Agile Practice Guide

Freitag, 28. November 2025 18:42

General Ideas

- Agile is a part of Lean
- Agile Methods work best for uncertain, fast changing work
- Uncertainty has the two degrees of Requirements Uncertainty and Technical Uncertainty

Definable Work:	• Routine, predictable tasks • e.g. Production processes
Uncertain Work:	• Unpredictable tasks with changing requirements • e.g. Software engineering

4 Core Values

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

12 Principles of the Agile Manifesto

1. Customer satisfaction is the main priority
2. Change in requirements is welcome
3. Frequent delivery is key
4. Collaboration on a daily basis
5. The environment should support motivation
6. Communication should be Face-to-face
7. Progress is measured by working software
8. A pace of development should be sustainable
9. Focus on technical excellence and good design
10. Simplicity: only doing what is necessary
11. Teams need to be self-organizing
12. Regularly reflect on how to improve effectiveness

4 Types of lifecycles

Different types of ways to approach projects, depending on frequency of delivery and degree of change.

Life cycles each have the stages: Analyze, Design, Build, Test & Deliver
The types differ from how often they are performed and if there are feedback loops between them and during them.

Predictive	**Iterative**
• Low frequency of delivery • Low degree of change • Fixed plan and scope • One fixed delivery at the end • e.g. construction	• Low frequency of delivery • High degree of change • Some big changes but only after long cycles • Design and research projects • e.g. developing a new car design

Incremental	**Agile**
• High frequency of delivery • Low degree of change • Small steps towards a bigger goal • e.g. releasing software modules	• High frequency of delivery • High degree of change • Small deliverables in short time intervals • e.g. developing an app or technology

Flow based approaches do manage the increments by limiting the Work in Progress as well.

Implementing Agile

Servant Leadership

= a way to empower teams.

- Shift from managing organization to facilitating collaboration
- Build a team of motivated people. Create an environment to support them to get the job done.

Purpose:	Defining the "why" with the team.
People:	Creating an environment where the people can succeed.
Process:	Reflecting on processes but they don't need to be perfect.

Tasks of a Servant Leader:

- Promoting self-awareness
- Listening
- Serving those on the team
- Helping people grow
- Coaching
- Promoting safety, respect, and trust
- Promoting the energy and intelligence of others

Attributes of successful Agile Teams

- Dedicated people
- Cross-functional team members
- Colocation or ability to manage
- any location challenges
- Mixed team of generalists and
- specialists
- Stable work environment

Agile Roles

Cross-functional team members:	• Team members with all types of skills • Designers, developers, testers, …
Product owner:	• Guiding the direction of the product • Collect feedback from customers and teams on the product • Create the backlog
Team facilitator:	• Servant Leader • project manager, scrum master, project team lead or team coach

Agile Environments

Project Charter

A project charter needs to include:

- Vision: why a project is done?
- Who benefits and how?
- Definition of Done
- A definition on how to work together

As long as the team understands this, there is no need for a rigid process for chartering.

Retrospectives

= regular reflection on how the team can become more effective

- Retrospect after a release, after a few weeks, when stuck or after a milestone
- Do not blame
- Find root causes of problems and adjust behavior

Backlogs

= A list of all the work to be done

- User Stories: Descriptions of product properties and functions in the form of stories
- Impact Mapping: The practice of comparing stories with their impact on the user experience to prioritize

Daily Standups

= 15min daily meetings to present current progress

- Everyone presents
  - What they did?
  - What they do next?
  - Which problems and risks they face?

