



UNIT IV – PROJECT EXECUTION AND RISK MANAGEMENT

This unit focuses on how the project is carried out after planning, how risks are identified and controlled, how quality is maintained, and how the project is closed. Execution is where the real work happens, and risk and quality management ensure that the output is reliable and meets expectations.

1. Project Execution Phase

The execution phase begins once the planning is complete and the project manager receives approval to start the actual work. This is the stage where the project's deliverables are developed and the team performs the activities outlined in the project plan.

Execution involves coordinating people, resources, and tasks to ensure the project moves forward as planned.

1.1 Key Activities During Execution

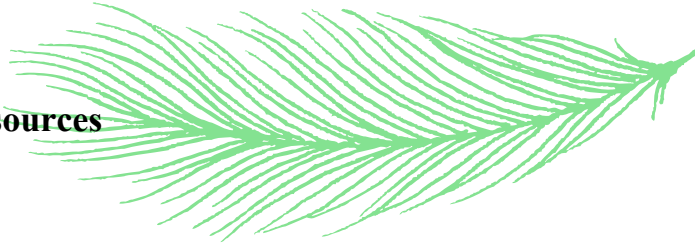
a. Team Coordination

The project manager assigns tasks, manages workloads, ensures collaboration among team members, and resolves conflicts. Strong leadership is essential in this phase.

b. Managing Communications

The project manager is responsible for consistent communication with stakeholders. This includes progress reports, meetings, updates, and feedback collection.

c. Managing Resources



Resources such as manpower, materials, equipment, and tools must be allocated and monitored.

d. Ensuring Quality

The team follows standards and procedures to ensure that the deliverables meet quality expectations.

e. Managing Risks and Issues

Risks identified in the planning phase are tracked, monitored, and controlled during execution.

f. Implementing Approved Changes

Any approved change requests are integrated into the project plan without disrupting workflow.

Execution is the most resource-intensive stage, requiring close supervision, timely decision-making, and effective communication.

2. Project Risk Management

Risk management is a systematic process of identifying, analyzing, and responding to project risks. A risk is an uncertain event that may positively or negatively impact the project.



Risk management ensures early detection and handling of issues so the project stays on track.

2.1 Types of Risks

a. Negative Risks (Threats)

Events that could harm the project, such as delays, cost increases, resource shortages, or system failures.

b. Positive Risks (Opportunities)

Events that could benefit the project, such as completing early, saving money, or gaining additional resources.

A good project manager addresses both threats and opportunities.

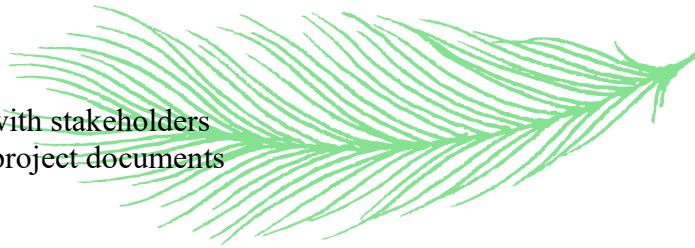
3. Risk Identification

Risk identification involves listing all possible uncertainties that might affect the project. This is usually done early, but risk identification is continuous throughout the project.

Common techniques include:

- Brainstorming sessions with the team
- Expert judgment
- Lessons learned from previous projects
- SWOT analysis (Strengths, Weaknesses, Opportunities, Threats)
- Checklists

- Interviews with stakeholders
- Reviewing project documents



Risk identification results in a **risk register**, which is a document that contains all identified risks with descriptions.

4. Qualitative and Quantitative Risk Analysis

Once risks are identified, they must be analyzed to understand their priority.

4.1 Qualitative Risk Analysis

This method assesses risks based on:

- Probability (likelihood of occurring)
- Impact (effect on scope, cost, or schedule)
- Urgency

Risks are often categorized using a **Probability-Impact Matrix**, which helps classify them as:

- High priority
- Medium priority
- Low priority

Qualitative analysis is faster and suitable for most projects.

4.2 Quantitative Risk Analysis



This method uses numerical values to measure the potential effect of risks. It is usually applied to high-priority risks.

Techniques include:

a. Decision Tree Analysis

Helps compare different action paths based on cost and outcomes.

b. Monte Carlo Simulation

Uses random variables and repeated calculations to estimate the probability of different outcomes.

c. Expected Monetary Value (EMV)

Calculates the expected financial impact of risks.

Quantitative analysis provides deeper insight but requires more data and expertise.

5. Risk Response Planning

Once risks are analyzed, the project manager creates a **Risk Response Plan** that defines actions to address each risk.

