TIME: DATE: / /

Different Agile Approaches and Their Comparative Analysis:

Agile software development is an iterative and incremental approach that emphasizes flexibility, collaboration and customer feedback. Various frameworks and methodologies fall under Agile, each with distinct features.

Below is a comparative analysis of some popular Agile approaches:

① Scrum:

How it works:

(i) In Scrum, the work is divided into fixed-length iterations called sprints (typically 2-4 weeks)

(ii) Teams hold regular stand-up meetings and review progress in ~~spir~~sprint planning, reviews and retrospectives.

(iii) A 'Product Owner' manages the backlog and a Scrum Master facilitates the process.

Applicability:

(i) Best for projects with rapidly changing or unclear requirements.

(ii) Suitable for software development, marketing campaigns or product design.

Effectiveness in Terms of Costs:

Moderate: Focuses on frequent delivery, reducing wasted efforts. However, sprints require experienced teams to avoid overruns.

Example

A mobile app development company uses Scrum to deliver incremental app features every two weeks, allowing clients to review progress and request changes.

② Kanban:

How it works:

(i) Visualize workflows using a Kanban board with columns representing stages (e.g. "To Do", "In Progress", "Done").

(ii) Focuses on limiting work in progress (WIP) to optimize task flow.

Applicability:

(i) Ideal for teams working on continuous processes, such as operations, support or maintenance projects.

(ii) Flexible for tasks that don't have fixed iterations or timelines.

Effectiveness in Terms of Costs:

Low to Moderate: Reduces bottlenecks and increases team efficiency but may require investments in tools or training.

Example

An IT support team uses Kanban to manage tickets. As tickets move through stages (e.g. 'Received', 'Investigating', 'Resolved') the team visualizes progress and identifies bottlenecks.

③ Extreme Programming (XP)

How it Works:

(i) Focuses on engineering practices like pair programming, test-driven development (TDD) and frequent releases.

(ii) Prioritizes continuous customer feedback and refactoring

Applicability:

(i) Best for teams in software development with high technical complexity and the need for high-quality code.

(ii) Effective when requirements evolve rapidly.

Effectiveness in Terms of Costs:

Moderate to High: High focus on quality reduces long-

term costs but require significant upfront investment in practices like TDD.

Example

A startup uses XP to build a SaaS application. The team delivers working features every week while continually refactoring the codebase to maintain quality.

④ Lean Software Development:

How it works:

(i) Adapts Lean manufacturing principles to software development, focusing on eliminating waste, delivering quickly and continuous learning.

(ii) Encourage empowering the team and deferring decisions until necessary.

Applicability:

(i) Useful for startups or companies with tight budgets looking to maximize value with minimal resources.

(ii) Fits projects requiring fast prototyping and delivery.

Effectiveness in Terms of Costs:

Low to Moderate:

Cost-effective as it eliminates waste and focuses on only building what is necessary.

Example

An e-commerce company applies Lean principles to launch a Minimum Viable Product (MVP) in six weeks, gathering customer feedback to guide further development.