# PMI-ACP notes

# Domain

| Domain | % |
|---|---|
| Domain I. Agile Principles and Mindset | 16 |
| Domain II. Value-driven Delivery | 20 |
| Domain III. Stakeholder Engagement | 17 |
| Domain IV. Team Performance | 16 |
| Domain V. Adaptive Planning | 12 |
| Domain VI. Problem Detection and Resolution | 10 |
| Domain VII. Continuous Improvement (Product, Process, People) | 9 |

# Domain I. Agile Principles and Mindset

Explore, embrace, and apply agile principles and mindset within the context of the project team and organization.

## The Agile Manifesto

On February 11-13, 2001, The Lodge at Snowbird ski resort in Utah witnessed a meeting between seventeen advocates of lightweight methodologies, seeking to discuss and identify any common ground for software development.

## Four Core Values of the Agile Manifesto

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

## Agile 12 Principles

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.

## AGILE Declaration of Interdependence

Agile and adaptive approaches for linking people, projects and value, with focus on Agile Leaders. To achieve these results

1. We **increase return on investment** by making continuous flow of value our focus.
2. We **deliver reliable results** by engaging customers in frequent interactions and shared ownership.
3. We **expect uncertainty** and manage for it through iterations, anticipation, and adaptation.
4. We **unleash creativity and innovation** by recognizing that individuals are the ultimate source of value, and creating an environment where they can make a difference.
5. We **expect uncertainty** through group accountability for results and shared responsibility for team effectiveness.
6. We **improve effectiveness and reliability** through situationally specific strategies, processes and practices.

## Benefits of Using Agile

- Increase wallet share quickly
- Reduce cost of making a change
- Speed up delivery
- Generate value from the user's perspective
- Reduced Risk
- Ensure visibility and transparency
- Increasing quality
- Increased motivation in the team

### Technical Debt

Consists of deficiencies in the code, technical documentation, development environments, 3rd-party tools, and development practices, which makes the code hard for the team to change.

Paying down technical debt by simplifying or optimizing design improves productivity hence increases velocity.

## Scrum

### Pillars of Scrum

- Transparency/Visibility
- Inspection
- Adaptation

### Roles in Scrum

- Development Team: Self-organizes to get the work done
- Product Owner: Responsible for the business value of the project
- Scrum Master: Ensures that the team is functional and productive and following scrum

### Scrum Artifacts

- Product backlog : Ordered/ Prioritized list of ideas for the product
- Sprint Backlog: Set of work from the product backlog that the team agrees to complete in a sprint, broken into tasks
- Increment of Product: Required result of every sprint. It is an integrated version of the product, kept at high enough quality to be shippable

### Scrum Ceremonies [Planned Opportunities for Inspection and Adaptation]

- Sprint Planning [Before Sprint]: Team meets with the product owner to choose a set of work to deliver during a sprint
- Daily Standup [During Sprint]: Team meets each day to share struggles and progress
- Sprint Review [After Sprint]: Team demonstrates to the product owner what it has completed during the sprint.
- Sprint Retrospective [After Sprint]: Team looks for ways to improve the product and the process.

## Extreme Programming (XP)

Extreme Programming, also commonly abbreviated as XP was created by Kent Beck in the 1990s. XP is described as a lightweight software-development discipline that organizes people to produce higher-quality software more productively.

### Core Values in Extreme Programming

- **Communication:** All team members should be free to discuss any aspect of a project without fear of a negative reaction.
- **Simplicity:** XP projects are characterized by simple design and code that is flexible enough to accommodate changes as the requirements evolve. “You aren’t gonna need it” (YAGNI)
- **Feedback:** Quick and timely feedback keeps everyone updated on the project.
- **Courage:** In XP, team members need to have the courage to modify a system as user requirements change and in order to make continuous improvements.
- **Respect**: XP teams have collective ownership and hence everyone has accountability for the design, code, success of a build, or passing of a regression test suite

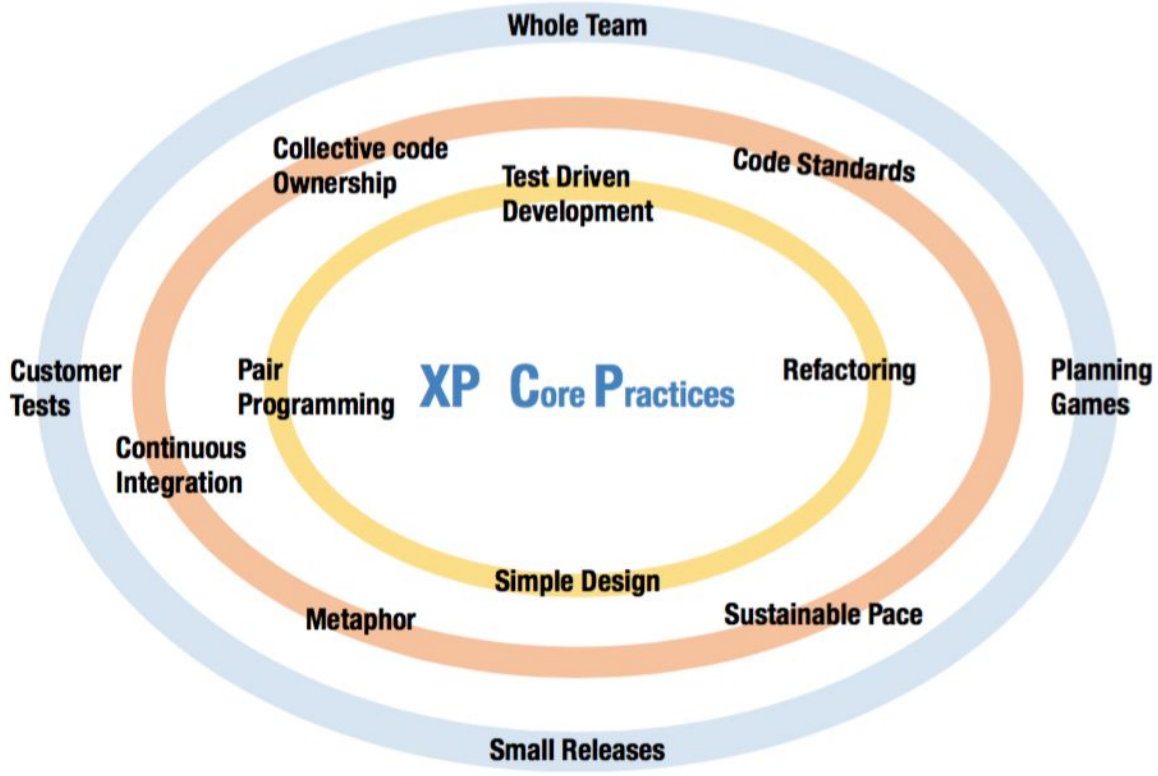
### XP Roles

Roles on mature XP team are not fixed and rigid, every one contributes their best, But initially fixed role help in learning new habits

