Development of Large Systems
Ola 4
Agile Process

By Kristofer, Mads, Michael and Søren

SAFe:

What is meant by “Essential SAFe”?

“Simplicity-the art of maximizing the amount of work not done-is essential” - The Agile Manifesto.

Essential SAFe is about providing the minimal necessary elements for an Agile Release Train to deliver a solution and is the simplest starting point for implementation of the framework.

What is an ART – why is it important?

Agile Release Train (ART) is supposed to be the heart of Essential SAFe. ARTs are virtual organizations that are formed to span functional boundaries, eliminate unnecessary handoffs and steps, and accelerate value delivery by adopting the SAFe’s Lean-Agile Principles and practices.

There are 7 core competencies in SAFe – what are they?

The Foundation element, which includes the Lean-Agile Mindset, Core Values, SAFe Principles, Implementation Roadmap, the role of the SPC, and two core competencies:

- Lean-Agile Leadership – Describes how Lean-Agile Leaders drive and sustain organizational change and operational excellence by empowering individuals and teams to reach their highest potential.
- Continuous Learning Culture – Describes a set of values and practices that continually encourage individuals—and the enterprise as a whole—to increase knowledge, competence, performance, and innovation.

Two delivery core competencies:

- Team and Technical Agility – Describes the critical skills and Lean-Agile principles and practices that high-performing Agile teams and Teams of Agile teams use to create high-quality solutions for their customers.
- Agile Product Delivery – A customer-centric approach to defining, building, and releasing a continuous flow of valuable products and services to customers.

What are 3 benefits of SAFe? Provide details of the benefits?

Faster Time-to-market
One of the benefits of scaling agile with SAFe is improved time-to-market. By aligning cross-functional teams of agile teams around value, leading enterprises can meet customer needs faster. Leveraging the power of Scaled Agile Framework helps them make quicker decisions, communicate more effectively, streamline operations, and stay focused on the customer.

Improvements in Quality
Built-in quality is one of the core SAFe values: it highlights the importance of integrating quality into every step of the development cycle. In this way, scaling agile with SAFe benefits organizations by shifting quality from a last-minute focus to the responsibility of everyone.

Increase in Productivity
SAFe provides measurable improvements in productivity by empowering high-performing teams and teams of teams to eliminate unnecessary work, identify and remove delays, continuously improve, and ensure they are building the right things.

What could be the downsides of SAFe?

Some downsides to using SAFe are:

- SAFe ties several concepts to the higher SAFe levels even though they are needed at all of the levels.
- SAFe is getting more complicated with each release.
- SAFe has caused much confusion with its overloading, re-defining and misusing existing terms and concepts.
- For all of its terms, there is no explicit equivalent to the minimum-business-increments (MBI) in SAFe.
- Overly complex strategies and portfolio management.
- SAFe is used with companies much smaller than it was intended for.

What in SAFe is the difference between a user story and an enabler story?

Enablers are backlog items that extend the architectural runway of the solution under development or improve the performance of the development value stream. Enablers are captured in backlogs as a type of Epic, Capability, Feature, or Story.
They are used primarily for exploration, architecture implementation, refactoring, building infrastructure, and addressing compliance. While their type is unique, they are managed similarly to customer-facing backlog items.

Enablers bring visibility to all the work necessary to support the efficient development and delivery of future business requirements. Enablers are used to explore ideas, improve architecture, strengthen infrastructure, and manage compliance. Since enablers result in the production of tangible outputs, they must be visible. They are treated like all other backlog items—subject to visibility, prioritization, incremental delivery, measurement, and feedback.

Enablers can be used to define any activity that improves the value stream in support of foreseeable business needs. These activities generally fall into one of four categories:

- Exploration – support research, prototyping, and other activities needed to develop an understanding of customer needs, including the exploration of prospective Solutions and evaluation of alternatives
- Architectural – used to build Architectural Runway, which allows smoother and faster development through the Continuous Delivery Pipeline (CDP),
- Infrastructure – support the creation and optimization of the development and runtime environments that host the systems used to build, validate, deploy, and operate solutions
- Compliance – facilitate managing specific compliance activities, including Verification and Validation (V&V), audits and approvals, and policy automation

Architectural enablers can also address problems with the resiliency of deployed solutions. After implementation, these enablers often reflect Nonfunctional Requirements (NFRs) imposed on future backlog items. NFRs often originate as architectural enablers and grow as a set over time.

How can XP be used in SAFe teams? (what roles does it play and what stages)

Extreme programming is a software development methodology that's part of what's collectively known as agile methodologies. XP is built upon values, principles, and practices, and its goal is to allow small to mid-sized teams to produce high-quality software and adapt to evolving and changing requirements.