Risk responses differ for threats and opportunities.



5.1 Responses for Negative Risks (Threats)

a. Avoid

Eliminate the risk entirely by changing the plan.

Example: Avoiding a feature that relies on unreliable technology.

b. Mitigate

Reduce the probability or impact of the risk.

Example: Adding extra testing to reduce the chance of system failure.

c. Transfer

Shift the risk to a third party.

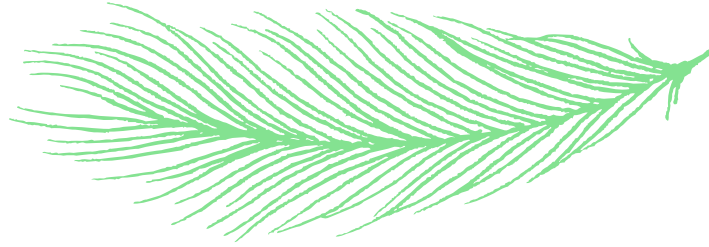
Example: Buying insurance or outsourcing high-risk tasks.

d. Accept

Acknowledging that the risk exists but taking no action.

Example: If the risk has low impact or cannot be controlled.

5.2 Responses for Positive Risks (Opportunities)



a. Exploit

Ensure the opportunity happens.

Example: Assigning top talent to complete work early.

b. Enhance

Increase the probability or impact of the opportunity.

Example: Improving quality to attract more customers.

c. Share

Partner with another organization to benefit from the opportunity.

Example: Joint development of a product.

d. Accept

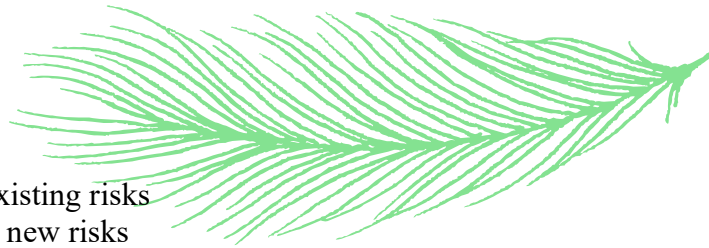
Taking advantage of the opportunity if it occurs.

6. Risk Monitoring and Control

Risk monitoring is ongoing during the execution phase.

It includes:

- Tracking existing risks
- Identifying new risks
- Keeping the risk register updated
- Reviewing risk responses
- Checking if mitigation strategies are effective



Risk reviews should be conducted regularly during meetings to keep the team prepared for uncertainties.

7. Project Quality Management

Quality management ensures that the project deliverables meet predefined standards and satisfy the customer's expectations. It is not about achieving perfection but ensuring that the final product is fit for purpose.

Quality management includes three key processes:

7.1 Quality Planning

Here, the project team determines which quality standards are relevant and how to achieve them. It includes identifying metrics, acceptance criteria, and quality assurance methods.

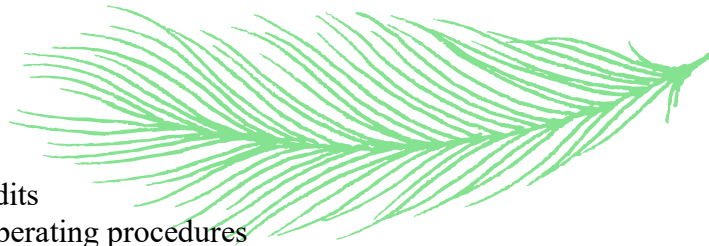
7.2 Quality Assurance (QA)

Quality assurance focuses on **processes**.

QA ensures that the team follows the correct methods, procedures, and standards during the project.

Examples:

- Process audits
- Standard operating procedures
- Quality training



7.3 Quality Control (QC)

Quality control focuses on **deliverables**.

It involves inspecting and testing the product to ensure it meets quality requirements.

Examples:

- Code reviews
- Testing
- Inspections
- Measurements

QC helps detect defects early and prevent rework.

8. Cost of Quality (COQ)

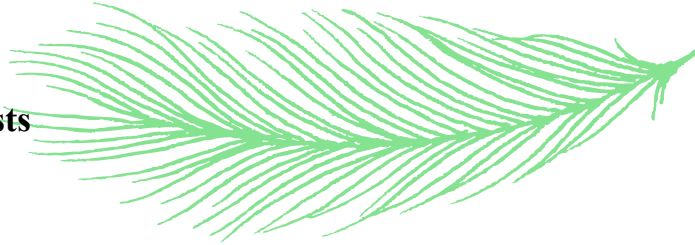
COQ represents the total cost of ensuring quality. It is divided into:

a. Prevention Costs

Cost to prevent defects.

