Allstacks

WHITEPAPER

10 Common SAFe Mistakes and How to Avoid Them

SAFe is known as an opinionated framework, and it is. But what's mentioned less often is how many people have opinions about SAFe, including that it's a struggle to get just right.

Allstacks is here to help.

At a time when modern software development efforts have never been under more pressure to release safe code more quickly, deliver value to the business, and keep increasingly fickle customers delighted, the SAFe framework is often held up as a tried and true way to make it all happen. But SAFe is complex and best suited for only certain types of companies. Here's a quick look at the goals of SAFe and Agile, followed by a look at common implementation mistakes and how to avoid them.

What is Agile?

As Agile is the foundation of SAFe, it's worth having a brief refresher. In the early 2000s, the Agile Manifesto was published with the hope of fundamentally changing the way software was developed. At the time, development and operations were completely siloed, moved very slowly, and it's safe to say weren't exactly embracing new thinking.

Agile broke development down into small, digestible pieces, and put the focus on how people and teams worked together. Agile also addressed another huge issue for software development teams – the ability to more easily adapt to change – by bringing common language and practices to the process. Agile's "project management" approach to software development has been wildly successful: as of 2022, 86% of teams report they use Agile for software development. And, according to the State of Agile Survey from digital.ai, Agile popularity is rising in non-tech departments as well.

Agile Manifesto

Values

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

Principles

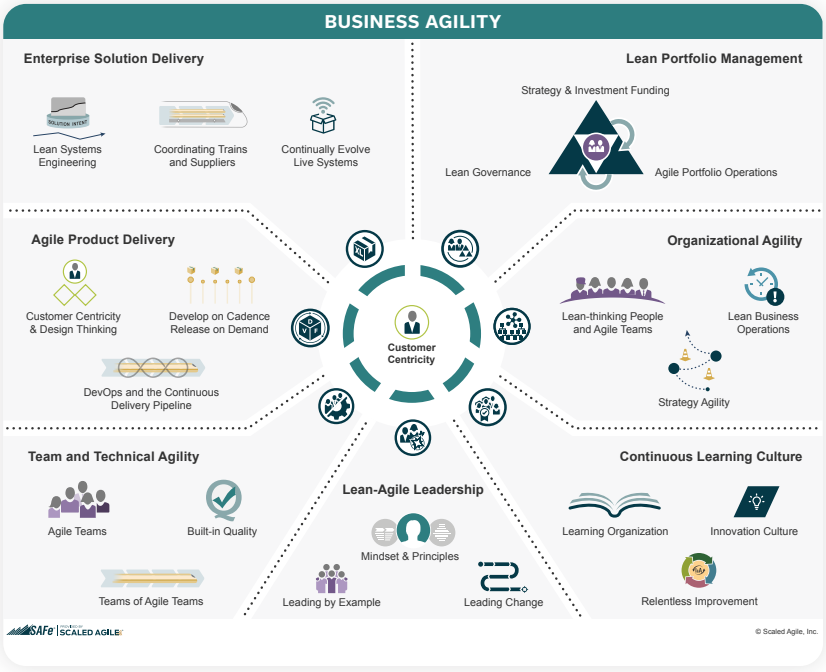
1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of 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.

What is Lean?

The concept of “lean” software development was conceived shortly after the publication of the Agile Manifesto. Lean software development borrows heavily from concepts in use in manufacturing, and it’s made up of 7 basic principles designed to make software development more efficient. Lean is strongly focused on helping organizations achieve true business agility and it works nicely partnered with the Agile focus on breaking down projects and emphasizing human contributions. Some consider Lean an extension of Agile, and its philosophies have helped shape the Scaled Agile Framework.

What is SAFe?

It’s easiest to think of the Scaled Agile Framework, or SAFe, as a way to efficiently implement Agile and Lean in organizations looking to grow over time. At its essence, SAFe is a collection of values, principles, and configurations that provide guidance and guardrails for teams looking to scale Agile. In short, there are lots of rules to follow.