What sets XP apart from the other agile methodologies is that XP emphasizes the technical aspects of software development. Extreme programming is precise about how engineers work since following engineering practices allows teams to deliver high-quality code at a sustainable pace.

Extreme programming is, in a nutshell, about good practices taken to an extreme. Since pair-programming is good, let's do it all of the time. Since testing early is good, let's test before the production code is even written.

With SAFe a quick and adaptive development environment is prioritized and it shares this with XP, though is there a difference between the two that may create problems as XP is much more focused on the technical aspects of a development project. This can be mediated by using XP as a means to an end within the already existing work flow provided by Essential SAFe.

SAFe is a Lean Agile framework – what does Lean mean here?

SAFe is based on ten immutable, underlying Lean-Agile principles. These tenets and economic concepts inspire and inform the roles and practices of SAFe:

- **Take an economic view:**

Delivering the best value and quality for people and society in the shortest sustainable lead time' requires a fundamental understanding of the economics of building systems.

- **Apply systems thinking:**

Deming observed that addressing the challenges in the workplace and the marketplace requires an understanding of the systems within which workers and users operate.

- **Assume variability; preserve options:**

Traditional design and life cycle practices encourage choosing a single design-and-requirements option early in the development process.

- **Build incrementally with fast, integrated learning cycles:**

Developing solutions incrementally in a series of short iterations allows for faster customer feedback and mitigates risk.

- **Base milestones on objective evaluation of working systems:**

Business owners, developers, and customers have a shared responsibility to ensure that investment in new solutions will deliver economic benefits.

- **Make value flow without interruptions:**

The third principle in Lean Thinking is to make value flow without interruptions.
Doing so requires an understanding of what flow is, what the various properties of a flow system are, and how these properties can accelerate or impede the flow of value through any particular system.

- **Apply cadence, synchronize with cross-domain planning:**

Cadence creates predictability and provides a rhythm for development. Synchronization causes multiple perspectives to be understood, resolved and integrated at the same time.

- **Unlock the intrinsic motivation of knowledge workers:**

Lean-Agile leaders understand that ideation, innovation, and employee engagement are not generally motivated by individual incentive compensation.

- **Decentralize decision-making:**

Achieving fast value delivery requires decentralized decision-making.
This reduces delays, improves product development flow, enables faster feedback, and creates more innovative solutions designed by those closest to the local knowledge.

- **Organize around value:**

Many enterprises today are organized around principles developed during the last century. In the name of intended efficiency, most are organized around functional expertise.
But in the digital age, the only sustainable competitive advantage is the speed with which an organization can respond to the needs of its customers with new and innovative solutions.
These solutions require cooperation amongst all the functional areas, with their incumbent dependencies, handoffs, waste, and delays.
Instead, Business Agility demands that enterprises organize around value to deliver more quickly.
And when market and customer demands change, the enterprise must quickly and seamlessly reorganize around that new value flow.

In PI planning risk is measured using ROAM – what does ROAM mean?

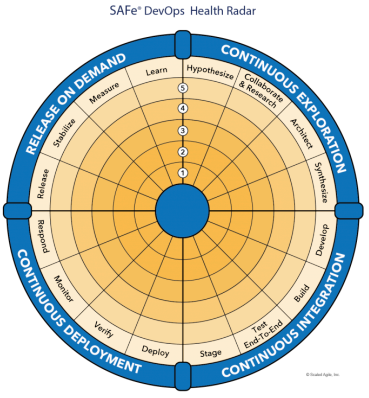
The ROAM board created during PI planning can be reviewed during the ART Sync to ensure those responsible for owning or mitigating a risk take the necessary actions. The ART may also record any new items that may have arisen after planning and ROAM them. ROAM consist of four parts: Resolved, Owned, Accepted and Mitigated

How does the Product Owner fit into the ART team? Focus on the role in relation to the developers – what does the PO do?

The Product Owner (PO) is the Agile team member primarily responsible for maximizing the value delivered by the team by ensuring that the team backlog is aligned with customer and stakeholder needs.

Its primary role in relation to the developers of a flow is to guide them towards the intended product, so that they properly prioritize implementations. It is also their role to improve communication between the developers and others within a project.

What is the SAFe DevOps health assessment and how might it be useful?



It's a tool called the SAFe DevOps Health Radar. The health radar was built to help Agile Release Trains assess their ability to Release on Demand.

The maturity in each sub-dimension is scored as: Sit, Crawl, Walk, Run, or Fly.
After the ART or solution train ascertains the problem areas based on its health radar.
The next step is to improve the flow, and the assessment provides suggestions on what the next level of maturity entails.
We're taking a system's view where it's better to move from a "crawl" across the radar to a "walk" across the radar, than improve to "fly" in one sub-dimension while leaving the rest at "crawl." This concept is important because the best approach is to improve the overall flow of value across the entire Continuous Delivery Pipeline, not just in one sub-dimension.

