.
7. **Team capacity (availability)**
Each team member calculates their capacity based on personal availability, allocation to this and other projects, productive time for tasks in this iteration each day.
8. **Issues and concerns**
Check in on any currently known issues and concerns and record as appropriate.

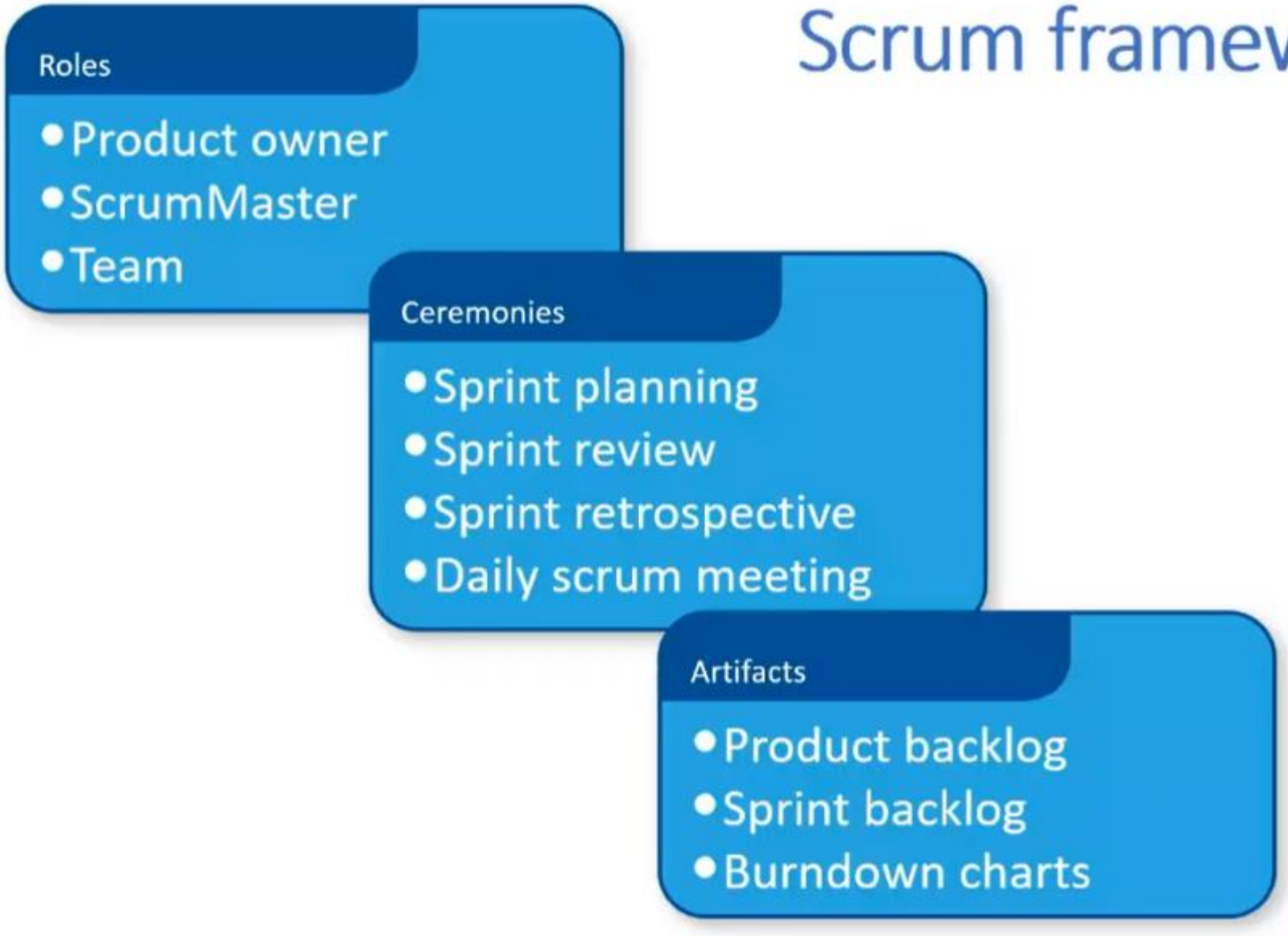
Cont…

1. **Review and update Definition of Done**
Review the Definition of Done and make any appropriate updates based on technology, skill, or team makeup changes since the last iteration.
2. **Stories or items from the product backlog to consider**
Present proposed product backlog items to be considered for the iteration backlog.
3. **Tasking out**
Delivery team determines tasks, signs up for work, and estimates tasks they own. Product Owner answers clarifying questions and elaborates acceptance criteria as appropriate; Scrum Master facilitates collaboration.
a. Tasks, b. Estimates, c. Owners
4. **New issues and concerns**
Check in on any new issues and concerns based on tasking out and record as appropriate.
5. **Dependencies and assumptions**
Check in on any dependencies or assumptions determined during planning and record as appropriate.
6. **Commit!**
Scrum Master calls for a *fist of five* on the plan. Agile team and Product Owner signal if this is the best plan they can make given what they know right now and commit to moving to the next level of planning—daily.

Cont…

1. **Communication and logistics plan**
Review and update communication and logistics plan for this iteration.
2. **Parking lot**
Process parking lot—all items should either be determined resolved or turned into action items.
3. **Action items/plan**
Process action plan—distribute action items to owners.
4. **Retrospect the meeting**
Because we want these meetings to be useful for everyone, we solicit feedback on the meeting itself.
5. **Close**
Celebrate a successful planning meeting!

Scrum framework



Scrum Roles

Product Owner

- Possibly a Product Manager or Project Sponsor
- Decides features, release date, prioritization

Scrum Master

- Typically a Project Manager or Team Leader
- Responsible for enacting Scrum values and practices
- Remove impediments / politics, keeps everyone productive

Project Team

- 5-10 members; Teams are self-organizing
- Cross-functional: QA, Programmers, UI Designers, etc.
- Membership should change only between sprints

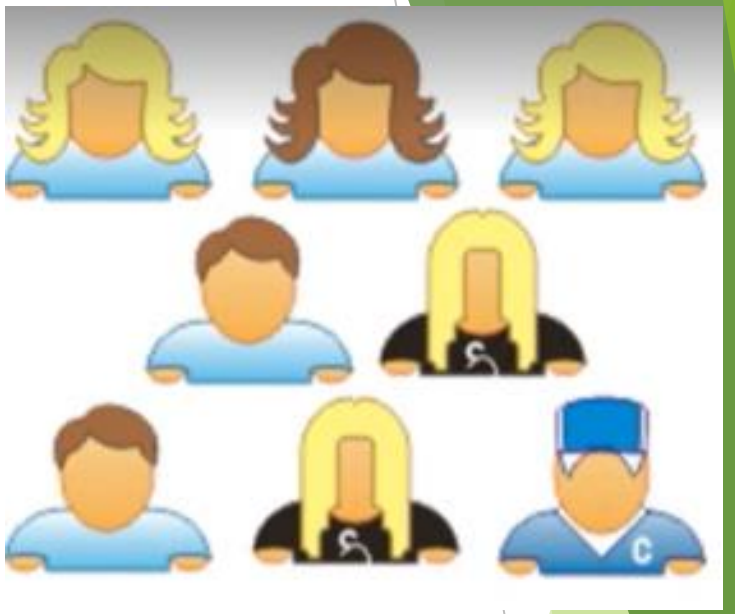
Product Owner

- Define the features of the product
- Decide on release date and content
- Be responsible for the profitability of the product (ROI)
- Prioritize features according to market value
- Adjust features and prioritize every iteration, as needed
- Accept or reject work results

The Scrum Master

- Represents management to the project
- Responsible for enacting Scrum values and practices
- Removes impediments
- Ensure that the team is fully functional and productive
- Enable close cooperation across all roles and functions
- Shield the team from external interferences

The Team



- Typically 5-9 people
- Cross-functional:
  - Programmers, testers, user experience designers, etc.
- Members should be full-time
- Teams are self-organizing
  - Ideally, no titles but rarely a possibility
- Membership should change only between sprints

Development Team

- They are self-organizing. No one (not even the Scrum Master) tells the Development Team how to turn Product Backlog into Increments of potentially releasable functionality;
- Development Teams are cross-functional, with all the skills as a team necessary to create a product Increment;
- Scrum recognizes no titles for Development Team members, regardless of the work being performed by the person;
- Scrum recognizes no sub-teams in the Development Team, regardless of domains that need to be addressed like testing, architecture, operations, or business analysis
- Individual Development Team members may have specialized skills and areas of focus, but accountability belongs to the Development Team as a whole.

Scrum Master Service to the Product Owner

- Ensuring that goals, scope, and product domain are understood by everyone on the Scrum Team as well as possible;
- Finding techniques for effective Product Backlog management;
- Helping the Scrum Team understand the need for clear and concise Product Backlog items;
- Understanding product planning in an empirical environment;
- Ensuring the Product Owner knows how to arrange the Product Backlog to maximize value;
- Understanding and practicing agility
- Facilitating Scrum events as requested or needed.

Scrum Master Service to the Development Team

- Coaching the Development Team in self-organization and cross-functionality
- Helping the Development Team to create high-value products;
- Removing impediments to the Development Team's progress;
- Facilitating Scrum events as requested or needed
- Coaching the Development Team in organizational environments in which Scrum is not yet fully adopted and understood.

