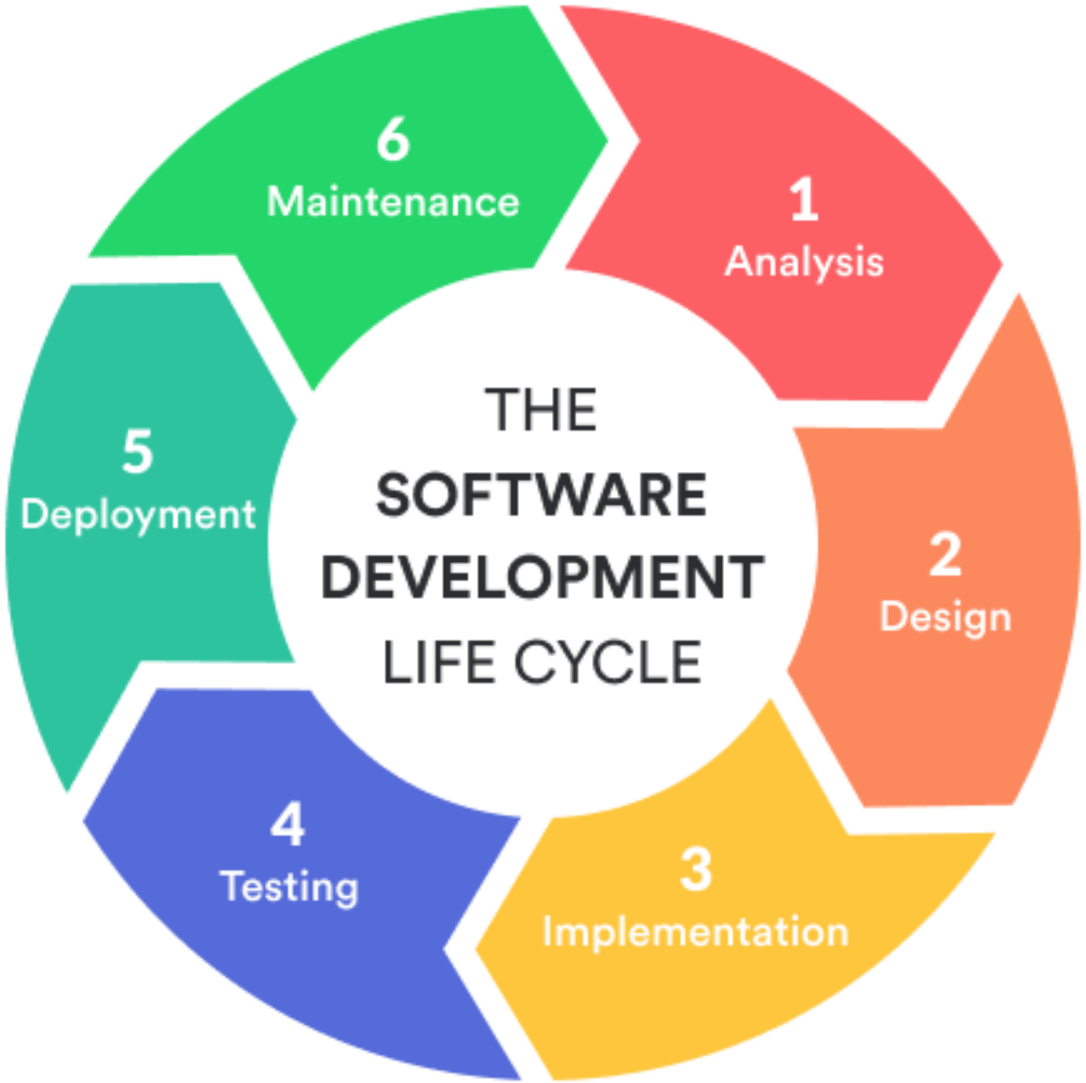
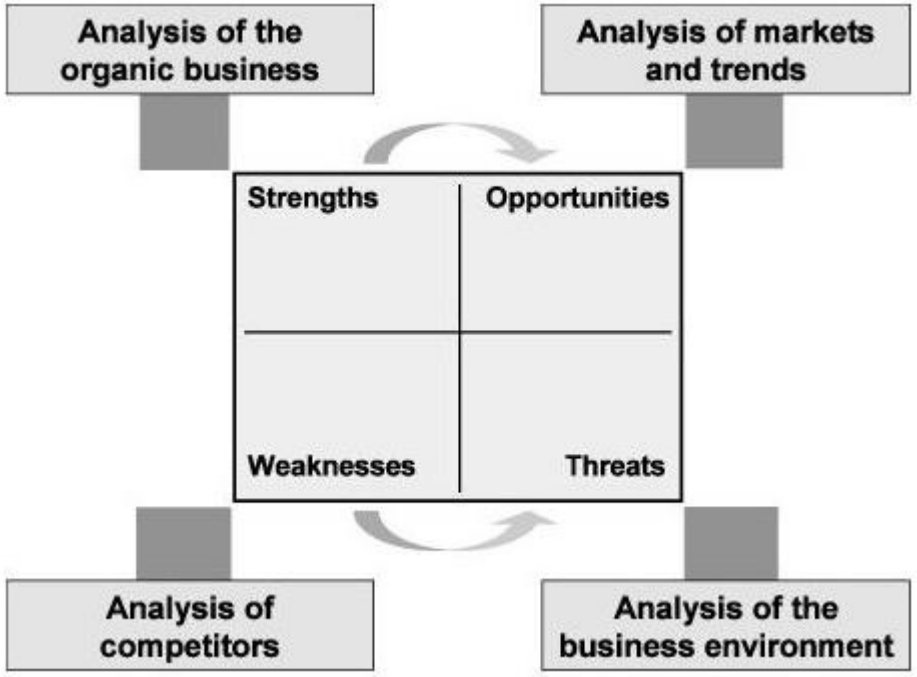

AGILE


6
Maintenance
1
Analysis
5
Deployment
THE
SOFTWARE
DEVELOPMENT
LIFE CYCLE
2
Design
4
Testing
3
Implementation

Planning and Requirement Analysis

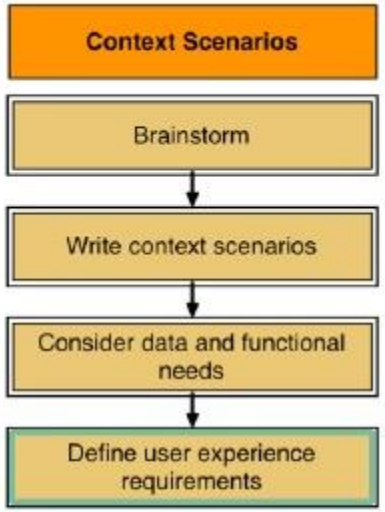


Requirement analysis is the most important and fundamental stage in **SDLC**. It is performed by the senior members of the team with inputs from the **customer**, the **sales department**, **market surveys** and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

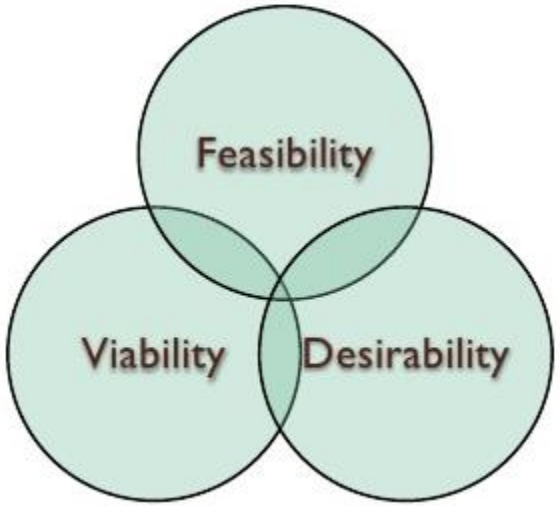
Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an **SRS** (**Software Requirement Specification**) document which consists of all the product requirements to be designed and developed during the project life cycle.

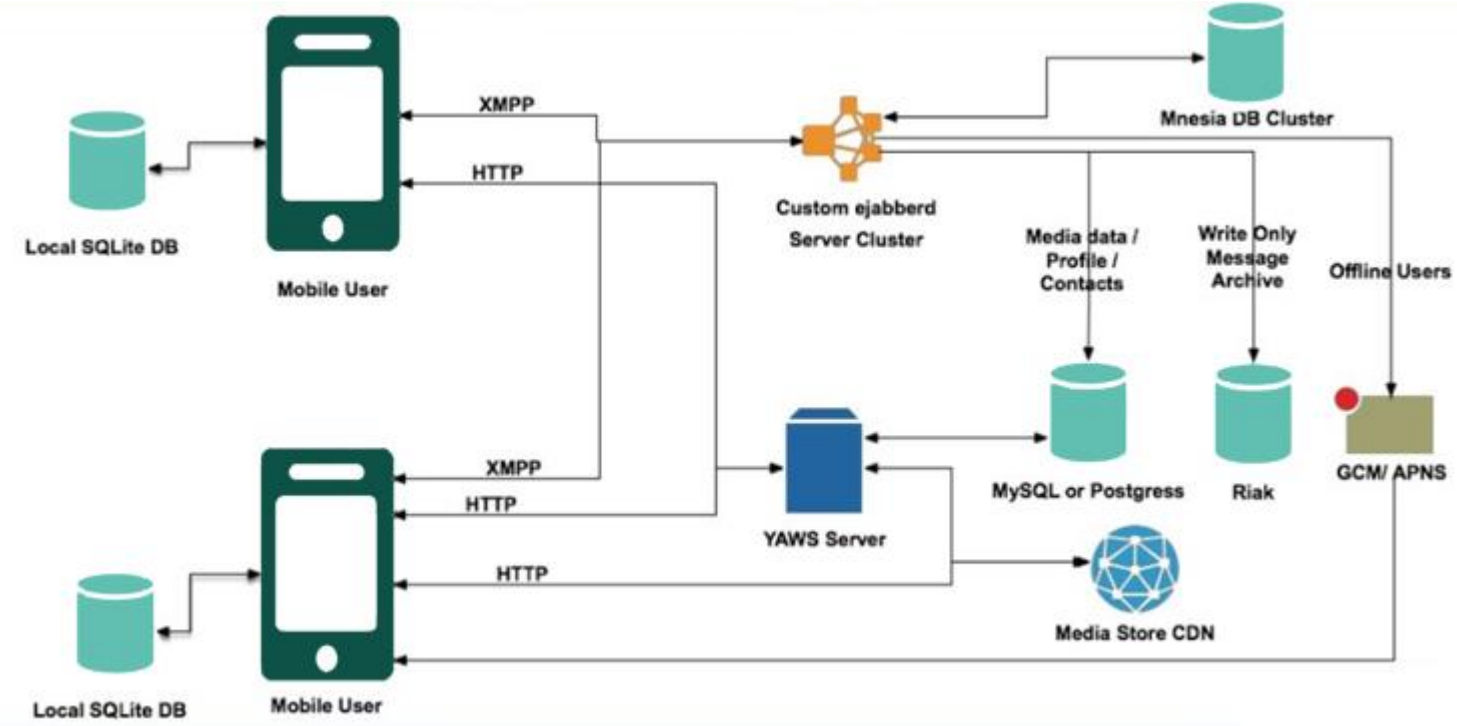


Activity:

Define a coherent user experience, where user and business needs converge as innovative solutions are defined.



Designing the Product Architecture



SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a **DDS - Design Document Specification.**

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

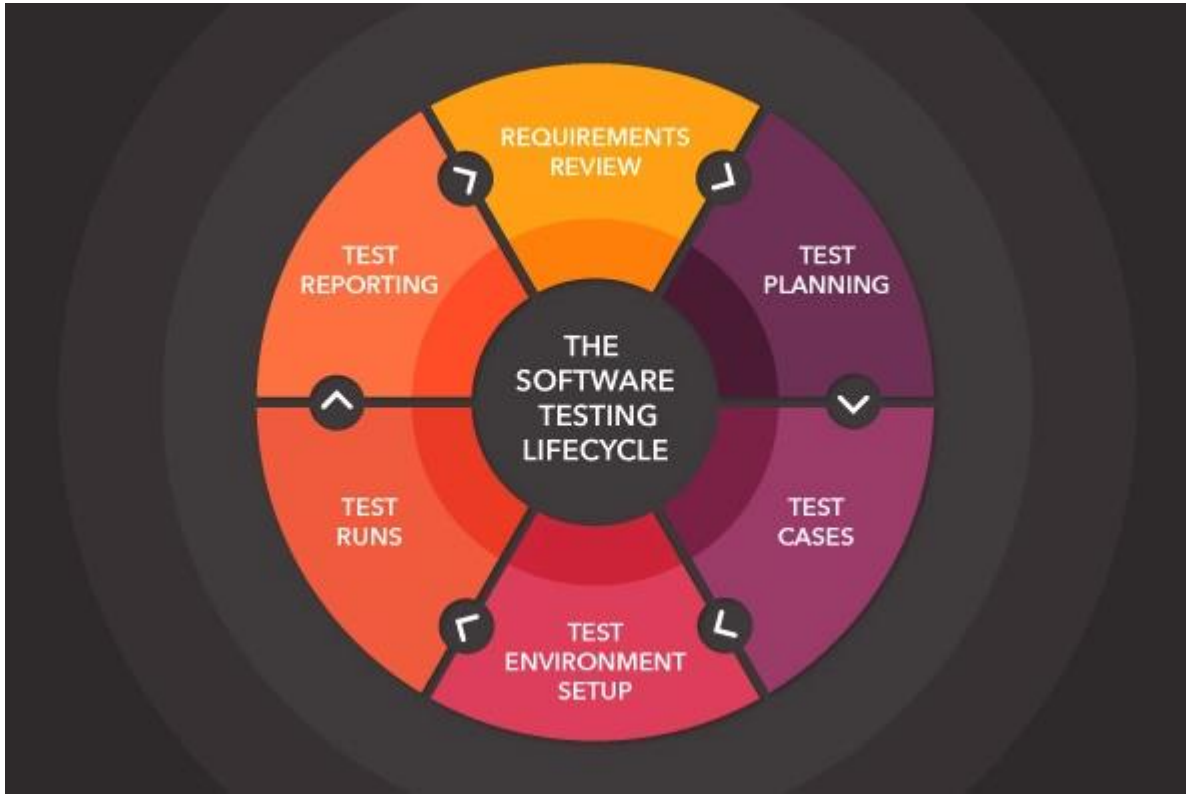
A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per **DDS during this stage**. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Testing the Product

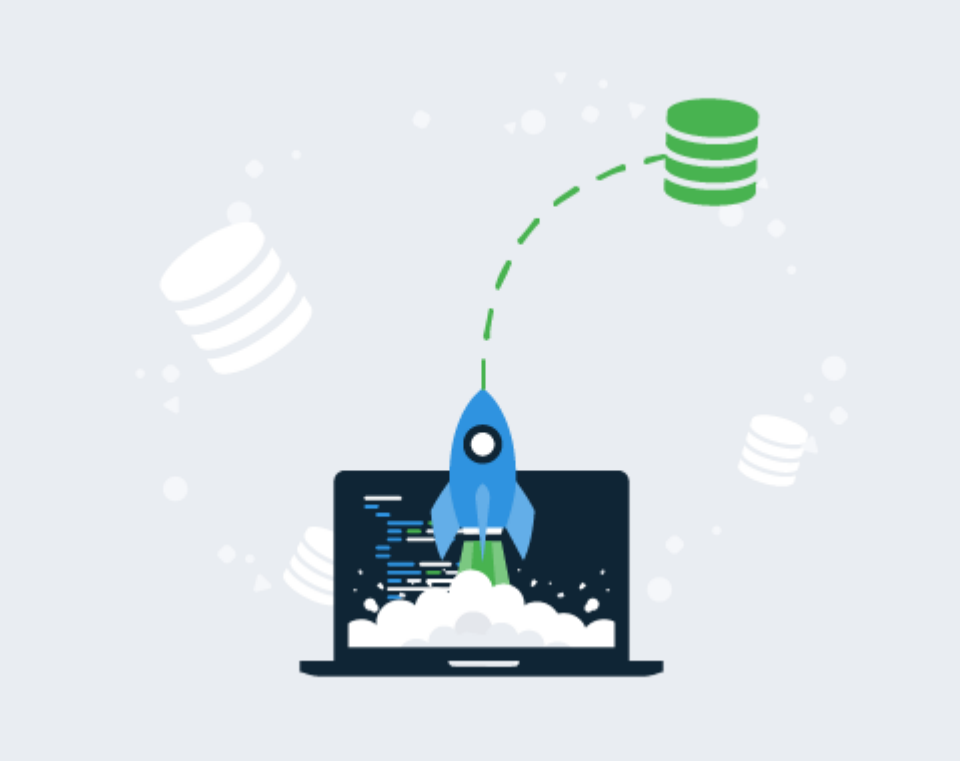


This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

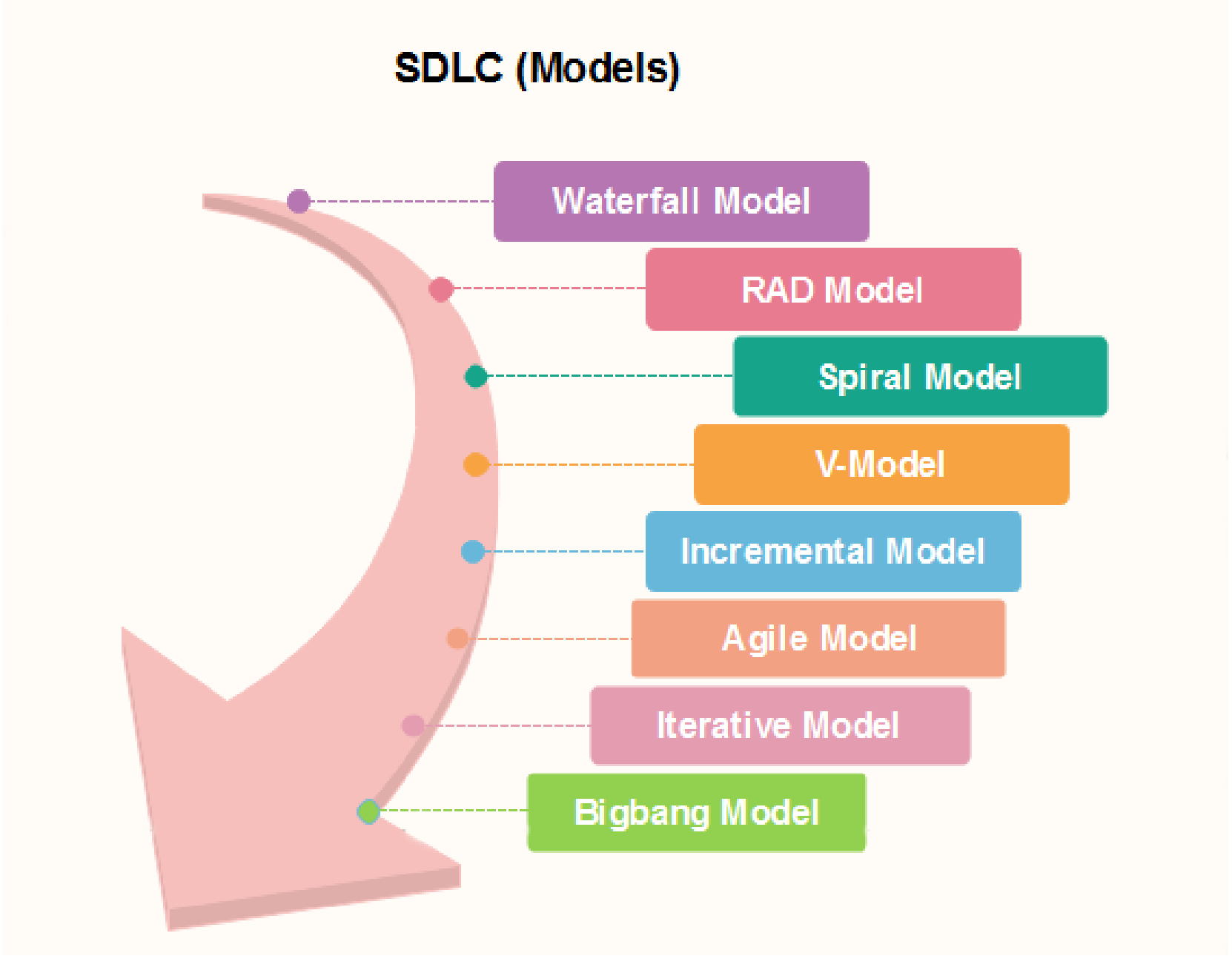
Deployment in the Market and Maintenance