Example: Training, quality planning.

b. Appraisal Costs



Cost of inspecting deliverables.

Example: Testing, audits.

c. Failure Costs

Cost if the product fails.

Divided into:

- **Internal failure** – defects found before delivery
- **External failure** – defects found by the customer

Managing COQ is essential to balance cost vs quality.

9. Controlling Changes (Change Management)

Change is inevitable in projects. Change management ensures that any modification to scope, timeline, or cost is reviewed and approved through a formal process.

Typical change control steps:

1. A change request is submitted
2. Impact on scope, cost, and schedule is analyzed
3. Review by Change Control Board (CCB)
4. Approve or reject the change
5. Update project documents and baseline

This prevents uncontrolled scope creep.



10. Project Issues

Issues are problems that have already occurred (unlike risks, which are future uncertainties). Issues must be resolved quickly because they directly affect the project.

Examples include:

- Team conflicts
- Technical failures
- Resource shortages
- Delays in delivery

Project managers maintain an **issue log** to track and resolve issues.

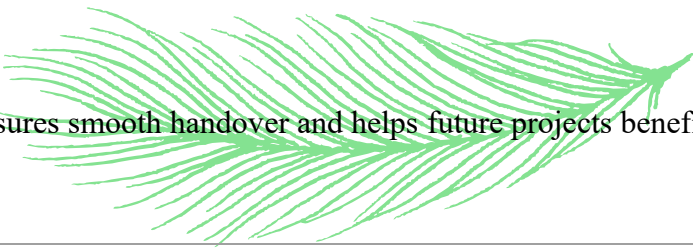
11. Project Closing

Project closing is the final phase of the project life cycle. It ensures that all project work is complete and the deliverables are accepted.

Activities include:

- Confirming project completion
- Getting acceptance from the customer
- Closing contracts with vendors
- Releasing resources
- Preparing final project reports
- Documenting lessons learned
- Conducting a post-mortem or retrospective

A proper closing ensures smooth handover and helps future projects benefit from lessons learned.



12. Project Post-Mortem

A post-mortem is a meeting conducted after project completion to analyze successes, failures, and improvement areas.

The goal is to identify:

- What went well
- What went poorly
- Why certain issues occurred
- How to avoid similar problems in future projects

This knowledge is valuable for organizational learning.



UNIT V – AGILE FOUNDATIONS AND FRAMEWORKS

(Full Descriptive Notes)**

Agile has become one of the most popular approaches in modern project management. This unit helps you understand the foundations of Agile, the principles behind it, key Agile frameworks, and how Agile compares to traditional (Waterfall) and hybrid approaches. You will also learn about Lean, Kanban, XP, Scrum basics, user stories, story points, and prioritization techniques.

This unit forms the core of Agile knowledge required for your exam.

1. Introduction to Agile

Agile is a flexible and iterative approach to project management that focuses on delivering value quickly and responding to change. It breaks work into small increments rather than attempting to define everything upfront, as in traditional models.

Agile was developed to address limitations of waterfall systems, where rigid planning often failed to adapt to changing customer needs.

Agile is best suited for projects where:

- Requirements evolve over time
- Customer feedback is important
- Teams need flexibility
- Speed of delivery matters

Examples include software development, app development, marketing, product design, etc.



2. Agile Values (From the Agile Manifesto)

The Agile Manifesto, created in 2001 by software developers, outlines four core values:

1. Individuals and interactions over processes and tools

Agile emphasizes team collaboration and communication rather than strict adherence to tools or rigid processes.

2. Working product over comprehensive documentation

Agile prefers delivering usable output quickly instead of spending too much time on documents.

3. Customer collaboration over contract negotiation

Customers give feedback throughout the project, rather than only at the beginning and end.

4. Responding to change over following a plan

Agile welcomes changes, even late in the project, to deliver a better product.

These values guide all Agile frameworks, including Scrum and Kanban.



3. Agile Principles (12 Principles)

Agile is based on 12 guiding principles. These principles shape how Agile teams work and deliver value. Some of the most important principles include:

- Customer satisfaction through early and continuous delivery
- Welcome changing requirements
- Deliver working software frequently
- Collaboration between business and technical teams
- Build projects around motivated individuals
- Prefer face-to-face communication
- Working product is the primary measure of progress
- Maintain sustainable development
- Continuous attention to excellence
- Simplicity is essential
- Self-organizing teams
- Team reflects and adjusts regularly

These principles ensure that Agile focuses on customer needs, flexibility, quality, and teamwork.