Scrum Master Service to the Organization

- Leading and coaching the organization in its Scrum adoption;
- Planning Scrum implementations within the organization;
- Helping employees and stakeholders understand and enact Scrum and empirical product development;
- Causing change that increases the productivity of the Scrum Team
- Working with other Scrum Masters to increase the effectiveness of the application of Scrum in the organization.

The Sprint

- The heart of Scrum is a Sprint, a time-box of one month or less during which a “Done”, useable, and potentially releasable product Increment is created.
- Sprints have consistent durations throughout a development effort.
- A new Sprint starts immediately after the conclusion of the previous Sprint.
- Sprints contain and consist of the Sprint Planning, Daily Scrums, the development work, the Sprint Review, and the Sprint Retrospective.

Cont…

- Each Sprint may be considered a project with no more than a one-month horizon.
- Each Sprint has a goal of what is to be built, a design and flexible plan that will guide building it, the work, and the resultant product increment.
- Sprints are limited to one calendar month.

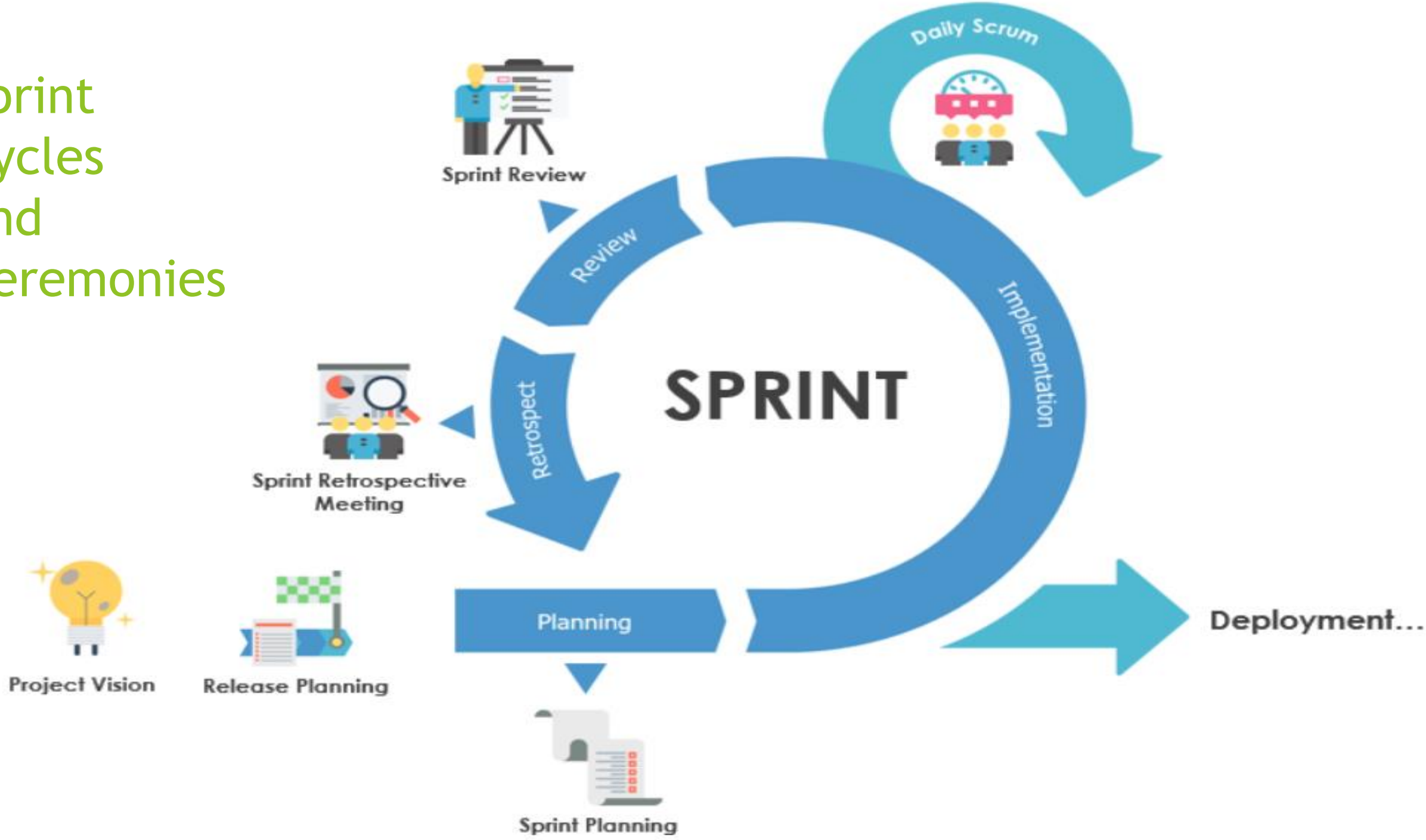
During the Sprint:

- No changes are made that would endanger the Sprint Goal;
- Quality goals do not decrease
- Scope may be clarified and re-negotiated between the Product Owner and Development Team as more is learned.

CANCELLING A SPRINT

- A Sprint can be cancelled before the Sprint time-box is over.
- Only the Product Owner has the authority to cancel the Sprint, although he or she may do so under influence from the stakeholders, the Development Team, or the Scrum Master.
- A Sprint would be cancelled if the Sprint Goal becomes outdated.
- This might occur if the company changes direction or if market or technology conditions change.
- In general, a Sprint should be cancelled if it no longer makes sense to the given circumstances.
- Due to the short duration of Sprints, cancellation rarely makes sense.

Sprint Cycles and Ceremonies



Sprint Planning

- Team selects items from the product backlog they can
  - commit to completing
- Sprint backlog is created
  - Tasks are identified and each is estimated
    - Collaboratively, not done alone by the Scrum Master
- High-level design is considered

Parts of Sprint Planning Meeting

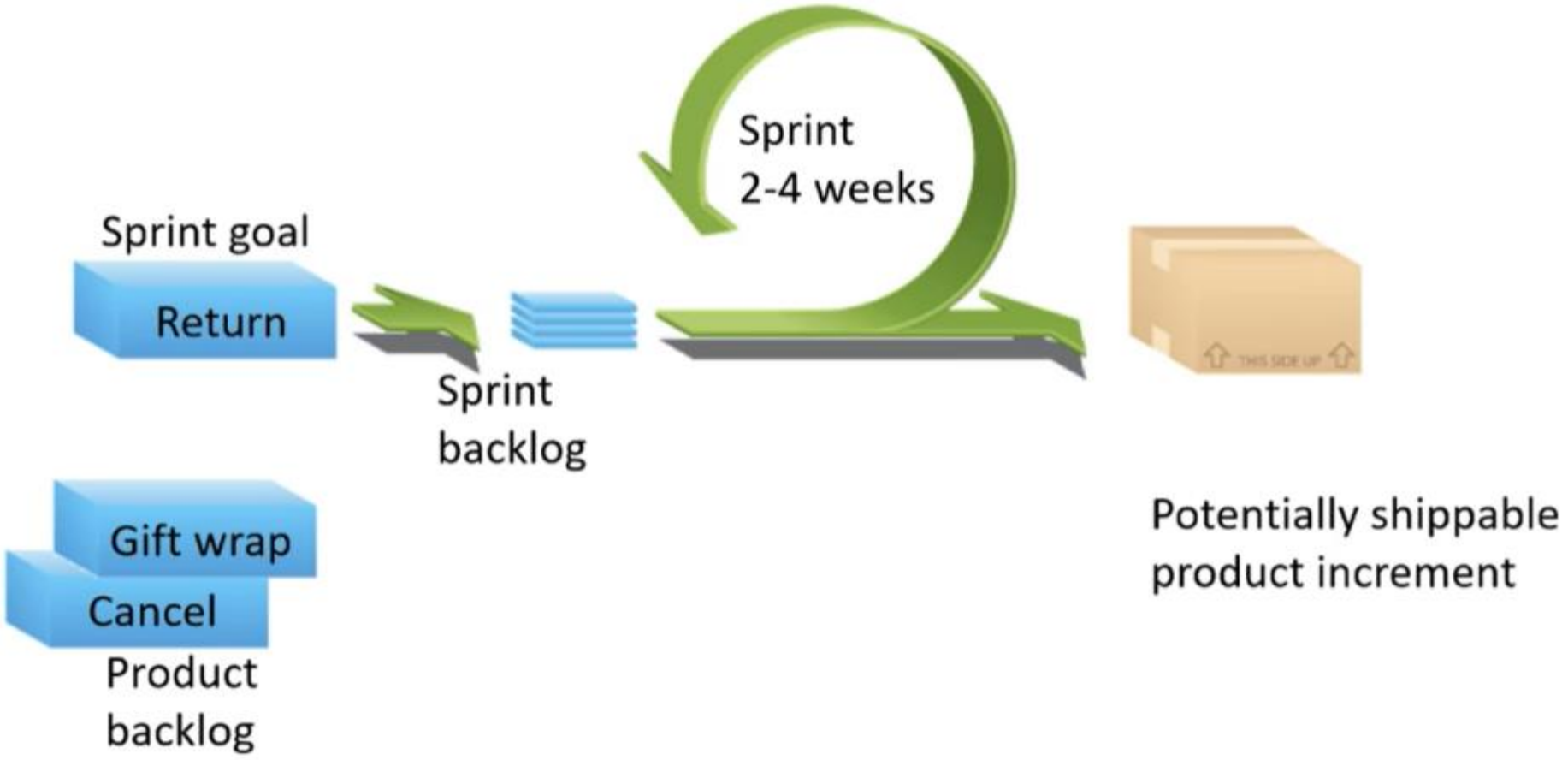
Planning Session 1

- Creating Product Backlog
- Determining the Sprint Goal.
- Participants: Product Owner, Scrum Master, Scrum Team