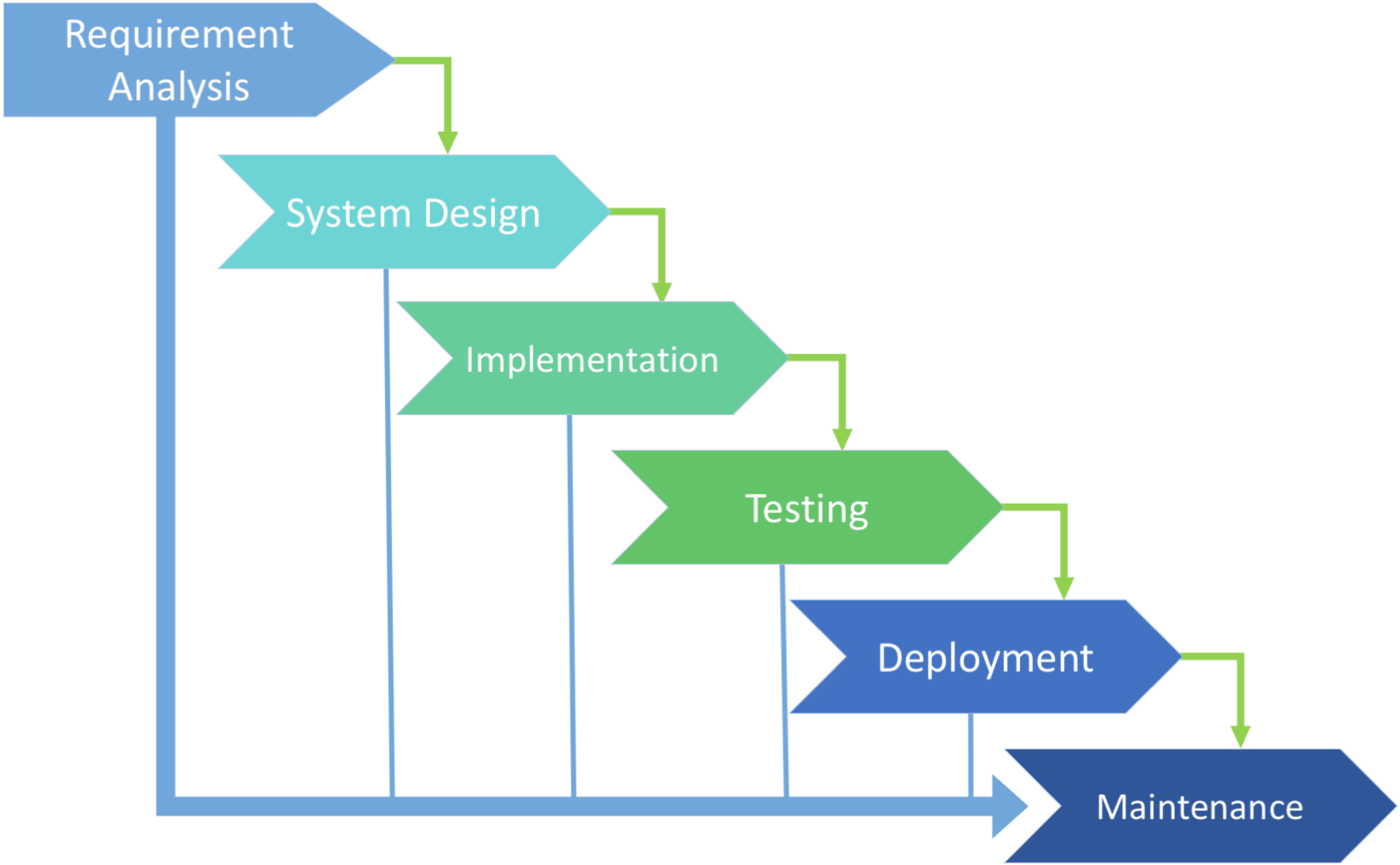
Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

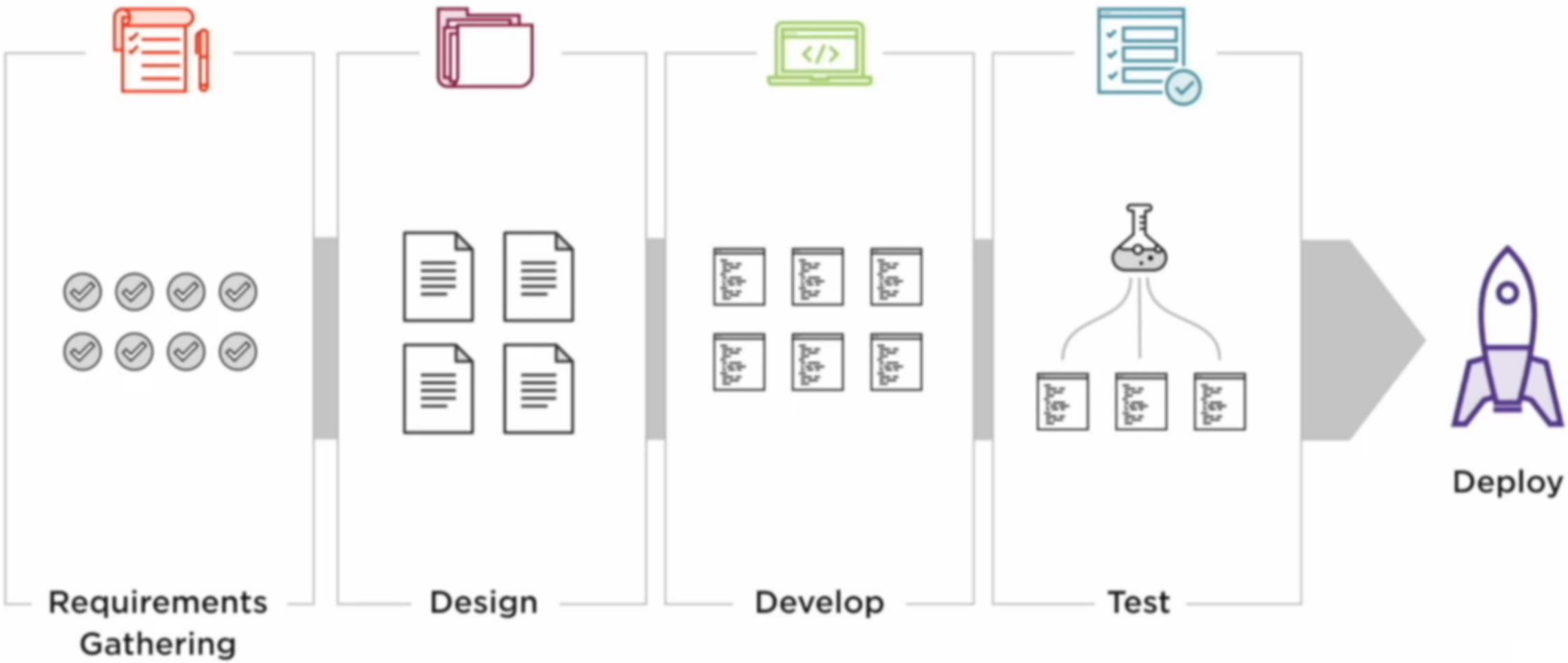
Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

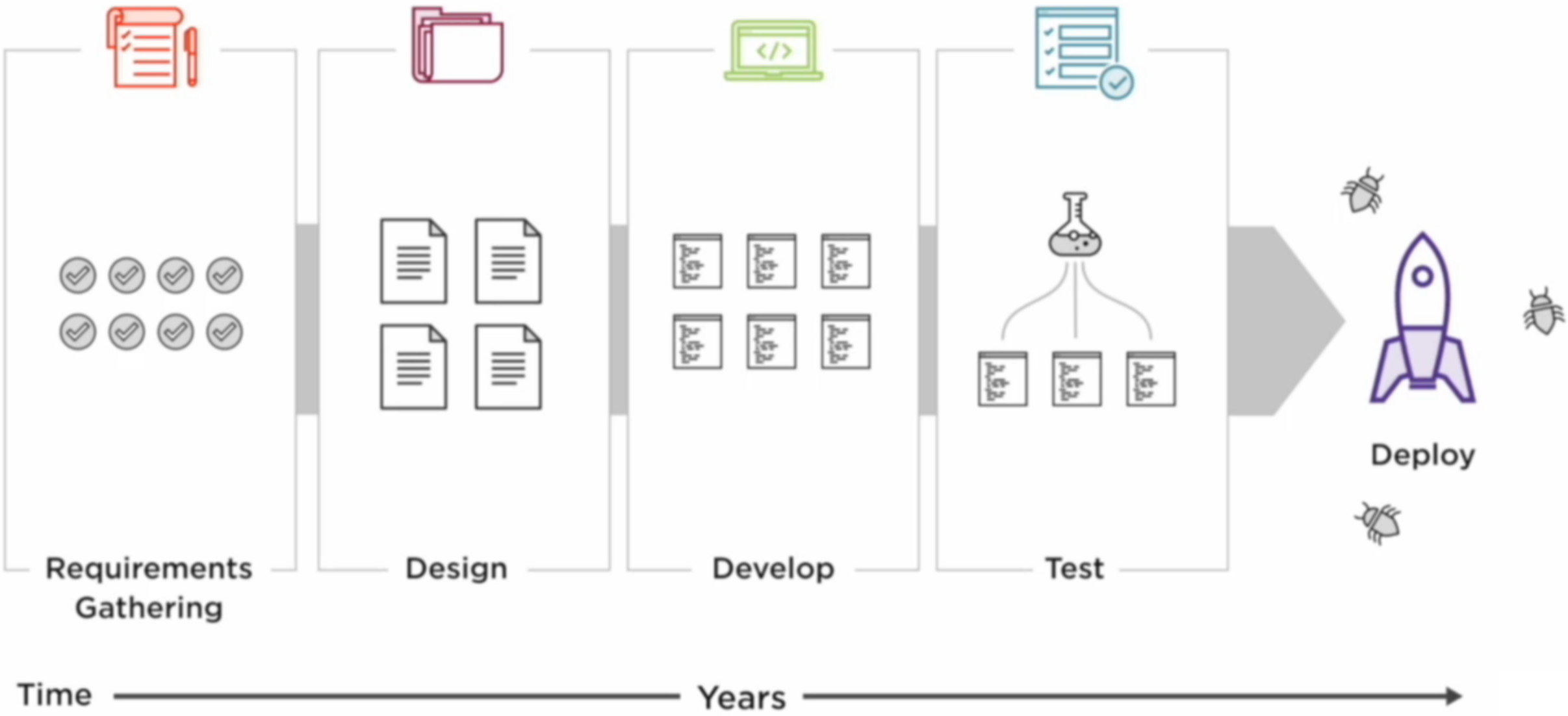


SDLC (Models)

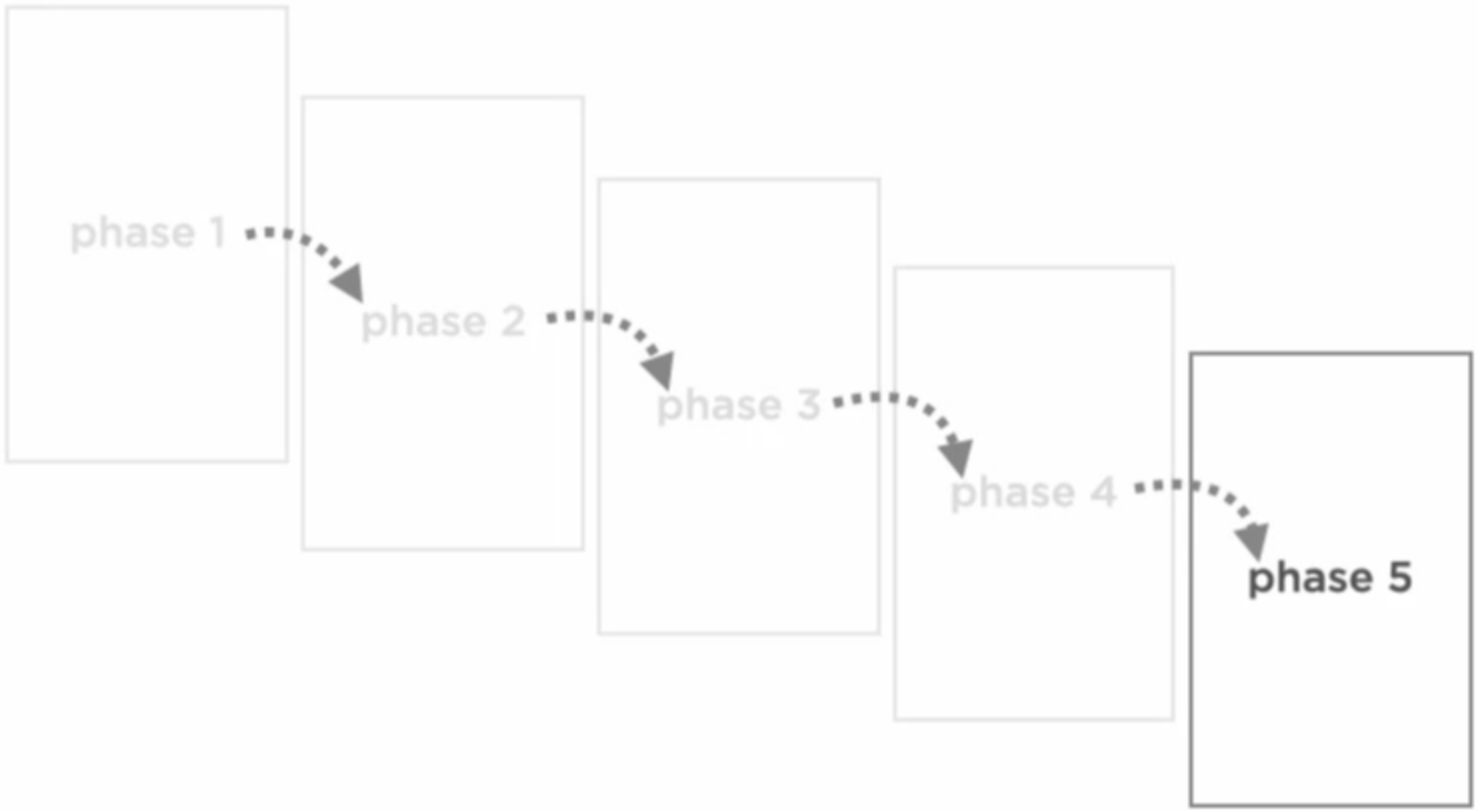

Waterfall Model
RAD Model
Spiral Model
V-Model
Incremental Model
Agile Model
Iterative Model
Bigbang Model

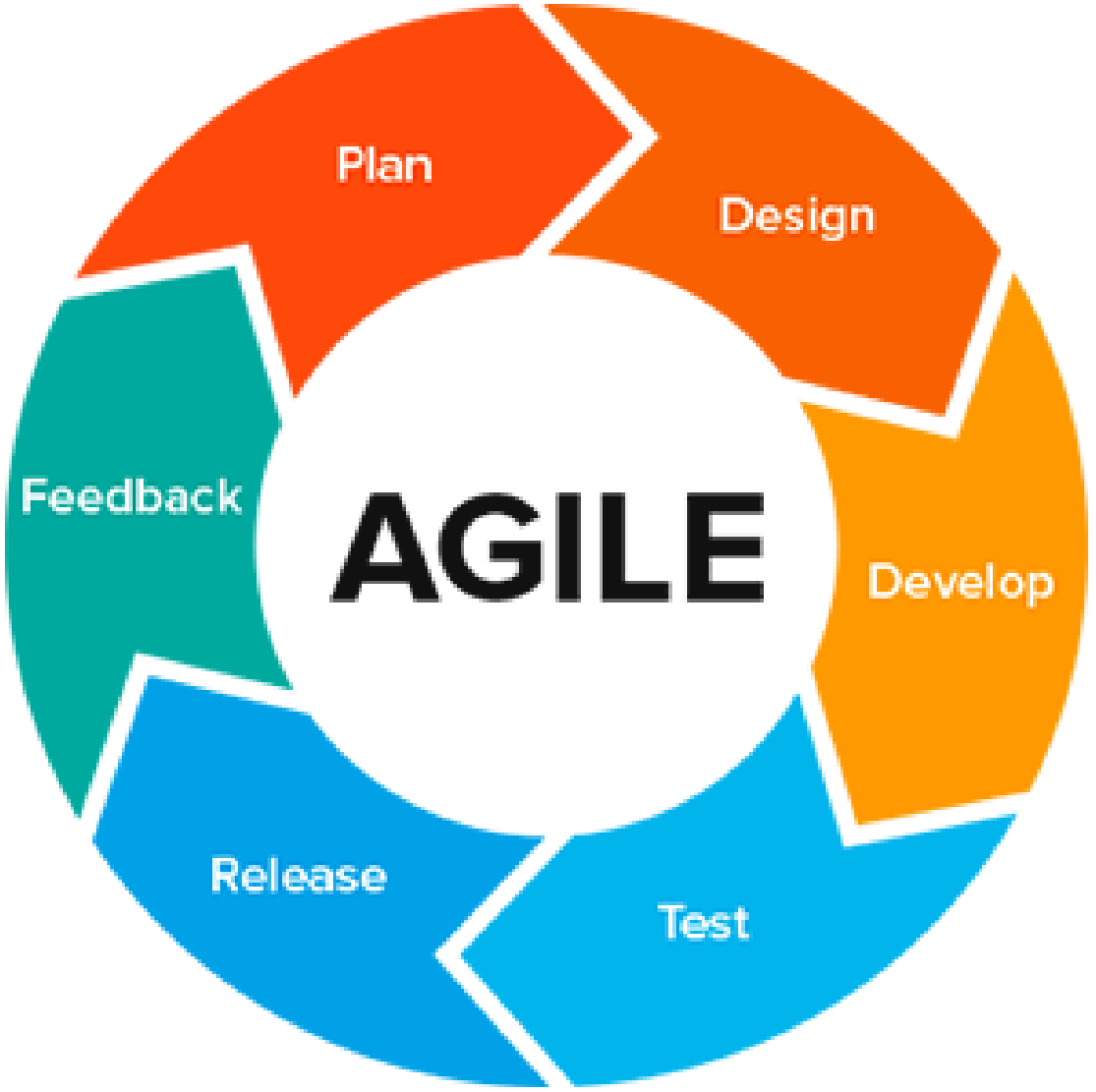

Requirement Analysis
System Design
Implementation
Testing
Deployment
Maintenance


Requirements Gathering
Design
Develop
Test
Deploy


Requirements Gathering
Design
Develop
Test
Deploy
Time
Years

Waterfall Model / Waterfall Methodology




Plan
Design
Develop
Test
Release
Feedback
AGILE

Agile Development

“Agile software development is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams”

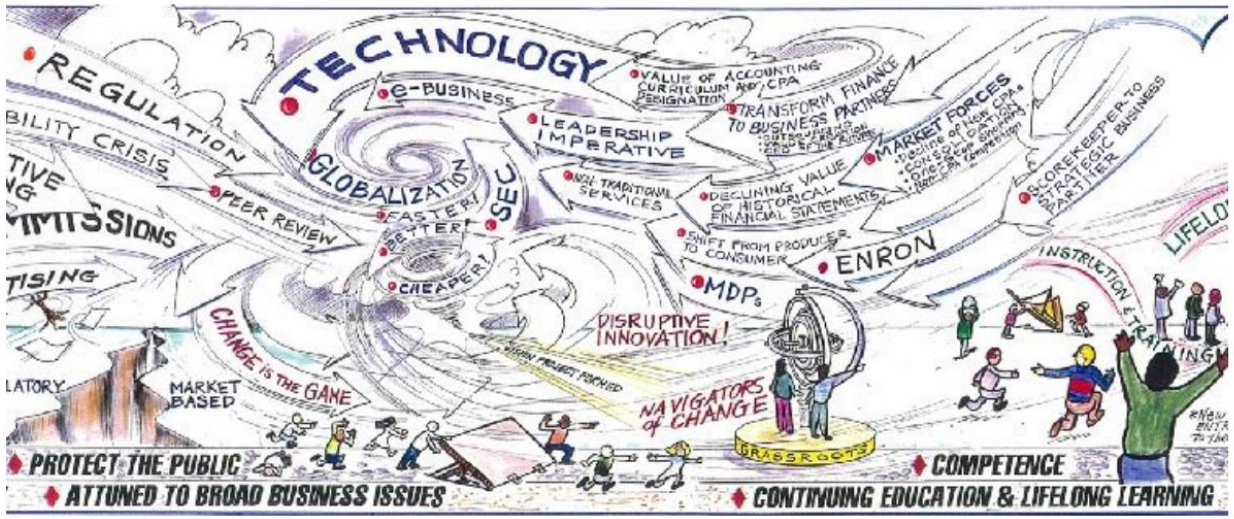
Why Agile?

Volatile

Uncertain

Complex

Ambiguos



QUALITY GUARANTEED?

Agile in a mindset or a way of thinking that guided by 4 values and 12 principles

www.agilemanifesto.org

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over **processes and tools**

Working software over **comprehensive documentation**

Customer collaboration over **contract negotiation**

Responding to change over **following a plan**

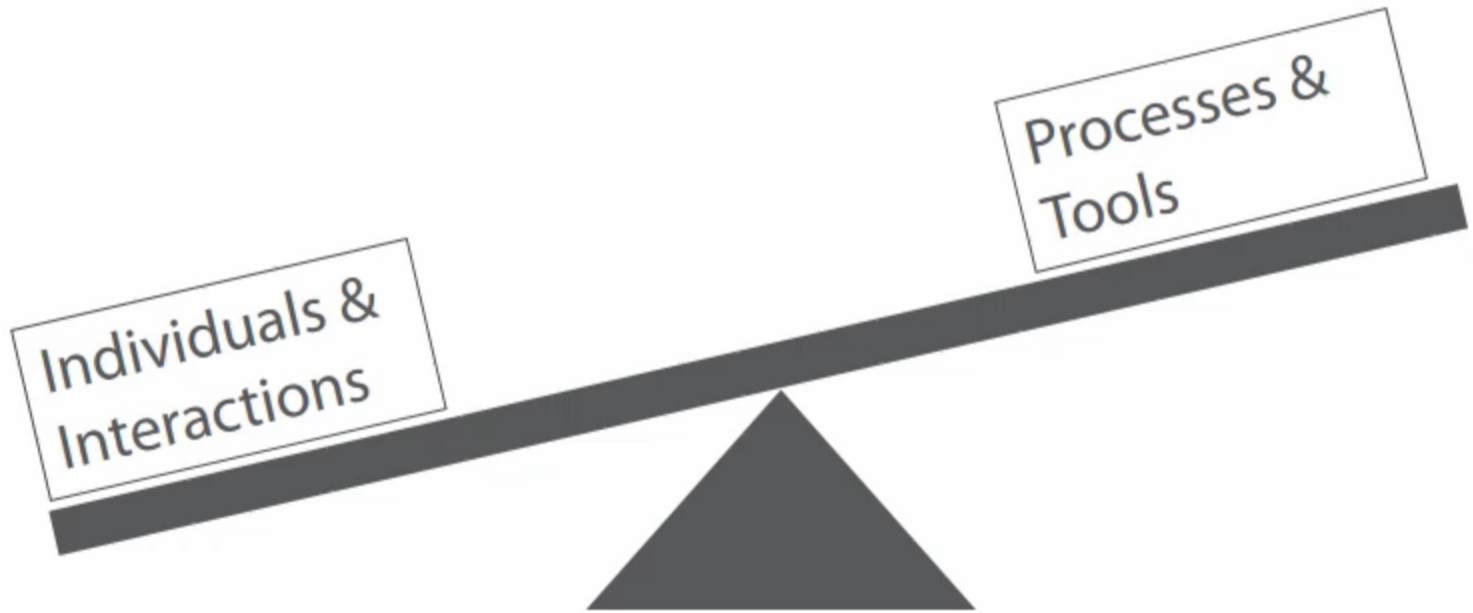
That is, while there is value in the items on the right, we value the items on the left more.

