ECE 811 – SOFTWARE ENGINEERING

AGILE METHODOLOGY – BEGINNERS'S STUDY GUIDE/REVISION

---

1. WHAT IS AGILE METHODOLOGY?

1. **Definition**
   Agile is an iterative approach to software development that emphasizes flexibility, collaboration, and customer feedback. Instead of a rigid plan, work is broken into small increments delivered in short cycles (iterations/sprints).
2. **Core Idea**

- Adapt to change over following a fixed plan.
- Deliver working software frequently (weeks, not months).
- Collaborate closely with customers and teams.

2. AGILE MANIFESTO: FOUR 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.

3. KEY FEATURES OF AGILE METHODOLOGY

1. **Iterative & Incremental Development:**
   - **Feature:** Projects are broken down into small, manageable units of work (often called iterations, sprints, or increments).
   - **Benefit:** Delivers a potentially shippable piece of working software at the end of each iteration (typically 1-4 weeks). This allows for early and frequent delivery of value, reduces risk, and enables continuous learning and adaptation.
2. **Customer Collaboration & Feedback:**
   - **Feature:** Continuous, close collaboration with the customer/product owner throughout the project lifecycle, not just at the beginning and end.
   - **Benefit:** Ensures the product evolves to meet the customer's *actual* and *current* needs, incorporates feedback rapidly, and delivers higher satisfaction. Replaces rigid contracts with collaborative partnerships.
3. **Adaptability & Embracing Change:**

- **Feature:** Welcomes changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- **Benefit:** Allows the team to respond effectively to market shifts, new information, or evolving user needs without derailing the entire project plan.

4. **Self-Organizing & Empowered Teams:**
   - **Feature:** Cross-functional teams (developers, testers, designers, etc.) organize themselves around the work. They decide *how* to best accomplish tasks, rather than being directed top-down.
   - **Benefit:** Boosts ownership, motivation, creativity, and leverages the collective intelligence of the team, leading to better solutions and faster problem-solving.
5. **Continuous Delivery & Integration:**
   - **Feature:** Frequent integration of code changes (often multiple times per day) and automated testing to ensure the software is always in a potentially releasable state.
   - **Benefit:** Reduces integration issues, enables rapid feedback on quality, minimizes bugs, and allows for faster deployment of new features.
6. **Face-to-Face Communication:**
   - **Feature:** Prioritizes direct, synchronous communication (e.g., daily stand-ups, co-location, video calls) over extensive documentation.
   - **Benefit:** Facilitates faster decision-making, clearer understanding, immediate clarification, and stronger team cohesion.
7. **Sustainable Pace:**
   - **Feature:** Focuses on maintaining a consistent, manageable workload for the team over the long term. Avoids burnout by not relying on overtime or unsustainable "crunch" periods.
   - **Benefit:** Leads to higher productivity, better quality work, improved team morale, and long-term project stability.
8. **Continuous Improvement (Kaizen):**
   - **Feature:** Regular reflection and adaptation of the team's processes through ceremonies like retrospectives at the end of each iteration.
   - **Benefit:** Teams constantly learn, identify inefficiencies, experiment with improvements, and refine their way of working to become more effective.
9. **Working Software as Primary Measure of Progress:**
   - **Feature:** Progress is measured by the delivery of functional, tested software that provides value to the user, *not* by comprehensive documentation, strict adherence to a plan, or completion of isolated phases.
   - **Benefit:** Provides tangible evidence of progress, focuses effort on real value, and reduces the risk of building something that doesn't work or meet user needs.

4. POPULAR AGILE FRAMEWORKS

FRAMEWORK	KEY FEATURES	ROLES
Scrum	- Sprints (2–4 weeks) - Daily stand-ups - Sprint review/retrospective	Product Owner Scrum Master Dev Team
Kanban	- Visual workflow (Kanban board) - Work-in-progress (WIP) limits - Continuous delivery	No fixed roles; flexible
XP (Extreme Programming)	- Pair programming - Test-driven development (TDD) - Continuous integration	Developers, Coach

5. KEY AGILE PRACTICES

- **User Stories:** Short requirements written from the user's perspective:
  "As a [user], I want [feature] so that [benefit]."
- **Sprints:** Time-boxed iterations (usually 2–4 weeks) to deliver a working increment.
- **Daily Stand-up:** 15-minute meeting for each member to answer:
  What did I do yesterday? What will I do today? Any blockers?
- **Retrospectives:** End-of-sprint meetings to reflect on improvements.
- **Backlog Grooming:** Prioritizing and refining the task list.
- **Burndown Charts:** Track progress toward sprint goals.

6. AGILE VS. WATERFALL COMPARISON

AGILE	WATERFALL
Iterative, flexible	Linear, sequential phases
Changes welcomed mid-project	Changes costly after planning
Customer involved throughout	Customer feedback late in the process
Short delivery cycles	Single delivery at project end

---

7. BENEFITS & CHALLENGES OF USING AGILE

Benefits

- Faster time-to-market.
- Adaptability to changing needs.
- Higher customer satisfaction.
- Improved team morale.

Challenges

- Requires cultural shift (collaboration > hierarchy).

- Hard to predict long-term timelines.
- Needs experienced team members.

8. GETTING STARTED WITH AGILE

1. **Start Small:** Pilot Agile with one team/project.
2. **Training:** Learn core concepts (Scrum/Kanban courses).
3. **Tools:** Use Jira, Trello, or Azure DevOps for backlog tracking.
4. **Ceremonies:** Implement daily stand-ups and retrospectives.
5. **Hire a Coach:** Bring in an Agile expert for guidance.

9. AGILE GLOSSARY

- **MVP (Minimum Viable Product):** Basic version of a product with core features.
- **Epic:** Large user story broken into smaller tasks.
- **Velocity:** Team's average work completed per sprint.
- **Impediment:** Blockers preventing progress.
- **Sprint Goal:** Short objective for a sprint.