Planning Session 2

- Participants: Scrum Master, Scrum Team
- Creating Sprint Backlog

Process involved in Sprint Planning

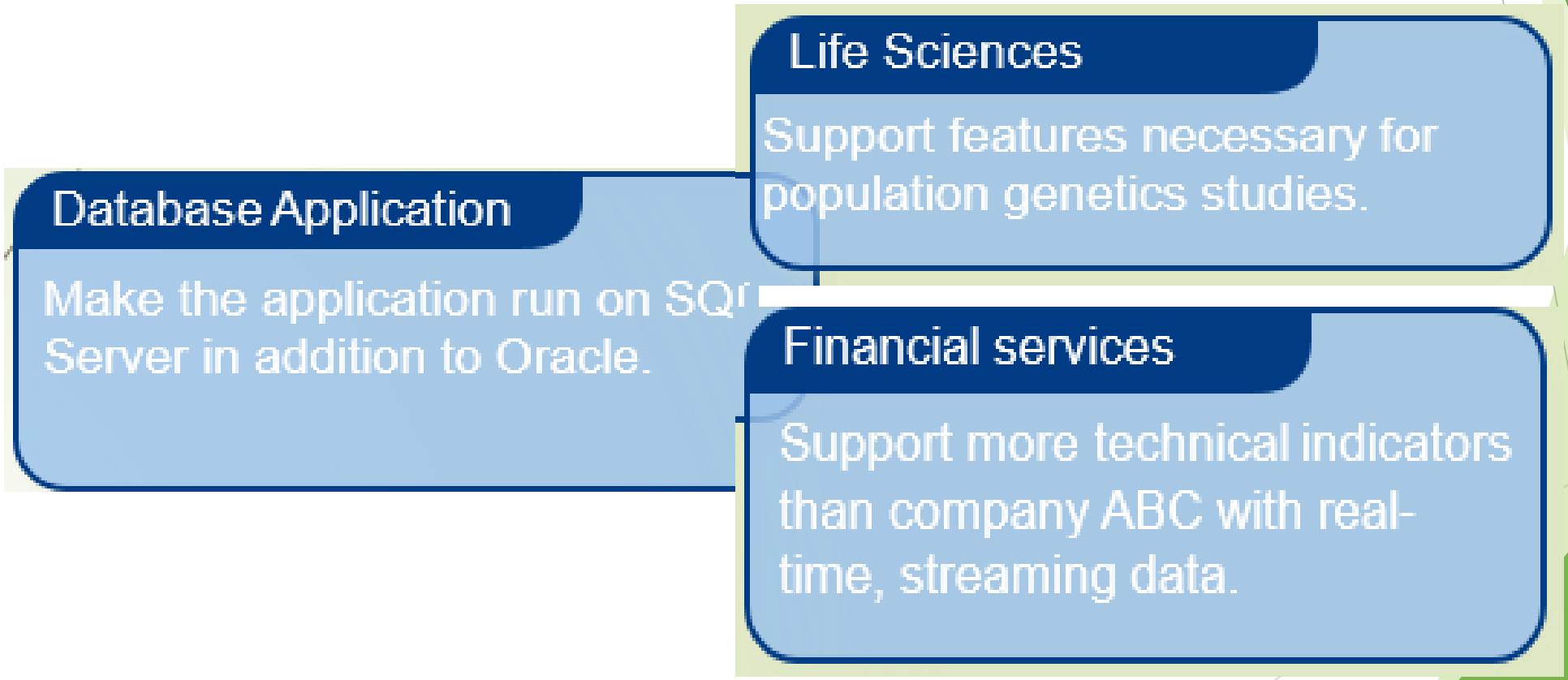


Sprint Planning Answers,

- **WHAT CAN BE DONE IN THIS SPRINT?**
- **HOW WILL THE CHOSEN WORK GET DONE?**

Sprint Goal

- A short statement of what the work will be focused on during the sprint



Sprint Review

- Team presents what it accomplished during the sprint
- Typically takes the form of a demo of new features

or underlying architecture

- Informal
  - 2-hour prep time rule
  - No slides
- Whole team participates
- Invite the world

Sprint Retrospective

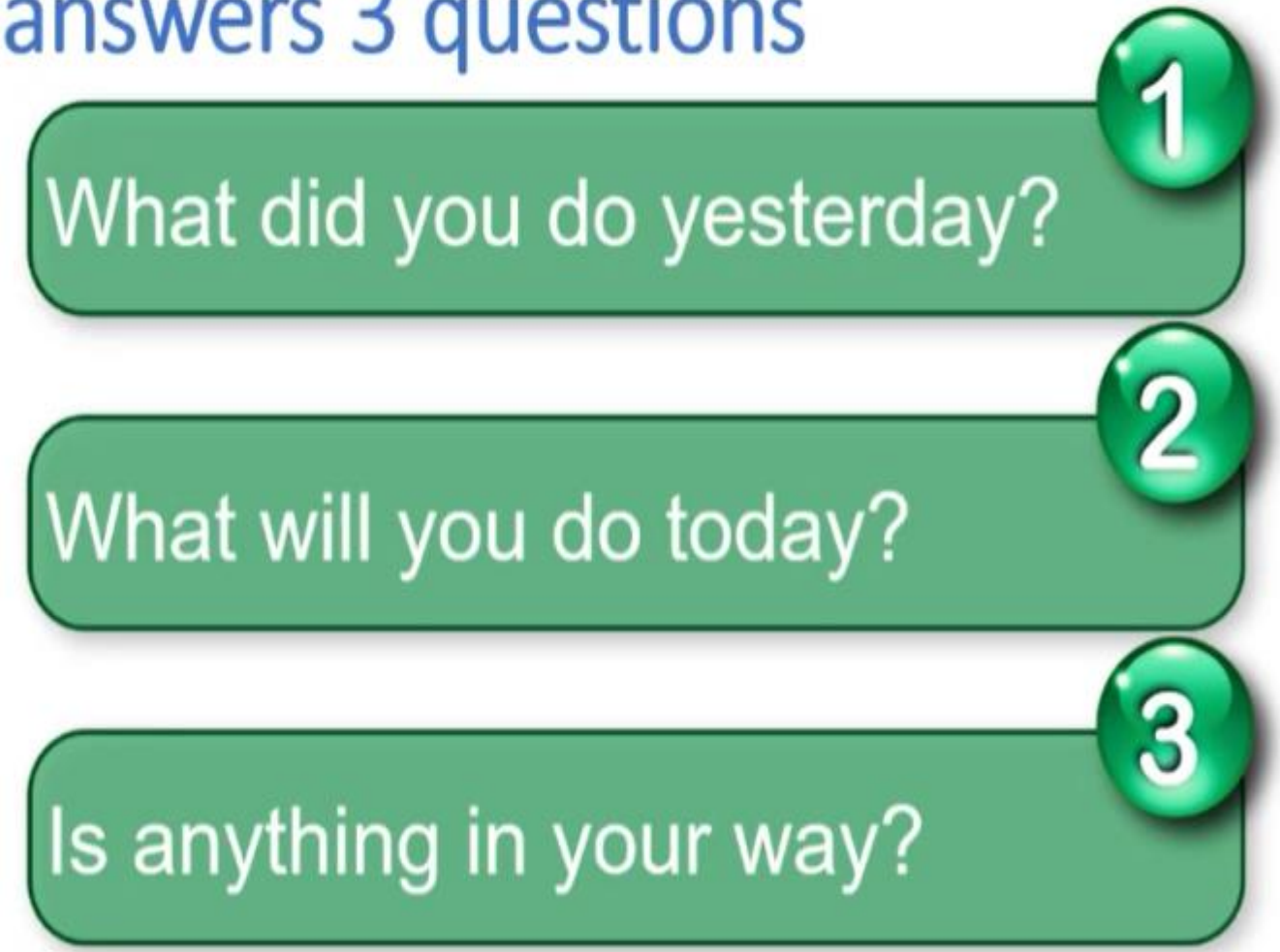
- Periodically take a look at what is and is not working
- Typically 15–30 minutes
- Done after every sprint
- Whole team participates
  - ScrumMaster
  - Product owner
  - Team
  - Possibly customers and others

Start/ Stop/ Continue

- Whole team gathers and discusses what they'd like to:

Daily Scrum

Everyone answers 3 questions



Cont…

- **Parameters**
  - Daily, ~15 minutes, Stand-up
  - Anyone late pays a $1 fee

- **Not for problem solving**
  - Whole world is invited
  - Only team members, Scrum Master, product owner, can talk
  - Helps avoid other unnecessary meetings

Scrum Artifacts

- Scrum has remarkably few artifacts
  - Product Backlog
  - Sprint Backlog
  - Burn down Charts