The Agile Manifesto 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

The Agile Manifesto Values

These statements are meant to be relative, not absolute. For example:



The Agile Manifesto Values

“Individuals And Interactions Over Processes And Tools”

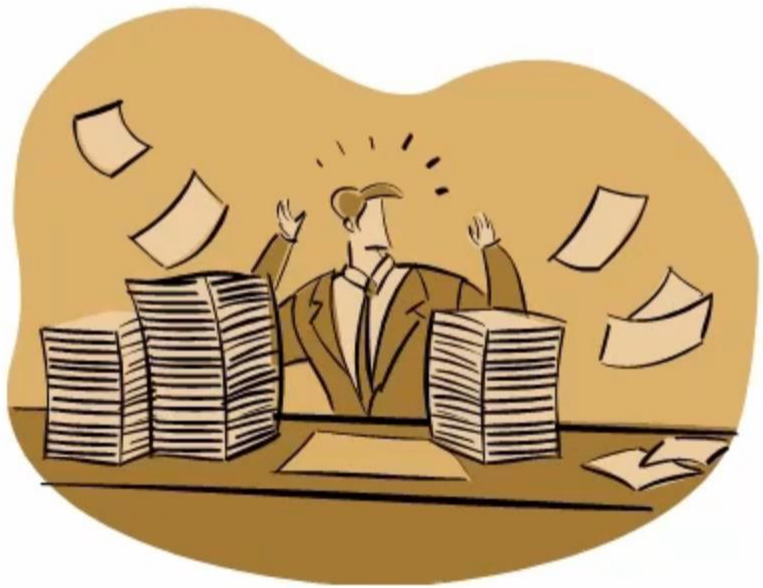
Does not mean that there are no processes and tools in an agile project

versus

The Agile Manifesto Values

“Working Software Over Comprehensive Documentation”

Does not mean that there is no documentation at all in an agile project



versus

The Agile Manifesto Values

"Customer Collaboration Over Contract Negotiation"

Does not mean that contracts are incompatible with an agile approach

versus

The Agile Manifesto Values

“Responding To Change Over Following A Plan”

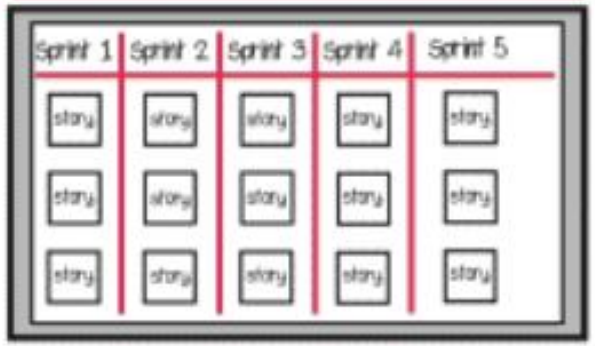
Does not mean that agile projects are totally unplanned

versus

1 Satisfy the **customer**

Welcome **change**

Deliver **frequently**



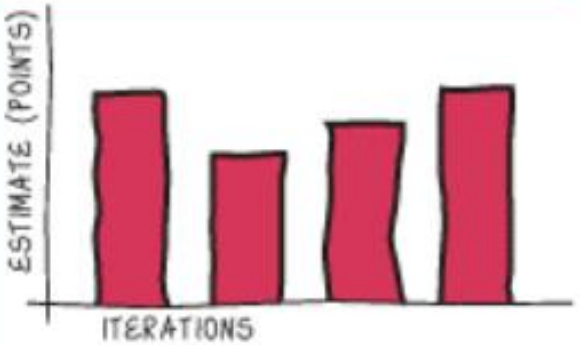
4 Work **together**

5 Trust and **support**

Face-to-face **conversation**

Working **software**

8 Sustainable **development**



9 Continuous **attention**

10 Maintain **simplicity**

11 Self-organizing **teams**

12 Reflect and **adjust**

WHAT IS AN AGILE ENVIRONMENT?

1: FOCUSED ALIGNMENT

Aligning our vision, values, and strategy, while discovering better ways to organize ourselves to focus on executing against measurable and impactful outcomes.

2: INCLUSIVE COLLABORATION

Striving for an environment of high trust, transparency, and safety, while bringing the energy of our best selves towards collective success through fun, collaboration, and shared responsibility.

3: CUSTOMER-CENTRICITY

Empathizing with our customers and stakeholders to solve their immediate problems by seeking to deliver quality and timely solutions.

4: GROWTH THROUGH LEARNING

Investing the time for personal and professional growth, experimentation towards innovative ideas, and solving our most significant challenges together with the help of teams and leaders.

5: RESILIENT TO CHANGE

Changing and evolving who we are as an organization by quickly sensing, responding, and adapting to any threats, risks, opportunities, or needs that will benefit our customers, stakeholders, and business.

Agile Methodologies

Kanban

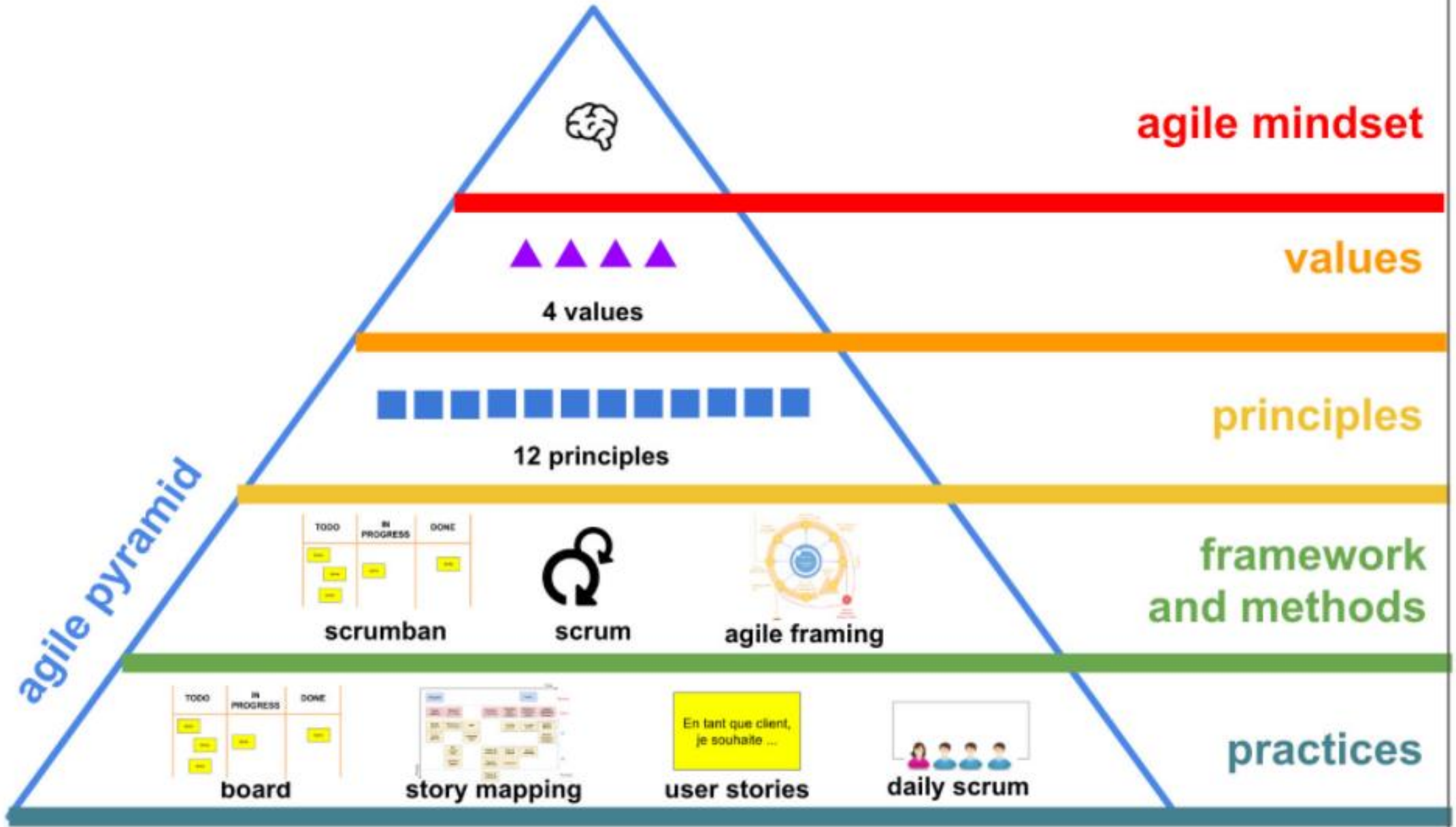
Acceptance Test Driven Development

Agile Modeling

Test Driven Development

Velocity Tracking

	Scrum	Kanban
Origin	Software development	Lean manufacturing
Idealogy	Learn through experiences, self-organize and prioritize, and reflect on wins and losses to continuously improve.	Use visuals to improve work-in-progress
Cadence	Regular, fixed-length sprints (i.e. two weeks)	Continuous flow
Practices	Sprint planning, sprint, daily scrum, sprint review, sprint retrospective	Visualize the flow of work, limit work-in-progress, manage flow, incorporate feedback loops
Roles	Product owner, scrum master, development team	No required roles


agile pyramid
agile mindset
values
4 values
principles
12 principles
framework
and methods
TODO
IN PROGRESS
DONE
scrumban
scrum
agile framing
practices
TODO
IN PROGRESS
DONE
board
story mapping
En tant que client, je souhaite ...
user stories
daily scrum

Benefits of a More Agile Approach

Adaptability

Time-to-market

Reduced costs

Customer satisfaction

Organizational agility

Adaptability

An alternative way for companies to manage projects

- Using a "one-size fits all" approach is *not* likely to achieve optimum results

- Agile offers an alternative approach that is particularly well-suited for projects that have high levels of uncertainty

Time-to-market

The potential to significantly accelerate delivery of valuable functionality

- An Agile approach isn't necessarily always the fastest time-to-market, but that is generally true
- There are many ways to use an Agile approach to accelerate time-to-market

Reduced Costs

Opportunities to reduce the costs and overhead associated with projects

- There is a lot of opportunity to reduce the costs of documentation and other overhead in a project

- Use Lean thinking and value-stream analysis to identify sources of "waste" to reduce costs

Customer Satisfaction

Produce higher value solutions that are more well-aligned with user needs

- Engaging customers more directly in the project to provide feedback and inputs as the project progresses rather than:

- Relying heavily on documenting requirements upfront prior to the start of the project

Organizational Agility

Build much higher levels of collaboration, trust, and shared responsibility

- Break down barriers between organizations
- Develop a spirit of shared ownership and responsibility with the business users

Why Is Systems Thinking Important?

Defined Process

Traditional Plan-driven Project

Empirical Process

Agile Project

Scrum Overview

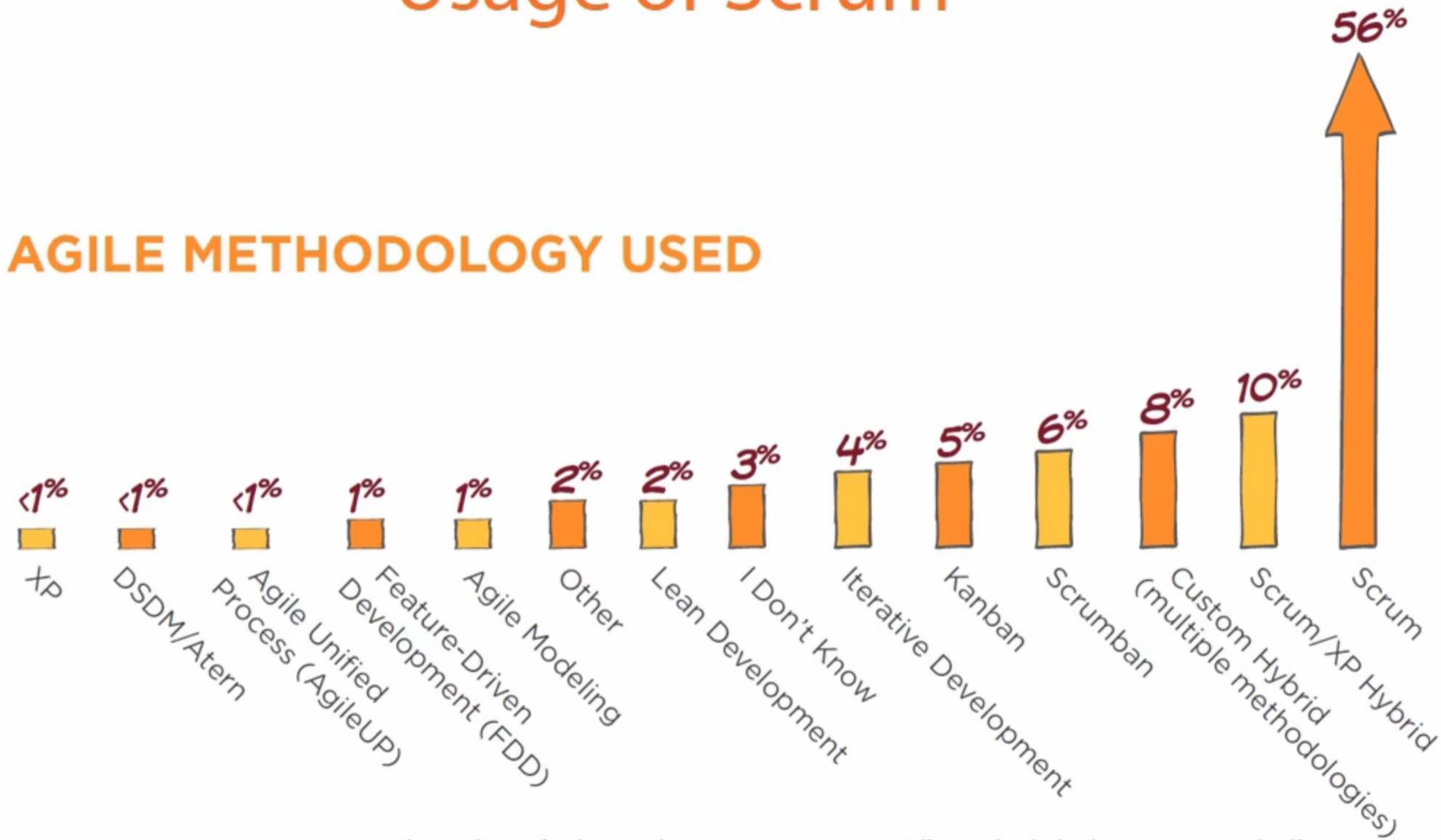
1986
Takeuchi and Nonaka
Coined in The New Product Development Game

1993
Jeff Sutherland
First Project Easel Corp

1995
Ken Schwaber and Jeff Sutherland
Formalized First Project Easel Corp

2002
Scrum Alliance formed

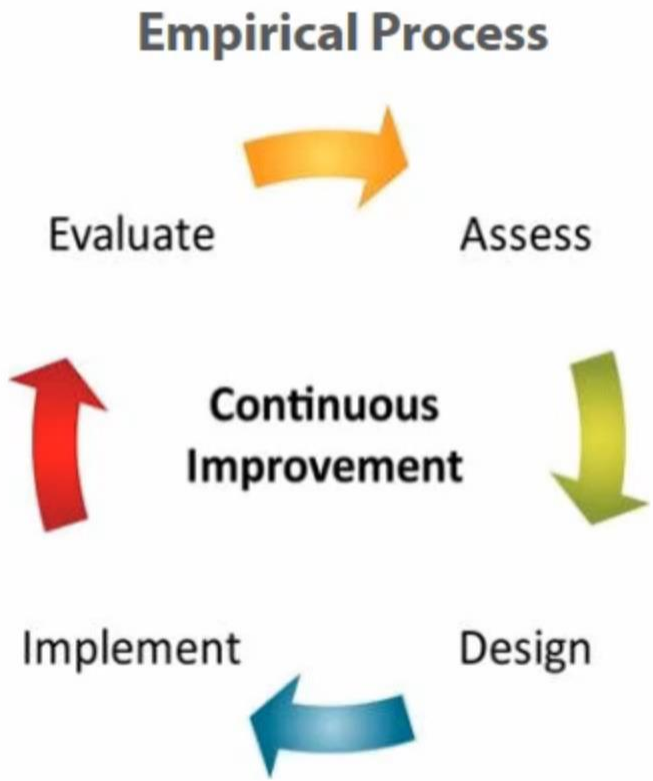
Usage of Scrum



VersionOne, Ninth Annual State of Agile Survey, https://www.versionone.com/pdf/state-of-agile-development-survey-ninth.pdf

Scrum is an Example of an Empirical Process

- The process is adaptive
- Changing the requirements is encouraged as the project progresses

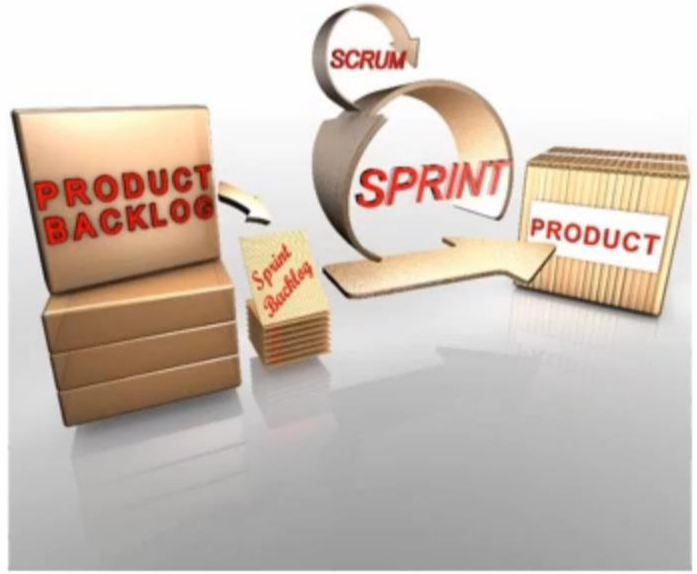


Adaptivity in Scrum

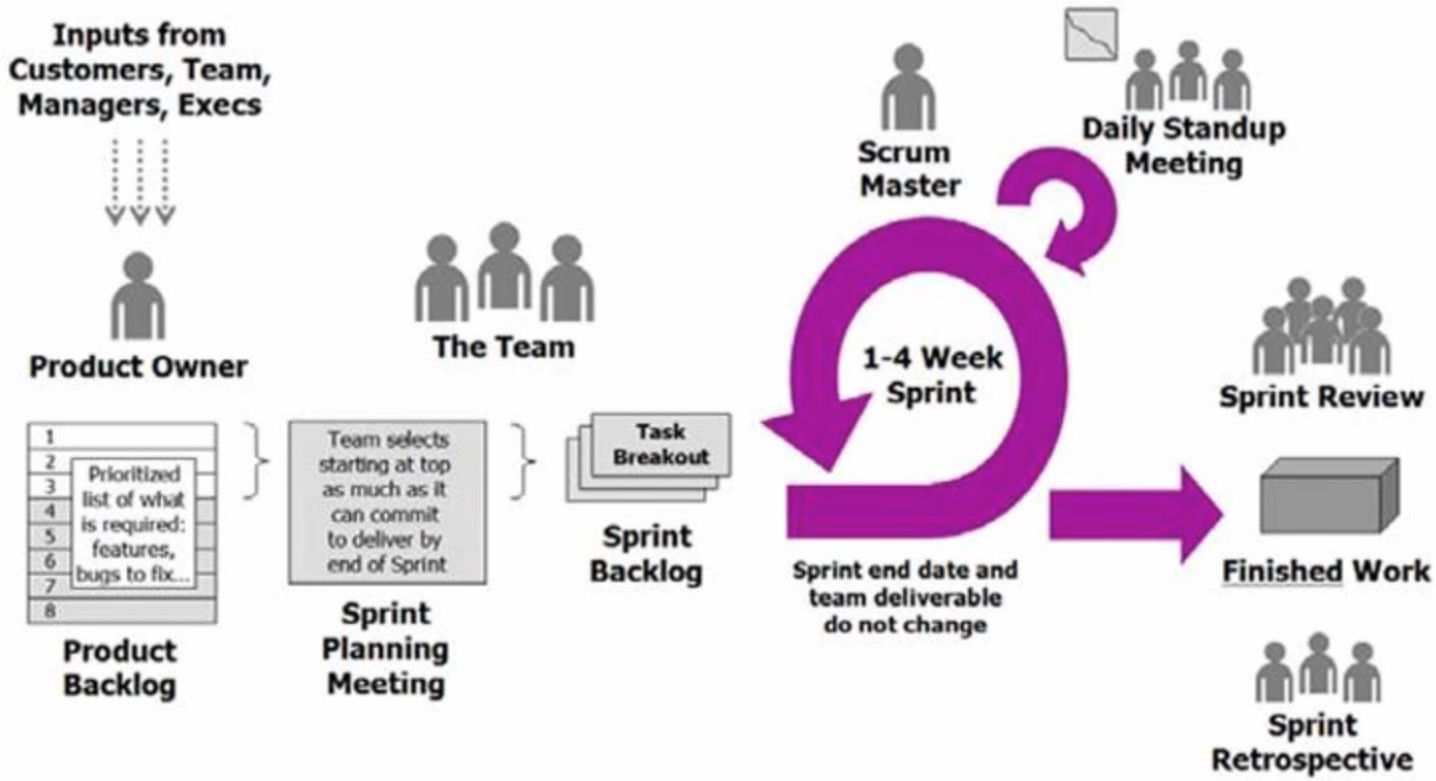
Scrum is adaptive in two ways:

