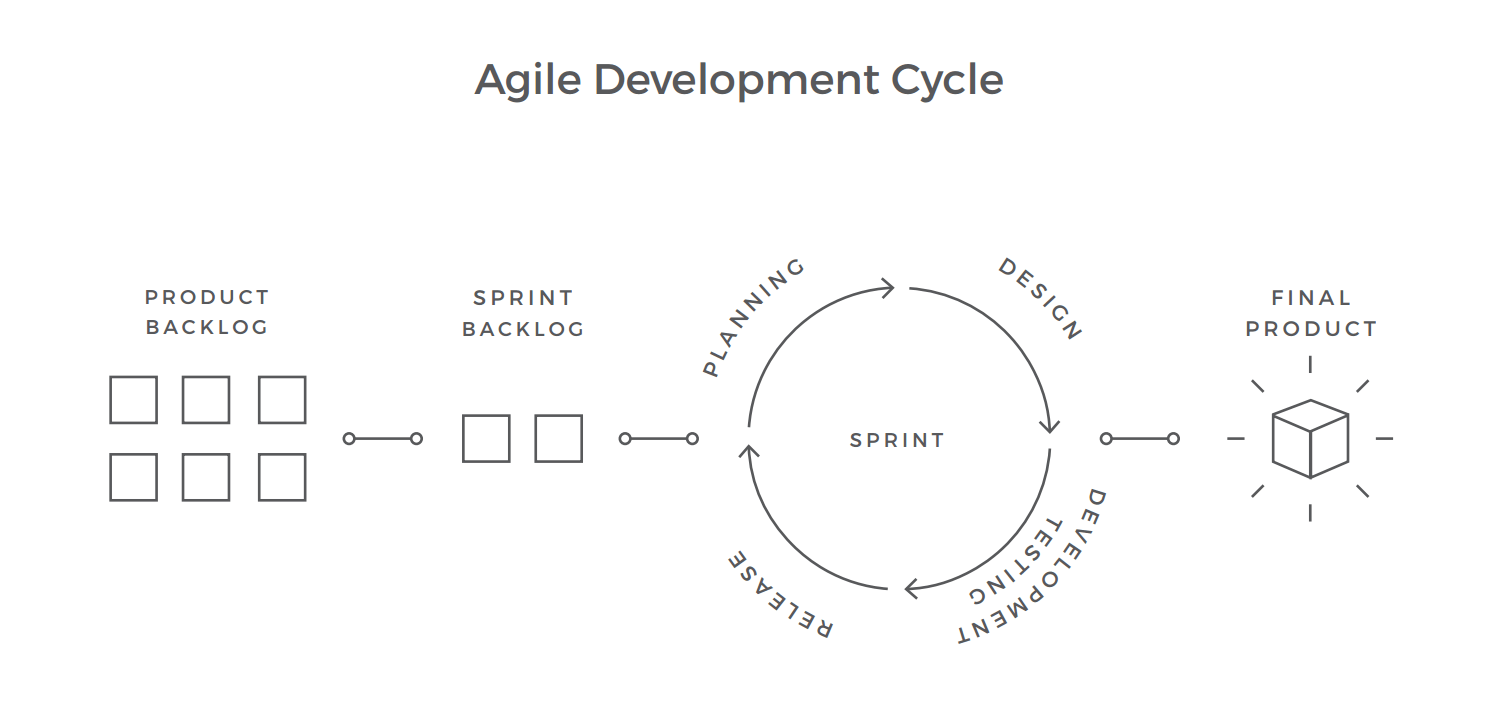
Q.1) List out the key principles in Agile Development model and Explain about different methodologies in Agile Software Development?

Answer:

Introduction to Agile Development

Agile software development (Agile) is a collection of software development methodologies that promote adaptive planning, evolutionary development and delivery, continuous improvement, and a time-boxed period of time to complete a body of work. Software development is dynamic by nature, and Agile encourages rapid and flexible response to change. Because adaptability is central to its conceptual framework, teams using this approach are well-equipped to respond to changes throughout the development cycle.



Values

Emerging from the Agile Manifesto were the following set of core values:

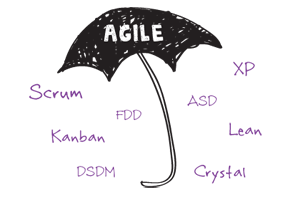
* Individuals and interactions over processes and tools
* Working software over comprehensive documentation
* Customer collaboration over contract negotiation
* Responding to change over following a plan

The following principles are based on the Agile Manifesto:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Differrent methodologies in Agile Software Development:

Agile is an umbrella term for a vast variety of methodologies and techniques, sharing the principles and values. Each of them has its own areas of use and distinctive features. The most popular frameworks are Scrum, Kanban, Hybrid, Lean, Bimodal, and XP.



1. Scrum:

• The entire scope of work is broken down into short development

cycles — Sprints.

• The Sprint's duration is from one to four weeks.

• The team should strictly follow a work plan for each Sprint.

• People involved in a project have predefined roles.

It is based upon the systematic interactions between the three major roles: Scrum Master, Product Owner, and the Team.