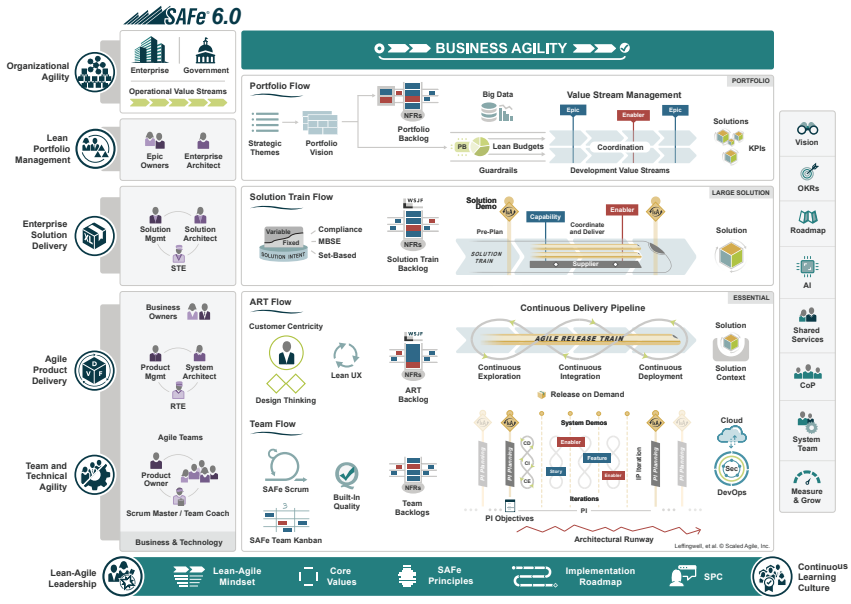


SAFE values

Five values are central to SAFe including alignment, built-in quality, transparency, program execution, and leadership.

1. **Alignment** focuses on bringing everyone involved in software development, including the business side, together at the same table.
2. **Built-in quality** stresses the need for code creators to decide when the product is ready; they should be the ones who own "the definition of done."
3. **Transparency** aims to make the development process, well, transparent to everyone in the organization, with the goal of eliminating surprise bottlenecks or delays.
4. **Program execution** encourages teams to adopt high-quality standards of production.
5. **Leadership** means management has to be involved with and supportive of any Agile efforts.

Without these values in mind, organizations trying to scale Agile can start to see their efforts fall apart... we'll get to that shortly.



SAFe principles

To help guide organizations as they scale Agile, SAFe offers 10 ways to approach the challenge. They are all laser-focused on making the entire software development process more efficient as well as tying it as closely as possible to business value and customer satisfaction.

The principles are:

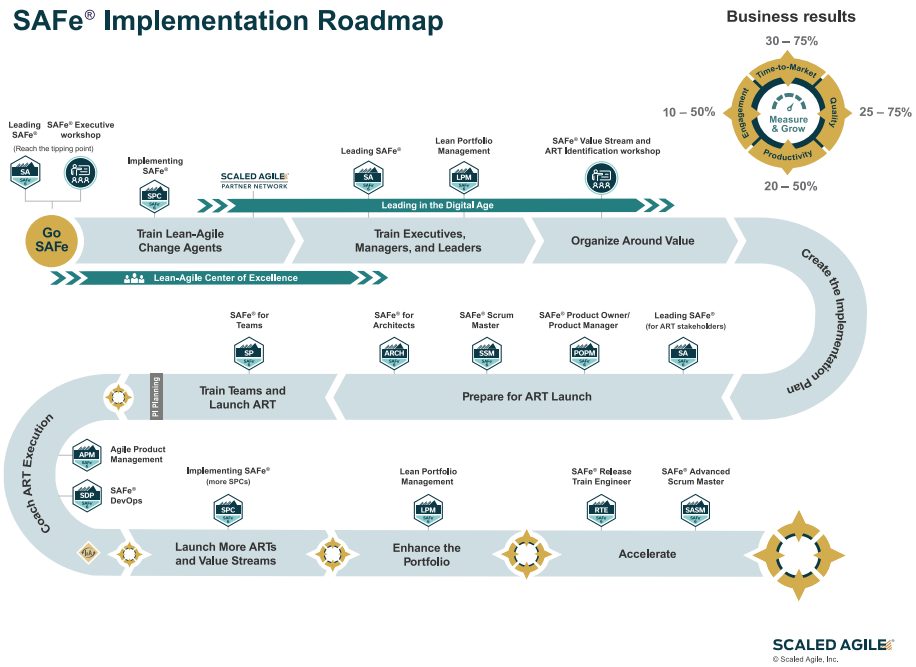
1. **Take an economic view**
2. **Apply systems thinking**
3. **Assume variability; preserve options**
4. **Build incrementally, with fast integrated learning cycles**
5. **Objectively evaluate working systems to create milestones**
6. **Make value flow without interruptions**
7. **Apply a cadence and synchronize with cross-domain planning**
8. **Unlock the intrinsic motivation of knowledge workers**
9. **Decentralize decision-making**
10. **Organize around value**