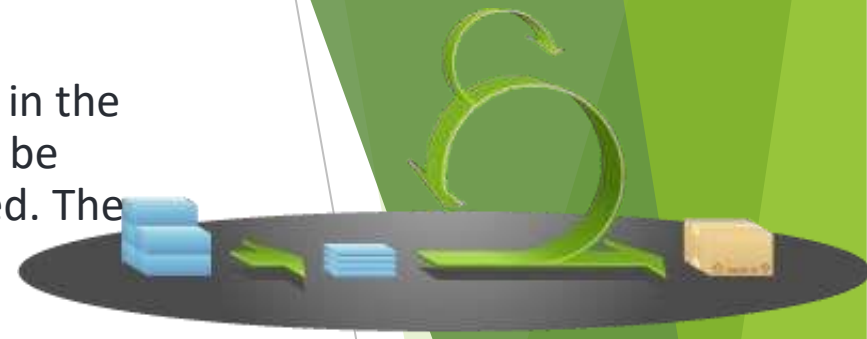
- Can be managed using just an Excel spreadsheet
  - More advanced / complicated tools exist:
    - Expensive
    - Web-based – no good for Scrum Master/project manager who travels
    - Still under development

Product Backlog

- The product backlog is an ordered list of everything that is known to be needed in the product. It sits outside of the sprint loop (meaning it contains work that will not be completed during the current sprint) but informs how your sprint will be planned. The product backlog is composed of feedback from:
  - The development team
  - Customer
  - Stakeholders

In general the Product backlog tends to mention the,

- The requirements
- A list of all desired work on project
- Ideally expressed as a list of user stories along with "story points", such that each item has value to users or customers of the product
- Prioritized by the product owner
- Reprioritized at start of each sprint



This is the product backlog

Significance of Product Backlog

- Think of a product backlog as a way of putting a brainstorm or a product plan into action. Undoubtedly developers will be approached by stakeholders (or customers) who have many ideas for improving the product. Not all the ideas are good and not all the ideas are valuable, but without an organized product backlog, it's difficult to differentiate between the great, valuable ideas and the ideas that would only be a waste of time. Here are some other benefits of the product backlog:

1. It's an organized list that's easily wrangled.
2. It's simple to prioritize.
3. It can be changed as priorities change.
4. It allows you to immediately see dependencies and order them.
5. It allows you to think about products in the long-term, not just in terms of immediate needs.

- In short, a product backlog allows you and your team to make systematic, smart improvements to a product over the long haul.

What's in the product backlog?

- The Scrum Guide is fairly prescriptive about what can be in the product backlog, which is helpful for keeping unnecessary items out. The product backlog contains:

  - ✓ Features
  - ✓ Functions
  - ✓ Requirements
  - ✓ Enhancements
  - ✓ Fixes

- It's not just a simple to-do list, though. Each item in the product backlog:

  - ✓ Adds value for the customer
  - ✓ Is prioritized
  - ✓ Is estimated

- There should be no low-level tasks in your backlog (like sending emails), and the backlog itself should be a living document that's regularly rearranged.

How to create a product backlog

- To create your product backlog, consider using a more flexible software solution. A flexible software that assists in creating a backlog template that gets updated on day-to-day basis is the easiest way to start building your scrum product backlog. It's a living document that's easy to share with stakeholders and rearrange however you'd like. The steps to start your scrum product backlog are as follows,

1. Add ideas to the backlog

- Stakeholders will typically be approaching you with ideas for product improvements

2. Get clarification

Once you're approached by a stakeholder with a product addition or fix, make sure you understand:

- The reason behind the addition or fix
- The amount of value it contributes to the product as a whole
- The specifications of the item

Contd…

3. Prioritize

- The backlog should have clearly defined, high-priority items at the top and vague items that are not a priority at the bottom. If an item has no value, it should not be added to the backlog.

4. Update the backlog regularly

- The backlog is a living document; make sure you're constantly prioritizing, refining, and keeping the backlog up to date.
- You may have hundreds of items in your backlog as ideas for product improvements are suggested. Some of these items may be discarded, but many of them will begin making their way up the backlog for further refinement and, ultimately, development.
- Get started quickly with a customizable product backlog template.

High Priority

PRODUCT BACKLOG

Low Priority

User Story	Story Point(s)	Priority
As a User, I am able to search for documents so I can find them more easily	2	1
As a site visitor, I can compare different types of accounts to see which account type suites me best	1	2
As a User, I can submit questions through the website so I know how to better use the product	1	3
As a site visitor, I am shown what I can do in the product so I know whether or not this product will fulfil my needs	2	4
As a User, I want to be able to retrieve documents that were deleted so I can reclaim documents that were accidentally got deleted	3	5
As a site visitor and User, I can sign up for newsletters to remain up to date on the product	2	6
As a user, I am notified when a new feature is released so I knew what are all the possible changes likely to be occurred	1	7
As a user I can change my username if desired	3	8
As an admin, I need the ability to update which team a user belongs to so I can make sure all teams are up to date	3	9
As a user, I can enable spell check, so I can be confident my final document has no spelling errors	4	10

Product backlog prioritization

- The product backlog itself is owned by the product owner. The product owner's job is to produce the very best product possible, so that means developing the most valuable additions to the software first. Since the product backlog is ranked in order of most valuable components, it would stand to reason that the most valuable addition would be at the very top. But that's not necessarily the case, as the most valuable addition likely has dependencies that need to be developed first.
  - **Higher-priority items** should be refined and have great value to the product.
  - **Mid-priority items** should be candidates for refinement (the process of detailing each task)
  - **Low-priority items** should not be a dependency and can be safely ignored until they are candidates for refinement.
- As items progress closer to the top of the list to be added to the next sprint cycle, they should be refined so they can be better acted upon. Here's a helpful tip: color-code each block to indicate an item is sufficiently refined and ready for sprint planning by coloring it green. You may wish to indicate mid-priority items with yellow and low-priority items with red. Or go crazy and make everything neon.

Product refinement

- Product refinement is the process of refining the tasks in the product backlog so that they’re clear enough to be action items instead of nebulous ideas.
- Say your team is developing a Medico app. One of the requests from stakeholders and customers has been to have an integrated background checker, so you add that to the product backlog. However, that’s not nearly defined enough to start assigning tasks for developing the background checker.
- Add necessary details right in each task of the product backlog so there’s never any confusion about what each item is. For instance, with our Medico app background checker, you can easily detail what kind of specialization you’ll be required with to provide a background check, what information should be gathered from the user to perform the background check, and what the ultimate goal of the background check should be. You can also easily add links, pictures, or any other information.

Sprint planning

- A scrum product backlog ultimately makes sprint planning much easier.
- After all, your to-do items are already defined for you in the backlog, and they can be easily moved onto your scrum board.
- Using each item's estimation allows you to determine how many action items can be added to the next sprint.
- Then it's a matter of following the sprint cycle guidelines to follow each item through to completion.
- A product backlog provides a bird's-eye view into upcoming items that can be added into the product, but its value really shines in its ability to organize, refine, and define action items.
- Ultimately, you'll be allowed to focus on systematically adding value into your product instead of trying to shift through the chaos.

SPRINT BACKLOG

Story 1

- Find existing Code
- Identify a Solution
- Update / Rewrite a Code
- Complete automated testing
- Complete testing
- Release Fix

Story 2

- Design the Pricing Web Page
- Publish pricing Web Page
- Write Code
- Complete automated testing
- Discuss CTA to include

Story 3

- Design a Feedback Form
- Write Code
- Complete testing
- Release Code
- Create a Platform to Store Feedback

Sprint Backlog

A sprint backlog is a list of work items your team plans to complete during a project sprint. These items are usually pulled from the product backlog during the sprint planning session. A clear sprint backlog prevents scope creep by clarifying exactly what your team will be doing—and not doing—during each sprint.

As the Scrum master or product owner, it's your responsibility to create the sprint backlog and distribute it to all project stakeholders. With the help of the product owner, you will choose backlog items based on priority. Then, document each task's needs in the form of user stories. These are software features written from the perspective of the end user within a workflow. Key points to be noted are,

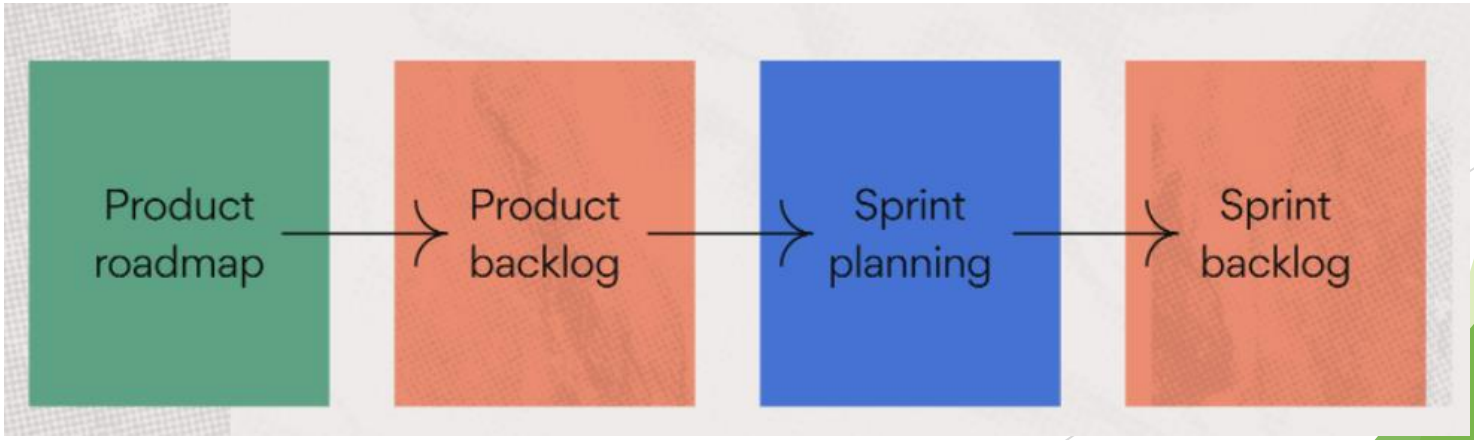
- Individuals sign up for work of their own choosing
- Work is never assigned
- Estimated work remaining is updated daily
- Any team member can add, delete change sprint backlog
- Work for the sprint emerges
- If work is unclear, define a sprint backlog item with a larger amount of time and break it down later
- Update work remaining as more becomes known

Purpose of a sprint backlog

- The purpose of a sprint backlog is to define work items to tackle within the sprint. This keeps information in one shared space in order to streamline communication and create one central source of sprint information.
- Items that are not in the backlog are not in scope. This creates a clear path, ensuring team members can focus on the task ahead to avoid building tings out of scope.

