# Introducing Scrum in SAFe

## Examine basic Agile development concepts

**The Agile Manifesto**

Individual and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following the plan

**Agile Manifesto principles**

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

*Customers are happier when they receive working software at regular intervals, rather than waiting extended periods of time between releases.*

1. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

*The ability to avoid delays when a requirement or feature request changes.*

1. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

*Scrum accommodates this principle since the team operates in software iterations that ensure regular delivery of working software.*

1. Business people and developers must work together daily throughout the project.

*Better decisions are made when the business and technical team are aligned.*

1. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

*Motivated teams are more likely to deliver their best work than unhappy teams.*

1. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

*Communication is more successful when development teams are co-located.*

1. Working software is the primary measure of progress.

*Delivering functional software to the customer is the ultimate factor that measures progress.*

1. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

*Teams establish a repeatable and maintainable speed at which they can deliver working software, and they repeat it with each release.*

1. Continuous attention to technical excellence and good design enhances agility.

*The right skills and good design ensures the team can maintain the pace, constantly improve the product, and sustain change.*

1. Simplicity--the art of maximizing the amount of work not done--is essential.

*Develop just enough to get the job done for right now. Identify the item which is not important for business and no need to do. (Must have vs nice to have)*

1. The best architectures, requirements, and designs emerge from self-organizing teams.

*Skilled and motivated team members who have decision-making power, take ownership, communicate regularly with other team members, and share ideas that deliver quality products.*

1. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

*Self-improvement, process improvement, advancing skills, and techniques help team members work more efficiently.*

* **Waterfall**: Based on Cost and Schedule will have Requirements to work it upon.
* **Agile**: Based on Features will decide Cost and Schedule of the project.
* Agile Teams show that dates matter and they meet their commitments.
* Business owners understand how priorities matter.
* Fix quality, not scope.

**Agile Frameworks**

* SAFe
* Scrum
* Crystal
* Kanban
* eXtreme Programming (XP)
* Features-Driven Development

**Agile Practices**

* Timeboxing
* User Stories
* Daily Stand-ups
* Frequent Demos
* Acceptance Test-Driven Development
* Information Radiators
* Retrospectives
* Continuous Integration

## Explore Scrum basics

**3 Pillars of Scrum**

* Transparency
* Inspection
* Adaption

**Scrum Values**

* Courage *(Don’t be in comfortable zone and challenge yourself)*
* Commitment *(For work)*
* Focus *(On goal instead of doing multiple things at same time)*
* Respect *(To each other in team)*
* Openness *(Involve and communicate)*

*Note: PO and SM will be chicken and Agile team will be called as Pig.*

**Iteration Basic**

* Iteration can be 1-4 weeks. SAFe advice for 2 weeks.
* Goal is to deliver working at the end of each iteration.
* Avoid adding scope once the iteration has begun.
* Team composition does not change during iteration, otherwise the velocity is invalid.

**Team Backlog**

* It represents opportunities, not commitments.
* Stories and Enablers may be estimated but estimates do not imply committed delivery.
* It has a single owner: team’s PO.
* Created by the Agile team, prioritized by the PO.
* Stories and Enablers for the next iteration are more detailed as compared with later iterations.
* Backlog items must have all 4 layers:
  + Presentation Layer
  + Business Logic
  + Data Access Layer
  + Data Sources & 3rd Party APIs.
* NFRs (Non-Functional Requirements): Regression testing, Performance testing, Security, Scalability and Maintainability.

**Scrum Events**

* Backlog Refinement (~1 hour): Prepare requirement for iteration planning.
* Iteration Planning (2-4 hours): Team commits to a set of goals to be delivered in the iteration.
* Daily Stand-up (<=15 mins): Team members sync regarding the progress of the iteration goals.
* Iteration Review (~1 hour): Deliverables reviewed with stackholders providing feedback.
* Iteration Retrospective (1-1.5 hours): Team reviews and improves its process before the next iteration.

|  |  |
| --- | --- |
| Backlog Refinement | Team (Own), SM and PO (Good to Have) |
| Iteration Planning | SM (Own), Team and PO (Must or Good to have) |
| Daily Stand-up | Team (Own), SM and PO (Optional) |
| Iteration Review | SM (Own), Team and PO (Must have) |
| Iteration Retrospective | Team (Own), SM and PO (Optional) |

**Scrum Master**

* Coaches team improvement using values, principles and best practices.
* Facilitates Scrum team events.
* Protects the development team.
* Helps to remove impediments. *(SM won’t remove blocks, just help to the team)*
* Is a Servant leader.
* *Unbiased and no delivery dependency with respect to coding.*

**Product Owner**

* The single voice of the customer and stakeholders in the team.
* Owns and manages the Team backlog.
* Defines and accepts requirements.
* Make the hard calls on scope and content.