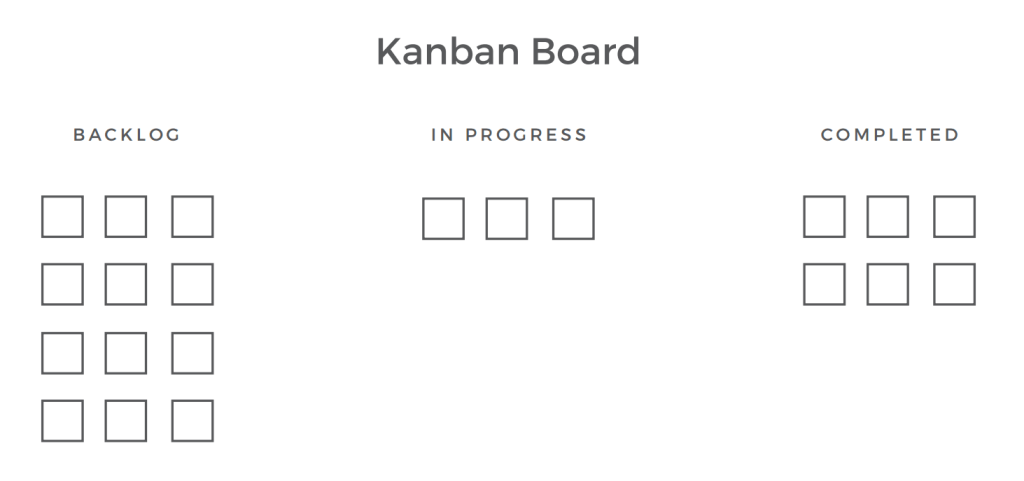
* Scrum Master is a central figure within a project. His principal responsibility is to eliminate all the obstacles that might prevent the team from working efficiently.
* Product Owner, usually a customer or other stakeholder, is actively involved throughout the project, conveying the global vision of the product and providing timely feedback on the job done after every sprint.
* Scrum Team is a cross-functional and self-organizing group of people that is responsible for the product implementation. It should consist of up to 7 team members, in order to stay flexible

**When to Use Scrum**

Scrum works well for long-term, complex projects that require stakeholder feedback, which may greatly affect project requirements. So, when the exact amount of work can’t be estimated, and the release date is not fixed, Scrum may be the best choice.and productive.

1. Kanban: Comprehensive Solution to Handling Work in Progress

Forty three percent of companies have stated that they use Kanban as one of the project management frameworks. Originating from a visual system of cards used in Toyota manufacturing as a production control method, Kanban is simple, yet powerful, approach to developing software products.



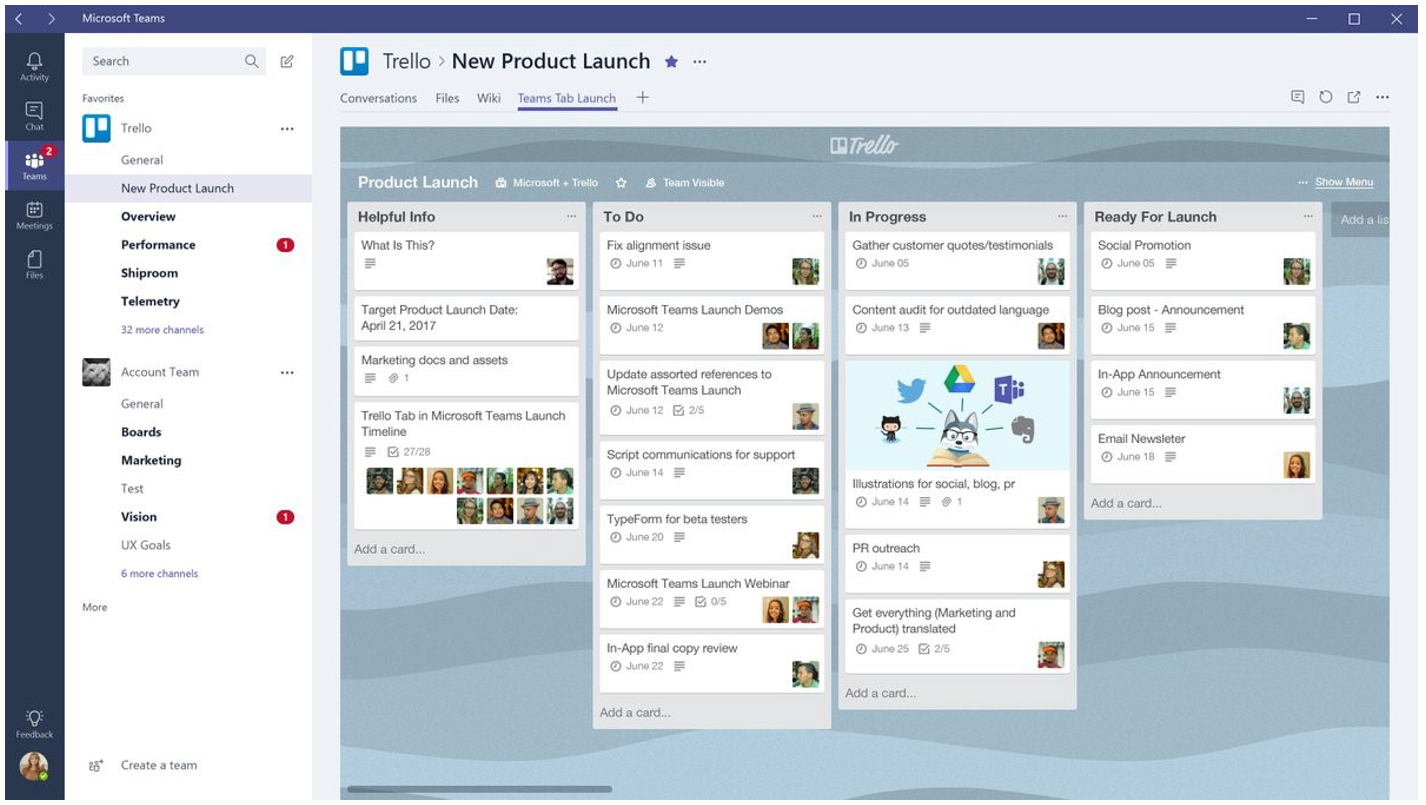
• Development is built on workflow visualization.