The Spotify Model:

Explain why The Spotify is much less formal than SAFe?

1. **Guiding Principles vs. Rigorous Structure**: The Spotify model is based on guiding principles rather than a strict framework. It provides lightweight structures like Squads, Tribes, Chapters, and Guilds to encourage autonomy and alignment within cross-functional teams. SAFe, on the other hand, is a well-defined, prescriptive framework with multiple layers (e.g., teams, programs, and portfolios) and specific roles, ceremonies, and artifacts designed to align large organizations to deliver effectively in an Agile way.
2. **Autonomy and Flexibility vs. Defined Roles and Responsibilities**: Spotify emphasizes autonomy, allowing teams to determine how they work best. For instance, squads can choose their own methods and tools. SAFe, however, prescribes specific roles like Product Owner, Scrum Master, and Release Train Engineer, along with a more defined hierarchy and responsibilities to ensure consistent processes across teams and departments.
3. **Emergent Framework vs. Formalized Framework**: The Spotify model emerged as an experimental approach that evolved organically within Spotify, making it more adaptable and less restrictive. SAFe is a formalized, structured framework developed externally by Dean Leffingwell, with comprehensive training and certification paths to ensure organizations adhere to SAFe practices.
4. **Scaling Philosophy**: Spotify's model focuses on scaling culture and alignment rather than scaling processes and practices. This means it relies heavily on creating a strong culture of trust, transparency, and continuous improvement. SAFe, in contrast, provides a process-driven approach to scaling Agile across large enterprises, with specific ceremonies like Program Increment (PI) Planning and Release Trains to synchronize work across multiple teams.
5. **Emphasis on Culture and Values over Procedures**: Spotify's approach is designed to support innovation and maintain a startup-like culture, even as the company grows. It encourages informal cross-functional interactions and values team independence and creativity. SAFe, however, seeks to bring consistency and standardization, making it well-suited to organizations that prioritize predictability, governance, and regulatory compliance.

While both approaches aim to facilitate Agile scaling, SAFe's formality and structure make it suitable for large organizations needing predictable outcomes, whereas Spotify's model is ideal for companies that prioritize flexibility, autonomy, and a strong team culture.

How is it a way of working with fewer events and more focus on team and team coordination?

The Spotify model emphasizes flexibility and team autonomy, focusing on informal collaboration rather than formal ceremonies. Unlike traditional frameworks, it minimizes

mandatory events, letting teams decide on the meetings they need. Coordination happens through Squads, Tribes, Chapters, and Guilds, which foster organic knowledge sharing and alignment without strict scheduling. By reducing the number of structured, required events and focusing instead on informal, flexible coordination, the Spotify model allows teams to work with greater flexibility. This approach puts a stronger focus on organic team coordination, cross-functional knowledge sharing, and collaboration across the organization.

Both Scrum and Kanban can be used with the Spotify model. Consider using one of them for your exam project, or even a combination of them. What are the main differences between Scrum and Kanban?

Scrum and Kanban are both popular Agile methodologies used for project management, but they have distinct differences in structure, approach, and principles.

1. Structure and Framework

- **Scrum**: Operates on a structured framework with specific roles (e.g., Scrum Master, Product Owner, Development Team), events (e.g., sprint planning, daily stand-up, sprint review, and retrospective), and artifacts (e.g., product backlog, sprint backlog). Work is divided into fixed-length sprints (usually 2-4 weeks), and progress is reviewed at the end of each sprint.
- **Kanban**: Is a less structured, continuous workflow framework. There are no fixed roles, events, or defined timeframes; instead, it focuses on visualizing work using a Kanban board. Work items flow through various stages (columns on the board) until completion, and new tasks can be added whenever there's capacity.

2. Process Flow

- **Scrum**: Uses a sprint-based approach, where work is planned and executed in time-boxed iterations. Tasks are pulled from the product backlog and committed to a sprint, aiming to complete them by the end of the sprint.
- **Kanban**: Has a continuous flow. There are no time-boxed sprints; instead, work items are pulled through the system as soon as there's capacity, allowing teams to respond to changes more immediately.

3. Flexibility and Change Management

- **Scrum**: Changes to tasks within a sprint are generally discouraged to protect the sprint commitment. However, the backlog can be reprioritized between sprints, allowing for adjustments in the following sprint.
- **Kanban**: Is highly flexible. Tasks can be reprioritized anytime since work is pulled when capacity allows, making Kanban ideal for teams that need to accommodate frequent changes or have unpredictable workloads.

