Introduction to Agile & Scrum

Sury-D!

Introduction to Agile development & Scrum

70%. adopts Agile } as per PMI
77%. adopts Scrum }

25%. → Manufacturing companies use Agile & PMI

28%. more successful than traditional projects — Price waterhouse coopers

47% Agile transformation failure rate → Forbes
1# - Inexperience in implementing the methodology

"Agile is a mindset the requires culture change"

"Agile ~ driving a car" :)

"Recognizing when something is wrong is just as important as knowing how to do something right"!

"Pivot vs persevere"
(try something new) (keep doing the way you are)

Agile principles

Agile is an interactive approach to project management that helps teams be responsive and deliver value to their customers faster.

Defining characteristics:

→ Adaptive planning (once scope (or) big picture is known, we plan to deliver in steps to get feedback from customer)

→ Evolutionary development (again step by step)

→ Early delivery (initial product for feedback → pivot (vs) persevere)

→ Continual improvement ( )

→ Responsiveness to change ( )

Agile manifesto

→ Individuals & interactions over processes & tools.

→ Working SW over comprehensive documentation

→ Customer collaboration over contract negotiation

→ Responding to change over following a plan.

(while value in right is important, we value the items in left more)

Agile SW development

→ An iterative approach to SW development consistent with the agile manifesto.

→ Emphasizes flexibility, interactivity and a high level of transparency

→ Uses small, co-located, cross-functional, self organizing teams.

→ Build what is needed, not what is planned

— x —

Methodologies Overview

Traditional waterfall Development: (entry-exit criteria for every phase)

Requirements → Design phase → Implement/Develop/Code → Integration → Testing → Deployement

(Step by step process)

→ No provisions for changing requirements
→ No intermediate delivery, so no idea if it works
→ Each step ends when the next begins
→ Mistakes found in the later are more expensive to fix.
→ long lead time for the sw.
→ Teams work seperately, unaware of impact on each other.
→ People who know least about the code are deploying it into production

Extreme programming (XP)

→ 1996 → Kent Beck

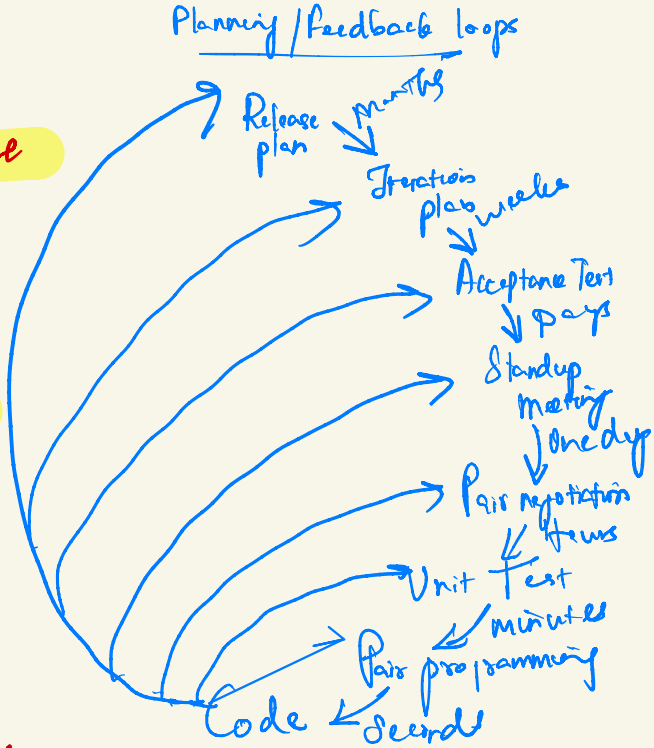
→ based on an interactive approach to SW development

→ Intended to improve SW quality & responsiveness to changing customer requirements.

→ One of the first agile methods

Values

- Simplicity
- Communication
- Feedback
- Respect
- Courage



Kanban (1970's - Tyoto) - billboard

→ A Japanese manufacturing system in which the supply of components is regulated through the use of an instruction card set along the production line.

Core principles

→ Visualize the workflow
→ Limit work in progress (WIP) (having focused team to perform the tasks needed)
→ Manage & enhance the flow.
→ Make process policies explicit. (everyone understands the definition of done)
→ Continously improve.

— x —

Working Agile:

→ Working is small batches (inspired from lean manufacturing)
→ Minimum Viable product (MVP)
→ Behaviour Driven Development
→ Test Driven Development
→ Pair programming

Working in small batches : → Delivering something useful quickly

It is better to work in small batches to know if we are making any mistakes at each phase of development, so we can adjust and continously improve based on the feedback recieved.

Minimum Viable product:

→ MVP is not the result of "phase 1" of a project.

→ MVP is the cheapest/easiest thing you can build to start testing your value hypothesis and learning.

→ The former focuses on delivery while the latter focuses on learning.

→ At the end of each MVP, you decide either pivot or persevere.