**Agile Team**

* Typically, 3-9 people (Excluding SM and PO)
* Everyone who is needed to define, build and test.
* Team members are only on this team (No shared or buffer resources)
* Self-organizing and accountable, collaborative, cross-functional, empowered.

Agile team responsibilities

* Listen and talk to people
* Seek and accepts help
* Work on items in an order set by the PO
* Be proactive and self-motivated
* Be honest (with yourself and others)
* Embrace “All sink or All swim”

## Position an Agile team in SAFe Enterprise

**SAFe Core Values**

1. Built-in Quality
2. Program Execution
3. Alignment
4. Transparency

**SAFe Lean-Agile principle**

* Take an economic view *(Earning/Profit after each iteration)*
* Apply system Thinking *(How all align together)*
* Assume variability; preserve options *(More clarify and understanding)*
* Build incrementally with fast, integrated learning cycles *(Integrate ASAP instead of doing late)*
* Base milestone on objective evaluation of working systems
* Visualize and limit WIP, reduce batch sizes and manage queue lengths
* Apply cadence, synchronize and cross-domain planning *(Release plan and duration of delivery should be same)*
* Unlock the intrinsic motivation of knowledge workers *(Keep environment healthy and motivated)*
* Decentralize decision-making *(Decision can be taken by team or management. Team can’t take decision for other team or for management)*

**Agile Release Train (ART)**

* A virtual organization of 5-12 teams (50 - 125+ individuals) that plans, commits and executes together.
* Program Increment (PI) is a fixed timebox; default is 10 weeks.
* Aligned to a common mission via a single Program Backlog.
* Operates under architectural and UX guidance. (UX: User Experience)
* *Business owners not part of ART. They create vision and mission.*
* *Product Management is chief PO and owns Product backlog. Product Manager reach to the customer and priorities the Product backlog.*

**ART Delivers Solutions**

* Define new functionality
* Implement
* Acceptance Test
* Deploy

**ART Roles**

* Release Train Engineer acts as Chief Scrum Master for the train.
* Product Management owns, define and prioritizes the Program backlog.
* System Architect/Engineering provides architectural guidance and technical enablement to the teams on the train.
* System team provides processes and tools to integrate and evaluate assets early and often.

*One system team for each train. Do automation if possible with respect to time. Integrate all team work and test it ASAP.*

* Business owners are the key stackholders on the ART.

**ART Program Events**

* PI planning (2 Days): Teams commits to a set of objectives to be delivered in the PI.

*Own by RTE.*

* ART Sync (1 hour): Train teams to sync regarding the progress of the PI.

*Own by RTE or Product Management. Every Week. (RTE, Product Management and POs)*

* System Demo (2 hours): Delivered reviewed with stackholders providing feedback.

*By system team once in iteration. (All and Stakeholders)*

* Inspect and Adapt Event (1/2 Day): The train reviews and improves its process before the next PI.

*Once in quarter or Each PI. Solution Demo. Present all matrix to stakeholders*

**Portfolio Level**

* Kanban system provides Portfolio visibility and WIP limits.
* Enterprise architecture guides larger technology decisions.
* KPIs support governance and improvement. (KPI: Key Performance Indicator)
* Value delivery via Epics

**Larger Solution Level**

* Coordinates development of large solution.
* Synchronizes multiple ARTs and suppliers.
* Integrates suppliers as partners.
* Managing solution intent.
* Value delivery via Capabilities.

**EPIC (MS Office) => Capability (MS Word || Business Use case) => Features => User Stories**

# Characterizing the role of the Scrum Master

## Examine the responsibilities of the Scrum Master role

* Coaches the team to create better solutions, improve business results, and better their processes.
* Facilitates team and program events.
* Helps to remove impediments.
* Assists the team in implementing SAFe and working with other teams who may or may not be using SAFe.
* Assists the PO in preparing and refining the backlog for PI and Iteration Planning.
* Coaches the team on the best ways to refine their backlog and create stories.
* Coordinates with other Scrum Master, the System Team and Shared services in the ART PI Planning events. *(Work with them throughout PI and each Iteration)*
* Coordinates with other Scrum Masters and Release Train Engineer in the Scrum of Scrums.
* Fosters normalized estimating within the team.
* *Helps teams operate under architectural and portfolio governance, System Level integration and System Demo.*

## Explore the characteristics of an effective Scrum Master

* Listens and supports team members in problem identification and decision-making. *(SM should be good listener)*
* Understands and empathizes with others.
* Encourages and supports the personal development of each individual.
* Motivates rather than uses authority.
* Think beyond day-to-day activities.
* Looks to help without diminishing the commitment of others.
* Coaches the team on Agile best practices.

