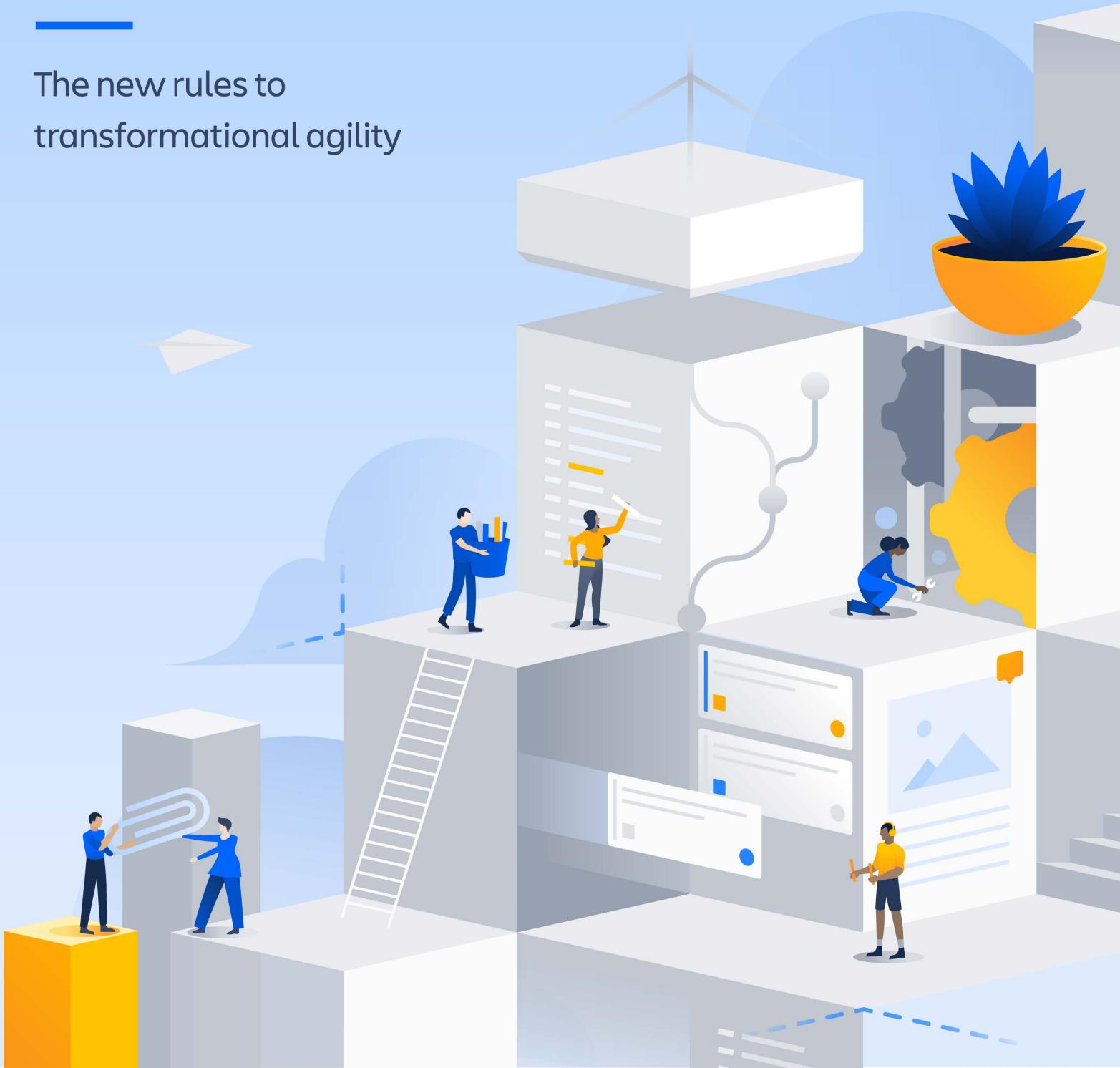


Beyond the basics of scaling agile

The new rules to
transformational agility



Introduction

The basics of agile at scale

- 02** Why are organizations scaling agile?
- 04** What is agile at scale?
- 05** Where are you on your agile at scale journey?