How popular is SAFe today?

According to the digital.ai State of Agile report, SAFe is the most popular Agile framework in use today: 53% of teams said they used SAFe in 2022, up from 37% in 2021. That dramatic jump in adoption is saying something as there are many frameworks, and there are more than 100 "flavors" of Agile in use as well.

What are the primary goals of SAFe?

At its core, SAFe aims to be a guidebook for organizations looking to expand Agile/Lean development principles throughout an organization. Modern software development is of course complicated, but it's also under constant pressure to adapt and move more quickly. By codifying basic concepts, SAFe is an effort to cut through the noise and give teams something to refer back to time and again.



10 common SAFe mistakes

All that said, SAFe itself is quite complicated and it can be difficult for fast-moving companies to get it right, as you'll see below.

1 Misunderstanding Agile and Scrum

If organizations begin the SAFe journey but don't fundamentally understand or embrace Agile or Scrum, they're in for a world of struggle. Companies need a deep understanding of the Agile Manifesto, and an organizational structure and acceptance of its principles and practices because SAFe simply won't work without it.

Also, SAFe leans heavily on Scrum, so teams looking to scale Agile will also need to be able to do the same. The bottom line: companies shouldn't even think of adopting SAFe until the cross-functional practices of Agile and Scrum are firmly established.

2 Interpreting the framework too literally

SAFe is a great playbook for growing organizations, but it's a journey, not a destination.

If teams adhere too rigidly to the many SAFe principles, they're actually missing the point. SAFe is meant to be adapted to a particular organization's needs so it can grow and evolve organically over time.

The best approach to SAFe is to wrap it in the context of business priorities and values, change it to suit the particular environment, and then embrace it flexibly, all while being prepared to continually tweak as needed. Implementing undiluted and unedited SAFe is **not** a recipe for success.

3 Jumping in too early

The Scaled Agile Framework literally requires scale—otherwise known as a larger business—to be successful. If an organization doesn't have more than five development teams, SAFe is likely to have too much overhead and complexity for it to make sense.

This can even be true in a proof-of-concept situation: the problems that SAFe was created to solve need big teams, lots of moving parts, and big problems to solve. Conducting a "small-scale" SAFe proof of concept isn't going to reflect what it can bring to an organization, and, in fact, might rule it out unnecessarily because it won't be seen as effective on a smaller scale. So it's key to turn to SAFe when an organization is already large and planning to grow.

4 Not giving it enough time

As we've made clear, SAFe is complicated, which means organizations need to give it a lot of time to prove its effectiveness.

First off, the framework is updated roughly every 18 months, and that can mean that just as teams are starting to feel comfortable, new suggested guidelines will be introduced.