4. Agile Frameworks Overview

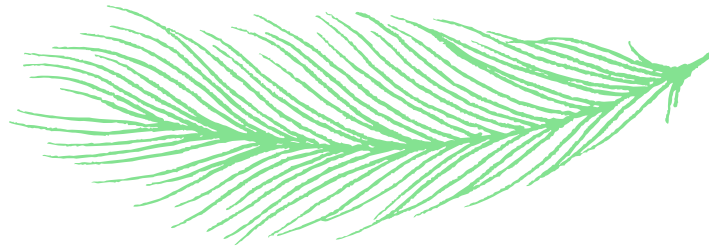
Agile is not a single method. It is a mindset with several frameworks and methodologies under it.

Popular Agile Frameworks:

1. **Scrum**
2. **Kanban**
3. **Extreme Programming (XP)**
4. **Lean Agile**
5. **Crystal Methodology**
6. **Feature-Driven Development (FDD)**

Your syllabus mainly focuses on:

- Scrum
- Kanban
- XP
- Lean Agile
- Hybrid models



5. Agile vs Waterfall vs Hybrid

Understanding these differences is important for exam questions.

Waterfall (Traditional / Predictive)

Characteristics:

- Sequential phases
- Requirements fixed at the start
- Heavy documentation
- Limited flexibility
- Best for stable and predictable projects

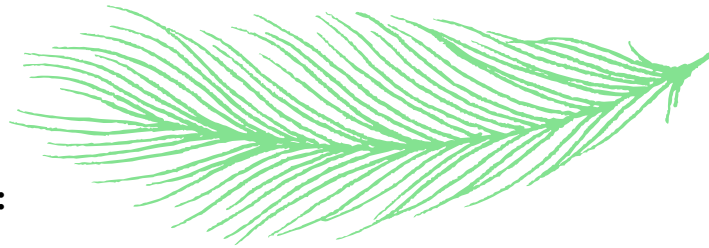
Pros:

- Clear structure
- Easy to manage
- Works well for construction, manufacturing, etc.

Cons:

- Difficult to adapt if requirements change
- Late feedback

Agile



Characteristics:

- Iterative and incremental
- Priorities can shift quickly
- Working product delivered frequently
- Heavy customer involvement
- Team collaboration is essential

Pros:

- Flexible and adaptable
- Faster delivery
- Better customer satisfaction

Cons:

- Needs skilled team
- Scope changes may require discipline

Hybrid Approach

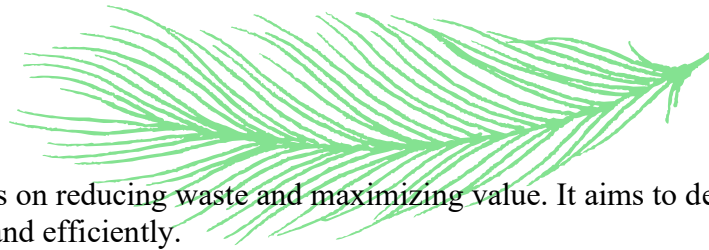
Hybrid combines the strengths of Waterfall and Agile.

For example:

- Planning and requirements using Waterfall
- Development and delivery using Agile sprints

Hybrid is used in organizations transitioning from traditional to Agile methods.

6. Lean Agile



Lean Agile focuses on reducing waste and maximizing value. It aims to deliver high-quality solutions quickly and efficiently.

Lean principles include:

- Eliminate waste (any activity that does not add value)
- Optimize flow (smooth workflow)
- Deliver fast
- Build quality into processes
- Respect people
- Improve continuously

Lean practices are widely used in software and manufacturing (Toyota Production System).

7. Kanban

Kanban is a visual method for managing work. It helps teams visualize workflow, reduce bottlenecks, and improve efficiency.

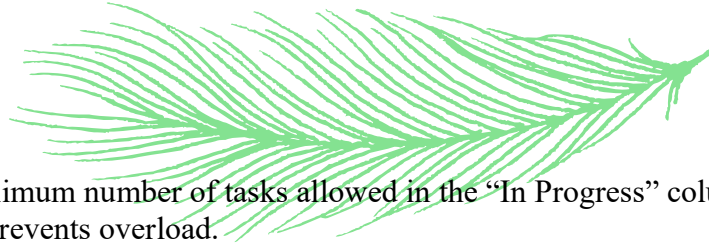
7.1 Kanban Board

A Kanban board usually has at least three columns:

- **To Do**
- **In Progress**
- **Done**

Cards represent tasks, and they move across the board as work progresses.