- **XP Coach**: plays a supporting role for the team’s success.
- **On-Site Customers**: are responsible for defining the user stories (requirements) and their acceptance criteria
- **Programmer**: the XP team consists of multi skilled teams of 4 to 10 developers that practice pair programming
- **Testers**
- **XP Tracker**: this role helps to keep track of the progress of the team at an iteration or release level

- **Sponsor:** the sponsor funds the project, hence is a very important stakeholder of the project.
- **Interaction Designers**
- **Architects**
- **Project managers**: He helps team work with rest of the organization
- **Product managers**
- **Human Resources**

## Core XP Practices




## XP Notes

**Pair Programming:** is XP technique in which two programmers work together at one workstation. The two programmers switch roles frequently. Two roles in pairs are

- Driver, writes code while the other,
- Observer or Navigator reviews each line of code as it is typed in.

XP chooses scope as primary means of planning, tracking and steering project. Time and cost are often fixed.
XP commitments are made AS LATE AS possible.
**Test-first programming**, one test at a time and continuous integration which integrates and tests a few hours' worth of changes at a time. Some example tools for test-first programming are

- Static Analysis
- Model Checking

**XP vs. SCRUM: SCRUM** teams don't allow changes in middle of sprint but teams do allow change.
**Coding Standards**: is an exercise in building consensus. You will learn how to disagree constructively.
**Planning Game:** It's XP Practice to write user stories and estimating them with goal of maximizing customer value.
**Cohesive Design**: means closely related concepts are at one place; it improves design quality and makes it easy to understand.
**Caves and Commons:** refers to the creation of two zones in the room.

- The Commons area is organized to maximize osmotic communication and information transfer. For this to make sense, the people in the room must be working on the same project. It is perfect for XP's single team of up to 12 people programming in pairs.
- The Caves portion of the room is organized to give people a private place to do e-mail, make phone calls, and take care of their need for separation.

## Lean

Lean has its roots in the manufacturing industry. The origin of Lean goes back to the late 1940s when a Japanese businessman Taiichi Ohno introduced what was called the Toyota Production System (TPS)
However, it was in 2003 that Mary Poppendieck and Tom Poppendieck first introduced the application of Lean principles to software development in their book called *Lean Software Development*.

### Lean Software Development Principles

- **Eliminate Waste**: Waste is anything that does not add value OR any delay that keeps customer from getting value when they want.
- **Build Quality In/ Build Integrity In**: Goal is to build quality from start rather than test it in later.
- **Create Knowledge/ Amplify Learning**
- **Defer Commitment/ Decide as late as possible**
- **Deliver Fast / Deliver as fast as possible**
- **Respect People/ Empower Team**
- **Optimize the Whole / See the whole**

### Forms of Waste **(the seven wastes is by the acronym TIMWOOD)**

- **Transport** - unnecessary movement and handling of goods and people.
- **Inventories** - storage of parts that are either excess or awaiting consumption and often runs the risk of degrading in quality or becoming obsolete.
- **Motion** - unnecessary motion of employees, artifacts, or equipment.
- **Waiting** - for an upstream process (e.g., instruction, approvals, etc.)
- **Overproduction** - of things that are not demanded by the actual users.
- **Over Processing** - doing non-value added tasks and relying on inspections rather than preventive measures up front.
- **Defects** - from a variety of sources including rework, scrap, or incorrect documentation.

### Lean 5S Tool for Improvement

- **Sort (Seiri)**: remove unnecessary materials from the workplace.
- **Set in order (Seiton)**: arrange all items in a proper sequence, which facilitates a smooth flow.

- **Shine (Seiso):** periodically inspect the workplace and keep it clean.
- **Standardize (Seiketsu):** follow standard practices and processes at the workplace.
- **Sustain (Shitsuke):** maintain order, discipline and good working conditions.

### Lean Notes

- As per Lean agile philosophy most errors come from system (Systematic in Nature) and NOT from people. Team should respect people and fix system errors.
- Lean principle of eliminating waste is similar to agile principle of simplicity-the art of maximizing the amount of work not done.

## KANBAN

Kanban is a pull-based method for software development that aims at minimizing work in progress (WIP), maximize continuous flow and provide visualization of work as it flows through various life-cycle stages from inception (conceptual phase) to production.

### KANBAN Principles

- Visualize the workflow
- Limit WIP
- Manage Flow
- Make Process Policies Explicit: directly related to visualization is a need for explicit agreement on policies and procedures.
- Implement Feedback Loops: this practice is about retrospective of the process itself
- Improve Collaboratively (using models & the scientific method)

### KANBAN Metrics

- **Task Completion Rate (TCR)**: Tasks completed per day
- **Task Add Rate (TAR)**: Tasks added or arrived per day.
- **Current Task Estimate (CTE)**: Total number of active and pending tasks.
- **Days to complete** = CTE / (TCR – TAR)

### KANBAN Notes

- KANBAN is Lean based methodology
- Pull System: is KANBAN scheduling system that signals what to produce and only produces item customer needs.
- Limited WIP: KANBAN has huge focus on limited work in progress (WIP)
- MMF (Minimal Marketable Features): is a feature that is minimal, because if it is any smaller, it would not be marketable. This concept originated in KANBAN and Lean.

## Dynamic Systems Development Method (DSDM)

The origin of DSDM methodology dates back to 1994 and was seen as an enhancement over the Rapid Application Development (RAD) method. It combines the project management and product management life cycle.

### Principles in DSDM

- Focus on the business need
- Deliver on time
- Collaborate
- Never compromise quality
- Build incrementally from firm foundations
- Develop iteratively
- Communicate continuously and clearly

- Demonstrate control

## Feature-Driven Development (FDD)

FDD is a lightweight Agile methodology that aims to build a software in increments of features or functionalities. These features directly represent value-added functionality that a user wants to use. The origin of FDD goes back to Jeff De Luca in the mid-1990s.