# Experiencing PI Planning

## Prepare to experience PI Planning

**Before PI Planning**

The program planning calendar can be set for a year in advance:

|  |  |
| --- | --- |
| **Program Level Calendar** | **Team Level Calendar** |
| * PI Planning * Demos * Inspect and Adapt workshops | * Iteration Planning * Iteration Review * Iteration Retrospectives |

**PI Planning**

Cadence-based (Calendar date based) PI Planning events are the pacemaker of the Agile Enterprise.

* Two days (Every 8-12 Weeks || 10 Weeks is typically)
* Everyone attends in person if at all possible.
* Product Management owns Features priorities.
* Development teams own Story planning and high-level estimates.

**Features have benefit hypothesis**

* Acceptance criteria is typically defined during Program Backlog refinement.
* Reflect functional and Non-functional requirements.

**Features are implemented by Stories**

Features are decomposed into Stories by the teams on the train:

* Teams on the train collaborate to deliver Features.
* Features are implemented incrementally via US.
* Teams demonstrate working increments of Features by delivering stories on a regular cadence.
* Stories fit in one iteration for one team.

*Note: SAFE recommended to deliver at-least one feature per iteration*

**User Story guidelines – The 3Cs**

* Card: Can be written on a card or in tool.
* Conversation: The details are in a conversation with the PO.
* Confirmation: Acceptance criteria confirm the story correctness.

**As a <User Role> I want <Activity> so that <Business Value>**

* **User Role** is the description of the person doing the action.
* **Activity** is what they can do with the system.
* **Business value** is why they want to do the activity.

**Acceptance Criteria**

* Acceptance criteria provide the details of the Story from a testing point of view.
* Acceptance criteria express the conditions that need to be satisfied for the customer.
* Acceptance criteria provide context for the team, more details of the story, help the team know when they are done.
* Acceptance criteria are written by the customer/PO and refined by the team during backlog grooming and iteration planning.
* User Acceptance Test Scenarios are a good starting point for acceptance criteria.
* Options to split a large Story into smaller Stories.

INVEST in a Good Story

I Independent

N Negotiable *(Something which we didn’t understand then split it)*

V Valuable

E Estimable

S Small *(2-3 Days work)*

T Testable

**Enablers**

They can represent different types of work:

* Exploration *(Spike – No need to estimate)*
* Architecture
* Infrastructure

Enabler Stories are considered like any other Story.

*Note: SPIKE are like POC, Not Estimate and won’t go to PROD.*

Below are 2 types of Enablers:

1. **Refactors**: Systematic approach to improve the system without changing observable system behavior. Ex: Improving Maintainability, Performance or Scalability.
2. **Spike**: Research activities to reduce risk, understand a functional need, increase estimate reliability or define a technical approach.
   1. Technical Spike: Researching a technical approach or unknown.
   2. Functional Spike: Researching how a user might use or interact with the system.

**Estimate Stories**

A story point is a singular number that represent:

* Volume *(How much is there?)*
* Complexity *(How hard is it?)*
* Knowledge *(What do we know?)*
* Uncertainty *(What’s not known?)*
* *Increase accuracy by including all perspectives*
* *Builds understanding*
* *Creates share commitment*

**Scrum Master’s Role in facilitating estimation**

* Make sure everyone participates
* Ensure relative estimates are used
* Focus the discussion on contested/challenged items
* Keep time spent estimating stories to a minimum.
* Identify SME who need to be present

Common Anti-Patterns

* Pressure by stakeholders to lower estimations
* Only a few people participate
* Not using the adjusted Fibonacci scale

## Create and review draft PI Plans

* The RTE facilitates the PI Planning event and kicks off the briefings.
* The executive, Product Manager, System Architect/Development Manager/UX role conduct their briefings to the entire Agile Release Train.
* The RTE briefly reviews the purpose of the meeting and presents the agenda, planning guidance and planning requirements.
* The executive presents the business context slides.
* The Product Manager presents the Vision and Features and benefits slides.
* The System Architect/Development Manager/UX role presents the Architecture, UX and Development manager briefing slides.