Behavior Driven Development (makes sure you are building the right thing)

→ Describes the behaviour of the system from the outside in

→ Used for integration testing. (against user/customer requirements)

→ Uses a single syntax to describe user stories and feature scenarios that both stakeholders and developers can understand.

Gherkin Syntax (from "cucumber" company) → A single syntax that developers and stakeholders can understand, given some set of preconditions when an event occurs, then some outcome is observed.

BDD feature scenarios :

As a < role >
I need < some functionality
so that < some business value >

Scenario:

Given a set of precondition
when an event occurs
Then some outcome is observed

Test driven Development : (makes sure you are building the thing right)

→ Tests the functions of the system from the inside out
→ Test cases drive the design.
→ Unit testing.

Workflow :

→ Write a test case and watch it fail
→ Write just enough code to make it pass
→ Refactor the code to make it better & repeat
→ Red, Green, Refactor.

Pair programming :-

→ Two programmers work together at one workstation
→ One writes the code
→ The other reviews each line of code as it is typed in
→ The two programmers switch roles frequently Eg - 20 mins
→ Code quality increases as defects are found earlier

Introduction to Scrum Methodology :-

Scrum methodology :-

Agile & Scrum :

Agile is a philosophy for doing work (not prescriptive)
Scrum is a methodology for doing work (adds process) in Agile fashion

Scrum :-

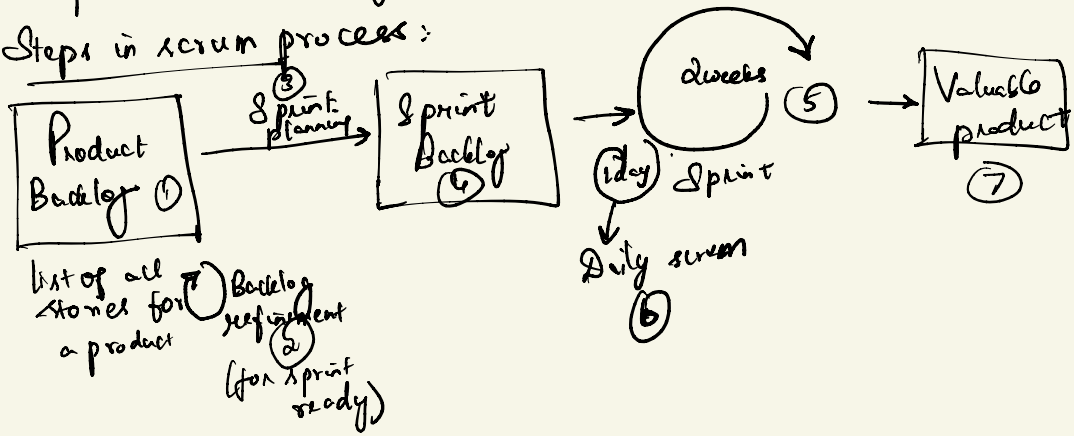
→ Is a management framework for incremental product development
→ Prescribes small, cross functional, self organizing teams
→ Provides a structure of roles, meeting, rules & artifacts
→ Uses fixed-length iterations called sprints
→ Has a goal to build a potentially shippable product increments with each iteration.

"Easy to understand ~~Difficult to~~ Master"!
_like a muscle memory

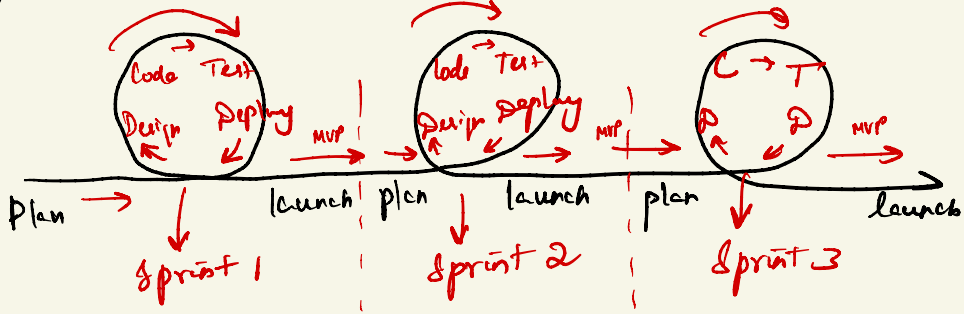
Sprint :-

→ A sprint is one iteration through design, code, test, and deploy cycle.
→ Every sprint should have a goal (or value)
→ Sprints are usually 2 weeks in duration (depends on project)

Steps in scrum process :



Agile development is iterative



Roles of Scrum :-

- → Product Owner
- → Scrum master
- → Scrum Team (whoever participates in prod. development)

Product Owner: (represents stakeholder, articulates the product vision & decides priorities - requirement, and readiness to ship)

- → Represents the stakeholder interests.
- → Liaison between Scrum team & stakeholder and their interest.
- → Articulates a product vision.
- → Is the final arbiter of requirements questions
- → Constantly re-prioritizes the product backlog, adjusting any expectations.
- → Accepts (or) rejects each product increment
- → Decides whether to ship
- → Decides whether to continue development (pivot vs persevere)

Scrum Master (coaches the team, promotes a co-operative environment & shields the team from interference, and unblocks impediments)

- → Facilitates the scrum process
- → Coaches the team.
- → Creates an environment to allow the team to be self organizing.