- Develop overall model
- Build feature list
- Plan by feature
- Design by feature
- Build by feature

## Crystal

Crystal is the name of a family of methodologies created by Alistair Cockburn in the mid-1990s. After years of research and looking into working practices of communication and community- based Agile projects,

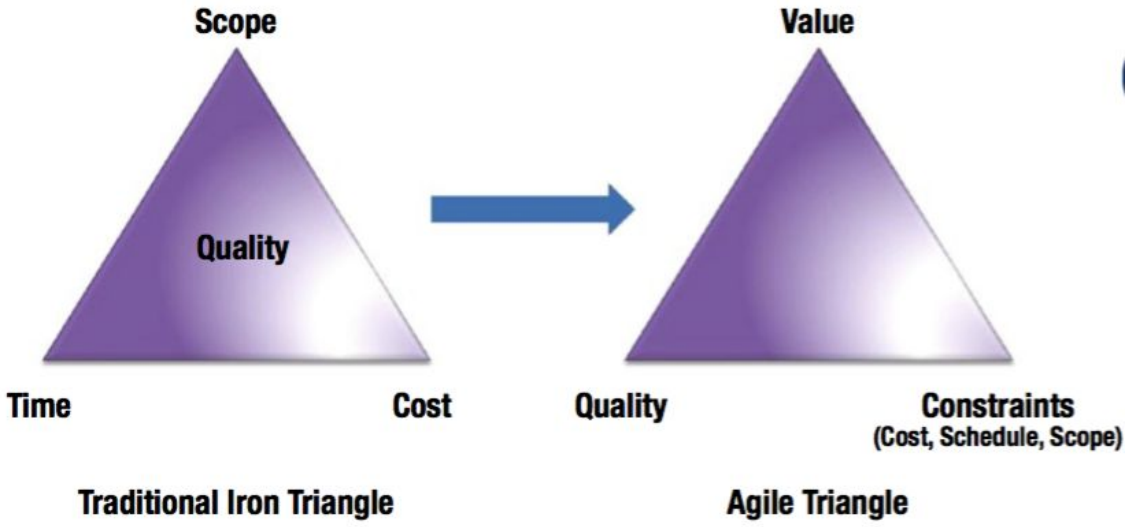
### Characteristics of Crystal

- Small teams interacting with each other
- Producing running code more frequently to the user
- Continuous improvements
- Using richer and osmotic communication within co-located team members
- Focus on the goals of the project and the individual tasks in hand.
- Providing the team an opportunity to pause and adapt their ways of working.
- Ease of access to business or expert user.
- Reduce intermediate work products.

# Domain II. Value-driven Delivery

Deliver valuable results by producing high-value increments for review, early and often, based on stakeholder priorities. Have the stakeholders provide feedback on these increments, and use this feedback to prioritize and improve future increments.

## Agile triangle




## Embedding Value-Driven Delivery in Agile Practices

- **Deliver Value in Increments:** Agile projects deliver in small increments with each increment consisting of the features of the products that are deemed to be of the highest business value.
- **Deliver Value Early:** Early deliveries help teams to understand customers requirements better, increase the confidence of stakeholders.
- **Value-Based Analysis:** Agile projects consider both the business value of work items as well as the cost of delivering them. This is called *Value-based analysis*
- **Prioritizing Collaboratively:** The product owner prioritizes units of work based on business value. This is collated in the product backlog. Each work item, based on its perceived complexity, is also estimated by the development team.
- **Minimizing Non-Value Added Work**
- **Frequent Review Based on Stakeholder Priorities:** At the end of every iteration, the team gets an opportunity to gain acceptance of the work done and solicit feedback from stakeholders
- **Focus on Quality:** Defects erode value and limit the usability of a product.
- **Focus on Nonfunctional Requirements:** Agile teams also need to be attentive toward non functional requirements that are like the quality attributes of the system and govern fitness for use.
- **Continuous Improvement:** Agile teams retrospect continuously.

## Determining Value at Project Initiation

Projects are typically authorized as a result of one or more of the following strategic considerations:

- Market demand or requests from customer segments
- Strategic opportunity to seek competitive advantage

- Improvement of operational efficiency
- Technological advancement
- Legal and regulatory compliance requirements Environmental consideration
- Social need or safety need

## Economic Models for Project Selection

A balance needs to be maintained between time, cost and scope

**Present Value (PV)** (choose max value)
The concept of present value is based on time value of money by factoring in a rate of interest or inflation. PV is commonly expressed by the following formula:

$$PV = FV / (1+r)^n$$

Where FV = future value, r = interest rate and n = number of time periods.

Example:
Project A has a future value of $300 after 3 years, at an interest rate of 10%.
Project B has a future value of $500 after 5 years, at an interest rate of 10%.

Project A has a PV of PV = 300 / (1 + 0.1)3 = 300 / 1.33 = $225
**Project B** has a PV of PV = 500 / (1 + 0.1)5 = 500 / 1.61 = **$311**

**Net Present Value (NPV)** (choose max value)
Net present value extends the concept of PV by factoring in the initial investment as well as the revenue stream of future.

Net Present Value (NPV ) = - Initial investment + ∑ Return of each year in today's terms.

Example:
Project A requires an initial investment of $1000 and produces a return of $300 for the next 3 years. Assume a steady interest rate of 10%.
Project B requires an initial investment of $800 and produces a return of $200 for the next 4 years. Assume a steady interest rate of 10%.

Project A has a NPV of NPV = - 1000 + 300 / (1.1)1 + 300 / (1.1)2 + 300 / (1.1)3 = - $254
**Project B** has a NPV of NPV =-800+200/(1.1)1 +200/(1.1)2 +200/(1.1)3 +200/(1.1)4 =**-$166**

**Payback Period** (choose the min)
Payback period is the number of time periods it takes to recover the investment in the project before it starts to accumulate profit.

**Internal Rate of Return (IRR)**(choose max value)
IRR is defined as the discount rate at which the project inflows and outflows are equal.

**Return on Investment (ROI) or Benefit Cost Ration (BCR)**(choose max value)
ROI is the ratio of net benefits of the project to it total cost.

## Agile charters

The project charter is one the earliest documents or artifacts that is produced in the project life cycle.
The acronym **W5H** is an easy way to remember the contents of a project charter.

- **What** is the scope of the project
- **Why** is the project being undertaken
- **Who** will be impacted because of the project
- **When** will the project start and end

- **Where** will the project occur or deliver
- **How** will the project be executed

## Product Vision and Elevator Pitch

The elevator pitch articulates the vision of the project in a clear, focused and compelling manner that is understood by one and all.
The product owner or business sponsor takes the lead and explains what the product serves and why it is deemed as valuable.
For defining the product vision, a few more Agile tools can be used as follows:

- **Product vision box**
- **Flexibility matrix**
- **Product data sheet:** this is a crisp one-page summary of the goals and objectives of the project that addresses the needs and expectations of the customer and the team.

## Cycle Time

Cycle time is defined as the average time it takes an item to get from the end of the queue to a state of being complete or done.
The goal of Lean and Agile is to ensure that the cycle time be kept at a minimum.