**Day 1**



**Day 2**



**Business Context – SWOT** (Presented by Executive for online bookstore)

|  |  |
| --- | --- |
| Strength   * The best software engineers in the industry. * A nimble/quick/agile organization. * Adopting SAFe * Collocated * Great Strategy | Weaknesses   * Difficulties finding qualified FTEs in DevOps. * No System Team in place |
| Opportunities   * Develop new online social experiences rather than copying existing one. * Accelerate global expansion through faster content translation. * Develop product offerings beyond books. * Build an advertising model. | Threats   * Amazon has clear dominance. * LinkedIn tech companies continue to grow. |

**Stretch Items** (Presented by RTE)

Stretch items do count in velocity/capacity:

* They are planned, and aren’t extra things teams need to do, “just in case team have time”.
* They are not included in the commitment, thereby making the commitment more reliable.
* If a team has low confidence in meeting a PI objective, encourage then to move it to stretch items.
* If an item has many unknowns, consider moving it to stretch, and put in early spikes.

*Note: Stretch item will take but not commit. If things are clear, then implement else leave it.*

**SMART Team PI Objectives** (Presented by RTE)

Team should write their PI objective in SMART format:

* **Specific**: State the intended outcome as simply, concisely and explicitly as possible.
* **Measurable**: It should be clear what a team needs to do to achieve the objective. Yes/No, Quantitative or provide a range.
* **Achievable**: Achieving the objective should be within the team’s control and influence.
* **Realistic**: Recognize factors that cannot be controlled. Avoid making assumptions.
* **Time-bound**: The time period for achievement must be within the PI and therefore all objectives must be scoped appropriately.

*Team Deliverables*

* *If a story has a dependency, put a red sticky on it describing the dependency.*
* *If a risk is broader in nature, put it on the risk sheet.*
* *If needed, allocate a percentage of capacity for unplanned activities such as maintenance and production support.*
* *You should have a velocity but not a load on the IP iteration (Innovation and Planning), since it should not contain any user value stories.*
* *Program risks are those that need to be escalated to the program level. They will be captured and “ROAMed” after the final plan review.*

**Scrum of Scrum**

* The SoS helps teams to be on track and facilitate early identification of risks.

## Finalize plans and establish business value

Possible changes during planning adjustments:

* Business priorities
* Adjustment to plan
* Changes to scope
* Movement of people

## Review final plans and commit to a set of PI Objectives

**Final Plan Review**

All Teams and Business Owners peer review all final plans. Find below agenda:

* Changes to velocity (capacity) and load
* Final PI objectives with business value
* Program risks and impediments
* Q & A

Building the final Plan

* Final plans are collected at front of the room.
* Final plans are reviewed by all teams.
* Business owners are asked whether they accept the plan.
* If so, the team’s plan and program risk sheet are brought to the front of the room.
* If not, the plans stay in place and team continues planning after the review.

**ROAMing Risks**

* **Resolved**: Has been addressed and no longer a concern.
* **Owned**: Someone has taken responsibility.
* **Accepted**: Nothing more can be done. If risk occurs, release may be compromised.
* **Mitigated**: Team has plan to adjust as necessary. *(Have plan but no bandwidth so need to adjust)*

**Confidence Vote**

After dependencies are resolved and risks are addressed, a confidence vote is taken at the team and Program level.

“Fist of Five” confidence vote. (Range of 1-5 || 1=No confidence and 5=Very high confidence)

**Retrospective for Planning meeting**

Retrospective helps to improve continuously the process:

* What went well
* What didn’t
* What we can do better next time

## Facilitate on effective PI Planning process

**Scrum Master’s role in team breakout**

* Ensure the team has a draft plan to present.
* Identify as many risks and dependencies as possible for the management review.
* Secure subject matter experts and Program Level stakeholders as needed by the team.
* Facilitate the coordination with other teams for dependencies.

Common Anti-Patterns

* No Plan or partial plan at the end of the timebox.
* Too much time is spend analyzing each story.
* Shared Scrum Masters and Product Owners are not available enough.
* Part-time Scrum Masters don’t have time to plan as part of the team.

**Scrum Master’s role in PI Planning**

* Maintain the timebox.
* Make sure the team builds a plan they can commit to.
* Ensure that the team is honest in their confidence vote.
* Facilitate the co-ordination with other teams, but don’t do it for the team.
* Act as a request buffer for a team that has a lot of dependencies.
* Manage the program board.
* Facilitate the retrospective.

Common anti-patterns

* Pressure is put on the team to overcommit.
* Team under-commits due to feat of failure.
* Over-planning ahead of time to make it more efficient loses the essence of PI Planning.
* The plan, rather than the alignment, become the goal.

# Facilitating Iteration Execution

## Experience an Iteration

## Plan the Iteration

* The team establish its velocity.
* The team clarifies the Stories.
* The team optionally breaks stories into tasks.
* The process continues while there is more capacity.