Solution adaptation

Process adaptation



Scrum Overview - Sprint

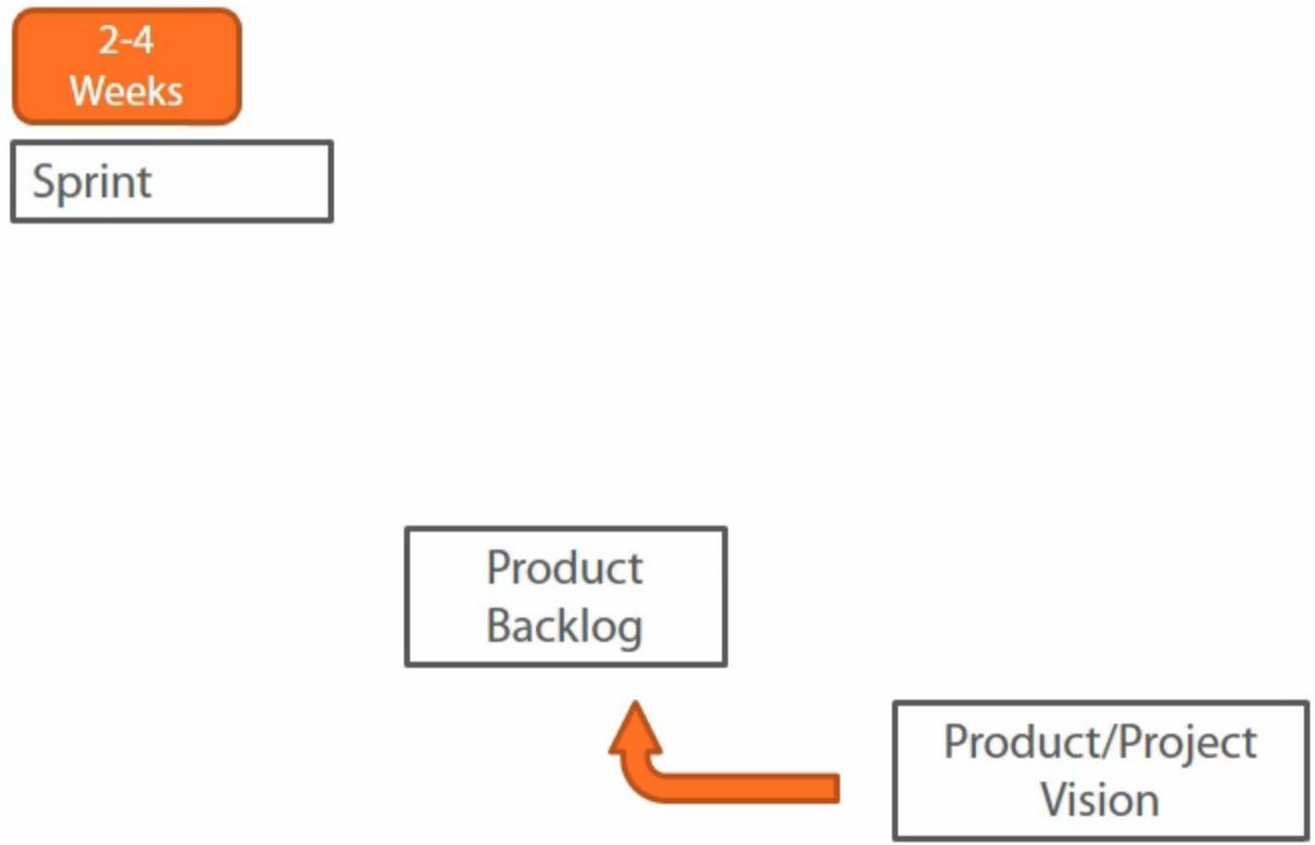


How the Scrum Process Works

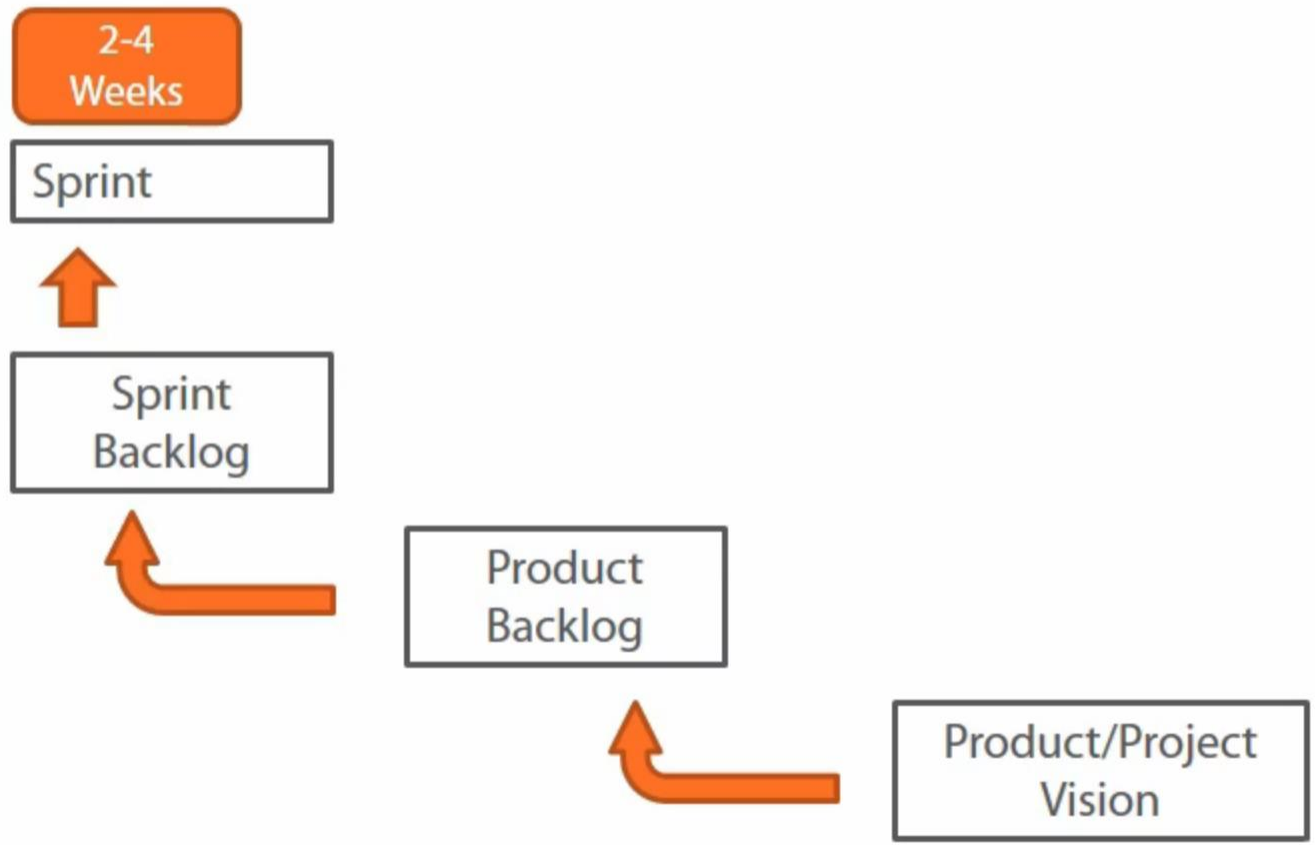
Product/Project
Vision

How the Scrum Process Works

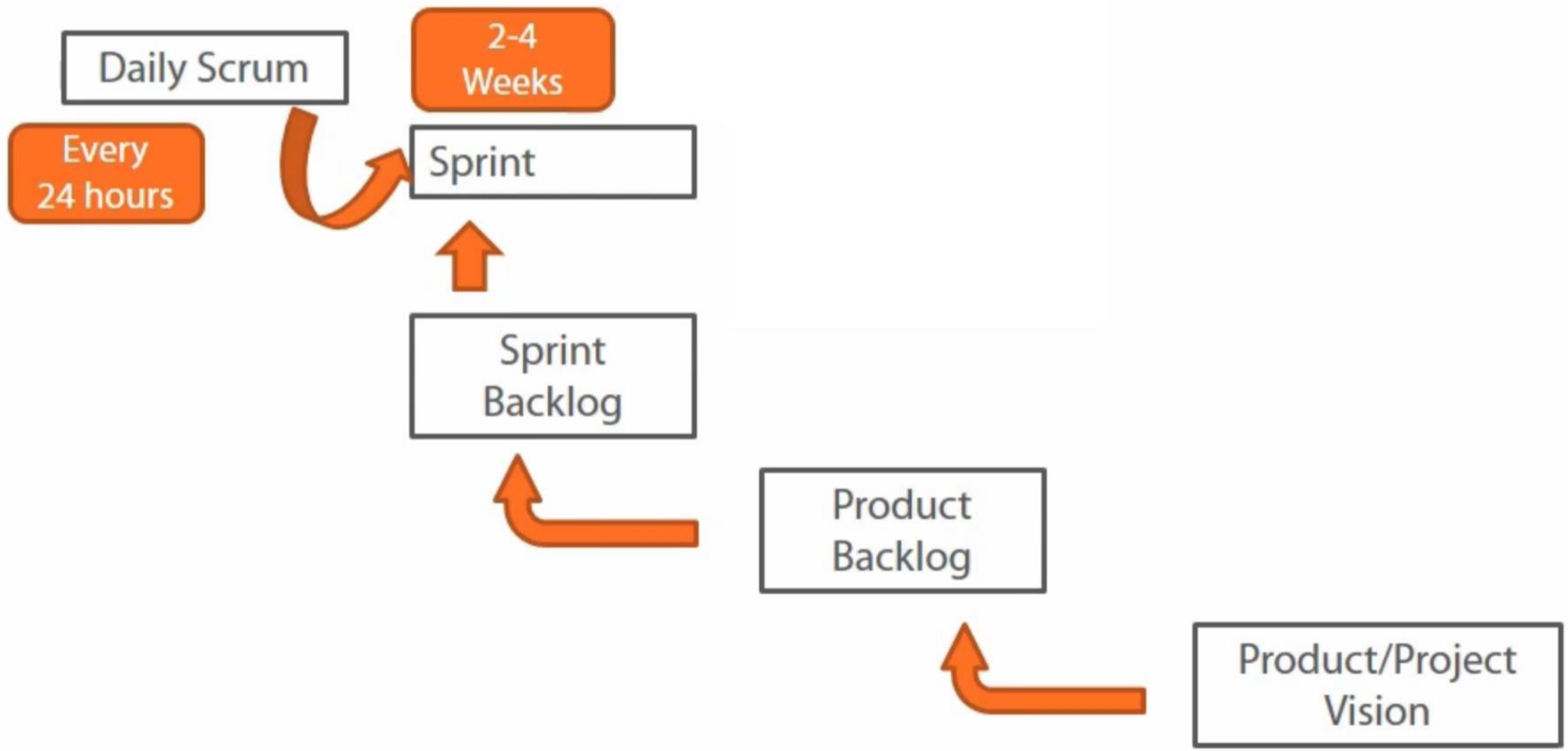
How the Scrum Process Works



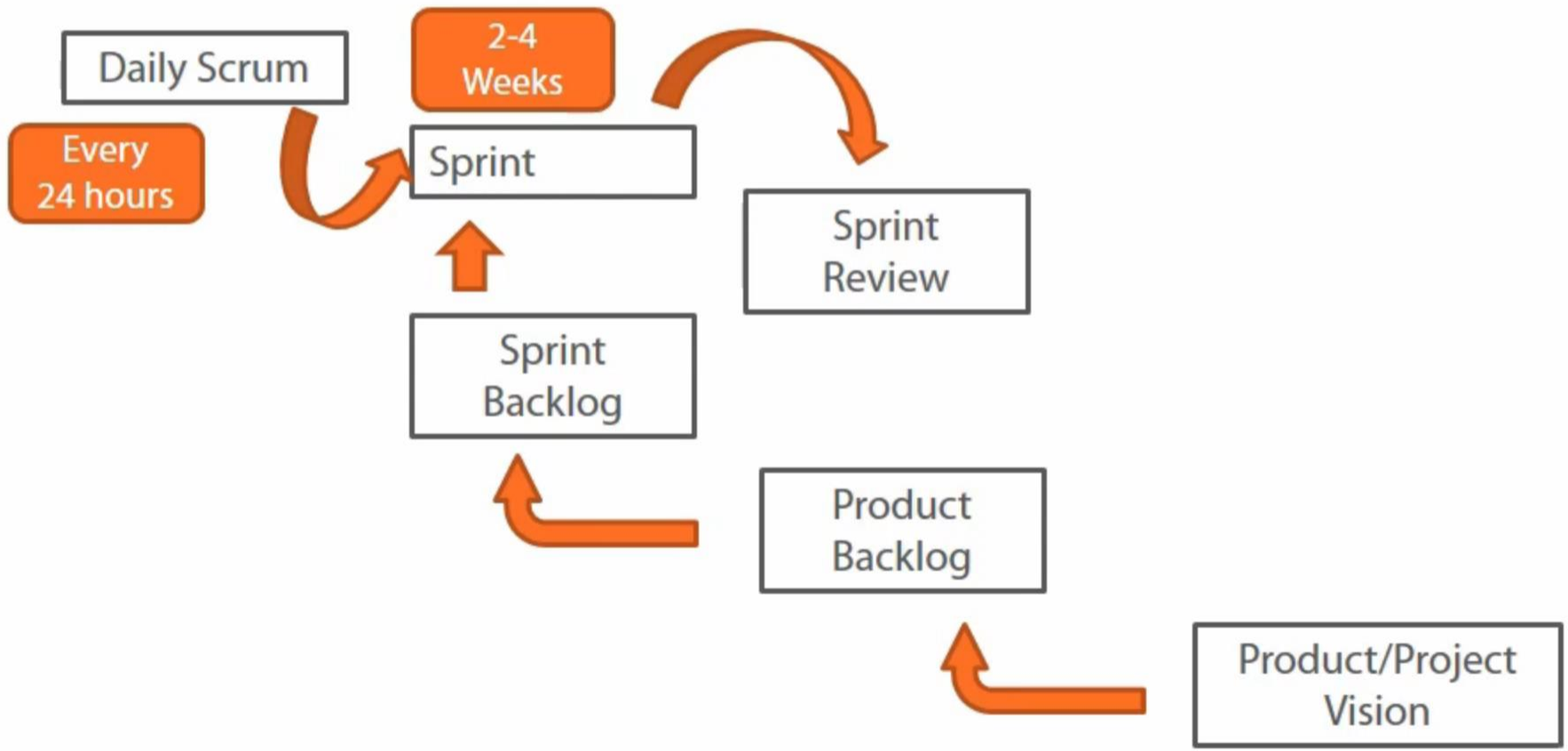
How the Scrum Process Works



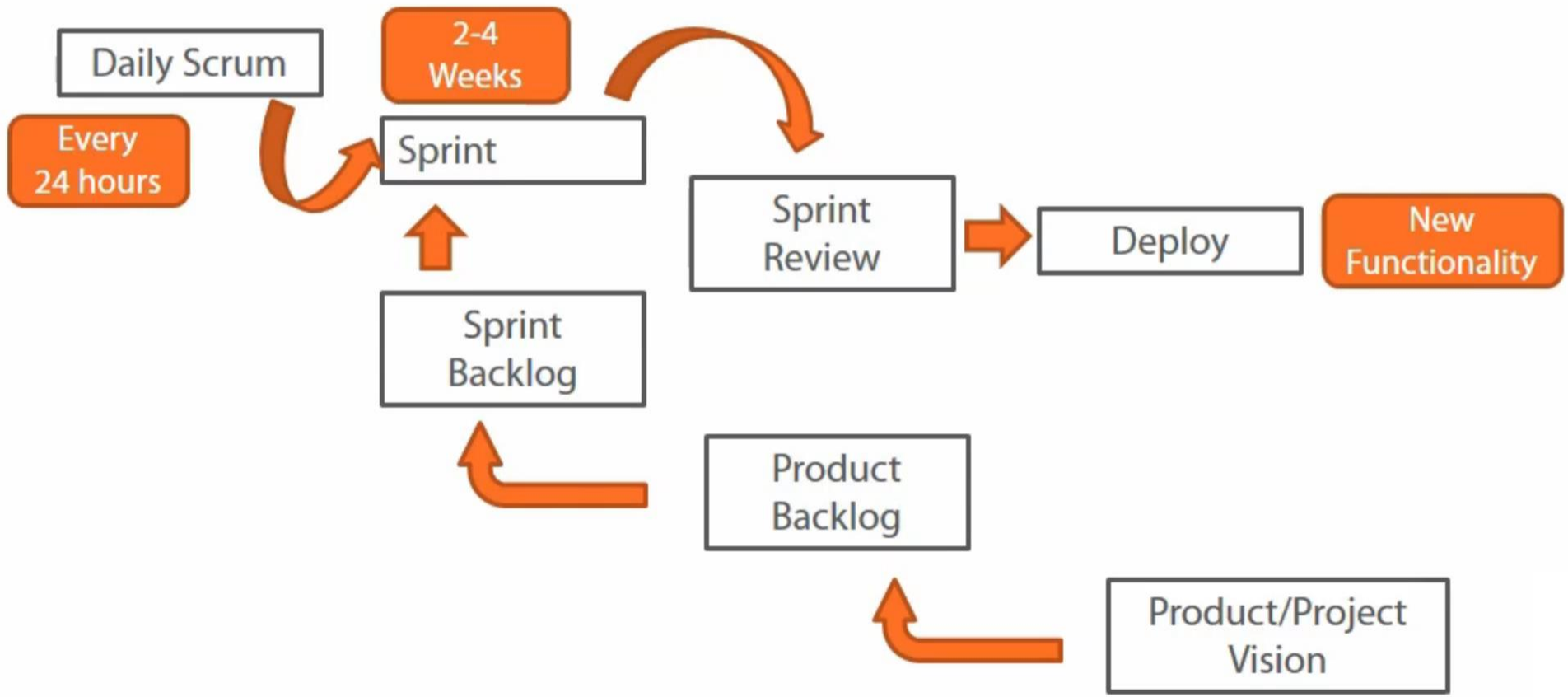
How the Scrum Process Works

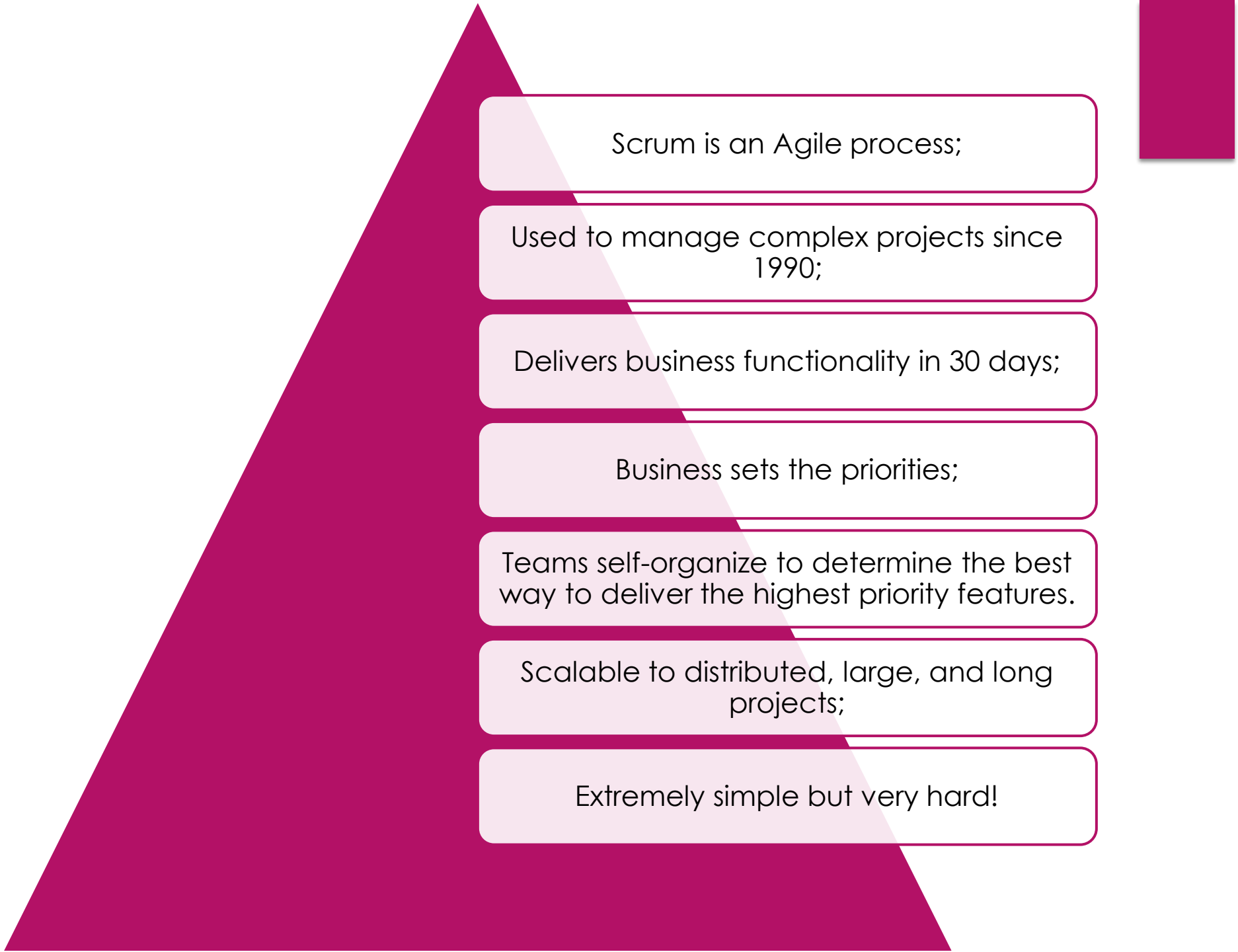


How the Scrum Process Works



How the Scrum Process Works




Scrum is an Agile process;
Used to manage complex projects since 1990;
Delivers business functionality in 30 days;
Business sets the priorities;
Teams self-organize to determine the best way to deliver the highest priority features.
Scalable to distributed, large, and long projects;
Extremely simple but very hard!