### Queueing Theory and Little's Law

The Little's Law states that the length of a queue (number of items in WIP state) is directly proportional to the duration of the time spent in the queue.

**Work in progress (WIP) = Lead Time * Throughput.**

where
Throughput = average rate at which work departs or is completed
and Lead Time = average time an item spends in the system

### How Do We Reduce Cycle Time

- Reduce variability in rate of arrival
- Reduce variability in rate of processing
- Limiting work in progress
- Removing blockers, waiting times
- Investing behind smart engineering tools and practices
- Investing behind a cross-functional team

## Value Stream Mapping

It looks at value from the customer's perspective and delivers them in the form of features in products and services

- **Identify the product** or service that **needs to be analyzed**.
- **Identify the steps**, queues, delays and information flows in the process to come up with the value stream map of the process.
- **Create flow by identifying and eliminating all forms of wastes** and their sources. Wastes imply delays, constraints, bottlenecks and non-value added tasks.
- **Create a new value stream map** of the future state by eliminating the wastes identified in the previous step. This leads to an efficient process that responds to customer pull.
- **Develop a plan** to reach the future state from the present state.
- Pursue perfection by **continuously reviewing the process** so as to find opportunities to optimize it further.

## Computing the Lead Time

total lead time = total duration of the value-added and the non-value-added tasks.
process cycle efficiency = Total value added time / total lead time

## Value-Based Prioritization Techniques

- **Numerical Assignment:** rank requirements in priorities of 1, 2, 3, or High, Medium, Low and so on.
- **Analytical Hierarchical Process (AHP):** takes a list of features and does a pairwise relative comparison based on some criteria.
- **100 point or Cumulative Voting Method:** is like an opinion poll for determining the priority of items in a group environment. Each participant in the group is given 100 points to be distributed as votes across the list of user stories (or product backlog items).
- **Monopoly Money:** is similar to the 100-point method.
- **MoSCoW:** prioritization technique has its origin in DSDM. It is an acronym that stands for the following:
    - **Must Have:** fundamental for the working of the system. Hence they have the highest priority.
    - **Should have:** important for the system to work correctly. Hence they have medium priority.
    - **Could have:** somewhat useful or desirable, but not necessary.
    - **Won't have:** cosmetic or 'nice-to-have', but are of least criticality.
- **Kano Analysis Model**
- **Wiegers' Method:** is a quantitative analytical approach to prioritization that makes a weighted computation of value, cost and risk associated with a requirement.
- **Balancing Risk and Value:** has introduced a 2 2 risk-to-value matrix mapping both value and risk from high to low and recommends the most suitable strategy at dealing with them during prioritization.

## Product Backlog

### Backlog Grooming or Refinement

The set of activities that the Product Owner, in collaboration with the development team, undertakes to manage the product backlog items (also abbreviated as PBI's) is called backlog grooming or backlog refinement.

### DEEP attributes of Product Backlog

DEEP is used to describe attributes of a product backlog:

- **Detailed appropriately:** PBI should have enough detail that can be used to convey the necessary information to the project team.
- **Estimable**: PBI's should have enough details that makes it estimable.
- **Emergent**: signifies that the product backlog is a dynamic list. It is expected to grow, change and get reprioritized over the life cycle of the project.
- **Prioritized** : It is the role of the product owner to ensure that the product backlog is prioritized with the PBI carrying the highest value featuring at the top of the backlog

## Agile Metrics and KPI's

**Velocity**

**Release Burndown charts:** burndown chart shows how much work is remaining at the beginning of each iteration and the likelihood that the project team will be able to achieve the iteration goal.

**Burnup charts:** are conceptually the opposite of a burndown chart displaying what has been done so far.

**Parking lot charts:** have their origin from Feature-Driven Development (FDD) methodology. To begin with, the stories for each release are grouped in themes. These themes are represented in rectangular boxes.
**Throughput**: tracks the number of completed work items in a week
**Takt Time:** Available time for production / customer demand
where Available time for production is the total number of hours employees are working minus any time for meeting, down times and breaks.
**Earned Value Management (EVM):** EVM techniques compare actual values measured against a planned or baselined value of cost or schedule.

- **Planned value** = Sum of estimates (in story points) of all the stories planned for the release.
- **Earned value** = Sum of estimates (in story points) of all the stories actually completed as part of the release.
- **Actual costs** = actual money spent behind resource and non-resource costs to implement the stories in the release.
- **Schedule Performance Index (SPI)** = Earned Value / Planned Value.
    - SPI < 1 signifies that the project is behind schedule.
    - SPI > 1 means the project is ahead of schedule.
- **Cost Performance Index (CPI)** = Earned Value / Actual costs incurred (money spent till date).
    - CPI < 1 signifies that the project is over budget schedule.
    - CPI > 1 means the project is under budget.

**Escaped Defects:** the number of defects that is leaked to the next stage of the process, the next sprint, or in the worst case into the hands of the customer.

# Domain III. Stakeholder Engagement

Engage current and future interested parties by building a trusting environment that aligns their needs and expectations and balances their requests with an understanding of the cost/effort involved. Promote participation and collaboration throughout the project life cycle and provide the tools for effective and informed decision making.

## Identifying Stakeholders

- people who are working on the project.
- sponsoring it or supporting it (like senior leadership team).
- providing requirements or intending to use the final product, service, or result of the project.
- marketing and advertising the product or the service.

### Analyzing Stakeholders Based on Power and Interest

Classify stakeholders based on their power and interest in the project and devise a strategy to deal with them appropriately.

### Analyzing Stakeholders Based on Engagement Levels

The stakeholder engagement matrix that is used to assess their current and desired engagement levels or outlook of the project. The engagement levels could vary as follows:

- Unaware: Stakeholders who are unaware of the project and potential impacts.
- Resistant: Aware of the project and resistant to change.
- Neutral: Neither supportive nor resistant.
- Supportive: Would like to see the change happen and has a positive outlook toward it.
- Leading: Actively engaged in ensuring the project is a success.

## Active Listening

- Level I – Internal Listening: although we hear what the speaker is saying and we focus on what it means to us.
- Level II – Focused Listening: we hear each individual word and how the speaker expresses them without allowing to be distracted by one's own thoughts and feelings.
- Level III – Global Listening: the listener not only focuses on the speaker, his words and emotions, but also picks up everything available with all senses.