7.2 Work-in-Progress (WIP) Limits



Kanban sets a maximum number of tasks allowed in the “In Progress” column. This reduces multitasking and prevents overload.

7.3 Principles of Kanban

- Visualize work
- Limit work in progress
- Manage flow
- Implement feedback loops
- Improve collaboratively

Kanban supports continuous delivery instead of fixed-length sprints like Scrum.

8. Extreme Programming (XP)

XP is a software-focused Agile framework that emphasizes high-quality code and continuous improvement.

Key XP practices include:

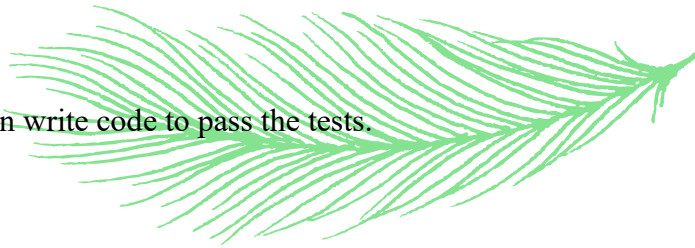
1. Pair Programming

Two developers work together on the same code.

One writes code, the other reviews.

2. Test-Driven Development (TDD)

Write tests first, then write code to pass the tests.



3. Continuous Integration (CI)

Code is integrated and tested several times a day.

4. Refactoring

Continuously improving existing code without changing functionality.

5. Short Iterations

XP delivers small increments quickly.

XP is widely used when quality and adaptability are top priorities.

9. Introduction to Scrum

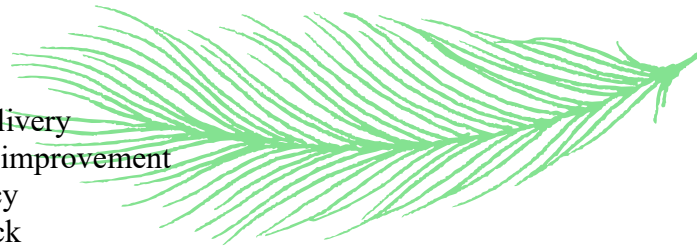
Scrum is the most widely used Agile framework. It is simple, structured, and effective for teams building complex products.

Scrum divides work into short cycles called **sprints**, usually 1 to 4 weeks long.

Scrum focuses on:

- Self-organizing teams

- Frequent delivery
- Continuous improvement
- Transparency
- Fast feedback



10. Scrum Roles

Scrum defines three main roles:

10.1 Product Owner

Represents the customer and stakeholders.

Responsibilities include:

- Maintaining and prioritizing the product backlog
- Clarifying requirements
- Defining the product vision
- Ensuring maximum value delivery

10.2 Scrum Master

Acts as a facilitator and coach for the team.

Responsibilities include:

- Ensuring Scrum rules are followed
- Removing obstacles
- Helping the team stay focused
- Supporting collaboration

The Scrum Master is not a manager. They help the team become more productive.

10.3 Development Team



A cross-functional group that builds the product.

Characteristics:

- Self-organizing
- Usually 5–9 members
- No hierarchy within the team

Their task is to deliver a working increment every sprint.

11. Scrum Artifacts

Scrum uses three primary artifacts:

11.1 Product Backlog

A prioritized list of features, requirements, bug fixes, and improvements.

Maintained by the Product Owner.

Characteristics:

- Constantly evolving
 - Items written as user stories
 - Prioritized based on value
-

11.2 Sprint Backlog

A list of tasks selected for the current sprint.

Created during Sprint Planning.



11.3 Increment

The working product produced at the end of the sprint.

It must meet the **Definition of Done (DoD)**.

12. Scrum Events (Ceremonies)

Scrum includes five ceremonies that structure team activities:

12.1 Sprint

A fixed time box (1–4 weeks) where the team works to deliver an increment.

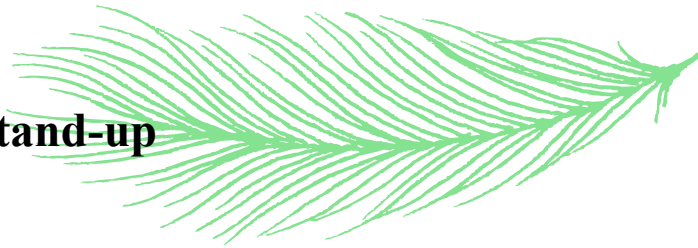
12.2 Sprint Planning

The team decides what to complete in the upcoming sprint.

This includes:

- Sprint goal
 - Tasks to include
 - Capacity planning
-

12.3 Daily Stand-up



A short daily meeting (15 minutes).