COURAGE
Scrum Team members have courage to do the right thing and work on tough problems

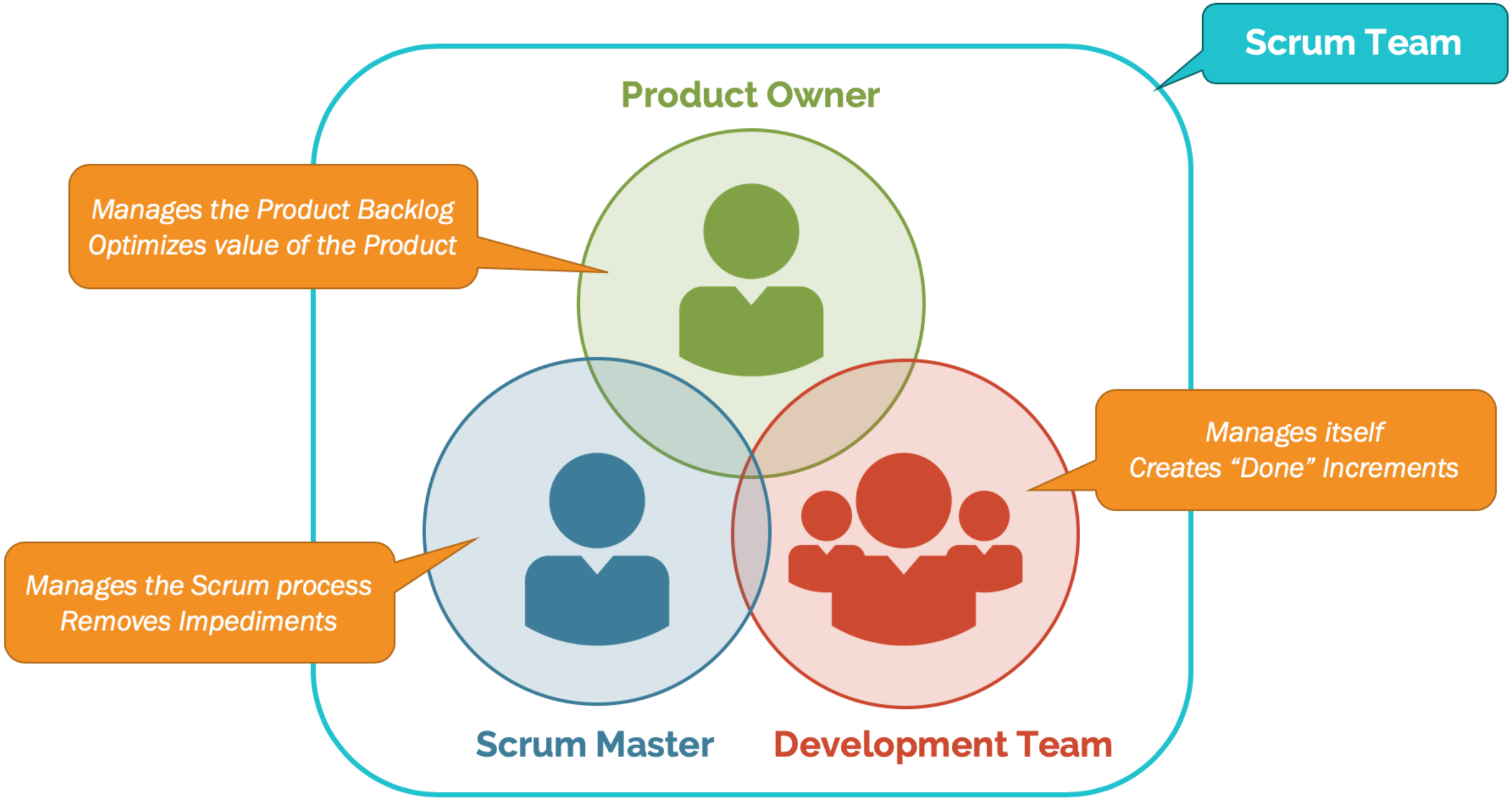
FOCUS
Everyone focuses on the work of the Sprint and the goals of the Scrum Team

COMMITMENT
People personally commit to achieving the goals of the Scrum Team

RESPECT
Scrum Team members respect each other to be capable, independent people

OPENNESS
The Scrum Team and its stakeholders agree to be open about all the work and the challenges with performing the work

SCRUM ROLES


Scrum Team
Product Owner
Manages the Product Backlog
Optimizes value of the Product
Manages itself
Creates "Done" Increments
Manages the Scrum process
Removes Impediments
Scrum Master
Development Team

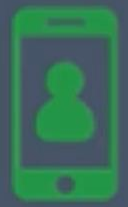
Product Owner

- Talk to the stakeholders and users to understand their needs and vision of the product.

Product Owner writes a simple description of the product feature from an end user's perspective (called user stories)

& prioritize these work items (or user stories)

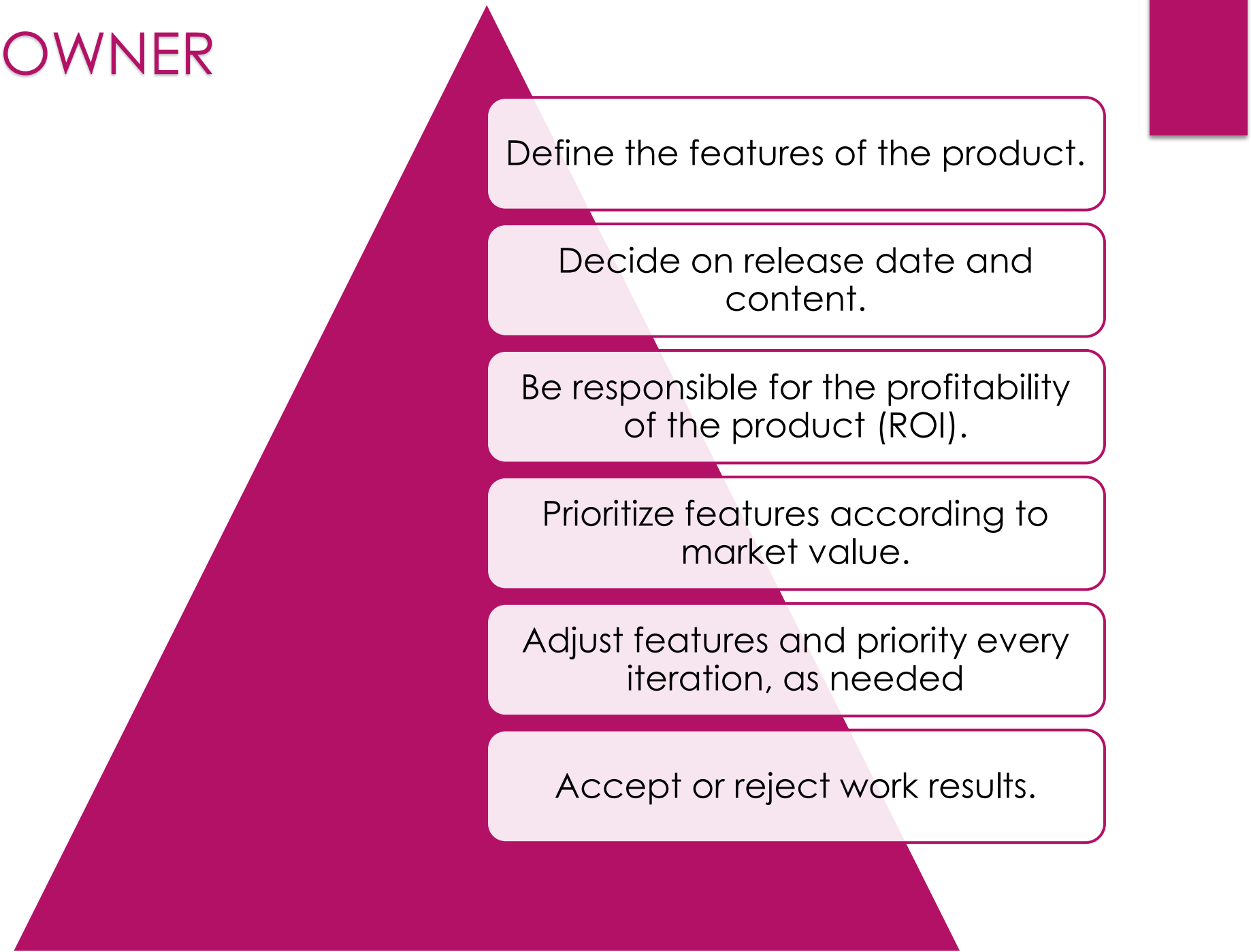
As a <End User Role>,



I want to <desired action>,

so that <desired benefit>.

PRODUCT OWNER

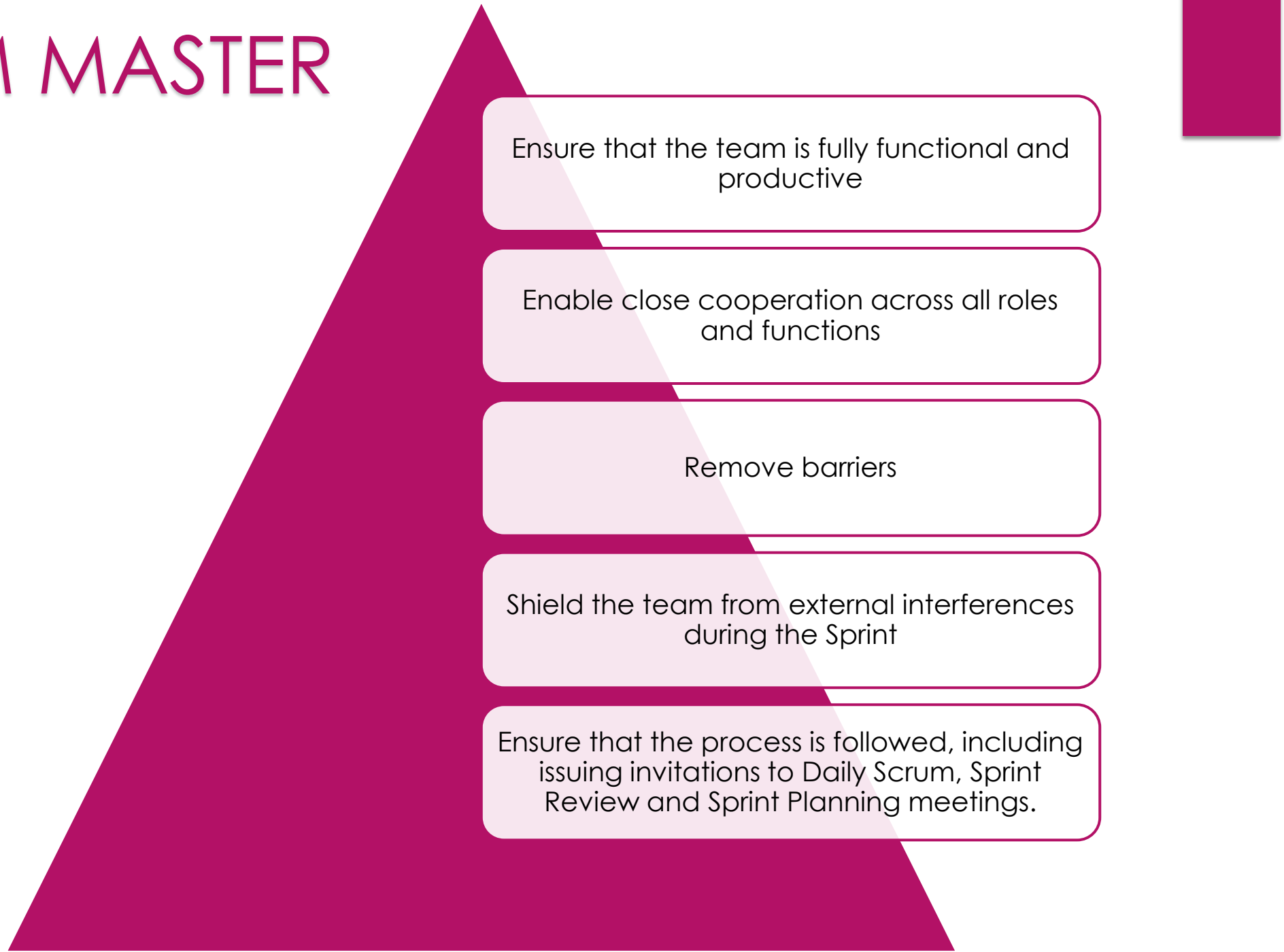


- Define the features of the product.
- Decide on release date and content.
- Be responsible for the profitability of the product (ROI).
- Prioritize features according to market value.
- Adjust features and priority every iteration, as needed
- Accept or reject work results.

Scrum Master

- Ensures Scum is understood and used skillfully by Scrum Team.
- Servant Leader
- Models Agile Mindset and Scrum Framework

SCRUM MASTER

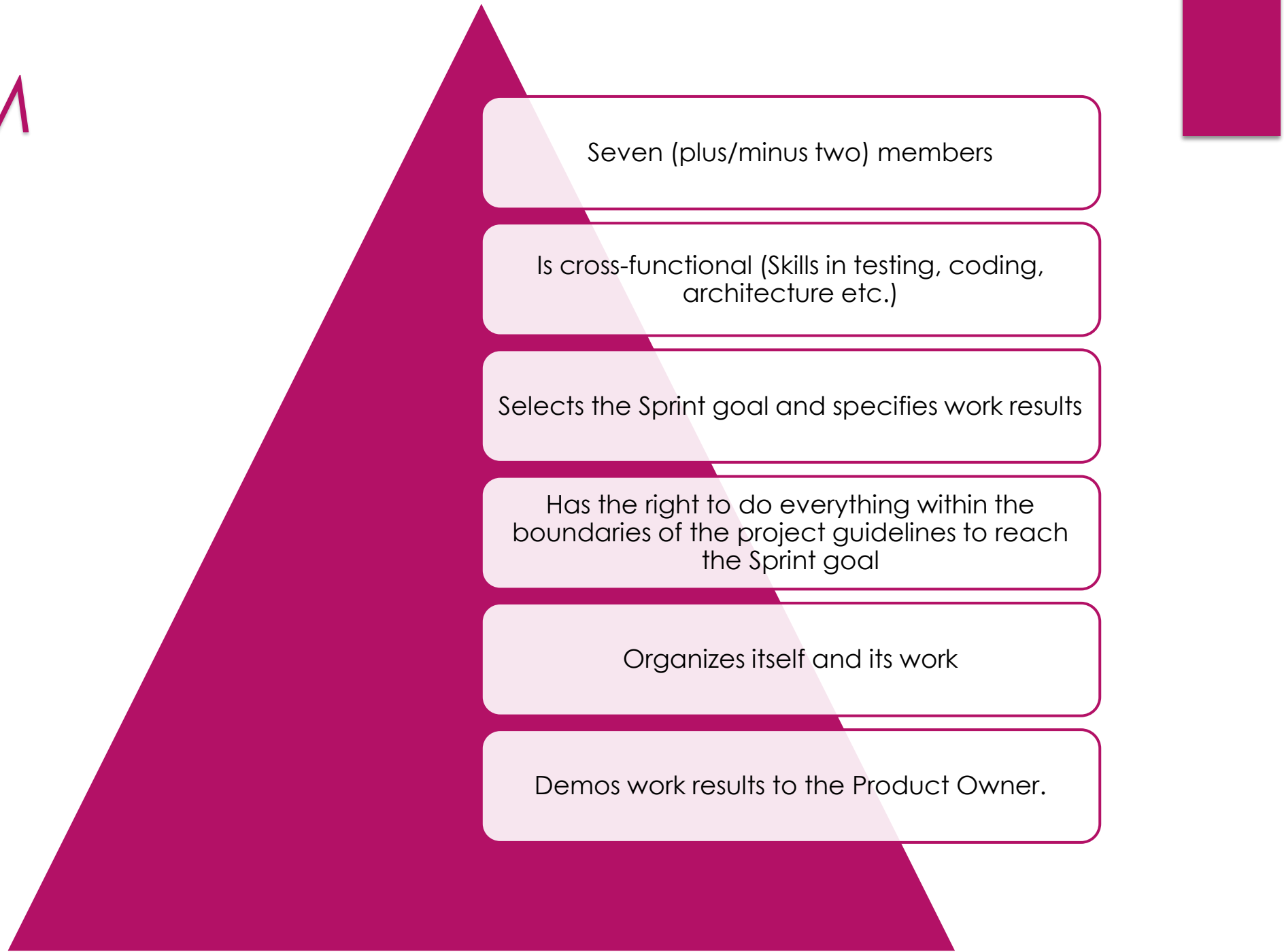


- Ensure that the team is fully functional and productive
- Enable close cooperation across all roles and functions
- Remove barriers
- Shield the team from external interferences during the Sprint
- Ensure that the process is followed, including issuing invitations to Daily Scrum, Sprint Review and Sprint Planning meetings.

The Development Team

- Build Product Increment
- Dev Team is cross functional
- Self Organizing
- Accountable as a team
- Three to Nine people

TEAM



- Seven (plus/minus two) members
- Is cross-functional (Skills in testing, coding, architecture etc.)
- Selects the Sprint goal and specifies work results
- Has the right to do everything within the boundaries of the project guidelines to reach the Sprint goal
- Organizes itself and its work
- Demos work results to the Product Owner.

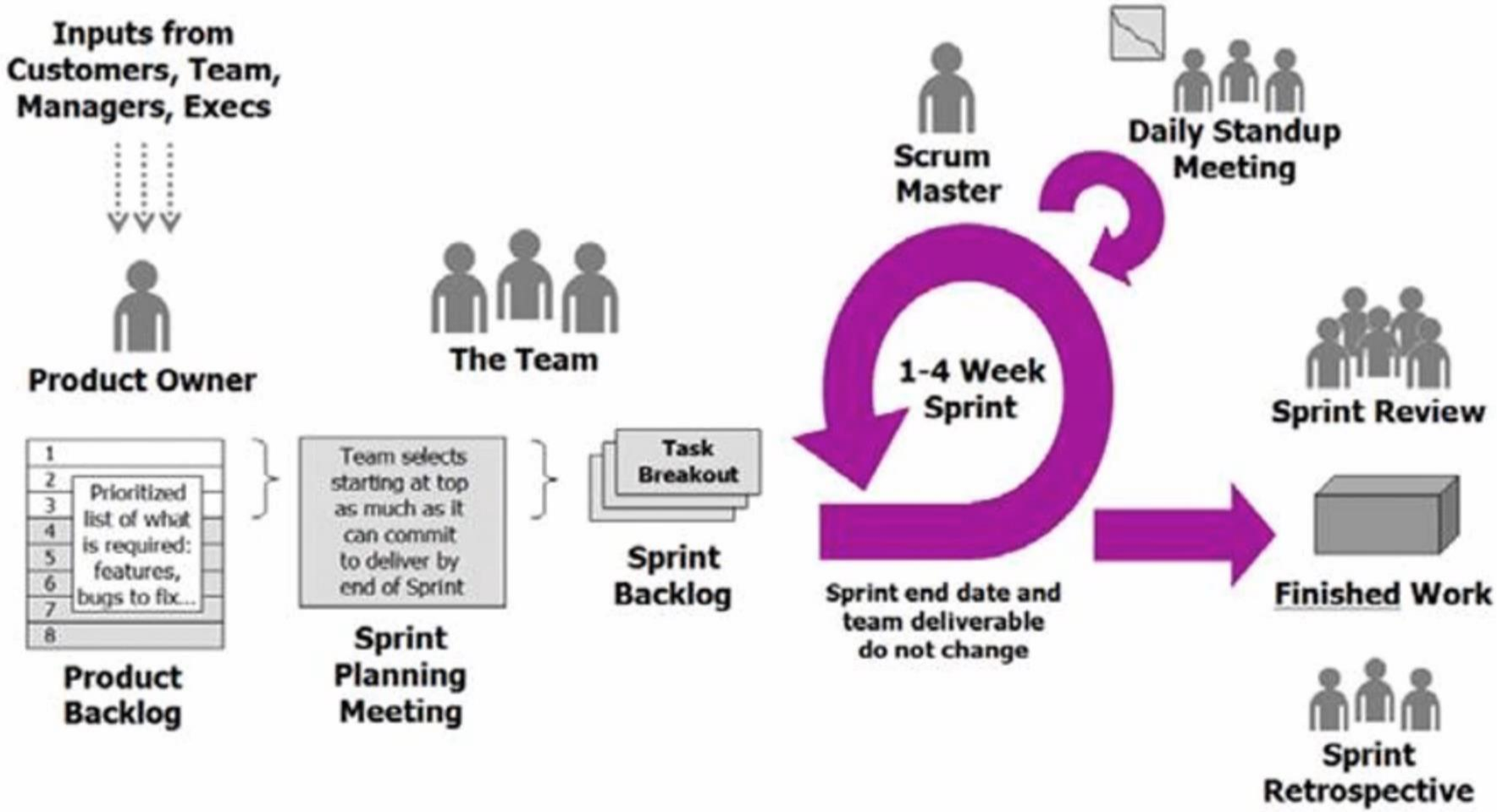
Development Team

A group of professionals who are capable of delivering a potentially releasable increment at the end of the sprint.