## Conflict Resolution Techniques

- **Problem Solving / confronting:** solving problems by examining alternatives
- **Compromise:** seeking common ground that brings in some degree of satisfaction for all parties.
- **Smoothing / accommodating:** emphasizing on areas of agreement rather than differences.
- **Collaborating:** considering multiple views and perspectives, ultimately leading to consensus and commitment.
- **Withdrawing / avoiding:** retreating from a conflicting situation.
- **Forcing:** pushing one's view over others, resulting in win-lose outcomes

## Styles of Group Decision-Making

- Command
- Consultative

- Consensus

## Methods of Reaching a Decision

- **Unanimity**: Everyone agrees on a single course of action
- **Majority**: Support from more than 50%
- **Plurality**: The largest block decides even if a majority is not achieved
- **Dictatorship**: One individual makes decision for the group

## Agile Leadership Styles

- **Servant leaders** or humble stewards of an organization:
    - Communicate the project vision
    - Uphold the principles of Agile
    - Protect the team from internal interruptions
    - Protect the team from external interruptions
    - Help to remove blockers
    - Provide logistics, encouragement and support
- **Adaptive Leadership:** building and sustaining a culture of continuous learning, flexibility and collaboration in an organization framework.
- **Participative Leadership:** also called democratic leadership is a management style that invites input from employees, team members and participants on a particular subject, but the leader finally retains the authority to make the final decision.

differentiates between “Doing Agile” and “Being Agile.” Doing Agile is mostly about processes, practices and tools that an Agile team needs. For example, iterative development is one of the basic practices that you expect out of a team “doing Agile.” To “be Agile,” enterprises and its leaders need to embrace the culture and behavior and deliver a continuous stream of solutions.

# Domain IV. Team Performance

Create an environment of trust, learning, collaboration, and conflict resolution that promotes team self-organization, enhances relationships among team members, and cultivates a culture of high performance.

## Team Selection: Cross-Functional and Generalizing Specialists

Cross-functional skills are invaluable for Agile project teams, because of the following benefits:

- It is easier to take shared ownership and group accountability for results.
- During the time of estimation, team members are forced to think what it will require to deliver the requirement to the customer.

## Interpersonal Skills

Collaboration, active listening, transparent and open communication, conflict management, negotiation, diversity, adaptability, flexibility, dealing with ambiguity, culture of continuous learning and improvement.

## Bruce Tuckman's Stages of Team Building

According to Tuckman's theory, teams move through the following stages progressively:

- **Forming**: the team members come together at the first instance. They are focused on themselves, their roles, responsibilities and goals rather than that of the team.
- **Storming**: begin to work on the projects, bring about their individualities, voice their opinions and at times, attempt to dominate. During this time, conflicts and disagreements are likely at play at various intensity levels. Some of these conflicts could be positive, but a few could be destructive and demotivating.
- **Norming**: set ground rules and emphasize what the professional behavior should be like. The team members attempt to resolve their conflicts, forgive and forget some nuances and get aligned to the shared goal of the project.
- **Performing**: This is the highest ('nirvana') stage where teams are motivated to achieve the goals of the project.
- **Mourning or Adjourning**: this is the stage where the project is over and team members get released.

## Shu-Ha-Ri Model

Used to depict the maturity of learning and application in a team environment

- **Shu**: which means learn. At this level the team member obeys the rules laid down by the master or instructor precisely. He focuses on doing the task well, based on the practices and conventions.
- **Ha**: which means detach. At this level the team member attains more maturity, gets deeper understanding, reflects on the rules that are at play, finds alternatives and exceptions and looks to "break" free from the rules.
- **Ri**: which means transcend. At this stage the practitioner has attained mastery. He has learned from others and is now thinking originally, progressing at his creative best while not ignoring reality and demands of everyday life.

## Dreyfus Model

- **Novice**: the team members have a very shallow understanding and need very close supervision.

- **Advanced beginner**: the team members are able to complete simple tasks without supervision.
- **Competent**: good understanding of the steps involved and are able to complete their tasks properly without supervision.
- **Proficient:** have gathered sufficient practice, experience and deep understanding of the subject. They see actions holistically (‘big picture’) and routinely achieves high standards of performance.
- **Expert**: the team members achieve excellence and go beyond existing interpretations, rules and guidelines.

## Situational Leadership Model

Created by Ken Blanchard and Paul Hersey depicts how an Agile leader needs to dynamically adjust his/her leadership style based on the competency and commitment levels of the team members being led.

- **Telling style**: the leader needs to exhibit a directing style giving clear instructions to accomplish the tasks and supervising the team closely.
- **Selling style:** the leader should use a selling / coaching role to convince team members and follow the specific directions given by him/her.
- **Participating style**: the leader plays a more participating role by blending with the team and supporting some of the decisions made by the team.
- **Delegating style:** the leader role suffices to be of a delegating nature.

## Systems Thinking

With the cross-functional and self-organized attributes of Agile teams, each team member is encouraged to see the big picture rather than stay bogged down on executing their particular tasks only.

## BART Analysis of Team

The acronym BART stands for **boundary**, **authority**, **role** and **task**. With this tool, one can actually look under the hood to find out about ambiguity, lack of clarity, organizational issues, or other nuances of team dynamics.

## Tacit knowledge

Is knowledge that cannot be adequately articulated by verbal means or documented. It is collective knowledge that an individual builds up over time through observation and practices, but difficult to communicate to others via words and symbols.

## Expert in Earshot

Is putting junior team member in line of sight and within hearing distance of more experienced team members.

## Caves and Commons

In order to maintain a balance between the need for real-time osmotic communication and quiet times in cone of silence, teams could create two zones in the team space as follows: Commons and Caves

## Contract Types for Traditional Projects

**Fixed-Price (FP) contracts**:

- Contract based on well defined and agreed specifications from the buyer.
- Seller plans for contigencies and charges a premium fee in lieu of the risk that it undertakes to complete the fixed scope in the contract.

**Cost-Reimbursable (CR) contracts**:

- Scope is uncertain, so the buyer agrees to reimburse the seller for all legitimate costs incurred to complete the work, plus a fee for the seller's profit.
- Considerable risk is involved on the buyer side as the costs are difficult to estimate at the outset.

**Time and Material (T&M) contracts:**

- A quick and simple form of contract which is a hybrid between FP and CR in terms of risks shared between the buyer and seller.
- Often used for staff augmentation where unit labour rate is fixed, but the duration of the contract is flexible.

## Contract Types in Agile Projects

## Fixed Price, but with Provision for Change in Scope in Future Iterations

the scope of the contract is reviewed before every iteration. Since Agile teams do not permit changes during the middle of the iteration, the scope can be considered fixed *during* an iteration making it suitable for a fixed-price, fixed-scope delivery in the timeboxed iteration.

## Contract with Premature Closure Clause

It is quite possible that the customer or user realizes that there is no value or ROI in continuing further with the project as outlined in the contract initially.

