LESSON 25

AGILE/SCRUM

KANBAN

- It visualizes the work of the development team (the features and user stories).
- It captures WIP limits for development steps: the circled values below the column headings that limit the number of work items under that step.
- It documents policies, also known as done rules, inside blue rectangles under some of the development steps.
- It also shows some Kanban flow management for the "User Story Preparation," "User Story Development," and "Feature Acceptance" steps, which have "In Progress" and "Ready" sub-columns. Each step's WIP limit applies to both sub-columns, preventing work items from overwhelming the flow into or out of those steps.

KANBAN BOARD

- To do
- Work in progress
- Done

AGILE SCRUM

- Agile - mindset
- Scrum - agile framework

AGILE VALUES

- Individuals and Interactions Over Processes and Tools
- Working Software Over Comprehensive Documentation
- Customer Collaboration Over Contract Negotiation
- Responding to Change Over Following a Plan

AGILE VALUES

- Individuals and Interactions Over Processes and Tools
  The first value in the Agile Manifesto is "Individuals and interactions over processes and tools." Valuing people more highly than processes or tools is easy to understand because it is the people who respond to business needs and drive the development process. If the process or the tools drive development, the team is less responsive to change and less likely to meet customer needs. Communication is an example of the difference between valuing individuals versus process. In the case of individuals, communication is fluid and happens when a need arises. In the case of process, communication is scheduled and requires specific content.

- Working Software Over Comprehensive Documentation
  Historically, enormous amounts of time were spent on documenting the product for development and ultimate delivery. Technical specifications, technical requirements, technical prospectus, interface design documents, test plans, documentation plans, and approvals required for each. The list was extensive and was a cause for the long delays in development. Agile does not eliminate documentation, but it streamlines it in a form that gives the developer what is needed to do the work without getting bogged down in minutiae. Agile documents requirements as user stories, which are sufficient for a software developer to begin the task of building a new function.
  The Agile Manifesto values documentation, but it values working software more.

- Customer Collaboration Over Contract Negotiation
  Negotiation is the period when the customer and the product manager work out the details of a delivery, with points along the way where the details may be renegotiated. Collaboration is a different creature entirely. With development models such as Waterfall, customers negotiate the requirements for the product, often in great detail, prior to any work starting. This meant the customer was involved in the process of development before development began and after it was completed, but not during the process. The Agile Manifesto describes a customer who is engaged and collaborates throughout the development process, making. This makes it far easier for development to meet their needs of the customer. Agile methods may include the customer at intervals for periodic demos, but a project could just as easily have an end-user as a daily part of the team and attending all meetings, ensuring the product meets the business needs of the customer.

- Responding to Change Over Following a Plan
  Traditional software development regarded change as an expense, so it was to be avoided. The intention was to develop detailed, elaborate plans, with a defined set of features and with everything, generally, having as high a priority as everything else, and with a large number of many dependencies on delivering in a certain order so that the team can work on the next piece of the puzzle.

