The Agile Manifesto and Mindset Study Guide

Readable Exam Notes

Table of Contents

1. The Four Agile Values & Manifesto

The Agile Manifesto highlights four key value pairs. Values on the left are preferred over those on the right, though the items on the right still have value in context.

Agile Value (Preferred)	Over
Individuals and interactions	Processes and tools
Working software	Comprehensive documentation
Customer collaboration	Contract negotiation
Responding to change	Following a plan

---

2. The Agile 12 Clarifying Principles

1. Satisfy the customer through early and continuous delivery of valuable software.

2. Welcome changing requirements, even late in development.

3. Deliver working software frequently, with a preference for shorter timescales.

4. Business people and developers must work together daily.

5. Build projects around motivated individuals and give them the environment and support they need.

6. The most efficient communication is face-to-face conversation.

7. Working software is the primary measure of progress.

8. Promote sustainable development at a constant pace.

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. Regularly reflect and adjust team behavior to improve effectiveness.

3. Life Cycle Selection and Comparison

Project life cycles differ based on requirement stability, complexity, and risk. Use predictive approaches for stable work and agile approaches for high change and uncertainty.

3.1 Life Cycle Comparison

Approach	Requirements	Activities	Delivery	Goal
Predictive	Fixed	Performed once for entire project	Single delivery	Manage cost / scope certainty
Iterative	Dynamic	Repeated until solution is correct	Single delivery	Correctness of solution
Incremental	Dynamic	Performed per increment	Frequent small deliveries	Speed of delivery
Agile	Dynamic	Repeated with feedback	Frequent small deliveries	Value delivery via feedback

Predictive / Waterfall Approach

- Plans are completed up front and executed sequentially.
- Best for high-certainty requirements and low expected change.
- Typical flow: Analyze → Design → Build → Test → Deliver.

Iterative Life Cycle

- Feedback is used on partially complete work to refine the solution.
- Optimized for learning and improving correctness over time.
- Useful when scope or requirements are evolving.

Incremental Life Cycle

- Delivers finished increments the customer can use immediately.
- Supports frequent small releases, one slice of value at a time.

Agile Life Cycle

- Combines iterative learning with incremental delivery.
- Feedback from each iteration drives planning for the next.
- Often uses Kanban boards and WIP limits to manage flow.

Hybrid Approaches

- Many real projects mix predictive and agile approaches.
- Examples: predictive contracts with agile delivery teams; predictive phases followed by agile iterations.

4. Definable Work vs. High-Uncertainty Work

Work can be thought of as a spectrum between clearly definable and highly uncertain. Choose a life cycle that matches where your project sits on this spectrum.

Definable Work

- Clear, repeatable procedures are known in advance.
- Similar to past projects (e.g., manufacturing, standard deployments).

High-Uncertainty Work

- High rate of change, complexity, or novelty (e.g., product design, R&D).
- Requirements and solutions evolve as the team learns more.

---

5. Agile Core Practices

5.1 Whole Team Approach

- Everyone needed for success is included on the team.
- Teams are small (often 3–9 people) and ideally co-located.
- High team dedication improves focus and throughput.
- Cross-functional teams deliver releasable increments regularly.
- Key roles:
- Product Owner – represents the customer and prioritizes the backlog.
- Team Facilitator / Scrum Master – servant leader who removes impediments.

5.2 Early and Frequent Feedback

- Short iterations deliver working features regularly.
- Stakeholders provide feedback early and often.
- Reduces rework and clarifies requirements.

5.3 Rolling Wave Planning

- Near-term work is planned in detail; future work remains high-level.
- Plans are progressively elaborated as information becomes available.
- Large features are broken down closer to when they will be done.

5.4 Daily Stand-Up

A short, timeboxed meeting (often 15 minutes) for the team to synchronize and highlight impediments.

- Typical questions:
- What did I complete since the last stand-up?
- What am I working on next?
- What blockers or impediments do I have?

5.5 Retrospectives

- Held at the end of an iteration to reflect on how work was done.

- Identify what went well, what did not, and what to change next time.
- Drives continuous improvement in both process and teamwork.

5.6 Release Planning vs. Iteration Planning

- Release Planning:
- Sets goals and scope for a group of iterations or a release.
- Involves business stakeholders and product management.
- Iteration Planning:
- The team selects and estimates user stories from the backlog.
- Scope is based on team capacity and historical velocity.

5.7 Collaborative User Story Creation

- Stories are created collaboratively by developers, testers, and business representatives.
- Cover functional / non-functional requirements and acceptance criteria.
- Follow the 'Three Cs': Card, Conversation, Confirmation.

5.8 Demonstrations / Reviews

- At the end of each iteration, teams demo completed features to stakeholders.
- Enables continuous feedback and alignment with customer needs.

5.9 Continuous Integration & Agile Testing

- Integrate and test changes frequently—ideally daily.
- Use automated builds and tests to catch defects early.
- Test at multiple levels: unit, integration, system, UAT, ATDD, TDD, BDD.
- Use spikes for research or experimentation when needed.

5.10 Servant Leadership

- Leaders focus on serving the team, not controlling it.
- Emphasize purpose (why), people (support), and process (improvement).
- Encourage self-organization and shared ownership of outcomes.

---

6. References & Further Study

For deeper learning, review the Agile Manifesto, the 12 principles, and topic-specific material on:
• Whole Team Approach
• Early & Frequent Feedback
• Rolling Wave Planning
• Daily Stand-Ups
• Release and Iteration Planning
• Collaborative Story Writing
• Continuous Integration and Agile Testing