## Fixed Fee and Not-to-Exceed Clauses

## Fixed Price per Story Point

## Target Cost Contract

The buyer and the seller mutually agree on a target cost and schedule of the project, which includes the profit margin of the seller.

## Contract Extension and Payment Based on Delivery and Acceptance

After each incremental delivery, the buyer pays out to the seller only when the acceptance test cases have been satisfactorily passed.

## Agile PMO

Here are some of the functions that the PMO (Project Management Office) can play in an agile organization:

- Agile transformation
- Agile adoption and training
- Agile coaching
- Agile governance and center of excellence
- Resource management
- Vendor management
- Audits and compliance
- Continuous improvement

# Domain V. Adaptive Planning

Produce and maintain an evolving plan, from initiation to closure, based on goals, values, risks, constraints, stakeholder feedback, and review findings.

## Aspects of Agile Planning

### Deming's Plan-Do-Check-Act (PDCA) Cycle:

Is the foundation of the principles of adaptive planning and continuous improvement in Agile.

### Progressive Elaboration/Rolling-wave Planning

Progressive elaboration is a technique where plans are detailed out as more detailed and specific information is available to the project team.
Rolling-wave planning is a form of iterative planning and progressive elaboration technique.

### Cone of Uncertainty

- Initial *Rough Order of Magnitude or ROM* estimate, estimate could be as high as +75% to -25%
- When the team is about to start implementation based on a detailed design, the estimates get more accurate and the range drops to ±20%, which is called *Budgetary estimate*.
- Then finally at later stages of development *definitive estimates* that are within a range of +10% to -5% can be produced.

### Just-in-time Planning

Agile methods follow a just-in-time (JIT) planning model. This concept originates from Lean manufacturing that eliminates all forms of wastes, prevents build-up of inventory and is pull-driven.

### Timeboxing

Is a fixed duration of time within which a task must be accomplished. Advantages of timeboxing:

- Maintain a sense of urgency
- Bring in some much needed operating rhythm in the team.
- Avoid the effect of *Parkinson's Law* – The law states that work tends to expand to fill all the time available for its completion.
- Avoid the effect of *Student Syndrome* – This is a form of procrastination and the analogy is with a student who will only start preparations for the exam when the deadline approaches,

### Levels of Planning - The Planning Onion

- **Strategy Planning:** this typically consists of a 3–5 year strategy that an organization charters out to serve its stakeholders.
- **Portfolio Planning:** in order to fulfill the vision laid down in the strategy plan, the organization chooses a portfolio of products and services designed to reach a particular market or market segment. This plan could span up to 2–3 years.
- **Product Planning:** with the product vision in hand, the next level of planning goes into creating the project charter and the product roadmap.
- **Release Planning:** the product roadmap is used to categorize requirements into themes by logically grouping them into usable set of features that can be released together.

- **Iteration planning:** the release plans have no details other than a list of stories to be done by a date.
- **Daily Planning:** the deepest level of Agile planning focuses on a single day.

## User stories

User stories describe requirements or functionalities of the software from a user's perspective.

**Card, Conversation and Confirmation**

**Epic:** an epic could simply be a large user story that can span several iterations.

**Feature:** a feature represents a high-level functionality required by the business. Similar to epics, features also need to be broken down into more manageable fragments like stories that can span several iterations to be delivered.

**Themes**: are set of related user stories that are combined together to speeds up estimation, planning releases, or financial planning

**Stories**: at the iteration level, it is the stories that are used as units for estimation, planning and implementation.

**Tasks and Subtasks**: in order to implement the story, all the things that a developer has to do is called as a task.

The process of breaking down epics into features, features into user stories and user stories into tasks is also called ***value-based decomposition***.

### Attributes of User Stories: INVEST

**Independent**: the user story should be independent and not overlapping with each other; otherwise it might lead to problems in planning and prioritization.

**Negotiable**: a user story from the product backlog is not rigid or a written contract, hence subject to elaboration and change.

**Valuable**: since the user stories are prioritized from the backlog, it is this value that drives prioritization.

**Estimable:** this estimate includes all the activities involved to meet the *definition-of-done* criteria for the story.

**Small:** ideally a user story should be sized such that it can be planned, prioritized and slotted into an iteration.

**Testable:** stories must be so written that they can be tested to confirm that they meet the user's expectation and satisfy the "doneness" criteria.

### Attributes of User Stories: SMART

**Specific**: user stories that are specific help to convey its purpose to the team and helps to keep it independent.

**Measurable**: user stories should be estimable (in terms of story points, for example) and measurable against the *definition of done.*

**Achievable**: user stories should be unambiguous and the team should be able to achieve the *doneness criteria* within the timeboxed iteration.

**Relevant**: user stories should add relevant value to the customer and contribute, incrementally, to the overall objective or goal of the project.

**Timebound**: stories should be such that they can be reasonably accommodated in the timebox that the team mutually agrees.

### Story-gathering Techniques

**Interviews**: Face-to-face interview sessions are a very rich form of communication and, if

the interviewees are carefully selected, this technique can be very useful to gather a lot of quality requirements.
**Surveys and questionnaires**: the audience is provided a set of questions to answer and the responses are analyzed.
**Voice of Customer (VOC)**: which is used to determine the customers' needs ranked in order of relative priorities and urgencies.
**User role modeling and Persona**

**Agile Prototyping and wireframes**
**Greenfield technique:** in this technique the users are asked to imagine an environment where there are no constraints, boundaries, or existing structures.
**Group creativity techniques:** Brainstorming, Nominal group technique, Multi-criteria decision analysis
**Group decision-making techniques:** Once the ideas are collected, decisions are made by following: Unanimity, Majority, Plurality, Dictatorship

### Innovation Games

- Buy a feature
- Product box
- Prune the product tree: this game starts by drawing a large tree that represents the core functionality and branches of various widths to represent components and features.
- Remember the future: the project stakeholders are told to imagine a situation in future where the product has been delivered and it has been several months or years that they have been using the same.
- Me and my shadow: In this exercise the team members literally sit next to the user and watches what they do and how.
- Sailboat
- Start your day: In this game the participants are asked to describe their monthly, weekly and daily activities relative to the usage of the product.
- My worst nightmare

### Best Practices for User Stories

- If the customer is not available for writing the user stories, consider **user proxies**.
- **Slice the cake vertically** – Instead of splitting stories along technical components, consider one that includes end-to-end functionality and touches each layer. The term Sashimi is used to describe this.
- **Splitting a user story vertically** also aids in building a '*tracer bullet*', which is a very thin slice of functionality built to illustrate end-to-end functionality that passes through all layers of an application and helps to determine feasibility and appropriate connectivity between interfaces.
- **Closed Stories** – Instead of being an ongoing activity, a story should be such that it achieves a specific and meaningful goal.
- **Write it in simple language** – Stories need to be described using simple and understandable language

## Estimation

### Units of Estimation

- Relative sizing
- T-shirt Sizes
- Ideal time: ideal time is the number of days or hours a task will take if you focus entirely on it and work on it without any interruptions
- Story points

### Advantages of story points over ideal days

- Story points are a “pure” measure of relative size, the estimate of a story in not related to the proficiency of the person developing it.
- It is faster to estimate in terms of story points. It is only the relative size of the story that matters in arriving at the estimate.