4. Measurement of Progress

- **Scrum**: Progress is tracked by sprint completion, typically measured through velocity (average story points completed per sprint). Scrum encourages tracking sprint burndown charts to monitor progress.
- **Kanban**: Tracks work-in-progress (WIP) limits and lead time (time taken to complete a task) or cycle time (time to move a task from one stage to another). Cumulative flow diagrams are often used to measure progress and identify bottlenecks.

5. Team Size and Stability

- **Scrum**: Works best with a stable team, as the sprint commitments depend on team capacity and velocity. It typically requires cross-functional teams and structured planning to ensure that sprint goals are met.
- **Kanban**: Is adaptable to various team sizes and compositions. It can support teams with varying capacities, making it suitable for maintenance or support teams where tasks and personnel may vary widely.

6. Focus and Goal Orientation

- **Scrum**: Each sprint has a set goal, with a focused sprint backlog aiming to complete specific tasks. This encourages a strong team focus on defined goals for each sprint.
- **Kanban**: Lacks specific sprint goals. Instead, the focus is on continuous improvement, optimizing flow, and keeping WIP limits under control to ensure tasks move smoothly from start to finish.

7. Continuous Improvement Practices

- **Scrum**: Uses retrospectives at the end of each sprint to identify improvements.
- **Kanban**: Encourages continuous process improvement by analyzing workflow and adjusting WIP limits, board structure, or policies to optimize throughput and reduce bottlenecks.

What is a Guild and how could you use Guilds in the exam project development process?

In Spotify's engineering model, a *guild* is an informal, cross-functional group of people with shared interests, goals, or expertise across different teams, known as *squads*. Guilds are open to anyone across the organization who wants to join, offering a space for knowledge-sharing, skill development, and the alignment of best practices in specific areas, such as design, testing, or DevOps. They are one of the elements that help Spotify scale culture and expertise horizontally, ensuring that information and skills flow across organizational boundaries without rigid formal structures.

Since our team is small and we naturally have a lot of shared interests and expertise,thinking of *guilds* less formally, we could still apply the idea of *guilds* to strengthen certain parts of our project:

Knowledge-Sharing: Even though we are all computer science students, each person likely has different strengths (e.g., front-end, back-end, testing). We could form informal “guilds” within the team based on these interests to let each person lead in their area, ensuring best practices and knowledge are shared as you develop.

Continuous Learning: If we encounter areas where our team feels less confident, we could create a mini guild to dive deeper into those topics together, share insights, and apply them to the project.

Standards and Consistency: We could set “guild” standards in code quality, documentation, or testing. These standards would help maintain consistency across the project, similar to how a guild’s guidelines work at Spotify.

Remote (distributed) Teams:

What are the main benefits of working in distributed teams?

One of the main benefits of working in distributed teams is the fact that it allows companies to hire skilled individuals from anywhere, even globally, which widely increases the potential teammates and creates a more diverse talent pool. Having a remote or partially remote team can also save cost when it comes to acquiring office space. Some people also find office conversations a distraction, which can be avoided when not seated close to colleagues.

What are the main challenges of working in distributed teams?

Having a remotely distributed team can result in several complications that wouldn't otherwise appear. If hiring abroad, language barriers can occur, which can cause confusion and miscommunication between colleagues. Team members are then also forced to account for time differences, and if the geographical distance is great enough, timing meetings that fit all members can become complicated. It also increases the difficulty of creating bonds and connections with colleagues. A small chat while refilling a coffee mug can go a long way to improve someone's mood and reduce stress. If in need of help, it is also easier asking a closely seated colleague, than it is sending a message that can be overseen/missed.

What practices can help a remote collaboration culture to stay healthy?

There are several tools and work approaches that can keep a remote collaboration culture healthy. Communication tools such as Slack, Zoom, Microsoft Teams and so on. These all provide an easy way of communication and allow easy access to colleagues despite not being physically close to one another. It is also important to keep expectations of the employees' availability limited to their work out. When being remote, it can easy be seen as always available since they are normally only a simple message away. This can be stressful and counter productive for a team which is why clear boundaries and specific work hours are needed to be in place. Having a way of tracking everyone's process is also vital to ensure no one is stuck without being able to ask for help.

What considerations do your group have on remote work in the exam project period?

We have set several of the above mentioned practices into use. We have a Discord communication space set up, where reaching out is simple and easy, and advice/responses from teammates are always within reach. We have also set up a Trello board, where tasks are created and tracked, ensuring work is being done in the right order and assignments are being completed on time. We also have regular meetings, to gather the troops and make sure everyone is on the right track.