AGILE MANIFESTO PRINCIPLES

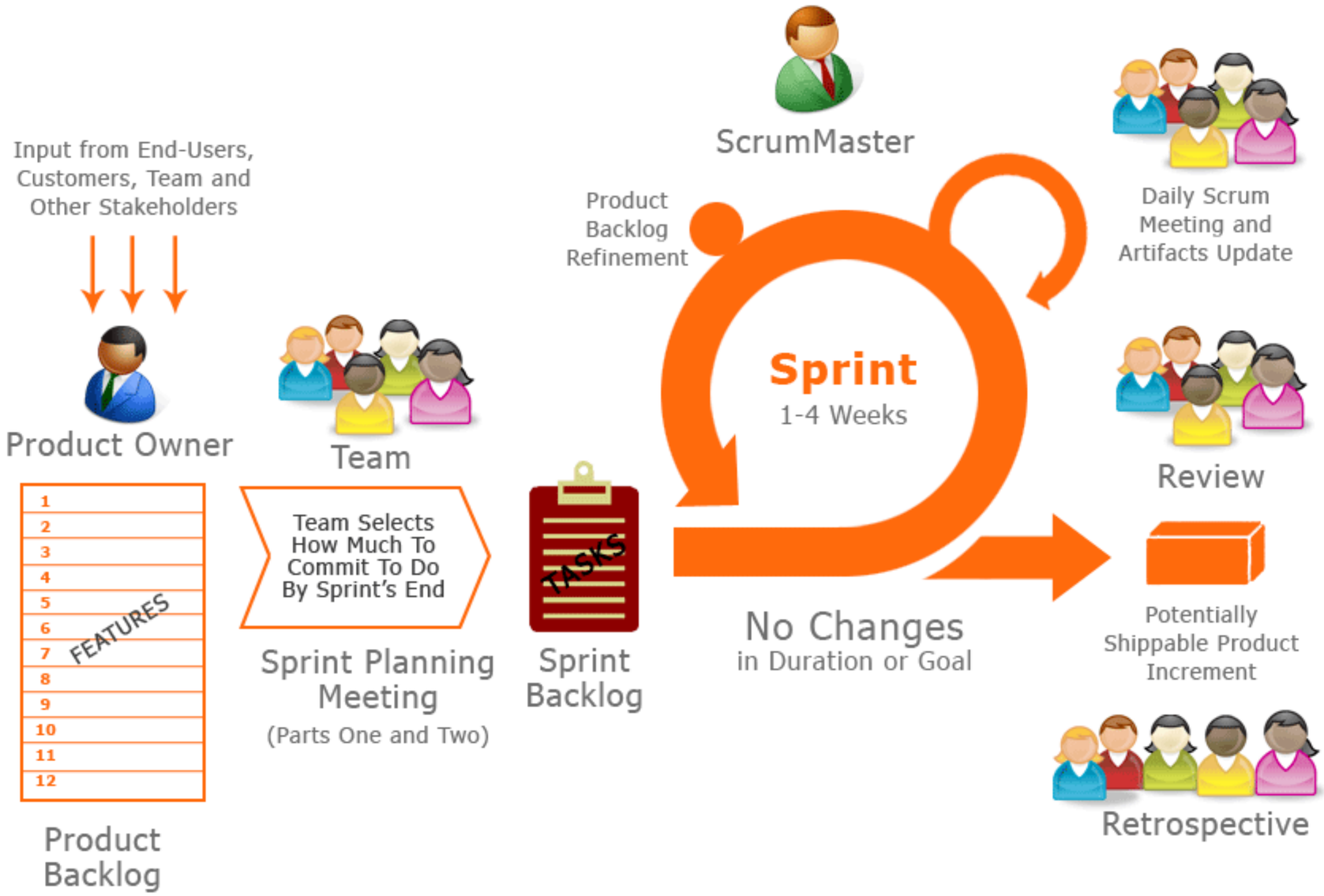
- Customer satisfaction through early and continuous software delivery – Customers are happier when they receive working software at regular intervals, rather than waiting extended periods of time between releases.
- Accommodate changing requirements throughout the development process – The ability to avoid delays when a requirement or feature request changes.
- Frequent delivery of working software – Scrum accommodates this principle since the team operates in software sprints or iterations that ensure regular delivery of working software.
- Collaboration between the business stakeholders and developers throughout the project – Better decisions are made when the business and technical team are aligned.
- Support, trust, and motivate the people involved – Motivated teams are more likely to deliver their best work than unhappy teams.
- Enable face-to-face interactions – Communication is more successful when development teams are co-located.
- Working software is the primary measure of progress – Delivering functional software to the customer is the ultimate factor that measures progress.
- Agile processes to support a consistent development pace – Teams establish a repeatable and maintainable speed at which they can deliver working software, and they repeat it with each release.
- Attention to technical detail and design enhances agility – The right skills and good design ensures the team can maintain the pace, constantly improve the product, and sustain change.
- Simplicity – Develop just enough to get the job done for right now.
- Self-organizing teams encourage great architectures, requirements, and designs – Skilled and motivated team members who have decision-making power, take ownership, communicate regularly with other team members, and share ideas that deliver quality products.
- Regular reflections on how to become more effective – Self-improvement, process improvement, advancing skills, and techniques help team members work more efficiently.