### Estimation techniques

- **Affinity estimation:** In the Affinity estimation technique the Agile team groups user stories or product backlog items (abbreviated as PBI’s) based on their similarities in complexity and size. This technique works well when 40 or more stories need to be estimated quickly.
- **Wideband Delphi:** In this technique a group of experts come together and anonymously provides estimates that are collated with an aim that a consensus will be reached.
- **Planning poker**

## Velocity

Iis used to measure how much the team accomplished during a particular interval.

### Computing Initial Velocity of the Team

- **Based on historical data:** if the project is similar to a previous one based on technology, domain, environment and team composition.
- **By running a few iterations:** if it’s possible to initiate a project, its best to run 2–3 iterations and predict the velocity based on the observed velocity of the team in motion.
- **Deduce based on expert judgment or hypothesis**

### Ways to Improve Velocity

- Continuously focus on refactoring code, so as to remove all traces of technical debt.
- Motivate the people to give in their best
- Ensure team members are fully focused and they can keep distractions at bay.
- Engage with the customer and solicit his presence so as to clarify requirements

## Types of release Planning

- **Date-driven release plan:** In this case the releases must be completed by a given date, but the scope of features included in that release is negotiable.
- **Functionality-driven release plan:** the aggregate of features are predetermined to be delivered together, but the team progress to deliver as soon as possible.

## Release burndown charts

- When work is completed by the development team, the top of the bar is lowered.

- When work is reestimated, the top of the bar moves up or down based on whether the estimate is increased or decreased respectively.
- When new work is added by the product owner, the bottom is lowered below the X-axis.
- When work is removed by the product owner, the bottom is raised and can even be raised above the X-axis.

# Domain VI. Problem Detection and Resolution

Continuously identify problems, impediments, and risks; prioritize and resolve in a timely manner; monitor and communicate the problem resolution status; and implement process improvements to prevent them from occurring again.

## Risk management

A risk is defined as an uncertain event or condition that, if occurs, has an effect on at least one project objective (e.g., on scope, cost, duration, quality, or customer satisfaction).

### Risk identification

There are a few dedicated ceremonies where risks are discussed explicitly:
Iteration planning, Daily Scrum meeting and Iteration retrospective

### Risk analysis

The main characteristics of risk are the probability, impact, frequency and timing of the risk

### Risk categorization

The acronym PESTLE: Political, Environmental, Social, Technological, Legal, Economic is used to categorize risks

### Probability impact matrix

Let us again refer to the two critical dimensions of risks: Probability, Impact

### Risk quantification using EMV

Agile teams use the concept of *Expected Monetary Value (EMV)* to obtain the value of the risk using the formula: ***EMV = Risk impact x Risk probability***

### Risk responses

The ultimate goal of risk management is to take proactive action to decrease the probability or impact of a risk. This is called a risk response strategy. There are four prevalent risk response strategies:

- **Avoid**: the project is driven in a direction that deviates it from the risk and its impact.
- **Transfer**: is shifted to a third party, often in lieu of a premium or fee charged by the third party in owning the liability of the risk.
- **Mitigate**: the aim is to decrease the impact and / or the probability of the risk.
- **Accept**: the risk is accepted – letting it happen, if it happens. However, the team can choose to allocate a contingency amount to it to cover the impact of the risk.

### Risk monitoring

- **Spikes**: agile teams use the term spike to refer to a time-boxed research activity. The outcome of the spike helps the team to check feasibility of architecture or design options, make better quality of estimates, fail fast ...
- **Checkpoints for frequent feedback**: the checkpoints help the team to make decisions at the last responsible moment. This follows the inspect-and-adapt style of Agile and Deming's Plan-Do-Check-Act cycle
- **Risk-adjusted backlog**
- **Risk burndown graphs**: similar to a release burndown chart, the risk burndown chart shows how the cumulative risk severities of the project are progressing over time.

## Quality control practices in Agile

- Incremental delivery: one of the characteristics of incremental delivery is simple emergent design
- Iterative delivery: frequent validation and verification
- Small releases: that eliminate prolonged development cycle and complex testing cycles.
- Eliminating non-value added activities.
- Limit WIP and swarm to resolve problems collectively
- Scrum teams collaborate with the product owner and the users to determine the acceptance test cases
- Definition of done, code reviews, coding conventions, pair programming, test-driven development, continuous build and integration, retrospectives
- Team involvement and participation in all ceremonies

# Problem resolution

## Process of problem solving

- Define the problem
- Analyse the problem, its origin, its context, its impact
- Identify a solution to address the problem
- Implement the solution
- Review and confirm that the problem is solved

### Techniques for problem solving

- **Divide and conquer**: breaking down complex problems into simpler and more manageable problems that are easier to resolve.
- **Expert judgement:** consulting with a subject-matter expert who has ‘been-there, done-that before’.
- **Simulation**: scaling down the complexity of the problem into a simpler model that is easier to control and trying out multiple permutations and combinations of scenarios.
- **Brainstorming / war room**: gathering all experts together in one place to come up with ideas, logical reasoning, debate, discuss. and plan action items.
- **Metaphors**: using analogy of the solution for a previous problem to resolve the current problem.
- **Trial and error:** trying different alternatives repeatedly until the resolution is obtained.
- **Spikes**: experimenting with alternate options to check out viability and feasibility of the solution.
- **Trend analysis**: trying to figure out a pattern in the system behavior.
- **Probing** – asking a series of questions to get to the bottom of the problem, rather than treating it based on symptoms.
- **Sandboxing**: solving the problem in a smaller and controlled environment, before applying the fix on the live system.

# Domain VII. Continuous Improvement (Product, Process, People)

Continuously improve the quality, effectiveness, and value of the product, the process, and the team.