• The current work (work in progress or WIP) is prioritized.

• There are no timeboxed development cycles.

• The team can change the work plan at any time.

No standard procedures within the process, as well as the fixed iterations, are required in Kanban, as opposed to Scrum. The project development is based on the workflow visualization through a Kanban board, usually represented by sticky notes and whiteboards or online tools like Trello.



When to Use Kanban:

Kanban is focused on doing small pieces of work as they come up. For example, if testers find errors in the product, developers try to fix them right away. Kanban, for instance, works well after the main release of the product.

1. Hybrid: Blend of Waterfall and Agile (Flexible Development and Thorough Project Planning)

• Agile and Waterfall complement each other.

• Agile software development is held under Waterfall conditions

(fixed deadline, forecasted budget, and thorough risk assessment).

For example, project planning can be done in sprints, testing can be incorporated in development, and feedback can be gathered regularly.

When to use Hybrid:

planning, requirements specification, and an application design of the project is not done properly.

1. Bimodal

• There are two separate modes of work — traditional (Mode 1) and

Agile (Modo 2).

• Two separate teams are working on projects with two different

goals.

• The Mode 1 team maintains IT system infrastructure.

• The Mode 2 team delivers innovative applications.

• Cross-team collaboration is important

When to use Bimodal:

If the company specializes in both long- and short-term projects that require different development and management approaches, Bimodal might be the right choice. This framework is about keeping the balance between maintaining IT system infrastructure and driving innovations.

1. Lean: Eliminating Waste in Software Engineering

• The framework promotes fast software development with less

effort, time, and cost.

• The development cycle is as short as possible.

• The product delivered early is being continuously improved.

• The team is independent and has a wider range of responsibilities

than those in Scrum, Bimodal, and Hybrid

• Developers can also formulate the product's concept

The application of Lean principles to software development was initially introduced by Mary and Tom Poppendieck in their book Lean Software Development: An Agile Toolkit. It includes the 7 basic principles:

1. Eliminate waste
2. Amplify learning and create knowledge
3. Decide as late as possible
4. Deliver as fast as possible
5. Empower the team
6. Build integrity/quality in
7. See the whole

When to Use Lean:

Lean works well for small, short-term projects due to their short life cycles. This approach is also appropriate if the customer can participate in a project realization as Lean requires ongoing feedback. Another important condition to the adoption of Lean is the whole team should work in one office to enable communication.

1. XP: Engineering Practices For Writing A Good Code

• The focus is on technical aspects of software development.

• XP introduces engineering practices aimed at helping developers

write a clear code.

• Product development includes consistent stages: coro writing,

testing, analyzing, designing, and continuous integration of code.

• Face-to-face communication within the team and customer

involvement in development are crucial.

When to Use XP:

Extreme Programming offers engineering practices and ideas that help development teams adapt to ever-changing requirements. The key features of this framework are a high rate of customer engagement and short iterative cycles that don’t exceed one week. Also, XP suggests developers make the simplest design possible and prioritize tasks.

The customer role evolves: it defines a Product Backlog and works together with a Development Team in the office until the project ends.

1. FDD: Feature Driven Develpoment

* FDD is model-driven, short-iteration process.
* It begins with establishing an overall model shape.
* FDD designs the rest of the development process around feature delivery using the following eight practices:
* Domain Object Modeling
* Developing by Feature
* Component/Class Ownership
* Feature Teams
* Inspections
* Configuration Management
* Regular Builds
* Visibility of progress and results

Conclusion

The Agile approach is often mistakenly considered to be a single methodology. Yet, there are dozens of methodologies and certain practices that have not been touched upon in this research.

References

1. <https://www.altexsoft.com/whitepapers/agile-project-management-best-practices-and-methodologies/>
2. <https://www.versionone.com/agile-101/agile-methodologies/>
3. <https://www.aha.io/roadmapping/guide/agile-development>