- The teams discusses ways to advance the project together

Systems that help Agile practice

Continuous Integration	Merge work frequently into main branch Retest to ensure system still works
Test at All Levels	Use unit, integration, and system tests Run smoke and regression tests as needed
Acceptance Test-Driven Development (ATDD)	Define acceptance criteria as a team Write tests first, code just enough to pass
Test-Driven Development (TDD) & Behavior-Driven Development (BDD)	Write tests before building Helps design and prevent defects
Spikes	Timeboxed research tasks Used for learning, estimating, or clarifying requirements

Measurement in Agile Projects

- Traditionally "**Traffic Light Measurement**" is used
  - Dividing the project into small steps
  - Assigning them times when they should be finished
  - Defining buffer to each increment
  - Tracking the progress.
    If the project is as expected, it is "green"
    If the buffer is being used unexpectedly it is "orange"
    If the buffer is exceeded, it is "red"
  - Progress is tracked visually in charts (Burndown or Burnup charts)
- Tracked are: Number of features, time passed, costs, ROI, WIP
- Baseline for measurement is estimates earned value per increment or Return on Invest (ROI)

- **Kanban Board:**
  =a visual tool to manage work.

Lean and Agile Frameworks

Scrum	• Single-team process. • Manages product development. • Timeboxes of 1 month or less. Called sprints. • Each sprint produces a potentially releasable increment of product. • Product owner maximizes product value. • Development team is cross-functional and self-organizing. • Team delivers working product. • Scrum master ensures Scrum process is upheld.
Extreme Programming (XP)	• Based on frequent cycles. • Distills best practices to simplest form. • Applies practice continuously throughout project. • Includes pair programming. • Focuses on collaboration and improvement.
Kanban Method	• Does not prescribe timeboxed iterations. • Teams not bound by timeboxes. • Work on highest priority item in backlog. • Focus on continuous delivery. • Emphasizes flexibility. • Increases productivity and quality.
Crystal Methods	• Family of methodologies. • Designed to scale. • Adapts to project size and criticality. • Adjusts rigor based on number of people and project importance.
Feature-Driven Development (FDD)	• Developed for large software development projects. • Six primary roles: project manager, chief architect, development manager, chief programmer, class owner, domain expert. • Based on five iterative processes: • develop an overall model • build a feature list • plan by feature • design by feature • build by feature.
Dynamic Systems Development Method (DSDM)	• Focuses on constraint-driven delivery. • Cost, time, and quality fixed at outset. • Eight guiding principles: • focus on business need • deliver on time • Collaborate • never compromise quality • build incrementally from firm foundations • develop iteratively • communicate clearly • demonstrate control.
Agile Unified Process (AgileUP)	• Offshoot of Unified Process for software projects. • Based on accelerated cycles. • Less heavyweight processes. • Focuses on iterative feedback. • Principles: • Simplicity • Agility • focus on high-value activities • tool independence • tailoring to fit • situationally specific.

Scaling Frameworks

Scrum of Scrums (SoS)	• Also called meta Scrum. • Two or more Scrum teams coordinate work instead one large team. • Each team sends a representative to regular meetings. • Ensures coordination and removal of impediments between teams. • Enables large-scale Scrum collaboration.
Scaled Agile Framework (SAFe)	• Scaling development work across the enterprise. • Taking an economic view • Applying systems thinking • Assuming variability

	○ Tasks in columns like **To Do**, **In Progress**, and **Done**. ○ Tasks get moved from one side to the other. ○ Work in Progress (WIP) is controlled by the number of tasks per column		• Building incrementally with fast learning cycles • Basing milestones on objective evaluation • Visualizing and limiting WIP • Reducing batch sizes, • Managing queues, • Applying cadence, • Synchronizing cross-domain planning, • Unlocking worker motivation, • Decentralizing decision making.
		Large Scale Scrum (LeSS)	• Extends Scrum to multiple development teams. • One product backlog and shared definition of done. • Formal split of sprint planning • Organic cross-team coordination • Overall cross-team refinement • Retrospectives focused on cross-team improvements.
		Disciplined Agile (DA)	• Process decision framework integrating agile best practices. • People-first with defined roles at multiple levels. • Learning-oriented with collaborative improvement. • Full delivery life cycle for fit-for-purpose results. • Goal-driven process tailoring. • Enterprise awareness for governance. • Scalable across complex programs.