→ Shields the team from external interference to keep it "in the zone". (eg from stakeholder questions)
→ Helps resolve impediments.
→ Enforces sprint timeboxes (how many mins for meeting)
→ Captures empirical data to adjust forecasts.
→ Has no management authority over the team

Scrum Team (small, dedicated, co-located, cross-functional & self managing)

→ A cross functional Team consisting of Developers, testers, Business analysts, Domain experts, others.

→ Self organizing: There are no externally assigned roles.

→ Self managing: They self assign their own work. (from kanban — board for eg)

→ Member: Consists of 7 ± 2 collaborative members

→ Co-located: Most successful when located in one team room, particularly for the first few sprints. (collaborate well)

→ Dedicated: Most successful with long-term, full-time members on specific project (not on multiple projects)

→ Negotiates commitments with the product owner — one sprint at a time.

→ Has autonomy regarding how to reach commitments.

Artifacts, Events, Benefits

Artifacts:

Product Backlog	Sprint backlog	Done Increment
→ All the stories for a product. → Any stories not in the current sprint, are in a product backlog → Some teams have a product backlog, an icebox & a release backlog (depends)	Stories that you're going to do in next 2 weeks, in the next sprint	→ product increment completed by end of sprint. → Should have a value

Events:

→ Sprint planning meeting (planning the sprint with - P.O, S.M & S.T to decide what to do next)

→ Daily scrum meeting (every day standup)

→ Sprint (2 week window)

→ Sprint review (demonstrate your increment to the stakeholder)

→ Sprint retrospective (what went well, what didn't go well, what can we change in future)

Benefits of Scrum

(if done well)

→ Higher productivity

→ Better SW Quality (TDD & BDD pair programming)

→ Reduced time to market

→ Increasing Stakeholder satisfaction

→ Better team dynamics

→ Happier employees

Scrum	Kanban
Cadence: Fixed length sprints	Continuous flow
Release methodology: End of each sprint	Continuous delivery
Roles: Product owner, Scrum master, Dev Team	No existing roles (optional agile coach)
Key metrics: Velocity	Key metrics: Cycle Time
Change philosophy: Team should strive to not make changes to the sprint forecast during the sprint	Change can happen at any time

Organizational impacts

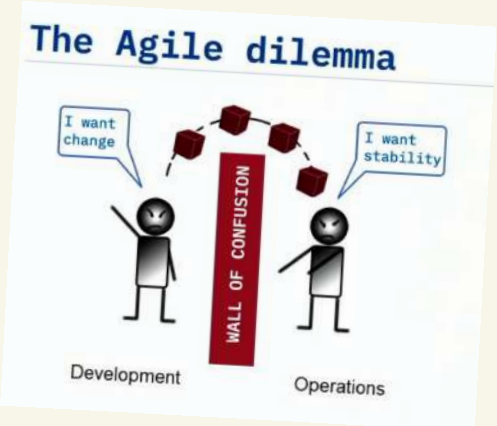
→ Proper organization is critical to success
→ Existing team needs to be reorganized

How teams should be aligned

→ Loosely coupled, tightly aligned.
(not lot of dependencies)
→ Each team has its own mission aligned with the business. (like a mini startup)
→ Team have end-to-end responsibility for what they build.
→ The long-term mission is usually around a single business mission.

Autonomy is important.

→ It's motivating — motivated people build better stuff
→ It's fast — decisions happen locally within a team
→ It minimizes handoffs & waiting, so teams don't get bogged down.



→ Conflicting views in organization will not yield success
→ Agile is a mindset where the culture of whole organization must change

Agile + Devops = Alignment
(DEV+OPS)

Agile Goals	Devops Goals
Develop SW faster ✓	Accelerate time to market ✓
Be responsive to changes	Improve IT's value by more closely aligning development, IT operations and the business
Obtain higher quality results ✓	Increase IT productivity ✓

Mistaking iterative for agile :

→ Agile is not a new version of a waterfall SW development Life Cycle (SDLC), where you do legacy development in sprints

→ Agile is not just the developers working in each sprint, it involves a cross functional team.

→ The agile manifesto does not include the term "Agile project management". (no project managers in agile)

→ Here many companies adhere to their waterfall planning & call it agile.

→ Simply doing iterative development is not agile unless you are being responsive to changes & delivering value often.

→ Project management is very different when working as agile team because self-managing teams assign work to themselves.