When is a sprint backlog created?

- Create a sprint backlog during the planning phase of a new project sprint. While you can update individual tasks with details and additional progress during the sprint, the backlog itself shouldn't alter during execution.
- The log is then stored in a shared space for stakeholders and Scrum masters to review during a retrospective meeting to evaluate what went well and what didn’t.



Sprint Backlog

No more than 300 tasks in the list

If a task requires more than 16 hours, it should be broken down

Team can add or subtract items from the list. Product Owner is not allowed to do it

Cont…

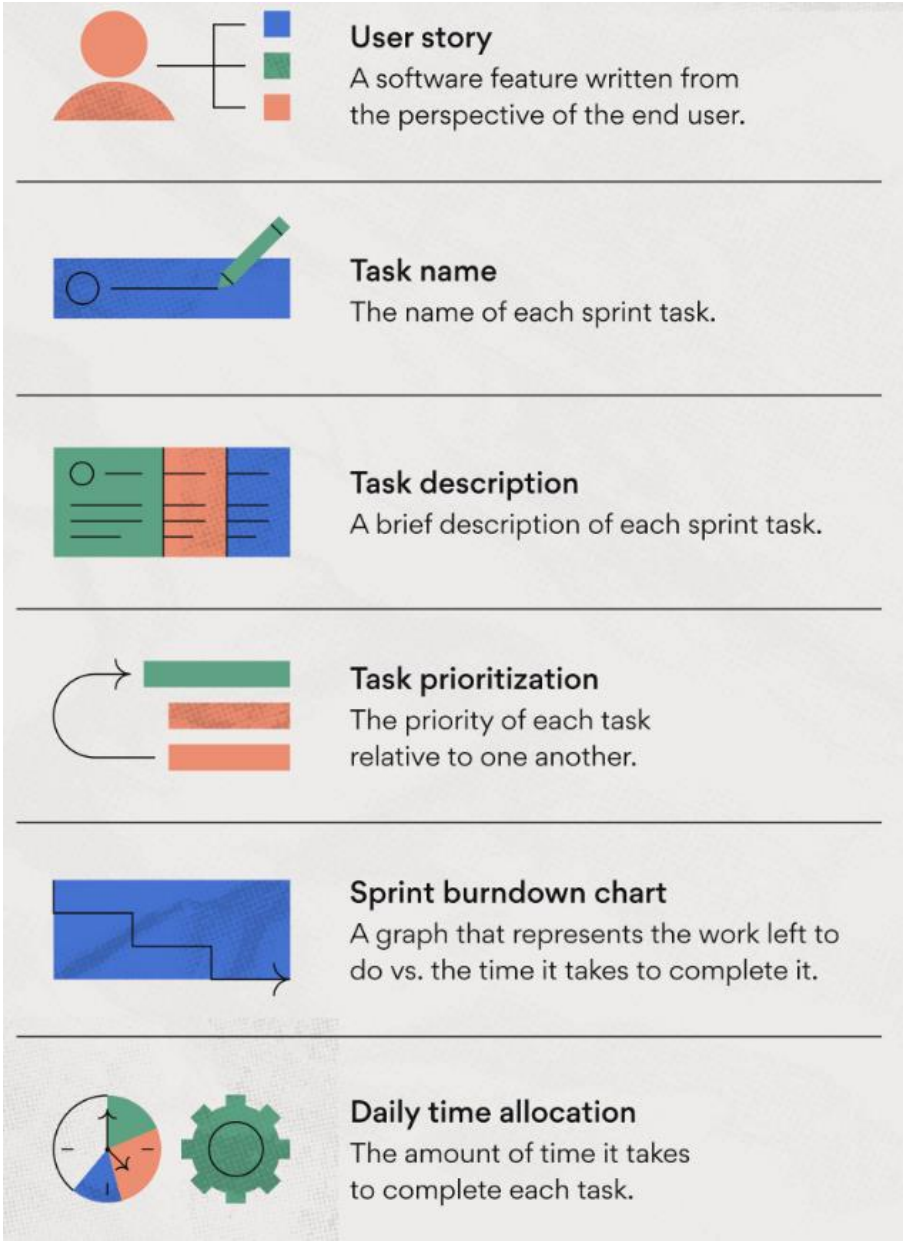
A subset of Product Backlog Items, which define the work for a Sprint

Is created ONLY by Team members

Each Item has it’s own status

Should be updated every day

Components of a Sprint Backlog



User story
A software feature written from the perspective of the end user.

Task name
The name of each sprint task.

Task description
A brief description of each sprint task.

Task prioritization
The priority of each task relative to one another.

Sprint burndown chart
A graph that represents the work left to do vs. the time it takes to complete it.

Daily time allocation
The amount of time it takes to complete each task.

Components of a Sprint Backlog

- **User story**- A user story is a software feature written from the perspective of the end user. It's an important piece to include in order to understand the effect each feature has on the end user.

- **Task name-** While obvious enough, keep your backlog organized by starting each task with a clear, action-oriented name. Ensure each task title starts with a verb—for example, "Design new mobile component for web app" is more descriptive than "New mobile component." This will help stakeholders quickly understand the backlog and deliverables that each team member is working on.

- **Task description-** Along with an actionable name, include a brief description of each task. This creates clarity around tasks so stakeholders are aware of upcoming steps.

Contd…

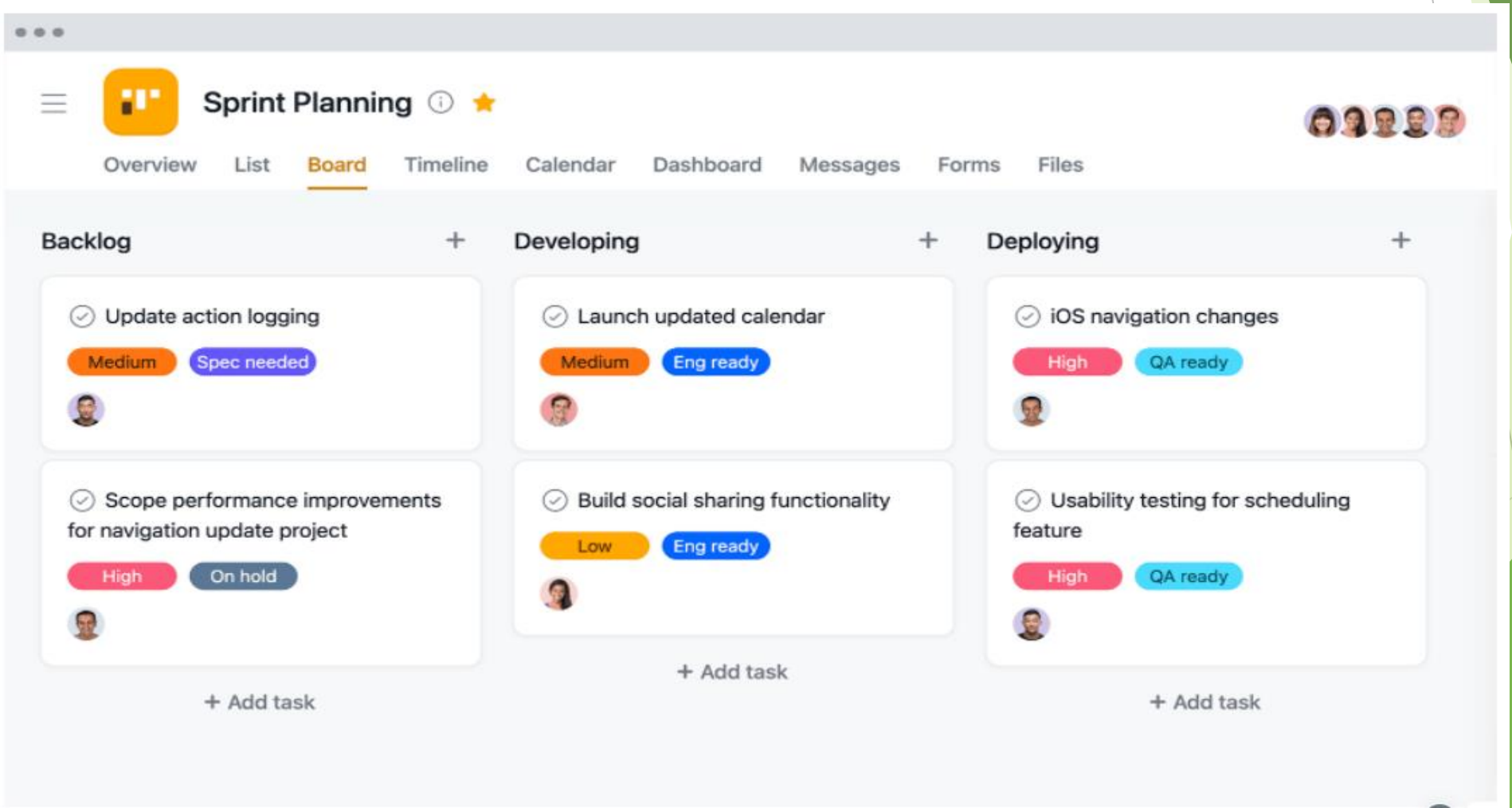
- **Task prioritization-** Since there are a number of tasks in a given project, it's important to prioritize your most important objectives. This ensures you meet deadlines and your sprint stays on track.