*Timebox: 4 hours*

*This meeting is by and for the team.*

*Subject matter experts (SMEs) may attend as required. PO will join if he/she is available.*

**Story analysis and estimation**

* PO presents stories in order of priority.
* Each story
  + - Is discussed and analyzed by the team
    - Has its acceptance criteria defined and refined.
    - Is estimated
* The process continues until the estimation of the stories has reached the velocity of the team.

**Iteration Goals**

Iteration goals provide clarity, commitment and management information. They serve three purpose:

* Align team members to a common purpose.
* Align Agile teams to common PI objectives and manage dependencies.
* Provide continuous management information.

**Commit to the Iteration Goals**

A team meets its commitment: By doing everything they said they would do.

Commitment: Too much holding to a commitment can lead to burnout, inflexibility and quality problems.

Adaptability: Too little commitment can lead to unpredictability and lack of focus on results.

*Team commitment are not just to the work. They are committed to other teams, the program and the stackholders.*

**Scrum Master’s role in Iteration Planning**

* Maintain Timebox
* Ensure that the team commits to the iteration Goals
* Verify that the PO or other managers don’t influence the team to overcommit.
* Challenge the team to exceed their previous accomplishments.
* Ensure the improvement items from the retrospective are put into effect.
* Ensure time is allocated for technical debt activities.

Common anti-patterns

* Delving too deep into technical discussions.
* Commitment is unrealistic.
* Velocity and load are exactly the same.
* Scrum Master is more focused on technical hat than facilitator’s hat.
* The team under-commits due to fear of failure.
* No time is reserved for support activities.

*If commitment reliability is more than 80% means, we are good.*

*Keep some bandwidth for support or ad-hoc activities.*

*Velocity should not be equal to load; team won’t challenge themselves.*

## Track the Iteration progress

**Daily stand-up (DSU)**

The DSU is not a daily status meeting for management. It is used to: *(DSU object is to plan for the day)*

* Share information about progress
* Coordinate activities
* Raise blocking issues

*Every day at the same time in front of team board.*

*Timebox of 15 mins*

*Not a problem-solving session*

*Update the board.*

* Many Scrum teams use Iteration burn-down charts.
* Burn-downs count the remaining effort (Stories, tasks etc.)
* As we don’t advocate tasks in SAFe, we prefer burn-ups and CFDs.
* *Burn-down work at program level. As team committed velocity but after that something adding new at the program level.*
* *Cycle Time: Start Analysis to completing the work.*
* *Lead Time: Concept to the Production. Will have many cycle time.*

**Collaboration with other teams**

* The team should integrate their work often with other teams in the program (multiple times per iteration)
* Work with the system team on automated system level tests.
* Join their Daily stand-up when important issues arise.
* Join their demo or planning.
* Work with the System Architect to better manage dependencies with other teams.

**Scrum of Scrums**

Programs continuously coordinate dependencies through a Scrum of Scrum (SoS).

* The SoS is a meeting for Scrum Masters and the Release Train Engineer to gain visibility into team progress and program impediments.
* It is typically held twice per week.
* It is timeboxed but is followed by a “Meet After” for problem-solving.

*Coordinate the implementation of Program Level improvement backlog items.*

**Scrum Master’s role in tracking Iteration progress**

* Facilitate mid-PI re-planning.
* Encourage the team to point out as early as possible if they think will miss Iteration goals or PI objectives. Communicate to and from the Scrum of Scrums.
* Encourage the use of engineering practices.
* Make sure defects are not pushed to the IP Iteration.
* Facilitate preparation for the next PI.
* Support release activities.

Common anti-patterns

* Team gets no input from Scrum of Scrums.
* Teams are unwilling to change or add objectives mid-PI.
* Scrum Master does all the synchronization, so team is incapable of doing it themselves.

*PO own rights to cancel the iteration.*

*Don’t wait for 5th Iteration (IP) for additional work and defect.*

## Refine the Backlog

* Timebox: 1-2 hours’ weekly
* Helps the team ‘sleep’ on new stories prior to Iteration Planning.
* Provides time to identify dependencies and issues that could impact the next iteration. Ensures that we have a ready backlog for Iteration planning.
* Agile team members are in attendance and actively engaged, subject matter experts and other teams’ members are invited as needed.

**Scrum Master’s role in backlog refinement**

* Maintain timeboxes.
* Maintain the right level of a deep backlog vs ready backlog for two iterations.
* Make sure all the team members participate.
* Invite the right SMEs.
* Hold the event at regular intervals.