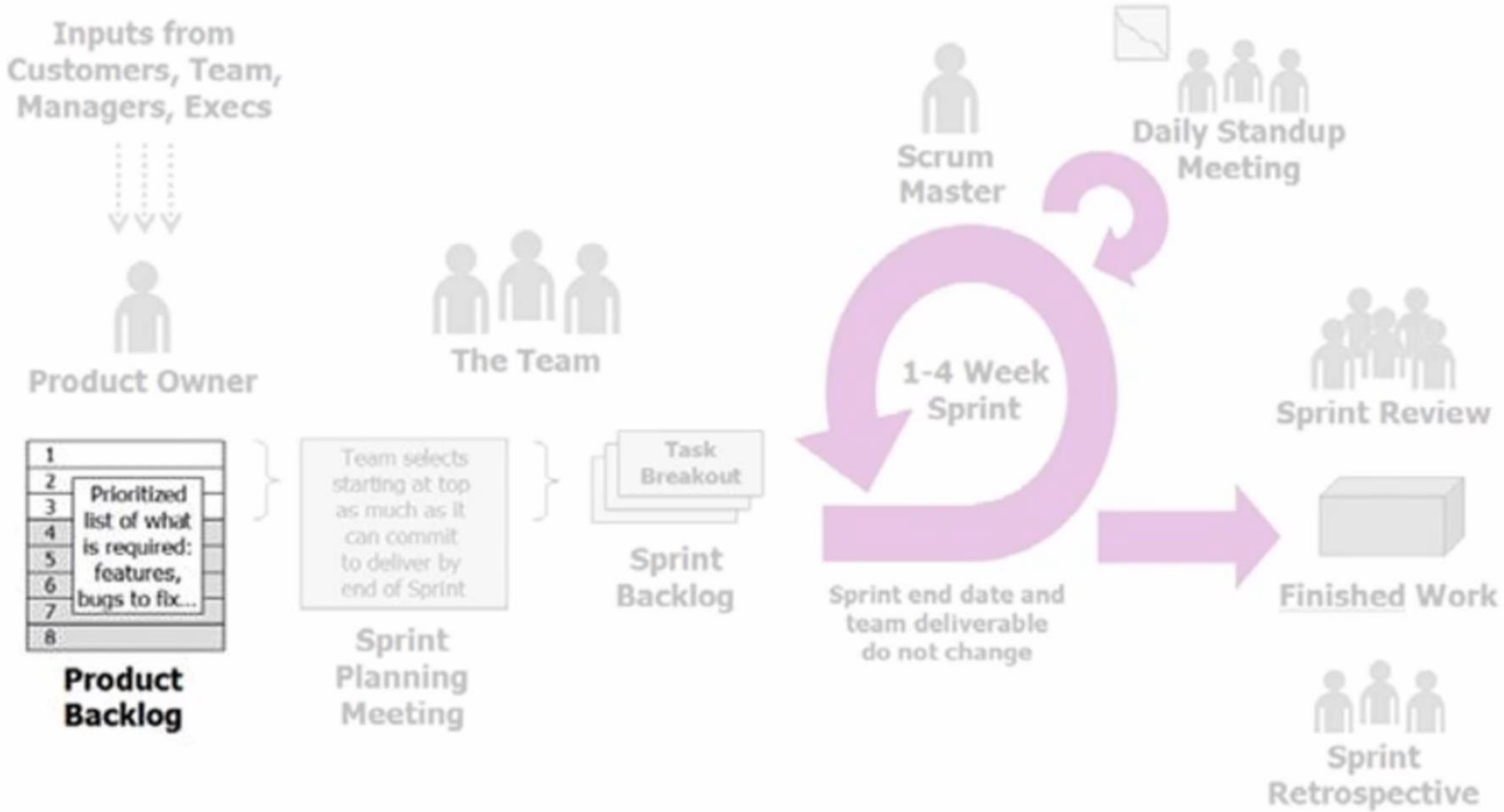
Sprints

- Fixed-length time-box 2 - 4 weeks in duration
- Development work gets done in the sprint
- Work is divided into small increments small enough to do in one sprint
- The time-boxing approach keeps length of sprint fixed and squeezes as many small increments of work into the sprint as possible

Scrum Overview – Product Backlog



Scrum Overview – Product Backlog



Product Backlog

- Dynamically changes throughout the project
  - Completed work is removed from the backlog
  - New work is added
  - The work is continuously re-prioritized as necessary
- Used to hold all work (including defects) that cannot be resolved in the current sprint

The Product Owner in a Scrum project owns responsibility for defining and maintaining the Product Backlog

The length of the Product Backlog can be limited for a very adaptive project or deeper for a more planned project

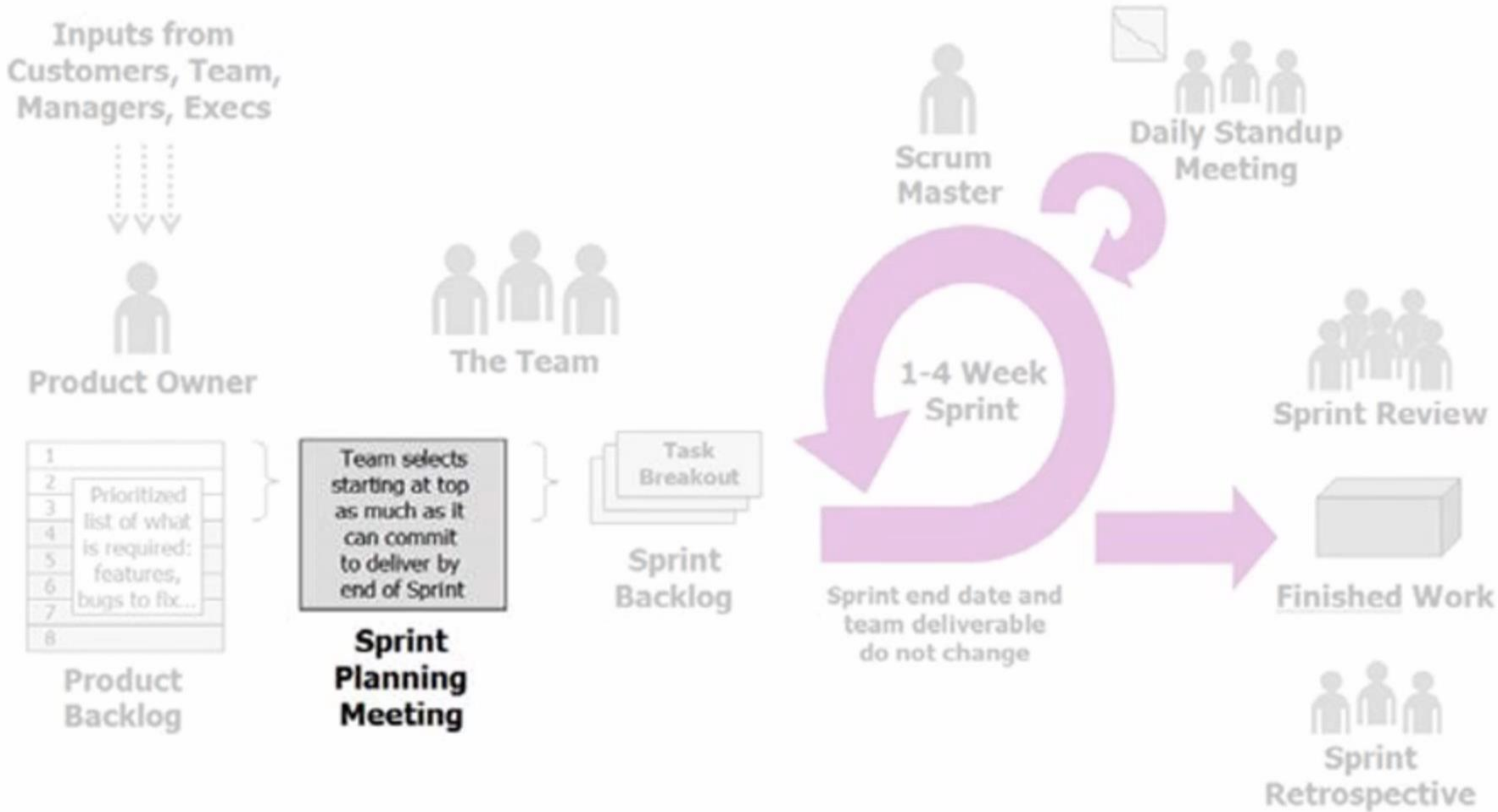
Product Backlog

Feature list (User stories)

Features, bugs, technical work, information...

Not required to be complete at the beginning

Scrum Overview – Sprint Planning



Sprint Planning Meeting

Part 1

Product Owner and team negotiate which stories will be taken into the sprint and define sprint goals

- Product Owner decides which stories are of highest priority
- Team can push back and voice concerns

Accepted stories are moved into the Sprint Backlog for development

Scrum Methodology – Scrum Meetings - http://scrummethodology.com/scrum-meetings/

Sprint Planning Meeting

Part 2

The team discusses the work to be done to complete the stories without the product owner and plans development tasks

Scrum Methodology – Scrum Meetings - http://scrummethodology.com/scrum-meetings/

The Product Backlog should be groomed and prioritized by the Product Owner prior to sprint planning

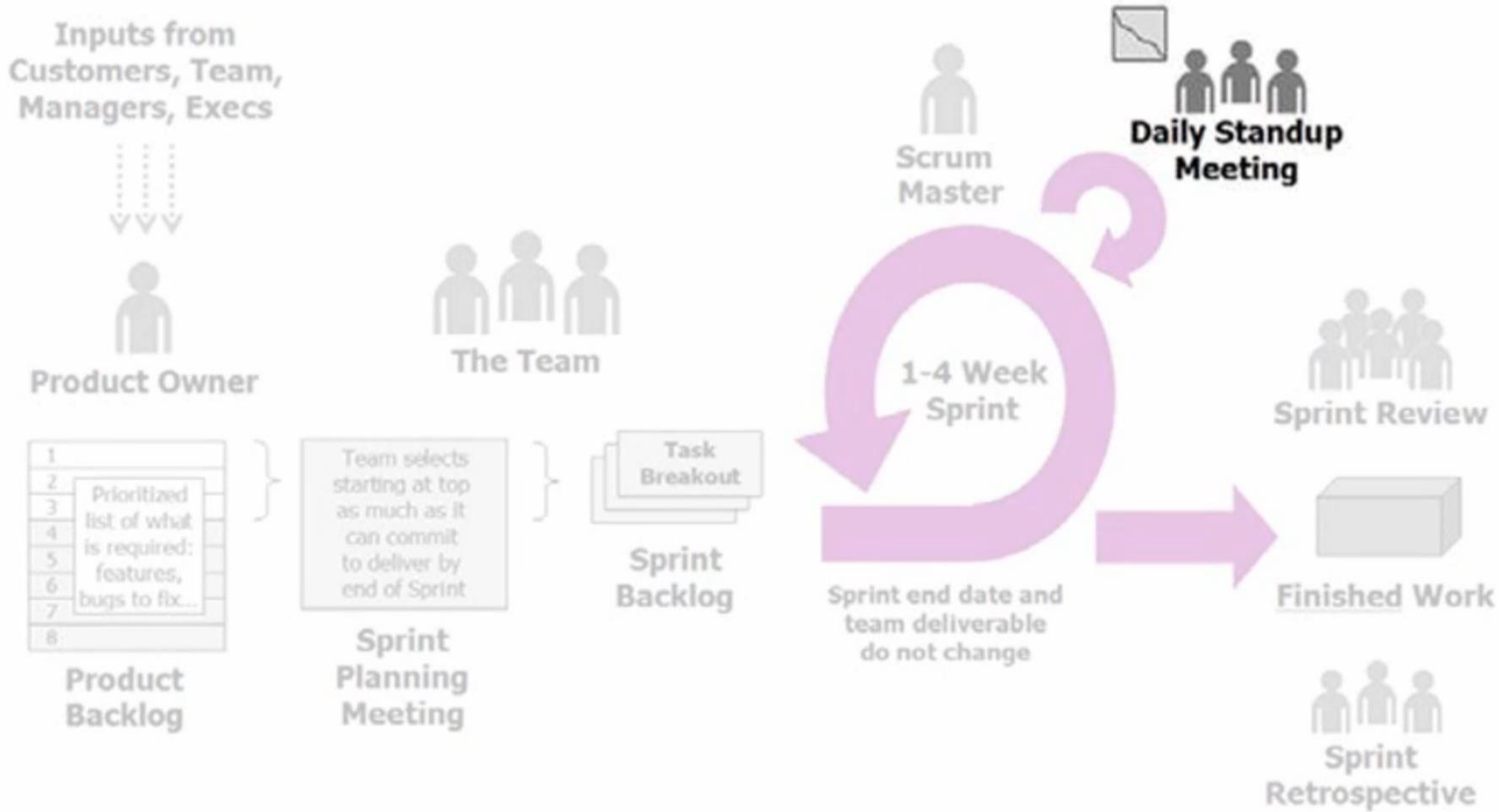
Sprint planning meeting is typically time-boxed to 4 hours

Sprint Planning Meeting

Goals of sprint planning:

- Product Owner and team set sprint goals
- Product Owner and team negotiate what will be taken into the sprint and those items move into the Sprint Backlog
- The team defines the tasks to be done to successfully complete the sprint (some tasks will be defined as the sprint progresses)

Scrum Overview – Daily Standup



Daily Standup Meetings

The Daily Standup is basically a check-in for everyone on the team to coordinate what's going on, monitor progress, and to identify any obstacles that may be inhibiting progress

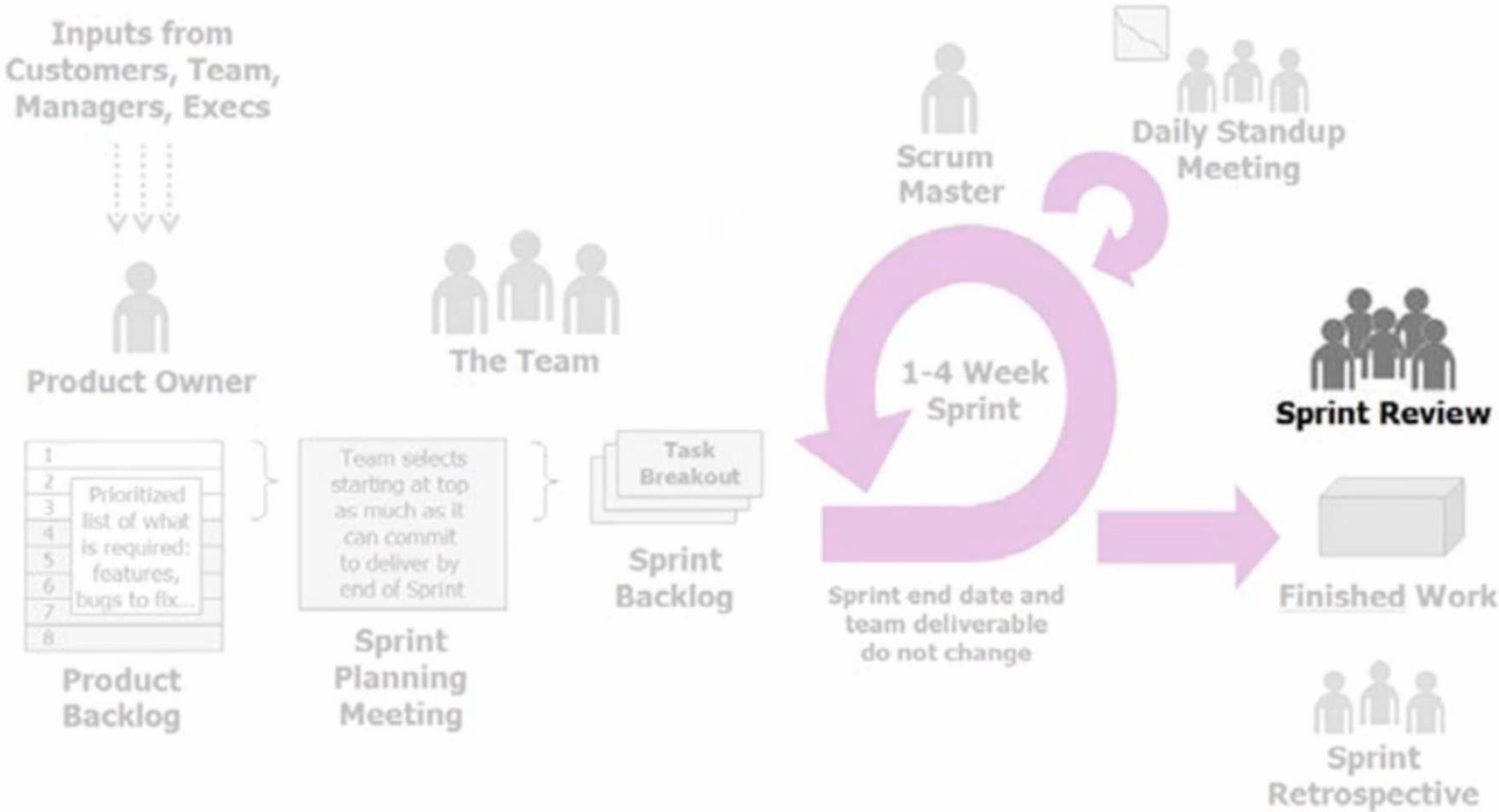
Each person on the team answers three questions:

- What did you accomplish yesterday?
- What are you going to accomplish today?
- What obstacles are in your way?

The meeting is facilitated by the Scrum Master

The meeting is typically done in front of an Agile Kanban board

Scrum Overview – Sprint Review



Sprint Review Meeting

Takes place at the end of every sprint

Purpose of the Sprint Review is for the product owner to review and accept the results of the sprint

Team presents completed work to the Product Owner for acceptance

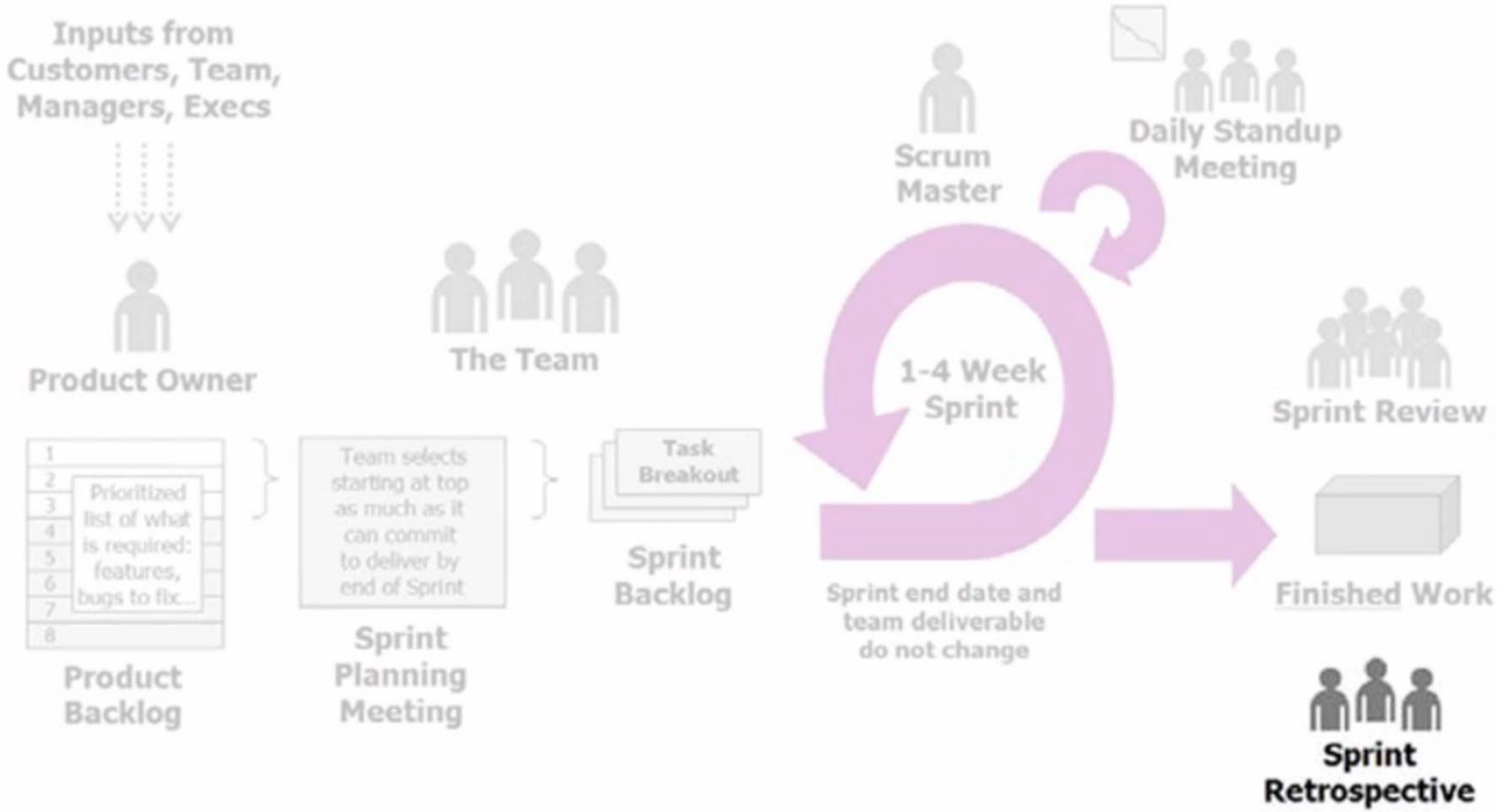
The Product Owner checks the work against the acceptance criteria to determine if the work is satisfactory or not

Scrum Methodology – Scrum Meetings - http://scrummethodology.com/scrum-meetings/

All defects in the product should typically be resolved prior to the Sprint Review unless the Product Owner has specifically approved deferring the resolution till a later sprint.

If the Product Owner requests significant changes or enhancements in the stories, those will typically be added to the Product Backlog and deferred until a future sprint.