- **Sprint burndown chart-** A burndown chart is a graph that represents the work left to do versus the time it takes to complete it. During a sprint, your team will use these charts to estimate how long each iteration will take.

- **Daily time allocation-** In order to track your time estimates against the actual time on your burndown chart, you need to track daily time allocations. Analyze how long each task takes in minutes or hours. At the end of the week, total up your weekly time allocations for each task to complete your burndown chart.

Creating a Sprint Backlog

- Since Scrum masters use a new backlog for each sprint, it's important to have a baseline to work off
- The key to creating a backlog is to make a blank template that you can use for each of your sprints. In your template, you should include columns for each of the functionalities listed above.



Sprint Backlog

USER STORY	TASKS	DAY 1	DAY 2	DAY 3	DAY 4	DAY 5	...
As a registered Member I can Update my Billing Information	Update security tests	6	6	4	0		
	Design a solution to alter the billing Structure	12	6	0	0		
	Write a test plan	8	8	4	0		
	Automate tests	12	12	10	6		
	Code the Solution	8	8	8	4		

Monitoring Sprint Progress

- At any point in time in a Sprint, the total work remaining in the Sprint Backlog can be summed.
- The Development Team tracks this total work remaining at least for every Daily Scrum

Increment

- The Increment is the sum of all the Product Backlog items completed during a Sprint and the value of the increments of all previous Sprints.
- At the end of a Sprint, the new Increment must be “Done,”
- An increment is a body of inspectable, done work that supports empiricism at the end of the Sprint.
- The increment is a step toward a vision or goal.
- The increment must be in useable condition regardless of whether the Product Owner decides to release
- Each Increment is additive to all prior Increments and thoroughly tested, ensuring that all Increments work together.

Artifact Transparency

- Scrum relies on transparency.
- Decisions to optimize value and control risk are made based on the perceived state of the artifacts
- The Scrum Master must work with the Product Owner, Development Team, and other involved parties to understand if the artifacts are completely transparent.
- There are practices for coping with incomplete transparency
- The Scrum Master must help everyone apply the most appropriate practices in the absence of complete transparency.

Definition Of “Done” (DOD)

- When a Product Backlog item or an Increment is described as “Done”
- Scrum Team members must have a shared understanding of what it means for work to be complete, to ensure transparency.
- This is the definition of “Done” for the Scrum Team and is used to assess when work is complete on the product Increment.
- It also guides the Development Team in knowing how many Product Backlog items it can select during a Sprint Planning.
- The purpose of each Sprint is to deliver Increments of potentially releasable functionality that adhere to the Scrum Team’s current definition of “Done.”

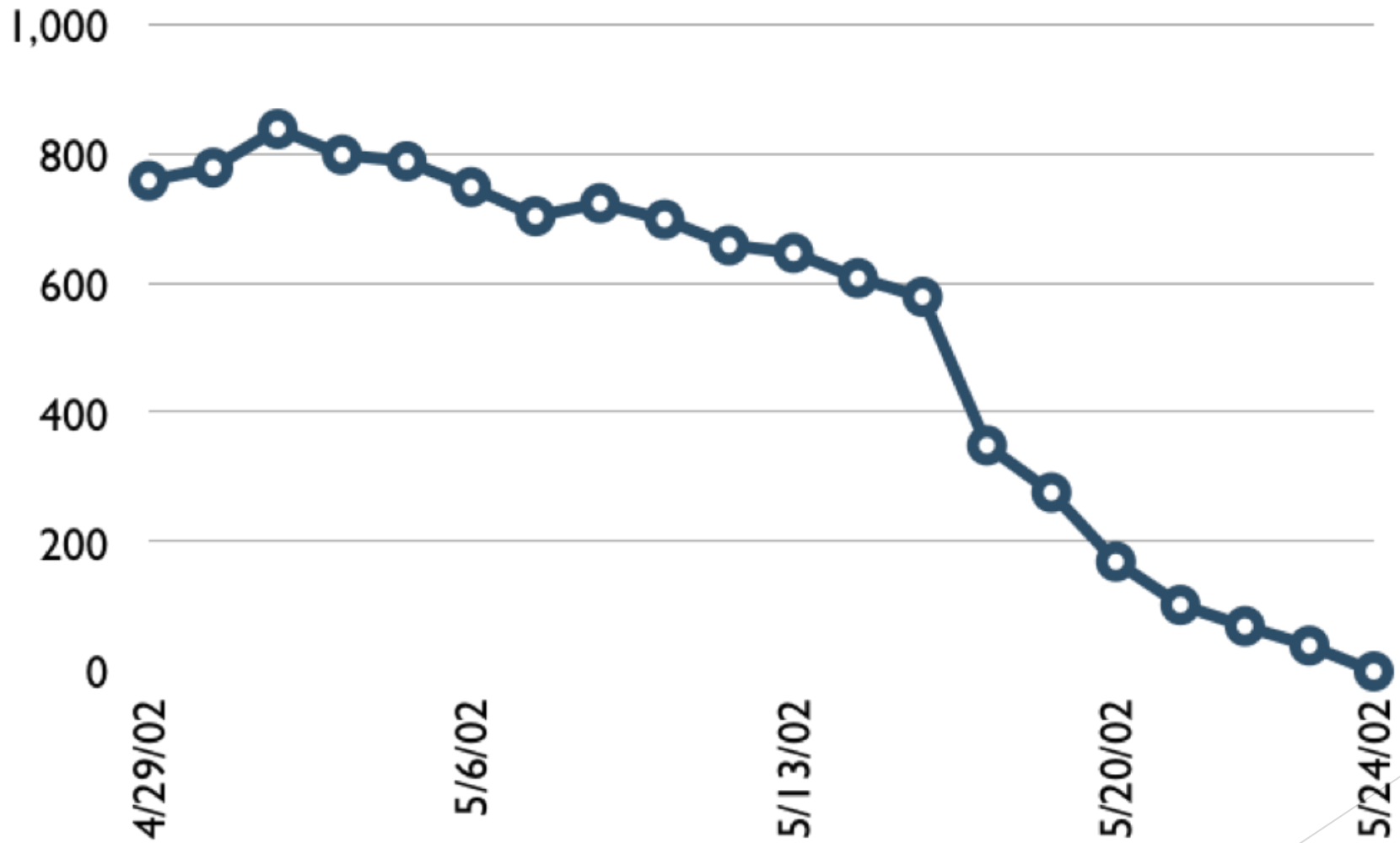
Cont…

- As Scrum Teams mature, it is expected that their definitions of “Done” will expand to include more stringent criteria for higher quality.
- New definitions, as used, may uncover work to be done in previously “Done” increments.
- Any one product or system should have a definition of “Done” that is a standard for any work done on it.

Burn down Chart

- A display of what work has been completed and what is left to complete
  - one for each developer or work item
  - updated every day
  - (make best guess about hours/points completed each day)
- variation: Release burn down chart
  - shows overall progress
  - updated at end of each sprint

Sample Burn down Chart



Project Velocity

- Velocity is a measure of the amount of work a Team can tackle during a single Sprint and is the key metric in Scrum.
- Velocity is calculated at the end of the Sprint by totaling the Points for all fully completed User Stories.
- Points from partially-completed or incomplete stories should not be counted in calculating velocity.
- Velocity should be tracked throughout the Sprint on the Sprint burn down chart and be made visible to all Team members.
- Velocity is a key feedback mechanism for the Team.
- It helps them measure whether process changes they make are improving their productivity or hurting it.