## Product improvement

### Continuous improvement of product quality and effectiveness

Verification: are we build the right product for the customer ?
Validation: are we building the product right

### Dissemination of knowledge

As products grow in complexity, it is important for organizations to ensure that knowledge is sticky within the project team. Here is a variety of ways in which Agile teams invest behind dissemination of knowledge among each other:

- Using forums for collaboration like brainstorming, group decision-making for planning, estimation, problem solving. and risk mitigation.
- Iteration demos to share knowledge between stakeholders on what the team achieved during an iteration.
- Information radiators to share project metrics and progress with stakeholders.
- Pair programming and pair rotation so that knowledge of the system implementation is shared between multiple team members, leading to collective ownership of code.
- Continuous build, integration, automated and exploratory testing to provide real- time feedback on the code.
- Conducting spikes to gain knowledge on unfamiliar technologies.

## Process improvement

**Kaizen**: Kai(change) + zen(for better) is characterized by many small and continuous changes that cumulatively result in big improvements over time. Similar to Deming's Plan-Do-Check-Act cycle.
**Process analysis:** Agile teams look to improving the efficiency and effectiveness of their processes by identifying and removing overheads, constraints, or anything that does not add value.
**Lean 5S technique:** The acronym 5S stands for sort, set in order, shine, standardize and sustain.
**Kanban Kata:** Kanban Kata is another continuous improvement strategy that uses a series of questions to help improve in small steps such that day-to-day work and improvements happen simultaneously.
**5 Why's technique:** The 5 Why's technique is used by teams to identify the root cause of a particular problem by asking a series of 'why' questions.
**Fishbone diagram:** Also called the Ishikawa diagram, is an extension of the 5 why's technique to determine the root cause of a problem.
**Pareto Diagrams (80-20 rule):** The rules state that 80% of the problems are due to 20% of the causes. Alternatively stated, 80% of the system errors can be removed by resolving 20% of the defects.

# Retrospective

**Styles of retrospectives**
**SAMOLO - Same as – more of – less of:**

- Same as: are those processes and practices that the team finds value and would continue to use.
- More of: are those processes and practices that the team find value, but feels it's not doing enough of.
- Less of: are those processes and practices that are team starting finding diminishing value and would like to discourage.

**Start doing – stop doing – keep doing**

- Start doing: are the activities that the team feels are not done, but is worth doing because they will generate better outcomes.
- Stop doing: are the activities that the team feel are not useful, generally disliked, or not worth doing, so should be stopped.
- Keep doing: are the activities that are liked, adding value and the team feels are worth continuing.

## Steps of a retrospective

There are five steps:
**Set the stage:** The first step in the retrospective meeting is to set the stage by getting participants comfortable to speak openly without any fear of retribution or potential conflicts in airing concerns : *Check-in, Focus on / focus off, ESVP, Working agreements.*
**Gather data:** Is to collate data such that team members can visualize what happened during the iteration. *Timeline, Triple nickels, Color code dots, (Mad, sad, glad), Locate strengths, Satisfaction histogram, Team radar, Like to like.*
**Generate insights:** Once the data is gathered, the third step in the retrospective meeting is to analyze the data and generate meaningful insights and conclusions from it. *Brainstorming, 5 why's, Fishbone, Prioritize with dots, Identify themes*
**Decide what to do:** Insights, once generated, needs to be translated into action. This fourth step of the retrospective focuses on the highest priority items and devises an action plan that can be implemented during the next iteration itself. *Retrospective planning game, SMART goals, Circle of questions, Short subjects.*
**Closing the retrospective:** The last step of the retrospective meeting is to close gracefully with expression of appreciations of the time devoted by the participants. *Appreciations, Helped/Hindered/Hypothesis, Return on time invested (ROTI)*

## Pre-mortem / pre-failure analysis

In a pre-mortem, the team members are asked to determine possible reasons what can go wrong or why the project could fail *in future*. Pre-mortems, logically, are similar to risk identification and analysis

# People

## Feedback methods

- Make the feedback constructive in nature.
- Positive feedback is easy to communicate. But negative feedback should be carefully delivered, otherwise it can get misconstrued very easily.
- Make the feedback relevant and related to the overall goals and objectives of the

individual and that of the project.
- Be specific, back up with sufficient data and give instances of observations and examples
- Praise in public, but criticize in private.
- Give the feedback at the appropriate time such that the issue can be addressed promptly.
- Let the feedback be regular.

# Agile adoption

## Agile hybrid models

Some organizations adopt a model that is more of a hybrid between traditional waterfall and Agile approaches. In some literature the word Water-Scrum-Fall can be found.

# PMI® Code of Ethics and Professional Conduct

## Purpose of the Code

PMI has created a practical framework consisting of four key values of honesty, responsibility, respect and fairness that drive ethical conduct.

## For Whom Does the Code Apply?

All members of the Project Management Institute (PMI®)
All who hold a PMI® certification like PMP®, PMI-ACP®, PMI-RMP®, PgMP®, CAPM® All who are pursuing a certification from PMI®
All volunteers of PMI® for chapter events like seminars and conferences

## Four Core Values of the Code

### Responsibility

The code states that responsibility is our duty to take ownership of the decision and actions we make or fail to make and their resulting consequences.

- We make decisions and take actions that serve the best interests of the society and the environment.
- We accept assignments and fulfill project and professional obligations that are commensurate with our qualifications and skills.
- We fulfill commitments dutifully, maintaining professional integrity at all times.
- Wherever we notice errors or omissions, we own up, communicate asis and take appropriate corrective actions promptly.

### Respect

The code states that respect is our duty to show a high regard for ourselves, others and the resources entrusted to us that include people, money, reputation, the safety of others and natural or environmental resources.

- Create an environment based on trust and confidence where diverse perspectives and views are encouraged, listened and valued.
- We respect cultural diversity and avoid any behaviors that are considered inappropriate or disrespectful.

### Fairness

The code states that fairness is our duty to make decisions transparently and act objectively with any partiality, bias, self-interest, or prejudice.

- We make decisions in a transparent and impartial manner.
- We try to steer away from potential conflict-of-interest situations, provide full disclosure to the authorities proactively and refrain from being involved or influencing or making any decisions in such a situation.
- We provide a fair ground and equal opportunities by allowing competitors access to the same resources (e.g., data and systems) that they are entitled to.
- We act fairly during evaluating and hiring of resources and do not base decisions on personal consideration or any discrimination.

**Honesty**

The code states that honesty is our duty to understand the truth and act in a truthful manner. Here are a few standards in this regard:

- We seek truth and create an environment where truth can prevail at all times.
- We do not mislead our stakeholders by giving them delayed, partial, incomplete, or inaccurate information.
- We provide communication that is accurate, reliable and timely even if that applies to delivering bad news or news that is expected to create a negative outcome.
- We do not shift blame to others when sharing bad news.