Scrum Overview – Sprint Retrospective



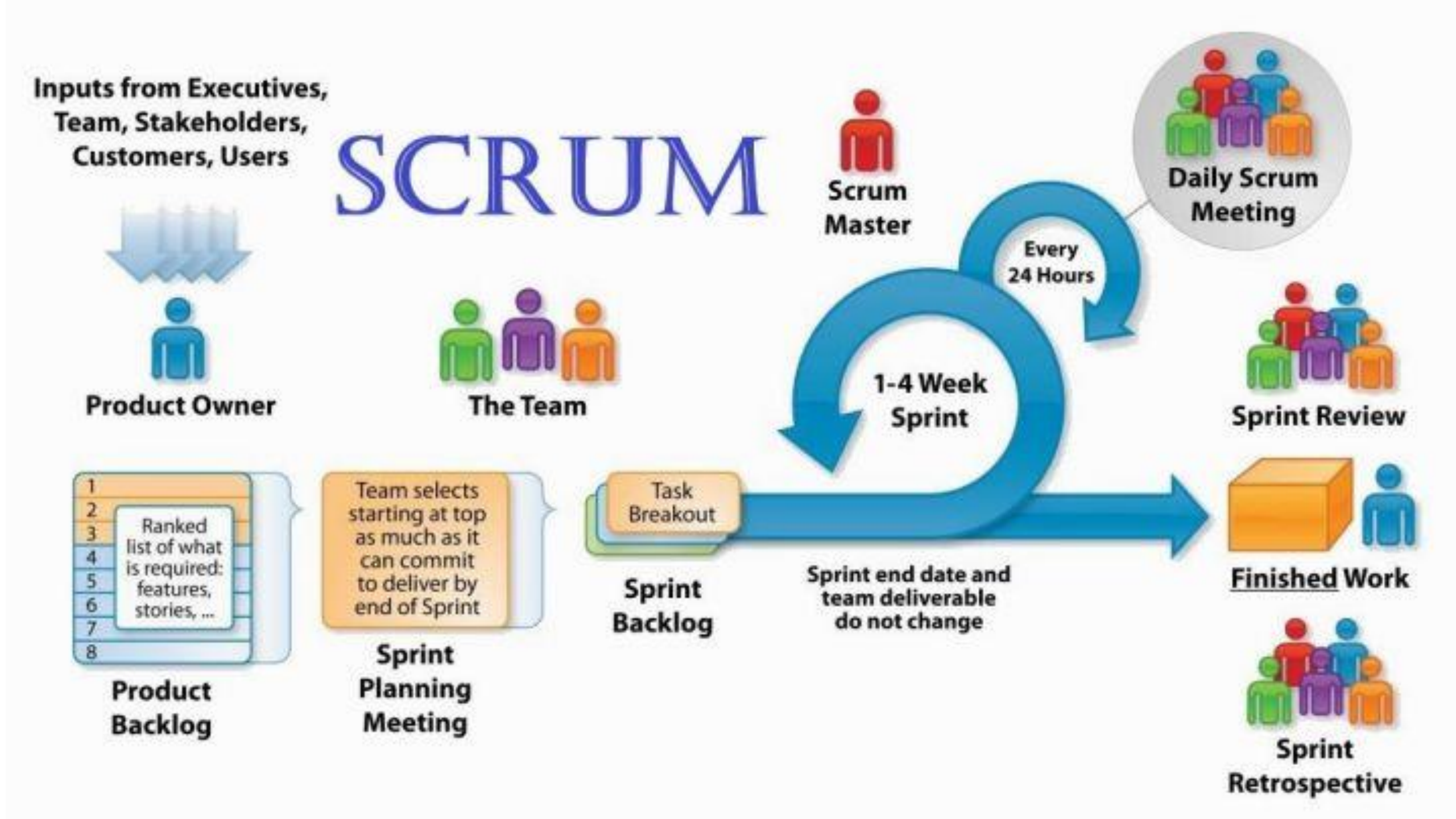
Sprint Retrospective Meeting

Takes place at the end of every sprint after the Sprint Review

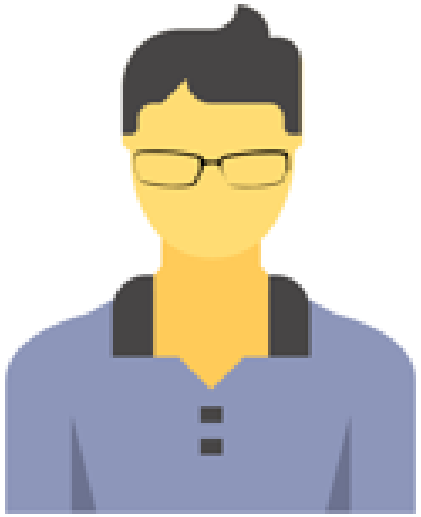
Purpose of the Sprint Retrospective is for the team and the Product Owner to reflect on:

- What went well and what didn't go well in the sprint, and
- Identify opportunities for improvement
- Sprint Retrospective is essentially a post-mortem to assess the performance of the sprint

Scrum Methodology – Scrum Meetings - http://scrummethodology.com/scrum-meetings/


Inputs from Executives, Team, Stakeholders, Customers, Users
SCRUM
Scrum Master
Daily Scrum Meeting
Every 24 Hours
Product Owner
The Team
1-4 Week Sprint
Sprint Review
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Team selects starting at top as much as it can commit to deliver by end of Sprint
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Product Backlog
Sprint Planning Meeting
Sprint Retrospective

Direct communication encouraged

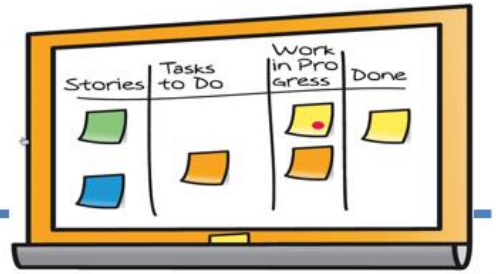


Product Owner:
Owns "what" is desired
And "why" it's desired

Scrum Master:
Keeper of Scrum
Process, facilitator

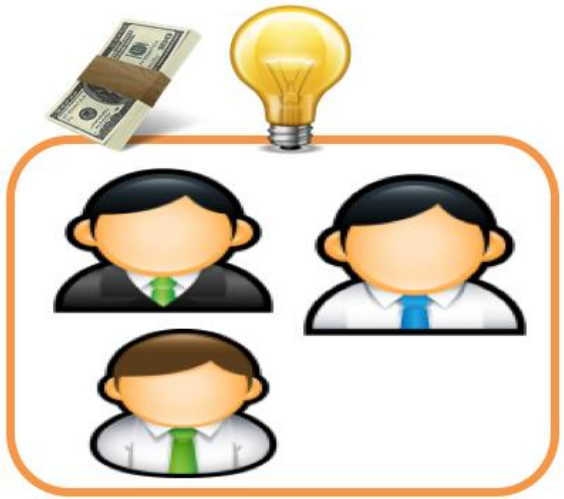
Scrum Team Members:
Owns "how" and "how quickly"
work is delivered

Taskboard



Burndown Chart

Scrum Members:



Stakeholders

Product Backlog

The Scrum Team

Product Owner

Scrum Master

Sprint Backlog

The Development Team

3-9 persons

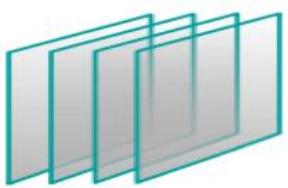
The Scrum Process:

Daily Scrum Meetings

Max 15 min.

Sprint Planning

Product Backlog



Sprint Backlog

Sprint

Sprint Review

Working increment of the software

Potential Benefits of Scrum

More business value sooner

Higher quality

Greater visibility

Improved productivity and discipline

Stronger teams with better morale

Higher efficiency & less waste

Scrum Challenges

Requires significant change

- Change in practices and skills
- Change in organization, planning, budgeting, HR
- Change in mindset and culture

Makes all dysfunction visible

- Doesn't fix anything; we have to do it
- If we don't address the problems, it will be painful
- Bad products will be delivered sooner, and doomed projects will fail faster

NOT CHECKED OUT

MIGRATION TOOL

Impl. migration 8d

BACKOFFICE LOGIN

Integr. with JBoss 2d

Impl GUI 1d

BACKOFFICE USER ADMIN

GUI design (CSS) 1d

Clarify requirements 2d

Impl GUI 6d

Write failing test 3d

CHECKED OUT

GUI spec 2d

Tapestry spike 1d 2d

Write failing test 2d

DONE! :o)

DEPOSIT

Write failing test 2d

Code cleanup 1d

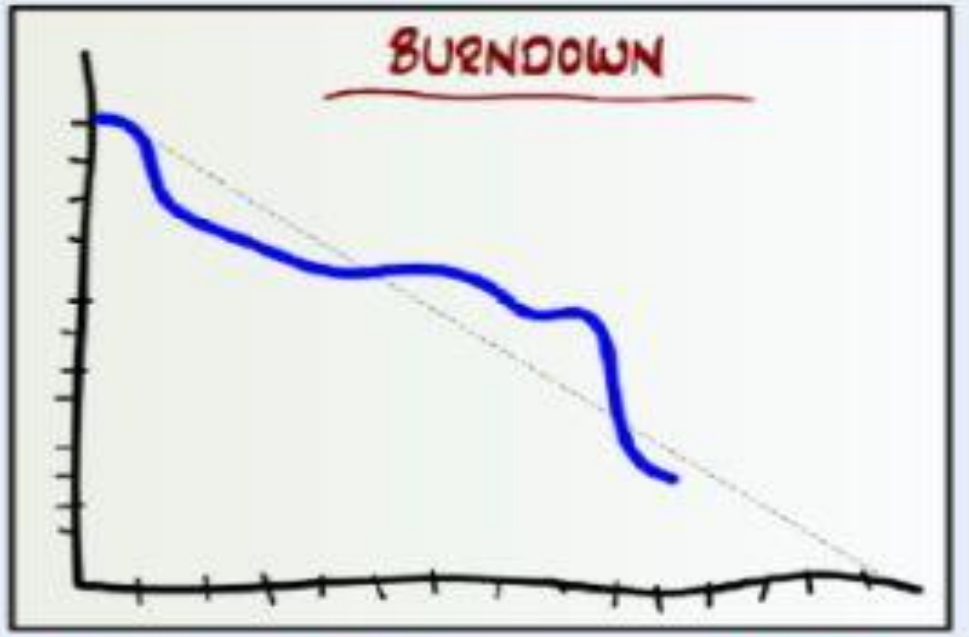
Integr test 2d 0.5d

DAO

DB design 2d 4d

Write failing test 2d 3d

SPRINT GOAL: BETA-READY RELEASE!



UNPLANNED ITEMS

Write failing test

Sales support 3d

Write whitepaper 4d

NEXT

WITHDRAW

AGILE CAREER PATH

Are You Confused?

PMI-ACP®

CERTIFIED LeSS PRACTITIONER

SAFe® Certified

S@S Certified Practitioner

Who Do You Want To Be?

Agile Tester

Agile Marketer

I want to do my job well!!!

Agile Technologist

Agile Product Leader

Agile UI/UX Expert

Agile Coach

Agile Leader

Basic Agile Knowledge

Path for Agile Coach

Basic Knowledge

Facilitation
Coaching
Servant Leadership

Lean
Building Teams
Transformations
Systems Thinking

Path for Agile Leader

Basic Knowledge

Agile Leader

Personal Leadership

Team Leaders

Organisation Leaders

Path for Agile Product Leader

Basic Knowledge

Advanced Knowledge

Path for Agile Project Managers

Basic Knowledge

Leadership Knowledge

Path for Agile Developers

Basic Knowledge

Agile Development

Path for Agile Tester

Basic Knowledge

Agile Testing

Path for
Agile Business Analyst

Basic Knowledge

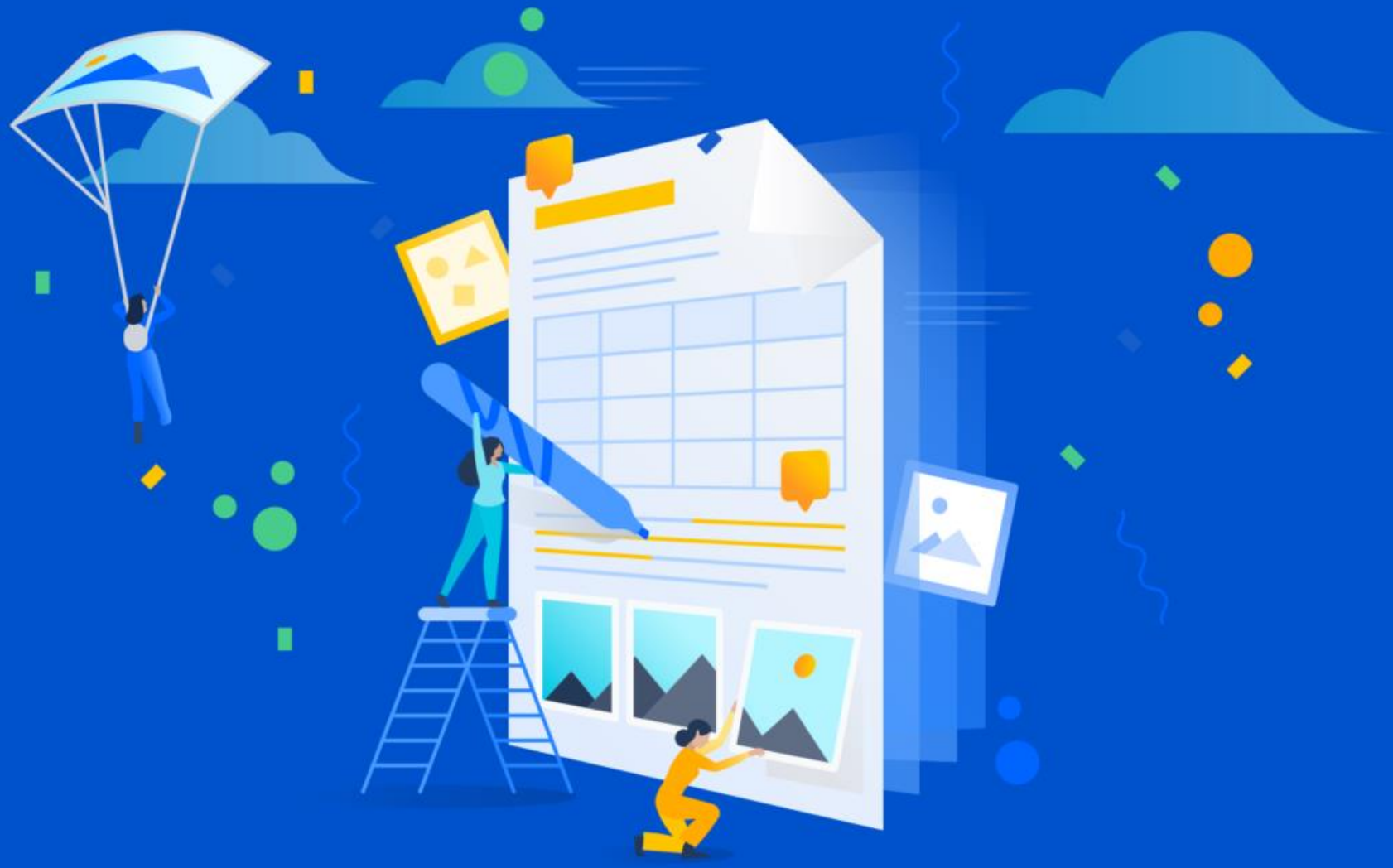
Agile Business Analyst

Path for Agile Operations

Basic Knowledge

Agile Ops

Overview



JIRA for Scrum

Issue tracker

Extensible platform

It just works!

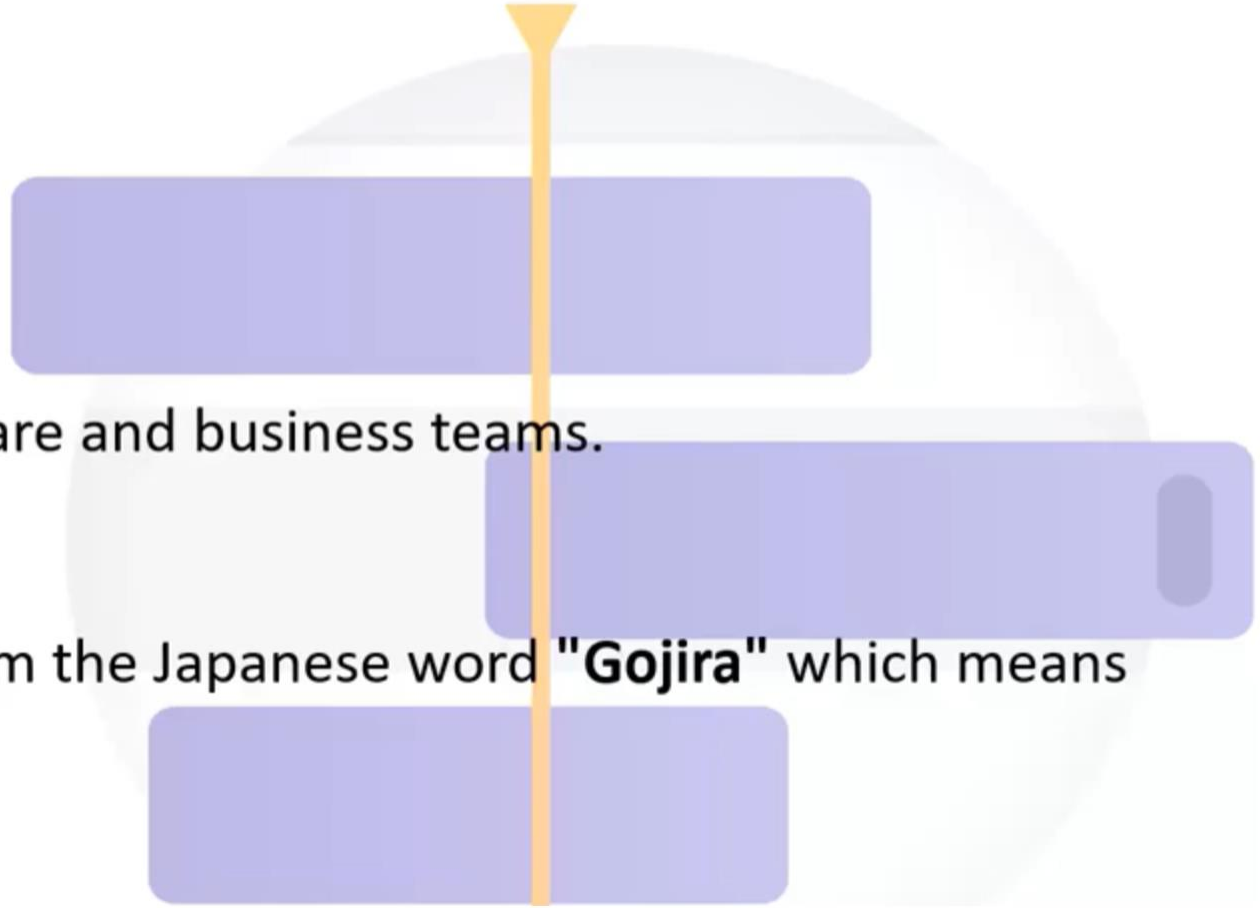
What is JIRA?

Jira is :

- ✓ Project management
- ✓ Issue-tracking
- ✓ and bug tracking tool - for agile software and business teams.

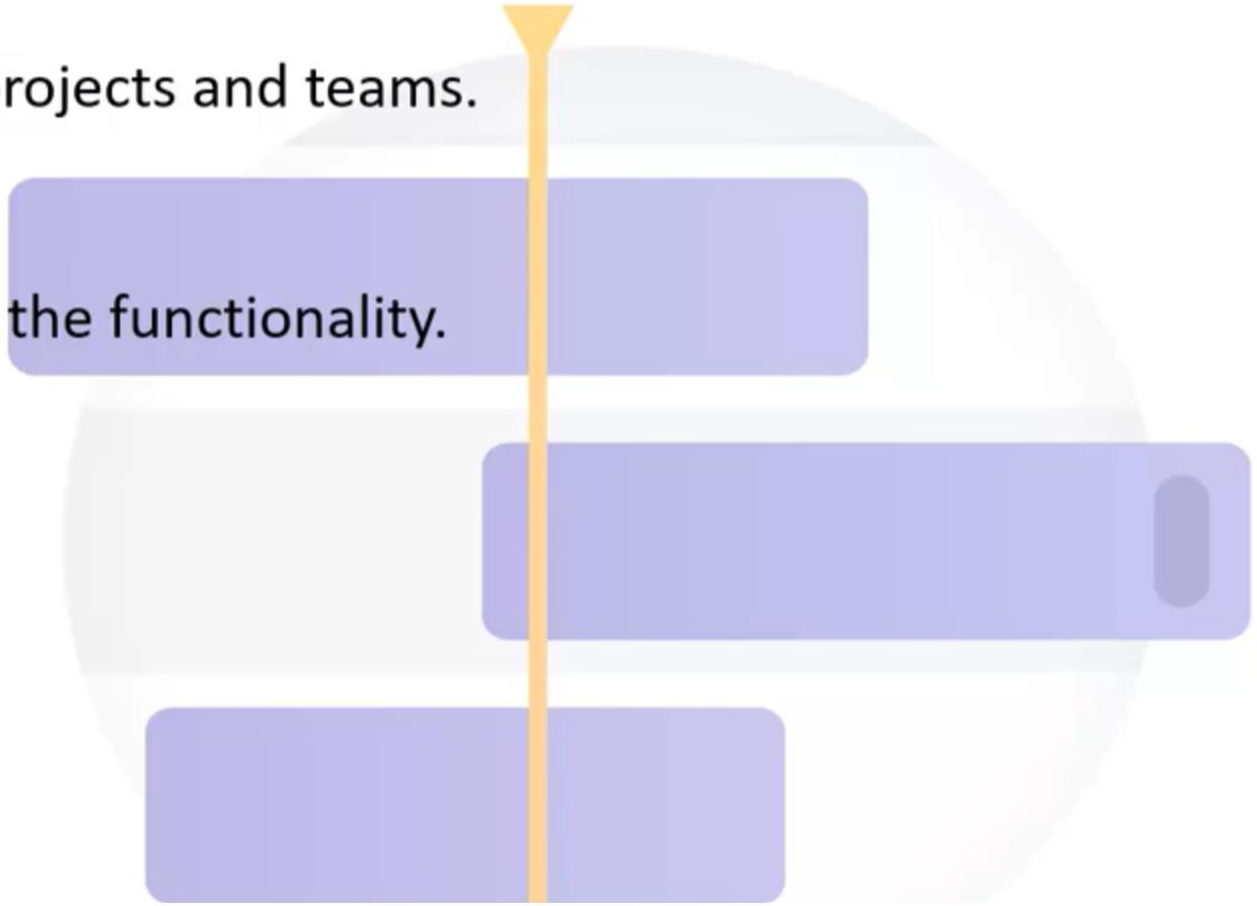
JIRA is made by Atlassian.