Common anti-patterns

* Arriving to the Iteration with non-ready Stories.
* Not doing the backlog refinement consistently.
* Team sees Stories for the first time during Iteration or PI Planning.
* Feature estimations impact Story estimation.

## Facilitate the iteration Review and System Demo

**Iteration Review**

* Timebox: 1-2 hours
* Provides the true measure of progress by showing working software functionality, hardware components, models, prototypes etc.
* Preparation for the review starts with planning.
* Teams demonstrate every Story, Enabler and NFR.
* Attendees are the team and its stackholders. *(Stakeholders mostly join System Demo instead of Iteration demo)*
* If major stackholders cannot attend, the PO should follow up individually.

**System Demo**

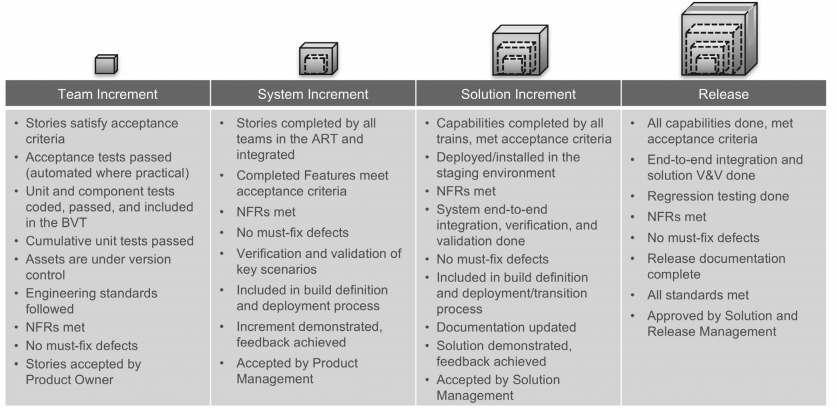
The System team/Product Management/Product Owners demonstrate the Solution increment to the ART stackholders.

What does ‘Done” mean

Goal for the end of every iteration:

* Working system
* No remaining work
* Everyone on same page regarding what was completed
* Standard followed
* Work accepted by the PO
* *NFRs too (Non-Function requirement)*

**SAFe Definition of DONE**



Benefits of a Definition of Done

* Aligns expectation among team members, Product owner and stackholders regarding what ‘Done’ means.
* Helps to maintain predictability.
* Avoid unfinished work that leads to technical debt.

*Building quality in:*

* + - *Ensures that every increment of the solution reflects quality standards.*
    - *Is required for high, sustainable development velocity.*

**Continuous System Integration**

Teams continuously integrate assets:

* Integrate every vertical slice of a User Story.
* Avoid physical branching for software.
* Frequently integrate hardware branches.

Test first: Automate now

* Automated tests are implemented in the same Iteration as the functionality.
* The team that builds functionality also automates the tests.
* Create an isolated automated test environment.
* Actively maintain test data under version control.
* Passing vs not-yet-passing and broken automated tests are the real iteration progress indicators.

**Scrum Master’s role in the team and System Demo**

* Begin to consider how and what to demo in Iteration Planning.
* Make sure the right participants are present.
* Ensure that the team celebrates its accomplishments and that stakeholders acknowledge them.
* Make sure different team members have the opportunity to demo.
* Ensure that the team is ready for the System Demo and coordinates with the System Team.

Common anti-patterns

* A lot of time is spent for the demo.
* Demo is mainly talk/slides as opposed to working software and/or hardware.
* Meeting doesn’t happen if PO is unavailable.
* PO sees things for the first time in the Team Demo.
* System Demo is not done because the Team Demo is enough.
* Team members are not invited to the System Demo, to save time.
* Demos that are interesting/relevant to Program Level Stakeholders.

## Continuously deploy with DevOps

**Six Recommended Practices for Continuous Deployment (CD)**

1. Maintain development and test environments to better match production.
2. Maintain a staging environment that emulates production.
3. Deploy to staging every Iteration.
4. Automate testing of features and nonfunctional requirements
5. Automate deployment
6. Decouple deployment from release

**What is DevOps**

An agile approach to bridge the gap between development and operations to deliver value Faster and more reliably.

Development Operations

* + - Create change - Create stability
    - Add or modify features - Create or enhance services

*DevOps (A mindset)*

**CALMR Approach to DevOps**

* **C**ulture: Establish a culture of shared responsibility for development, deployment and operations.
* **A**utomation: Automate the continuous delivery pipeline.
* **L**ean Flow: Keep batch sizes small, limit WIP and provide extreme visibility.
* **M**easurement: Measure the flow through the pipeline. Implement application delivery.
* **R**ecovery: Architect and enable low risk releases. Establish fast recovery, fast reversion and fast fix-forward.