Each member answers:

- What did I do yesterday?
- What will I do today?
- Any blockers?

This meeting improves transparency and coordination.

12.4 Sprint Review

A demonstration of what the team completed during the sprint.

Stakeholders attend and give feedback.

12.5 Sprint Retrospective

Team reflects on the sprint and identifies improvements for the next sprint.

Focus areas:

- What went well
 - What didn't
 - What can be improved
-

13. User Stories

Agile requirements are written as **user stories**.

A typical format:

“As a [user], I want [feature], so that [benefit].”

Example:

“As a customer, I want to reset my password so that I can recover access to my account.”

User stories are simple, understandable, and reflect user needs.

14. Story Points and Team Velocity

Story Points

A measure of complexity or effort.

They do not represent time but represent how difficult a task is.

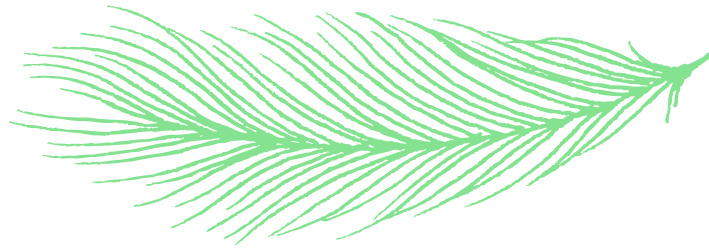
Teams commonly use Fibonacci numbers:

1, 2, 3, 5, 8, 13...

Velocity

The number of story points a team completes per sprint.

Helps predict how much work the team can handle.



Example:

If a team's velocity is 20 points, they aim to do around 20 points each sprint.

15. Setting Priorities

Agile uses several techniques to prioritize backlog items:

MoSCoW Method

- Must have
- Should have
- Could have
- Won't have

Value-Based Prioritization

Items with the highest business value are done first.

Cost of Delay

Tasks that create delay-related losses are prioritized.

Prioritization ensures maximum value delivery.



UNIT VI – AGILE PRACTICES AND ORGANIZATIONAL ADOPTION

(Full Descriptive Notes)**

Unit VI focuses on how Agile is applied in practice within teams and how organizations adopt Agile ways of working. While Unit V explains Agile foundations and frameworks, this unit goes deeper into real execution through sprints, backlog refinement, reviews, retrospectives, Kanban boards, and Agile leadership.

You will also learn what makes an Agile organization, how Agile structures work, and how distributed teams function in modern workplaces.

1. Introduction to the Sprint

A sprint is the core development cycle in Scrum. It is a fixed-length period, usually 1 to 4 weeks, during which the team completes a set of tasks from the product backlog. Every sprint must produce a **working increment** of the product.

Characteristics of a Sprint

- Fixed duration—does not change once started
- Carries a clear sprint goal
- Delivers usable output
- Includes planning, development, testing, review, and retrospective
- Team must not add new work once the sprint begins
- If the sprint becomes irrelevant or impossible, only the Product Owner can cancel it

Sprints create predictability and allow the team to focus on short-term achievable goals.

2. Sprint Planning



Sprint Planning marks the beginning of each sprint. The entire Scrum team participates, including the Product Owner, Scrum Master, and Development Team.

The purpose of sprint planning is to define:

1. **What the team will deliver** during the sprint
2. **How the work will be done**

Inputs to Sprint Planning

- Product backlog
- Team capacity (how many story points the team can complete)
- Previous velocity
- Sprint goal

Key Outcomes of Sprint Planning

- **Sprint Goal** – a short statement describing the purpose of the sprint
- **Sprint Backlog** – selected items (user stories) that the team commits to complete
- **Task Breakdown** – team breaks each story into smaller tasks

Sprint Planning ensures that everyone starts the sprint with a shared understanding of objectives and workload.

3. Product Backlog and Backlog Refinement

The **product backlog** is a prioritized list of everything that needs to be done for the product. It includes features, enhancements, bugs, spikes (research tasks), and technical work.



3.1 Characteristics of a Good Product Backlog

- Continuously updated
- Items ordered by value
- Each item has acceptance criteria
- Written in user story format
- Visible to all stakeholders

3.2 Backlog Refinement (Grooming)

Backlog refinement is an ongoing process where the Product Owner and the team:

- Break large items into smaller stories
- Add details
- Add acceptance criteria
- Estimate story points
- Reorder priorities
- Remove unnecessary items

Refinement usually takes about 5–10 percent of the team's time.

The goal is to keep the backlog clean, understandable, and ready for sprint planning.