Cont…

- While a Team's velocity will oscillate from Sprint to Sprint, over time, a well-functioning Scrum Team's velocity should steadily trend upward by roughly 10% each Sprint.
- Used to calculate the amount of work completed and it helps to plan the future sprint
- For the forecasting purpose the average of the last three sprints velocity should be used
- Without velocity release planning will be impossible

Agile Velocity Definition

- Agile Velocity help you to understand how long it will take your team to finish the whole backlog.
- In general, it takes few sprint to get to know the team velocity.
- Average of the amount of user story completed by your team for e.g past three sprint should be considered. Calculate the average of those 3 sprints and it is your team velocity.
- Assuming your team is completing 5-7 user stories in each sprint with total story points of 25-35. So, the average velocity of team in past three is 25 – 35

Calculating Sprint Velocity in Agile Development

- **Assuming that team is having one week sprint.**

<u>Calculating Sprint 01 Velocity in Agile Scrum</u>

- Assuming that in sprint 01, team gets committed to total of 5 user stories
- Story point of each of the user story = 8 story points
- Total story points committed by team in sprint 01 = 40 story points
- **By end of sprint 01 team completed 3 out of 5 user stories**
- Total user stories completed in sprint 01 = 3 user stories
- Total story points completed in sprint 01 = 3 x 8 story points -> 24 story points

Cont…

Calculating Sprint 02 Velocity in Agile Scrum

- Assuming that in sprint 02, team committed to total of 7 user stories
- Story point of each of the user story = 8 story points
- Total story points committed by team in sprint 01 = 56 story points

  By end of sprint 2 team completed 4 out of 7 user stories,

- Total user story completed in sprint 02 = 4 user stories
- Total story points completed in sprint 02 = 4×8 story points->32 story points

Cont…

Calculating Sprint 03 Velocity in Agile Scrum

- Assuming that in sprint 03, team committed to total of 9 user stories
- Story point of each of the user story = 8 story points
- Total story points committed by team in sprint 03 = 72 story points

By end sprint of 03, team completed 5 out of 9 user stories

- Total user story completed in sprint 03 = 5 user stories
- Total story points completed in sprint 03 = 5×8 story points->40 story points

Calculating Average Velocity of Sprints

- knowing the velocity of past three sprints, calculate the average of them.
- Number of User Stories completed in first 3 sprints being released by that team is,
  - Sprint 01: 24
  - Sprint 02: 32
  - Sprint 03: 40
- **So average velocity of sprint is: 24+32+40/32 = 32**
- That is the average velocity in past three sprints.
- This is the reference how a team will be finishing the user stories in a sprint.

Cont…

- successfully complete at least 3-5 sprints and check your completed story points in each of the prints, this will give you better visibility and help you in planning future sprints, when planning your sprint, velocity will provide you reference how much user stories can be completed in a sprint, by using this value you are sure that you are not over or under planning your sprint.
- So the total number of user story taken during a sprint should not be exceeding the average velocity of past sprints.

Velocity Graph

Add Active Sprints ☑

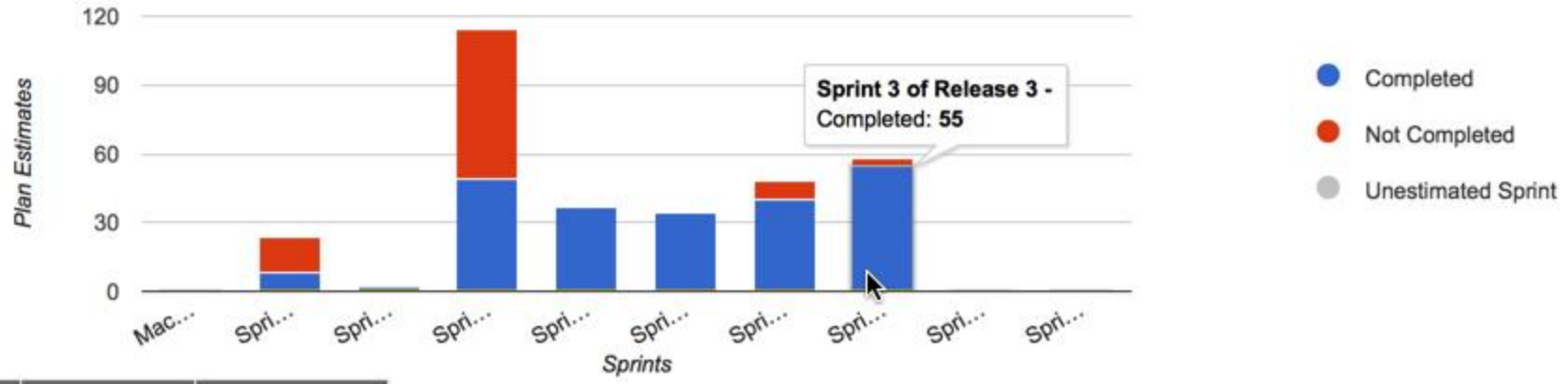
Number of Sprints 10

Average Velocity (10 sprints)

22

Best velocity : 55

Worst velocity : 0



Sprint	Commitment	Completed	Not Completed
Mac App - Sprint 1 *	0	0	0
Sprint 2 of Train 1	24	8	16
Sprint 1 of Train 1	2	2	0
Sprint 7 of Release 3 - Bug Fixing	114	49	65
Sprint 6 of Release 3	37	37	0
Sprint 5 of Release 3	34	34	0
Sprint 4 of Release 3	48	40	8
Sprint 3 of Release 3 -	58	55	3
Sprint 1 & 2 of Release 3	0	0	0
Sprint 2 of Rel. 2 - (30- 10 Mar/Apr)	0	0	0

Utilizing Average Velocity In Agile Development

Release Planning

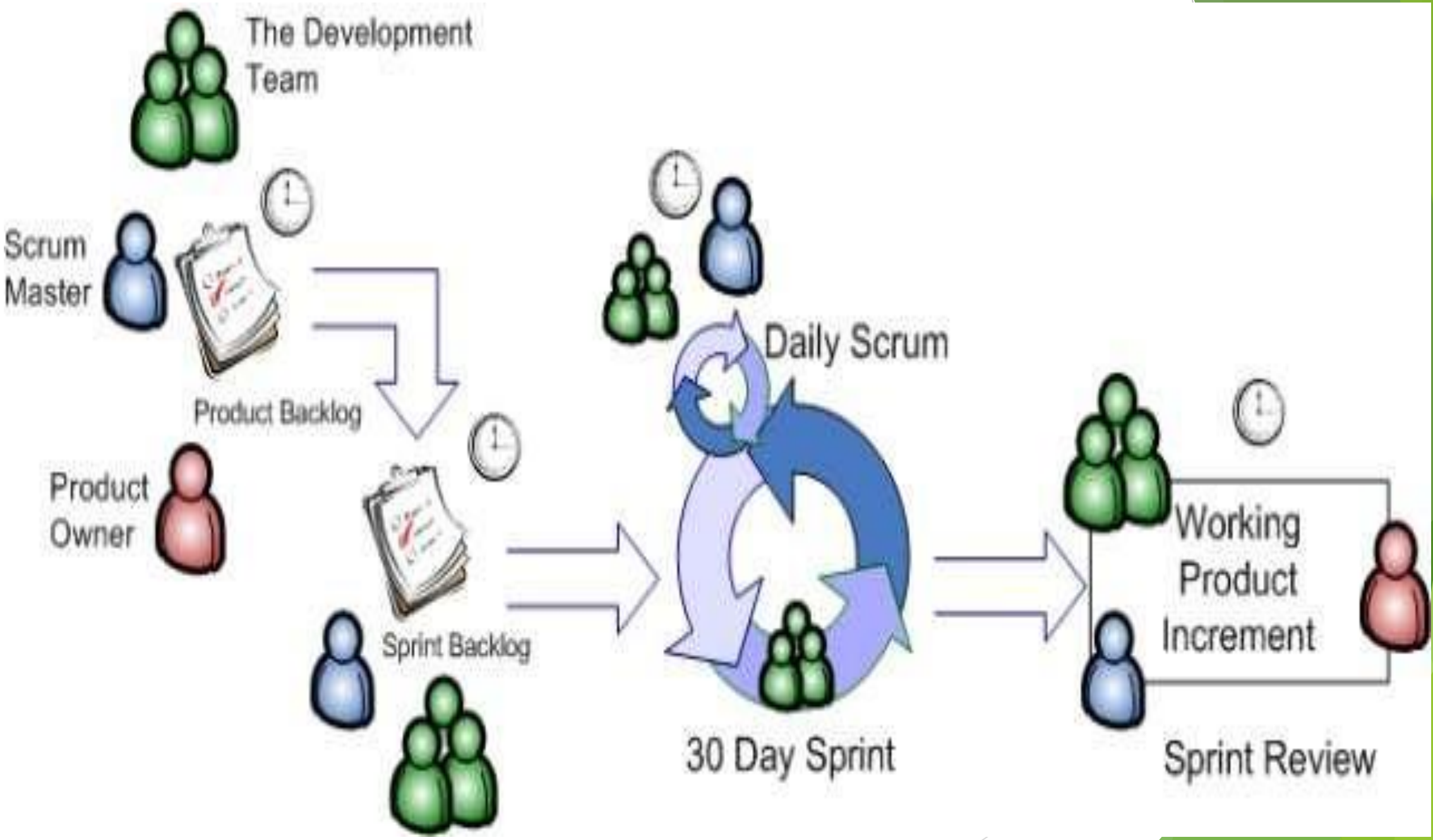
- Use average velocity for planning releases, for e.g, if you are a product owner, you need to know when you can deliver all the features.
- If you know the team's velocity you can put all user stories from product backlog in sprints to calculate how many sprint it will take to finish all the features.
- Assuming you have two weeks sprints, and your average team's velocity is 33.
- So you can put the swim lane of sprints to see how many sprints will take to complete all the features (putting user stories in sprints using the velocity of sprint)
- This is a useful way of predicting your time-line of delivering the final product with all the features.