## Release on Demand

## Facilitate relentless improvement

(Action of Retro)

Agile Teams continuously adapt to new circumstances and improve the methods of value delivery.

* Understand where you are
* Foster the culture of ‘improving everywhere’
* Use retrospectives as summary points but not as limitations.
* Supports continuous learning.
* Actively engage with other SMs to drive improvement on the Program level.

**Creative Iteration retrospectives**

**Simple**: Three columns and open discussion.

**Appreciation**: Has someone helped you or helped the team?

**One Word**: to describe the iteration.

**Individually**: Write Post-Its and the find patterns as a group.

**Rate**: Rate the iteration on a scale of 1-5 and the brainstorm how to make the next a 5.

**Scrum Master’s role in the improvement**

* Encourage improvement between retrospectives.
* Coach the team on problem-solving techniques.
* Retrospective
  + - Start by reviewing the results of the previous retrospective.
    - Make sure each person speaks.
    - Make sure the meeting ends with actionable improvement stories that are added to the backlog.
    - Write down what people are saying exactly.
    - Take program concerns to the RTE.

Common anti-patterns

* The only focus is on what to improve and not what to preserve
* Focus on problems that are outside of the team’s control.
* Failure to achieve results.
* Inviting people outside the team (especially management) to the retrospective.

# Finishing the PI

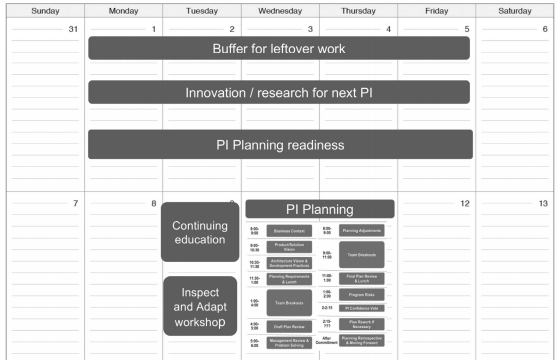
## Coach the IP Iteration

Innovation: Opportunity for innovation spikes, hackathons, and infrastructure improvements.

Planning: Provides for cadence-based planning and is an estimating guard band for cadence-based delivery.

Common anti-pattern

* Planning work for the IP Iteration in PI Planning.
* Leaving testing or bug fixing to the IP Iteration.
* Leaving integration of the whole system to the IP Iteration.



## Facilitate the Inspect and Adapt Workshop

Three Parts:

1. The PI system demo
2. Quantitative measurement
3. The Problem solving workshop

*Attendees: Teams and stakeholders*

*Timebox: 3-4 hours per PI*

**The PI System Demo**

* Often led by Product Management, Product Owners and System Team.
* Attended by Business Owners, Program Stakeholders, Product Management, Release Train Engineer, Scrum Masters and teams.
* Business value is assigned to the team’s PI objectives during or following the PI system Demo.

**Quantitative Measurement (Team PI Performance Report)**

* Planned total does not include stretch objectives.
* Actual total includes stretch objectives.
* % Achievement = Actual total/Planned total
* A team can achieve greater than 100% (As a result of stretch objectives achieved)
* Effort required for stretch objectives is included in the load (i.e. not extra work the team does on weekends)
* Individual team totals are rolled up into the Program Predictability Measure.

**The Problem - Solving Workshop**

The RTS facilitates the problem-solving workshop after a short retrospective. The workshop is done in organic or ad-hoc teams.

**Scrum Master’s Role in Inspect and Adapt**

* Facilitate the team preparation for the PI System Demo.
* Provide data
* Facilitate one of the teams in the problem-solving workshop.
* Help the RTE make sure improvement items are included during the PI.
* If using as-hoc teams for the I&A, then Scrum Masters may be participant rather than facilitators.

Common anti-patterns

* Only the PO presents in the PI system demo.
* No actionable improvement Features are created.
* Improvement items don’t enter the PI Planning process.
* Improvement items are not demoed in the PI System Demo.

# Coaching the Agile Team

## Act as a servant leader

Good Leaders must first become good servant.

A servant leader knows that his own growth comes from facilitating the growth of others who deliver the results.

**Behavior patterns of servant leader**

* Listen to and supports team members in the problem identification and decision-making.
* Understand and empathizes with others.
* Encourages and supports the personal development of each individual.
* Persuades rather than uses authority.
* Thinks beyond day-to-day activities.
* Seeks to help without diminishing the commitment of others.
* Is open and appreciates openness in others.