The name **"JIRA"** is actually inherited from the Japanese word **"Gojira"** which means **"Godzilla".**



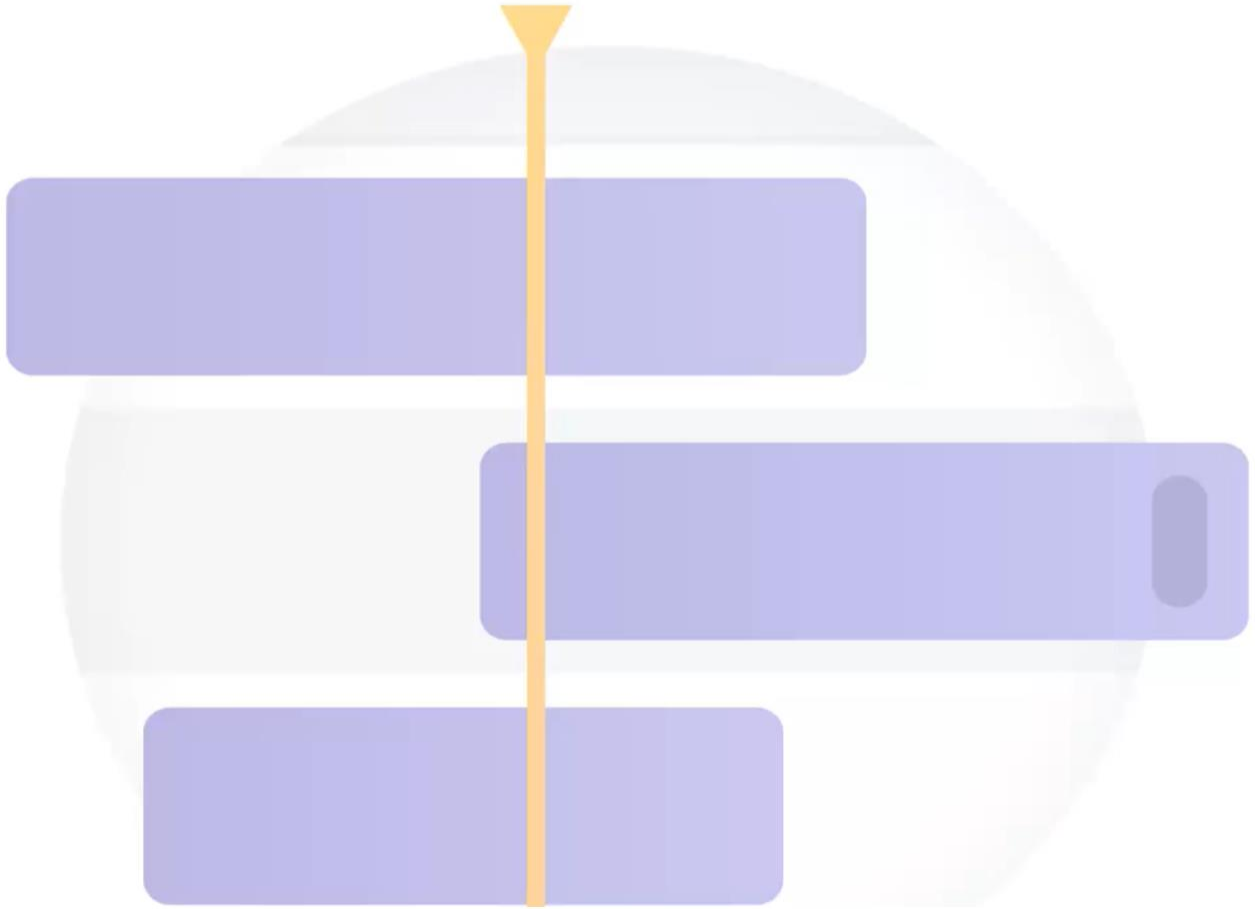
Why should we use JIRA?

- ✓ Everything that you need to manage projects and teams.
- ✓ Fully Customizable to your needs.
- ✓ Countless add-ons to further enhance the functionality.



What are the JIRA Products?

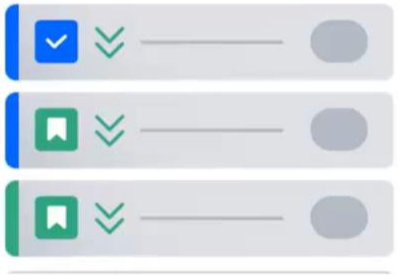
- Jira Core
- Jira Software
- Jira Service Desk
- Jira Ops



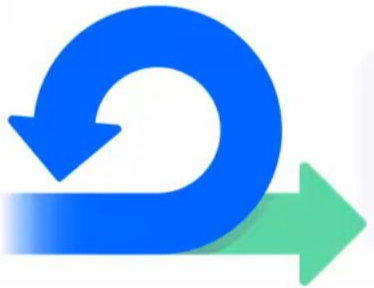
Which Agile Methodology used in JIRA?

- ✓ Jira Software is an agile project management tool that supports any agile methodology, be it scrum, Kanban, or your own unique flavor.

Scrum



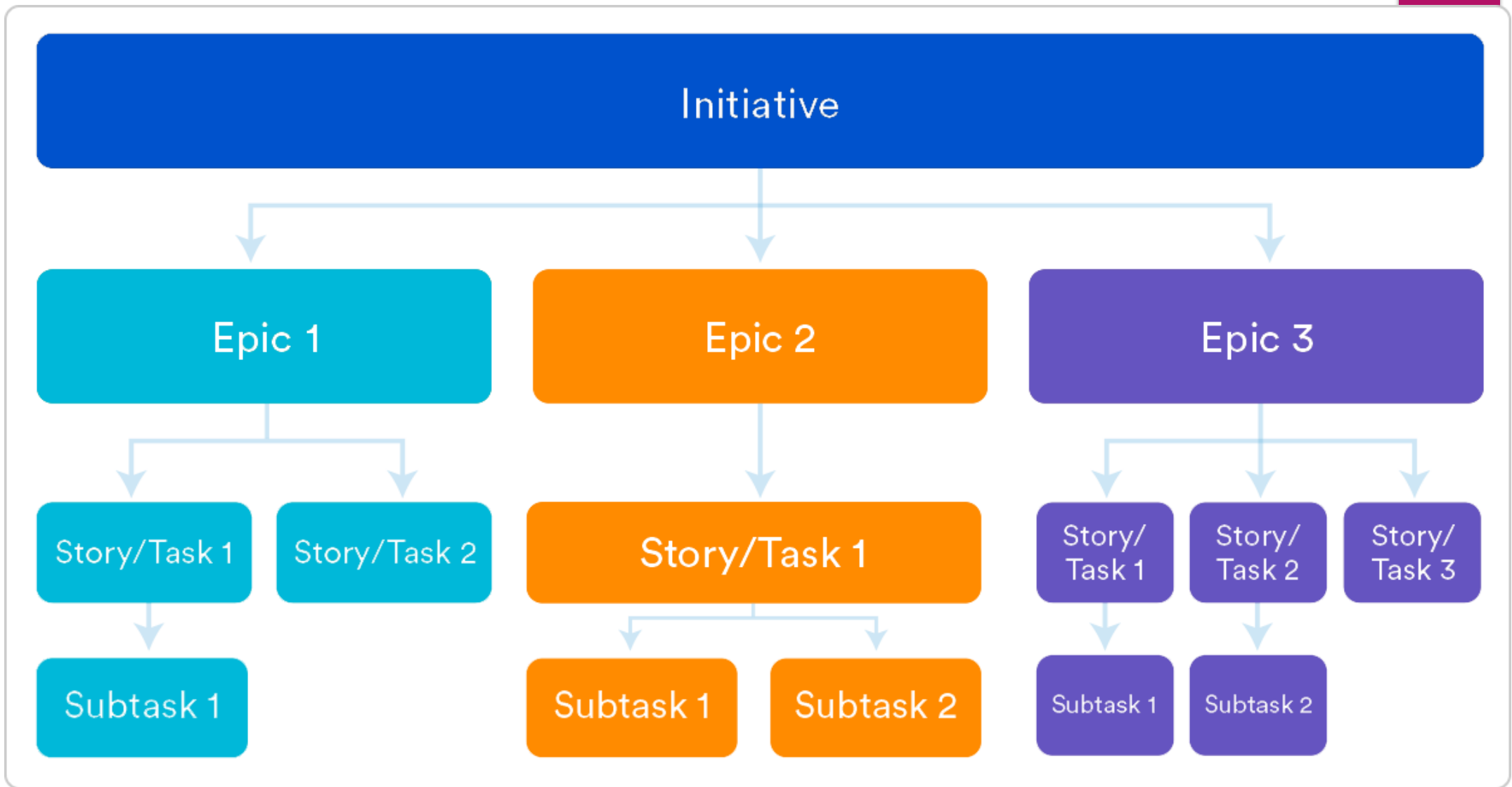
Kanban



Mixed Methodologies

Agile Scale

User Stories, Epics and Themes



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

Product Owner, Scrum Master, Team Member

Product Owner

- Manages the end-user and stakeholder expectation
- Prioritize backlog and provide clear requirements
- Provides clear testable requirements to the team

User Story Recipe:

As a <role> I want <feature>
so that <benefit>.

Some User Stories

As a traveller, I want to reserve a room

As a vacation planner, I want to see pictures of the hotels

A Note on Roles

Vacationer

Hotel Owner

Travel Agent

Trip Planner

Parent

Why User Stories Work Well

They are simple to write and understand

Software requirements is a communication problem

They elicit detail in conversation

Requirements analysis is effective when performed collaboratively

Full intent can rarely be modeled or represented 100%

The User Story Conversation

As a user with a reservation,
I want to cancel my reservation
So that I get a refund.

Does user get full or partial refund?

Refund to credit card or site credit?

How far ahead must reservation be cancelled?

Same for all hotels?

For all site visitors or can frequent travelers cancel later?

Confirmation provided to user? How?

Details as smaller sub-stories

AS A user with a reservation, I WANT to cancel my reservation SO THAT I get a refund

As a premium member, I want to cancel at the last minute with no penalty so that I get a full refund

As a non-premium member, I want to cancel up to 24 hours in advance so that I get a 50% refund

As a site member, I want an email confirmation of my cancelled reservation so that I can have a record of the transaction

Signs Stories are Working

- Focus shifts from writing to talking
- Stories are understood by customer and developer
- At estimation time, they are the right size
- Participative design is occurring
- Emphasis is on the user goals, not the system's attributes

Given/When/Then Criteria

AS A user with a reservation, I WANT to cancel my reservation SO THAT I get a refund

Given I am a premium member, when I cancel under 24 hours, then I incur no penalty.

Given I am a non-premium member, when I cancel less than 24 hours in advance, then I pay a 50% fee.

Given I am a site member, when I cancel my reservation, then I am emailed a confirmation.

Adding Snappy Titles

Title: User Cancels Reservation

AS A user with a reservation, I WANT to cancel my reservation SO THAT I get a refund

Scenario 1: User is PREMIUM member

Given I am a premium member, when I cancel under 24 hours, then I incur no penalty.

Scenario 2: User is TYPICAL member

Given I am a non-premium member, when I cancel less than 24 hours in advance, then I pay a 50% fee.

Scenario 3: User gets email confirmation

Given I am a site member, when I cancel my reservation, then I am emailed a confirmation.

The Whole Story on a Card

Title: User cancels reservation

Description: As a user with a reservation, I want to cancel my reservation so that I get a refund.

Success Criteria:

- Given I am a premium member, when I cancel under 24 hours, then I incur no penalty.
- Given I am a non-premium member, when I cancel less than 24 hours in advance, I pay a 50% fee.
- Given I am a site member, when I cancel my reservation, then I am emailed a confirmation.

Story Owner:

Business Value Estimate:

Development Effort Estimate:

ROI Estimate:

Estimates Are Necessary

To plan and proceed deliberately

To get a feel for costs

To calculate potential ROI

To understand the size of something

To know if work even can be done

To weigh options

Deadly Estimation Warning Signs

Estimates are given without looking at historical performance.

Someone other than the team is doing the estimation.

Estimates are treated as promises.

Estimates are rejected because they don’t fit an already existing plan.

“I just want to know when it will be done.”

“That’s bigger than it should be.”

“That’s smaller than it should be.”

The Typical Estimation Process

PM: Hey Bill, how long to _____?

PM to self: They always say that. So, 2.5 weeks. I'll make it 3.

PM out loud: Thanks Bill. I'll go write the specs now.

Dev to Self: I'm busy, that'll take 2 days I can't afford to lose. What can I say to make him go away?

Dev out loud: About a week.

Dev to self: I can stall those out for weeks.

How do we measure software work?

Lines of Code

Buckets per day

Cycles Per Month

Mega Jewels per nanosecond

Coffees per day

Rotations Per Minute

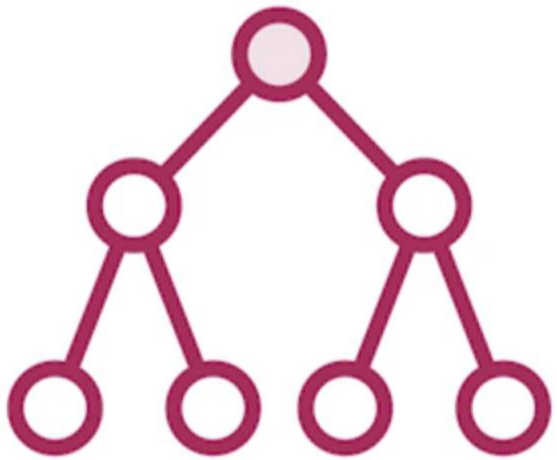
Feature Points

Kilowatts per hour

Hertz

Miles Per Hour

Story Points



- Very common way to estimate work
- Based on size and complexity, not duration
- Unitless and numerically relative
- Different for each team of estimators
- Points are additive, unlike time
- Based on historical reality
- Easy to use and understand

Using Story Points

Pile o'
User Stories

Defect A | Cost: 20

Defect B | Cost: 30

Requirement A | Cost: 100

Requirement B | Cost: 100

Requirement C | Cost: 30

Constraint A | Cost: 20

We can see right away

Which work items cost the most

Total cost of all the work

Total cost to an iteration

JIRA for Scrum

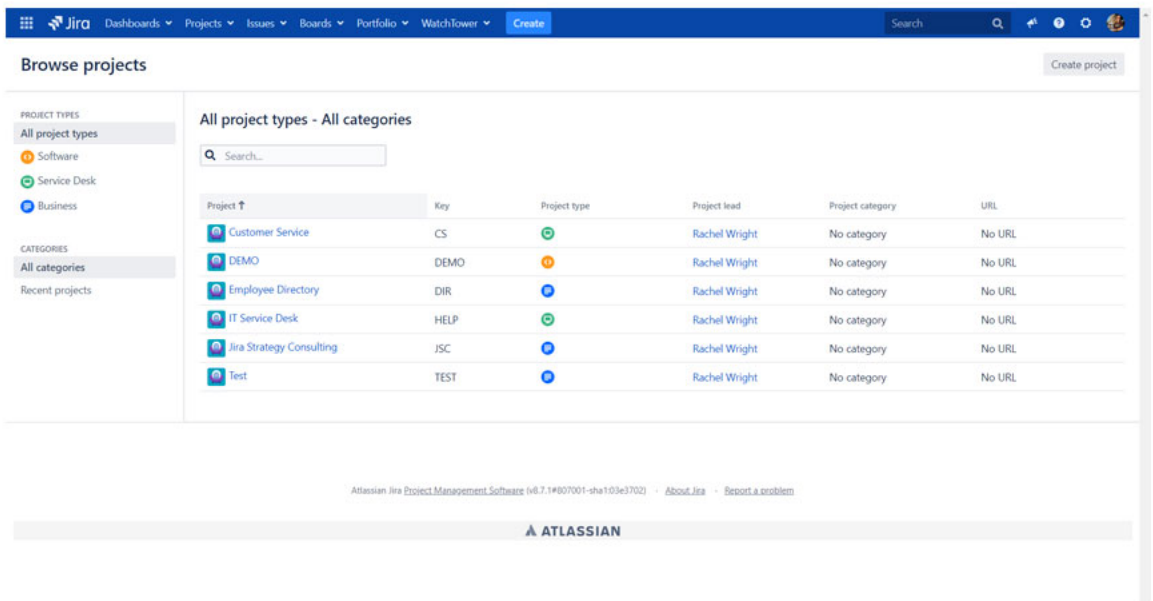
Issue tracker

Extensible platform

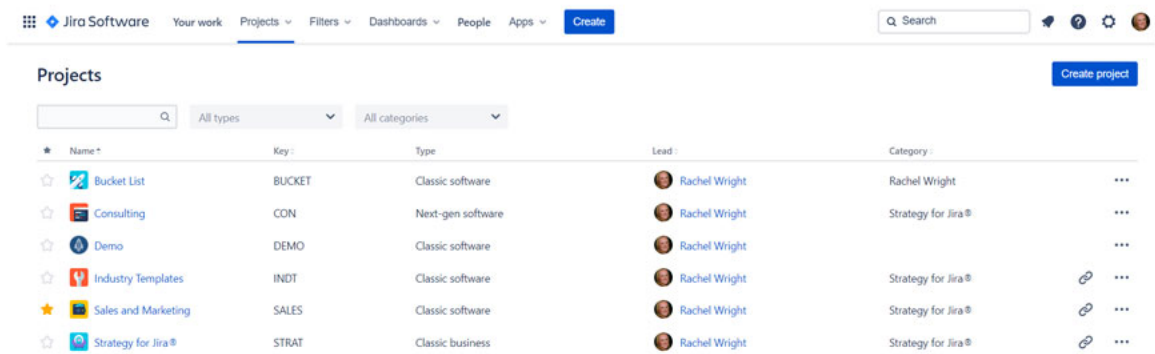
It just works!


VS

Jira Server



Jira Cloud



JIRA Core

JIRA Software

JIRA Service Desk

Web Page	https://www.atlassian.com/software/jira/core	https://www.atlassian.com/software/jira	https://www.atlassian.com/software/jira/service-desk
Before JIRA 7	JIRA	JIRA + JIRA Agile add-on	JIRA + JIRA Service Desk add-on
Use Cases	• General team project collaborations • Change Requests • Workflow Approvals • Task Management • On-boarding of employees • Tracking of advertising campaigns	• Bug Tracking • Product Management • Basic Software Development • Kanban Software Development • Scrum Software Development	• IT Helpdesk • Helpdesk Ticketing • Incident Management • Change Management • Problem Management
Targeted users	• Non technical teams • Business users • Legal teams • HR teams • Marketing teams • Finance teams • Operations teams	• Software developers • Scrum Masters • Project Managers • Quality Assurance Engineers	• Customers • Helpdesk Agents • Call Centre Managers • Non Technical users
Keywords	• Tasks • Approvals • Requests • Forms	• Backlogs • Epics • Sprints • Boards • Software development tool	• Tickets • Queues • Service Level Agreements (SLAs) • Knowledgebase integration

Bundled project templates	• **Business projects**
	  • Project Management
	  • Process Management
	  • Task Management
		• **Software projects**
		  • Scrum software development
		  • Kanban software development
		  • Basic software development
		• **Business projects**
		  • Project Management
		  • Process Management
		  • Task Management
			• **Service Desk projects**
			  • IT service desk
			  • Basic Service Desk
			• **Business projects**
			  • Project Management
			  • Process Management
			  • Task Management

	JIRA Core	JIRA Software	JIRA Service Desk
Simple business projects	✓	✓	✓
Workflow editing	✓	✓	✓
Powerful issue search	✓	✓	✓
Dashboards	✓	✓	✓
Basic reporting	✓	✓	✓
Agile boards		✓	
Backlog planning		✓	
Agile reports		✓	
Development tool integration		✓	
Release hub		✓	
Queues			✓
SLAs			✓
Confluence knowledge base integration			✓
Detailed service metrics			✓

Understand your deployment options

Cloud

- Get immediate access to our latest features with automatic upgrades
- No servers, no storage, no maintenance – we host your site for you
- Our comprehensive security, compliance, and privacy programs ensure your data is safeguarded
- Built on best-in-class core technologies and designed for strong performance
- Extend and customize your workflows with 1,000+ apps and integrations

Data Center

- Deploy on-premise or with IaaS vendors, like AWS and Azure, for complete control of your infrastructure and data
- Built to scale reliably as you grow with active-active clustering and an Atlassian-supported disaster recovery plan
- Advanced administrative tools and customization
- Easily manage planned and unplanned downtime with clustering, zero downtime upgrades and read-only mode

Server

- Host our products with your internal machines behind your firewall
- Complete customization and controls for system administrators
- Deploy the latest version when and where you want
- Manually implement upgrades when new product versions are released

References

- https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software
- https://www.atlassian.com/agile/scrum
- https://www.atlassian.com/agile/kanban
- https://www.atlassian.com/agile/project-management
- https://www.atlassian.com/agile/product-management

Questions?