SCRUM

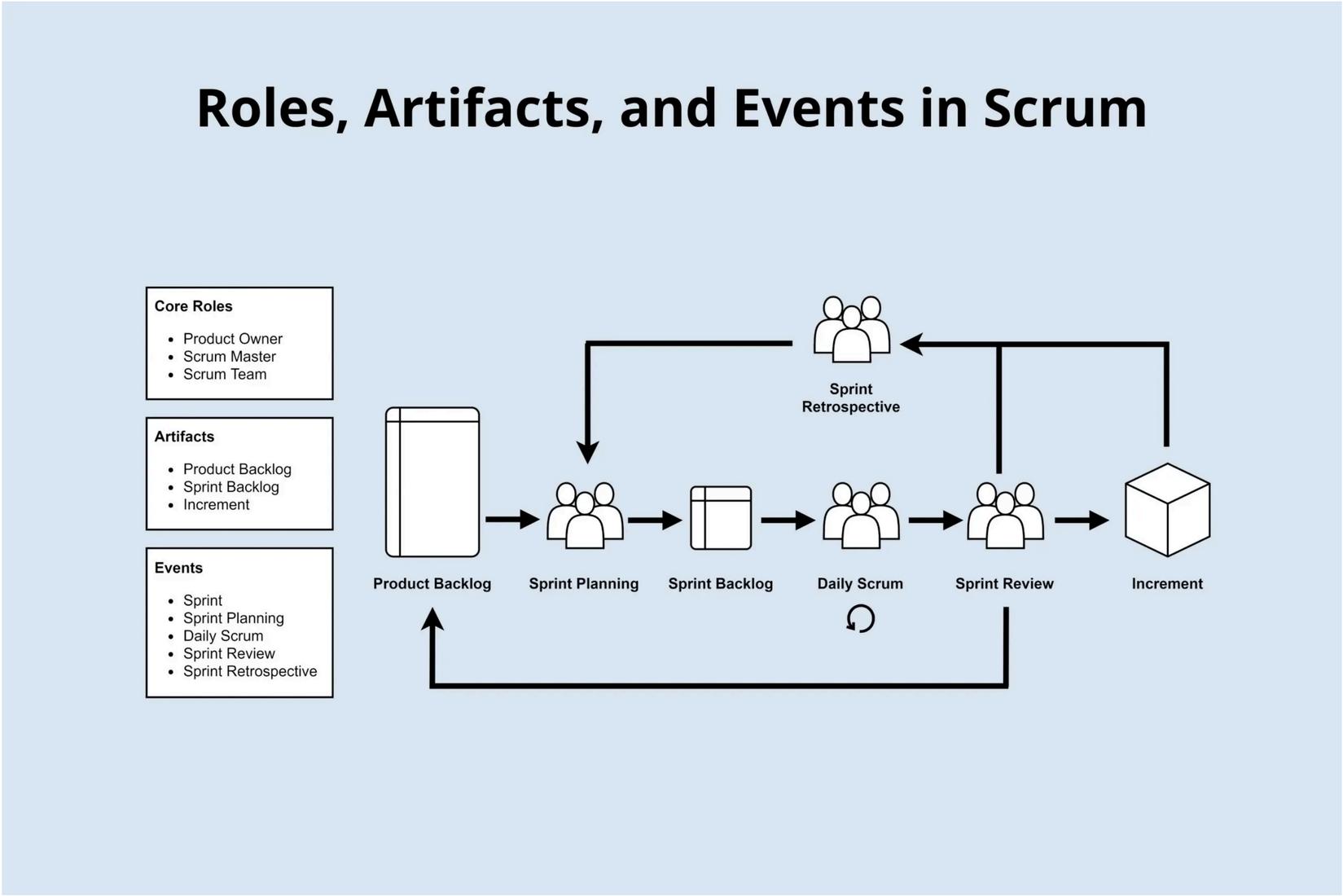
- **Scrum** is a **framework** that helps teams work together. ... Often thought of as an agile project management **framework**, **Scrum** describes a set of meetings, tools, and roles that work in concert to help teams structure and manage their work.

- Lightweight
- Simple to understand
- Difficult to master

IDEA


Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
Team
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Product Backlog
Team Selects How Much To Commit To Do By Sprint's End
Sprint Planning Meeting
(Parts One and Two)
TASKS
Sprint Backlog
ScrumMaster
Product Backlog Refinement
Sprint
1-4 Weeks
No Changes
in Duration or Goal
Daily Scrum Meeting and Artifacts Update
Review
Potentially Shippable Product Increment
Retrospective

SCEMA


Roles, Artifacts, and Events in Scrum
Core Roles
Product Owner
Scrum Master
Scrum Team
Artifacts
Product Backlog
Sprint Backlog
Increment
Events
Sprint
Sprint Planning
Daily Scrum
Sprint Review
Sprint Retrospective
Sprint Retrospective
Product Backlog
Sprint Planning
Sprint Backlog
Daily Scrum
Sprint Review
Increment

USER STORY

- In software development and product management, a **user story** is an informal, natural language description of one or more features of a software system. User stories are often written from the perspective of an end user or user of a system. They are often recorded on index cards, on Post-it notes, or digitally in project management software Depending on the project, user stories may be written by various stakeholders including clients, users, managers, or development team members.
- User stories are a type of boundary object. They facilitate sensemaking and communication; that is, they help software teams organize their understanding of the system and its context.

ESTIMATION

- Person-hours
- Story Points (sp)
- L, M, S

THERMS

- Spike
- Tech debt
- Definition of done (acceptance criteria)
- Velocity
- Capacity
- Burndown chart
- KT knowledge transfer

SCRUM

Pros

- **1. It creates a system of transparency.**
  Daily scrums do more than keep workers accountable to their assignments. It is also a way for a company to maintain their transparency with their clients.
- **2. It offers motivation on multiple levels.**
  Teams are motivated using this methodology because there are defined deadlines and expectations to meet. Individuals are motivated by the rewards that are offered (or at least should be offered) for meeting or exceeding expectations. This system creates a stronger set of knowledge work that can be presented to the client.
- **3. It provides continuous feedback.**
  Because this methodology requires daily check-ins for progress reports, there is always feedback offered at the team and individual level. This helps to make the project better in the long run.

Cons

- **1. It does not care about the final project deadline.**
  The scrum methodology uses personal deadlines to create a specific amount of work. It does not take the project deadline into account. The only real requirement is that each person or team meet expectations.
- **2. It requires a team environment.**
  Individuals can follow the concepts of scrum methodology, but this format is designed to work with a team of at least 3 people. It's only suitable for small teams as well. If there are 10+ people involved, it doesn't work as well.
- **3. It requires experience.**
  Feedback can be provided to teams and individuals through relevant experience only. If the individual or team that offers feedback is not experienced in the work being done, then the entire system breaks down.

LINKS

- https://www.scrum.org/resources/what-is-scrum
- https://agilemanifesto.org/