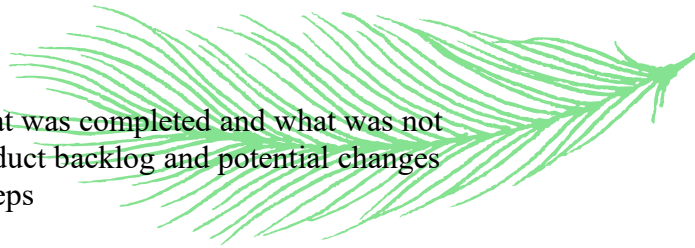
4. Sprint Review

The sprint review takes place at the end of the sprint. It is not just a demo but an interactive meeting between the Scrum team and stakeholders.

Purpose of Sprint Review

- Show the completed increment
- Gather feedback

- Discuss what was completed and what was not
- Review product backlog and potential changes
- Plan next steps



Stakeholders can provide suggestions, highlight missing elements, or request refinements. This ensures the team stays aligned with business expectations.

Sprint review is important because Agile welcomes continuous feedback.

5. Sprint Retrospective

The retrospective is one of the most important Agile practices. It happens immediately after the sprint review and before the next sprint begins.

Purpose

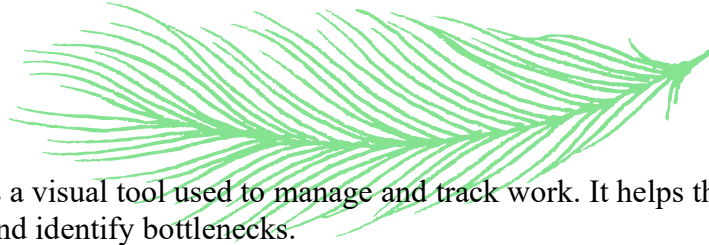
To improve the team's process, collaboration, and performance.

Key Areas Discussed in Retrospectives:

1. What went well
2. What didn't go well
3. What improvements can be made
4. How the team can work better in the next sprint

The team identifies **actionable improvements**, not just problems. Retrospectives help Agile teams grow and evolve continuously.

6. Kanban Board



A Kanban board is a visual tool used to manage and track work. It helps the team understand the flow of tasks and identify bottlenecks.

Common Columns

- **To Do** – work not yet started
- **In Progress** – work currently being done
- **Review / Testing** – optional column for quality checks
- **Done** – completed work

Why Kanban Boards Matter

- Brings clarity to workflow
- Improves transparency
- Helps teams focus
- Reduces multitasking
- Highlights blockers

Kanban boards can be physical (sticky notes on a board) or digital (JIRA, Trello, Asana).

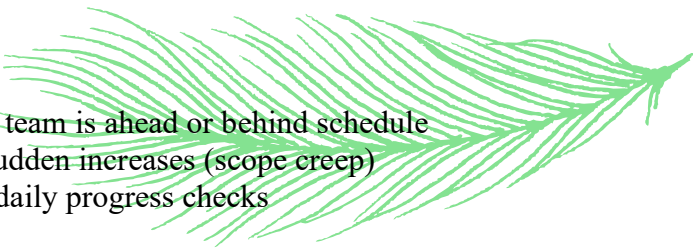
7. Burn Down and Burn Up Charts

Charts help track sprint progress and forecast completion.

7.1 Burn Down Chart

A burn down chart shows the amount of work remaining versus time.

How it helps:

- 
- Shows if the team is ahead or behind schedule
 - Highlights sudden increases (scope creep)
 - Encourages daily progress checks

The horizontal axis represents time, and the vertical axis represents story points or hours remaining.

7.2 Burn Up Chart

Burn up charts show the amount of work completed versus the total amount of work.

Benefits:

- Shows growth of scope
- Shows progress more clearly
- Useful for big picture monitoring

Both charts help teams make realistic decisions about timelines.

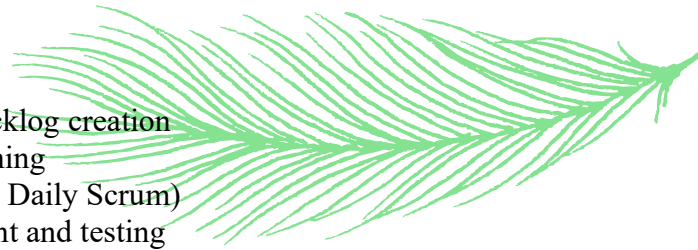
8. Scrum Process and Tools

Scrum uses tools and practices to manage work and maintain transparency. Common tools include:

- **Jira** – the most widely used tool for Agile project tracking
- **Trello** – simple, card-based workflow
- **Asana** – combines tasks and timelines
- **Azure DevOps** – preferred for enterprise-level projects
- **Monday.com** – visual workflow

Scrum Process Flow

1. Product Backlog creation
2. Sprint Planning
3. Sprint (with Daily Scrum)
4. Development and testing
5. Sprint Review
6. Sprint Retrospective
7. Repeat



These tools support the cycle by visualizing tasks, tracking progress, and maintaining backlog details.