Sprint Planning based on Velocity

- Normally average velocity is only used by product owner when doing the release planning
- Instead this should be use in sprint planning as well
- normally scrum team uses hours when calculating a user story,
- As an alternate story points should be use for estimation and average velocity should be use to make sure sprint is not over or under planned.

User Story points instead of Hours

- developers are pretty bad at estimating time, a very complex task can be done in 2 hours from the point of view of a developer, but to maturing it will take 8+ hours
- using the abstract value like story points which compare complexity in relative terms will be easier for developers, it takes off the actual time in hours and put the focus on relative complexity of task.
- We use the Fibonacci number from 1- 8 anything above that number needs to be divided into smaller user stories.
- using planning poker which give the edge to start a technical discussion among team during sprint planning.


The Development Team
Scrum Master
Product Backlog
Product Owner
Sprint Backlog
Daily Scrum
30 Day Sprint
Working Product Increment
Sprint Review

Acceptance Test and Verifying User Stories

- Functional testing of the user story by the software development team during the implementation phase
- QA then create acceptance test or scenarios based on the acceptance criteria to ensure sufficient test coverage
- Black-box system test
- Customer are responsible for verifying test scores to decides which failed test tests are of highest priority
- User stories are not considered complete until it has passed its acceptance tests

Acceptance Criteria

- Ensures the team knows when they are done
- Ensures the team does not forget important edge cases and considerations
- Produced through collaboration of the developers, testers and product owners (3 amigos)
- Created prior to development, during planning phase
- Expressed at a high level (conceptual, not detailed)
- Expressed in whatever form works best for the team…keep it minimal
- Considers edge cases and failure scenarios
- Keep it concise (minimum documentation needed by team…may be more for one team, less for another)

Acceptance Tests

- Defines behavior of system
- Ensures the feature works as expected
- Code implemented by developers and testers
- Test definition can include product owners or customers if you are using a DSL that decouples the definition of a test from its implementation code (like with Gherkin or FitNesse)
- Test definition can happen before development if using a DSL as mentioned above
- Test implementation occurs during development (ideally in a test-first manner)
- Tests are usually implemented in code, but if testing manually (hopefully only rarely), the “implementation” can be a list of steps/expectations

Roles

Product Owner: Set priorities

ScrumMaster: Manage process, remove blocks

Team: Develop product

Stakeholders: observe & advise

Key Artifacts

Product Backlog

- List of requirements & issues
- Owned by Product Owner
- Anybody can add to it

Sprint Goal

- One-sentence summary
- Declared by Product Owner

Sprint Backlog

- List of tasks
- Owned by team

Blocks List

- List of blocks & unmade decisions
- Owned by ScrumMaster

Increment

- Version of the product
- Shippable functionality (tested,

Key Meetings

Sprint Planning Meeting

- Hosted by ScrumMaster; ½-1 day
- In: Product Backlog, existing product, business & technology conditions

1. Select highest priority items in Product Backlog; declare Sprint Goal

2. Team turns selected items into

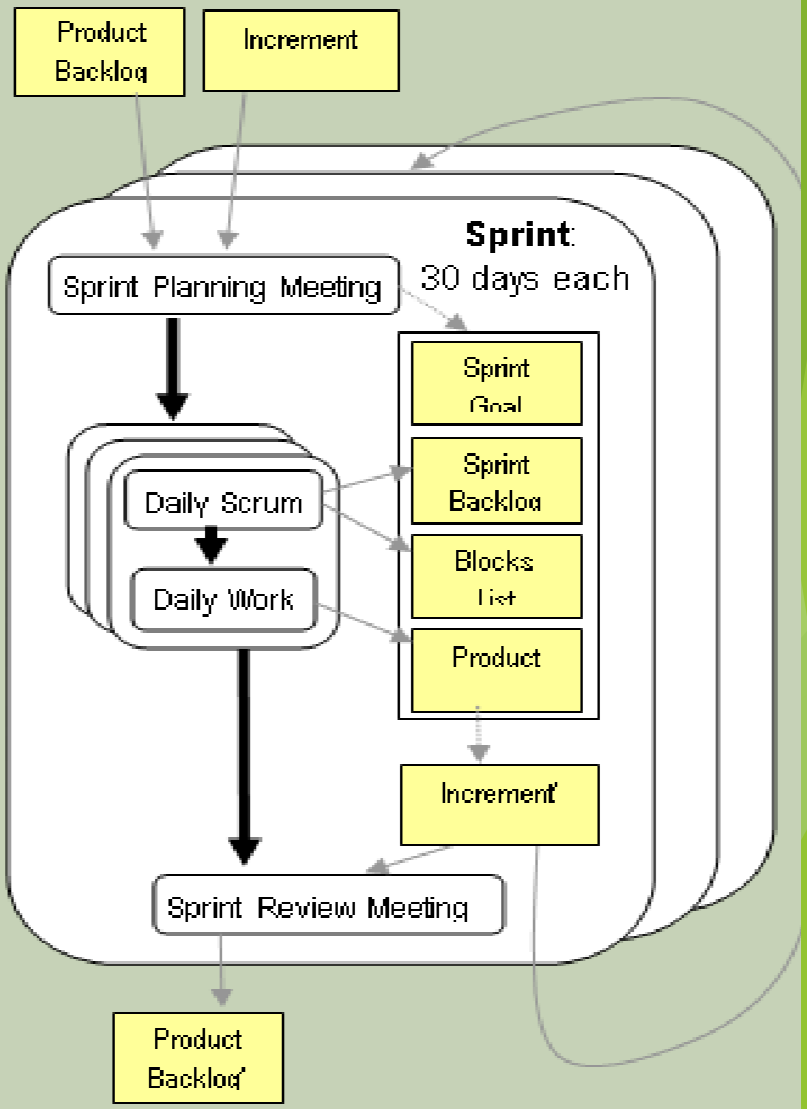
Daily Scrum

- Hosted by ScrumMaster
- Attended by all, but Stakeholders don't speak
- Same time every day
- Answer: 1) What did you do yesterday? 2) What will you do today? 3) What's in your way?
- Team updates Sprint Backlog;

Sprint Review Meeting

- Hosted by ScrumMaster
- Attended by all
- Informal, 4-hour, informational
- Team demos Increment
- All discuss
- Hold retrospective
- Announce next Sprint Planning

Development Process



Scrum Vs. Other Models

Process Comparison

	Waterfall	Spiral	Iterative	SCRUM
Defined processes	Required	Required	Required	Planning & Closure only
Final product	Determined during planning	Determined during planning	Set during project	Set during project
Project cost	Determined during planning	Partially variable	Set during project	Set during project
Completion date	Determined during planning	Partially variable	Set during project	Set during project
Responsiveness to environment	Planning only	Planning primarily	At end of each iteration	**Throughout**
Team flexibility, creativity	Limited - cookbook approach	Limited - cookbook approach	Limited - cookbook approach	**Unlimited during iterations**
Knowledge transfer	Training prior to project	Training prior to project	Training prior to project	**Teamwork during project**
Probability of success	Low	Medium Low	Medium	**High**

Pros & Cons

- Advantages
  - Completely developed and tested features in short iterations
  - Simplicity of the process
  - Clearly defined rules
  - Increasing productivity
  - Self-organizing
  - each team member carries a lot of responsibility
  - Improved communication
  - Combination with Extreme Programming
- Drawbacks
  - “Undisciplined hacking” (no written documentation)
  - Violation of responsibility
  - Current mainly carried by the inventors

Tools to Support the Agile Project Management

ATLASSIAN JIRA

- Atlassian JIRA is one of the best project management tools used by Agile teams.
- Ideal for most offices. Mainly- IT professionals, institutional designers, working in a shared environment.

Pros:

- It is highly extensible and extremely customizable as per your project needs.
- It is a mature and proven product used by thousands of big companies around the globe. So, it has a large and active community.
- Very helpful for start-ups as it is cheap for small teams.

Cons:

- It is hard to set up and takes a long time to learn using it properly.
- Too many features make it complex to use for some teams at times.

AGILO for Scrum

- Ideal for teams that require powerful communication.

Pros:

- Ideal for distributed teams
- Well priced
- Great communication system

Cons:

- No mobile app
- Cannot host more than one project
- Few people have found it difficult to learn

SPIRATEAM

- SpiraTeam® is a complete agile software development management system that manages your project's requirements, releases, iterations, tasks and bugs/issues.
- Designed specifically to support agile methodologies such as Scrum, Extreme Programming (XP), Dynamic System Development Method (DSDM) and Agile Unified Process (AUP) it allows teams to manage all their information in one environment.
- In SpiraTeam, each project has a dashboard home-page that summarizes all of the information regarding the project into a comprehensive, easily digestible form that provides a “one-stop-shop” for people interested in understanding the overall status of the project at a glance.
- It contains summary-level information for all types of artifact (requirements, test cases, incidents, etc.) that you can use to drill-down into the appropriate section of the application.

Scrum and Kanban Boards

Scrum

- Scrum is the most popular Agile development framework that has taken a highly iterative approach focusing on the key features and objectives. The best part about Scrum is that it is adaptable and relatively simple to implement.

Kanban

- Kanban was developed by Toyota to help increase productivity in factories. It is another framework for Agile based methodology. This approach gives a quick way to bring code to production. Teams can have a visual representation of their tasks and keep a transparency in the progress of their tasks by moving it through stages and identify where roadblocks occur.