## Facilitate effective SAFe Team events

**The Challenge with meetings**

Meetings can be challenging because:

* The purpose is not clear
* There are no actionable outcomes
* They may result in unproductive conflict
* They may be boring
* Conversation may divert from the agenda into deep discussion.

**Running successful meeting**

Scrum Masters can benefit from the best practices from amazing meetings from companies like Apple and Google.

* Prepare for every meeting, no matter how short.
* Advertise a clear purpose and agenda.
* Identify a Directly Responsible Individual (DRI) for agenda/action items.
* Expect participants to know why they are attending, what contributions they will make and expected outcomes.
* Leave with clear action items.
* Advertise and keep to timeboxes.
* Be prepared to challenge and be challenged.
* Ge5t participants moving, use manipulatives, engage kinetically.
* Establish default decision; decision should never wait for a meeting.
* Don’t bring a problem without bringing at least one possible solution.
* Review actions taken to meet commitment-enforce accountability.
* Use “yes, and... “instead of “no, but… “to keep inputs positive and flowing.
* Take frequent breaks.
* Go the extra mile to bring remote participants into the discussion.
* Use parking lots to avoid too much details and/or going off on tangents.
* Communicate beyond the meeting.

## Coach the Agile Team using powerful questions

Coaches don’t give people the answer. Instead, they guide people to the solution.

**Scrum Master as coach**

|  |  |
| --- | --- |
| Move away from… | Move toward… |
| * Coordinating individual contributions * Acting as a Subject Matter Experts * Driving toward specific outcomes * Knowing the answer * Directing * Talking about deadlines and technical options * Driving the ‘right’ (your) decisions * Fixing problems rather than helping others fix them | * Coaching the whole team to collaborate * Being a facilitator * Being invested in the team’s overall performance * Asking the team for answer * Letting the team find their own way * Guiding * Focusing on business value delivery * Doing the right thing for the business right now * Facilitating team problem-solving |

## Guide team collaboration and resolve conflicts

**Some truth about teams**

* Teams are far more productive than the same number of individuals.
* Face to face communication is extremely efficient.
* Teams work best when not interrupted.
* Products are more robust when a team has all the cross-functional skills necessary.
* When teams themselves make a commitment, they will probably figure out how to meet it.
* Changes in team composition can impact productivity.
* Peer pressure is the best individual motivator.

**SAFe helps the five dysfunctions**

1. Inattention to results: Results are empirically reviewed at the end of every Iteration and Release. Team retrospectives drive continuous improvement.
2. Avoidance of accountability: Stakeholders, peer pressure and review of results drive accountability.
3. Lack of commitment: Teams make shared commitment to each other and to the external stakeholders.
4. Fear of conflict: Scrum creates safe environment for conflict; the Scrum Master encourages discussion of disagreements. Shared commitment avoids individual conflict that occurs when objectives are not aligned.
5. Absence of trust: The environment is safe. The team shares commitment and goals, displays hyper-transparency and engages in retrospectives.

**Avoiding ideological/personal conflicts**

A leader should spend far more time helping things go right than dealing with things that are going wrong.

* Help others see their teammates as human begins with their own needs, cares, worries and objectives (instead of as obstacles).
* Help the team set a common vision, goals and values.
* Start gradually, dealing with long-term tension within the team.
* Educate the team on achieving consensus.
* Build ‘relentless collaboration’
* Master proven conflict-resolution techniques.

**Resolving conflicts**

Steps in resolving conflicts:

* Meet with the conflicting parties.
* Identify exactly what each party wants.
* Identify why each party needs what they want.
* Find out what the common goal is that ties these reasons together.
* Obtain agreement that the common goal is correct.
* Dig deeper and review the assumptions.
* Challenges each of the assumptions.

**Working Agreement**

Working agreements facilitate conflict management. Have them and keep them visible.

As a participant on this team, I agree and acknowledge that:

* I am committed to the team’s objectives and goals.
* I respect other people’s opinions, even when they contradict or conflict with mine.
* If we cannot reach agreement, I will seek and support a consensus decision.
* I will at all times avoid blocking my team from moving forward
* Whether or not the team decision coincides with mine, I will do my best to support.

**Achieving Consensus**

* Define why reaching consensus is important in this situation.
* Let people exchange thoughts. Begin with someone who disagrees and then ask someone who agrees to give his or her perspective.
* Decompose the disagreement. Identify precisely what parts of the idea they disagree with. Can a portion be removed or modified?
* If that doesn’t work, ask those who disagree to propose a modification to the idea or exchange alternative ideas.
* Continue exchanging thoughts and finding alternatives until you reach consensus or decide consensus is not possible. If consensus isn’t possible, make a majority decision and clarify that everyone will support this decision.