9. Creating a Kanban Board

Creating a Kanban board involves simple steps but requires consistency.

Steps:

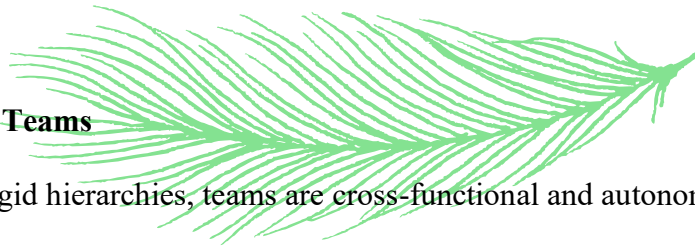
1. Identify the workflow stages (To Do, In Progress, Done)
2. Add tasks as cards or sticky notes
3. Assign team members to tasks
4. Add WIP limits to prevent overload
5. Move tasks across stages as work progresses

A good Kanban board is simple, accessible, and updated daily.

10. Agile Organization

An Agile organization is a company that embraces Agile values not just in development teams, but across the entire structure. This environment promotes flexibility, transparency, and quick decision-making.

Characteristics of Agile Organizations



1. **Networked Teams**

Instead of rigid hierarchies, teams are cross-functional and autonomous.

2. **Rapid Decision-Making**

Decisions are made at lower levels, reducing delays.

3. **Customer-Centric Culture**

Customer feedback is valued throughout the process.

4. **Continuous Learning and Improvement**

Teams adapt their methods based on retrospectives and market changes.

5. **Empowerment of Employees**

People are encouraged to take responsibility and propose ideas.

Agile organizations move faster and respond better to changing environments.

11. Agile Organizational Structure

Traditional organizations use a top-down structure. Agile organizations follow a **flat or network-based structure**.

Elements of an Agile Structure

- Cross-functional teams
- Team autonomy
- Short communication lines
- Shared ownership
- Flexible roles

In many Agile companies, leadership roles are more about coaching and supporting rather than controlling.



12. Five Trademarks of Agile Organizations

Well-known frameworks identify five traits common in successful Agile organizations:

1. Shared Purpose and Vision

Everyone understands and works toward a common goal.

2. Flexible and Networked Structure

Teams collaborate freely and adjust quickly to change.

3. Empowered Teams

Teams make decisions instead of waiting for approvals.

4. Rapid Learning Cycles

Teams reflect and adapt regularly using retrospectives.

5. Data-Driven Decision Making

Real-time analytics guide decisions rather than assumptions.

These traits make organizations extremely adaptive and competitive.



13. Agile Strategic Vision

Agile organizations create a strategic vision that is clear but flexible. It guides the product roadmap but allows changes based on customer feedback and market trends.

This vision helps:

- Prioritize work
- Align cross-functional teams
- Avoid confusion
- Encourage innovation

A strong vision creates unity across the organization.

14. Distributed Teams

Modern Agile teams often work across different locations or time zones. These are called **distributed teams**.

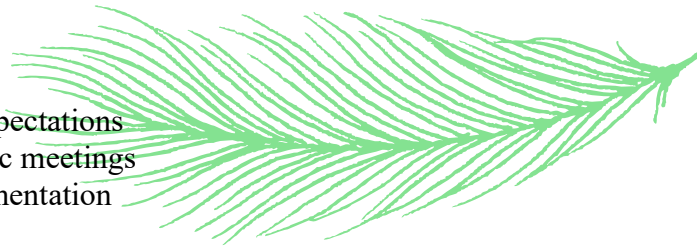
Challenges:

- Communication gaps
- Time-zone issues
- Cultural differences
- Tool dependency

Solutions:

- Use strong communication tools (Zoom, Slack)
- Overlap working hours

- Set clear expectations
- Regular sync meetings
- Clear documentation



Agile practices actually help distributed teams work smoother because of fixed meetings and transparency tools.

15. Agile Structures and Leadership

Agile leadership is different from traditional leadership.

Agile Leaders:

- Serve the team (servant leadership)
- Remove barriers
- Support experimentation
- Encourage collaboration
- Build trust
- Promote continuous learning

Agile leaders do not micromanage. They facilitate and empower teams.