Experts also suggest most companies need five years to really understand and strategically implement SAFe, and that doesn't include the need for continuing education along the way. With something this large and involved, patience and commitment will go a long way.

5 Playing the name game

In the enthusiasm of new SAFe-manship, teams may rearrange roles and titles to show that they're fully onboard. But this plan can actually backfire.

An organization's existing roles may not necessarily map directly to SAFe roles and titles, and simply changing what a person is called may, or may not, change the job function, meaning key tasks and responsibilities can get overlooked, or worse.

For example, product managers play a key role in SAFe, but there are usually no project, program, or delivery manager roles at all. The most successful SAFe organizations adopt the "what's in a name?" philosophy and don't get caught up in titles but rather in making sure each role has the right responsibilities in order to support cross-functional relationships.

6 Forgetting to share

Transparency is nearly always a good idea, but it's particularly critical in a complex framework like SAFe. Individuals and teams need to be empowered to make quick decisions, and transparency is what ensures the choices are correct.

The more an organization is working around transparency, and not resolving it, the less likely it is to be scaling successfully. Luckily, Agile, Scrum, and SAFe have some built-in options to help encourage transparency ranging from Kanban boards to retrospectives and paired programming.

7 Overthinking the options

It's so easy to overthink choices when it comes to an Agile transformation. From choosing the right tech tools to aligning with business values, the number of options is dizzying, which is why some organizations stall out on SAFe early.

As mentioned in common mistake #2 ("interpreting the framework too literally"): teams need to flexibly embrace the SAFe journey, which means jumping in, trying things, and not being afraid to make changes as often as necessary.

8 Not building from the ground up

No SAFe adoption plan will be successful if it is, or is perceived to be, coming from the C-suite down to the masses.

For starters, that approach is inherently anti-Agile; Agile is all about building a culture that supports "motivated individuals" and then standing back and letting the team organize solutions that best fit their individuals, processes, and tools.

SAFe brings a lot of changes to an organization, from roles and responsibilities to ways of working and even potentially new toolsets, so it's incumbent on management to explain, persuade, cheerlead, and join in the efforts as needed to get the transition started on the right notes.

9 Doing "partial" SAFe

As a sprawling framework with a lot of pieces, it's probably not surprising to hear that many organizations have implemented SAFe in a piecemeal manner. For some companies, that may be just a starting point, or it may be all that is needed at the time. But others can find themselves unintentionally stuck with lesser SAFe. Partial SAFe implementations can include:

- SAFe-in-a-box: trying to scale, but stuck in a silo
- "Small scale SAFe:" a proof-of-concept using too few teams that never goes anywhere
- "Mechanical" SAFe: It looks like SAFe processes, but the practices and culture aren't there.
- SINO (SAFe-in-name-only): Saying "We're doing SAFe" while not actually doing SAFe.

Every team has to start SAFe somewhere, just don't stop there – the full benefits come from a well-rounded scaled Agile effort.

10 Not organizing for success

Organizations truly committed to scaling Agile most likely need to reorganize to eliminate silos and promote true cross-functionality within teams. It's critical to think through areas that might become "tripping points" when scaling Agile (like isolated teams, a top-down culture, or even shadow IT efforts) because otherwise time will be spent working around those obstacles rather than scaling. This is not an easy step to take, but it is critical to true agility and a successful SAFe rollout.

The goal is successful SAFe!

...but organizations have to decide for themselves what that ultimately looks (and acts) like. Scaled Agile has an implementation roadmap that includes 14 articles and related graphics.

But as a good summation, here's what Scaled Agile indicates are the top eight "accelerators for change:"

1. **Create a sense of urgency**
2. **Build a guiding coalition**
3. **Form a strategic vision**
4. **Enlist a volunteer army**
5. **Enable action by removing barriers**
6. **Generate short-term wins**
7. **Sustain acceleration**
8. **Institute change**

It's a big ask of any organization at any stage, but patience, process, and flexibility can go a long way to help teams achieve their goals.