Understanding the principles & frameworks for scaling agile

- 07** Whether or not to use a framework
- 09** 7 essential principles for practicing agile at scale
- 16** An interview with Dom Price, Work Futurist at Atlassian

Best practices for implementing scaled agile

- 19** Understand the process
- 20** Assess where you are in your agile journey
- 21** Define the scope for your MVP
- 22** Make the pitch to leadership...and the teams
- 23** Decide on areas of alignment

How to use Atlassian tools for agile at scale

- 27** Putting the tools in the context of people, principles & practices

Conclusion

Additional Resources



Introduction

As agile adoption has increased over the last decade, many organizations have grown with agile and are scaling agile methodologies to help them plan, deliver, and track progress across their teams.

Others are looking to agile as a set of practices and principles that can help them transform to deliver better outcomes and execute against their strategy more quickly and effectively. But this is easier said than done.

In this guide, you'll learn:

- 💡 The basics of agile at scale
- 👥 How to understand your agile maturity
- 📅 The minimum viable principles for scaling agile
- 🕒 The changes required of your people to engage in an agile transformation
- 🌟 Best practices for scaling agility and tips to get started
- ⚡ How Atlassian's tools can support agile at scale



The basics of agile at scale

The why, what and where of scaling agility

Why are organizations scaling agile?

In today's landscape, you need to maintain a high level of innovation in order to compete and succeed in your market. Over the past two decades, software development teams have proven that practicing agile methodologies lets them deliver solutions to customers faster, with more predictability, and gives them the ability to pivot based on new information.

As your customers' needs have evolved and the dynamics of the modern workforce have changed, the stakes have been raised, and you must be able to accelerate and adapt at enterprise scale. You must now:

- Adapt to constantly evolving customer demands, driven by smaller and more nimble competitors
- Not just retain customers by meeting expectations, but delight them, as the costs for switching tools continue to go down
- Provide flexible, customizable solutions that feel cohesive and unified across an ever-expanding number of digital touchpoints
- Facilitate teams of teams working together as a single entity, even when distributed across timezones and geographies
- Shift from technology as a back office cost center to a strategic enabler of the rest of the business
- Enable other teams, not just software, to become more iterative, adaptable, and innovative in how they work



“The pace of change today is so high, there are no more three or five-year plans. You need to be able to respond to change. The people that survive in this disruptive world are those that can adapt, and those that can adapt and scale at speed.”

Shayne Elliot
CEO, ANZ Bank



While many organizations have implemented agile at the team level, it's difficult to realize the benefits across all levels of the organization. You may have hit a point in your journey where you're asking questions like:

- Is our organization equipped to define and track the investment made vs. value delivered across teams?
- Is the work we're delivering having a positive impact on our customers and helping us meet the company's key objectives?
- Will scaling agile throughout all levels of the organization help us achieve our goals?

In this whitepaper, we'll discuss how to answer these questions by learning about what agile at scale is, how to understand where you're at in your agile journey, and ways to get started.

Before we dive in, keep in mind that "agile at scale" is not the end goal. The end goal is to effectively execute your strategy. And we think scaling agile can help you do just that.



What is agile at scale?

Agile software development was the evolution of work practices beyond the traditional, linear waterfall development model. Instead of betting everything on a “big bang” release, an agile team delivers work in small, but consumable, batches. This results in faster time to market, quicker adaptation to customer needs and reduced risk. When applied “at scale,” agile ceremonies and rituals take on new contexts and there are different challenges in evolving from traditional to more agile ways of working. But the core principles are the same.

Scaling agile is a complex, multidimensional topic. It involves changing people (skills and mindset), practices (rituals and habits), and tools (used with consistency and discipline) to improve collaboration and the organization’s ability to execute against its strategy. Ultimately, these changes will create greater transparency and alignment around work, and help to hard-code the values and principles of agile into the entire DNA of an organization. The road to enterprise agility can be treacherous, but the journey is worth it.



Atlassian defines agile at scale as:

The ability to drive agile at the team level, while applying the same sustainable principles, practices, and outcomes at other layers of the organization.

For example: If agile teams do pre-sprint planning three days before a sprint begins, then I will want to do pre-quarter planning 2-3 weeks before the start of the quarter to ensure a properly prepared backlog across a team of agile teams.



“Teams and leaders come to me all the time and ask ‘Please make my team agile.’ My response is almost always some form of ‘Sure...but can you rephrase that without using the word agile?’

By asking this question, I can uncover what they really want. Once I have an understanding of their team’s goals and objectives, then we can discuss how to map the agile practices to support their desired outcomes. This in turn supports our company goals.

We try and encapsulate this in our overall vision statement of the Intuit Agile program:
Deliver world-class agility through enterprise-wide adoption of agile principles.

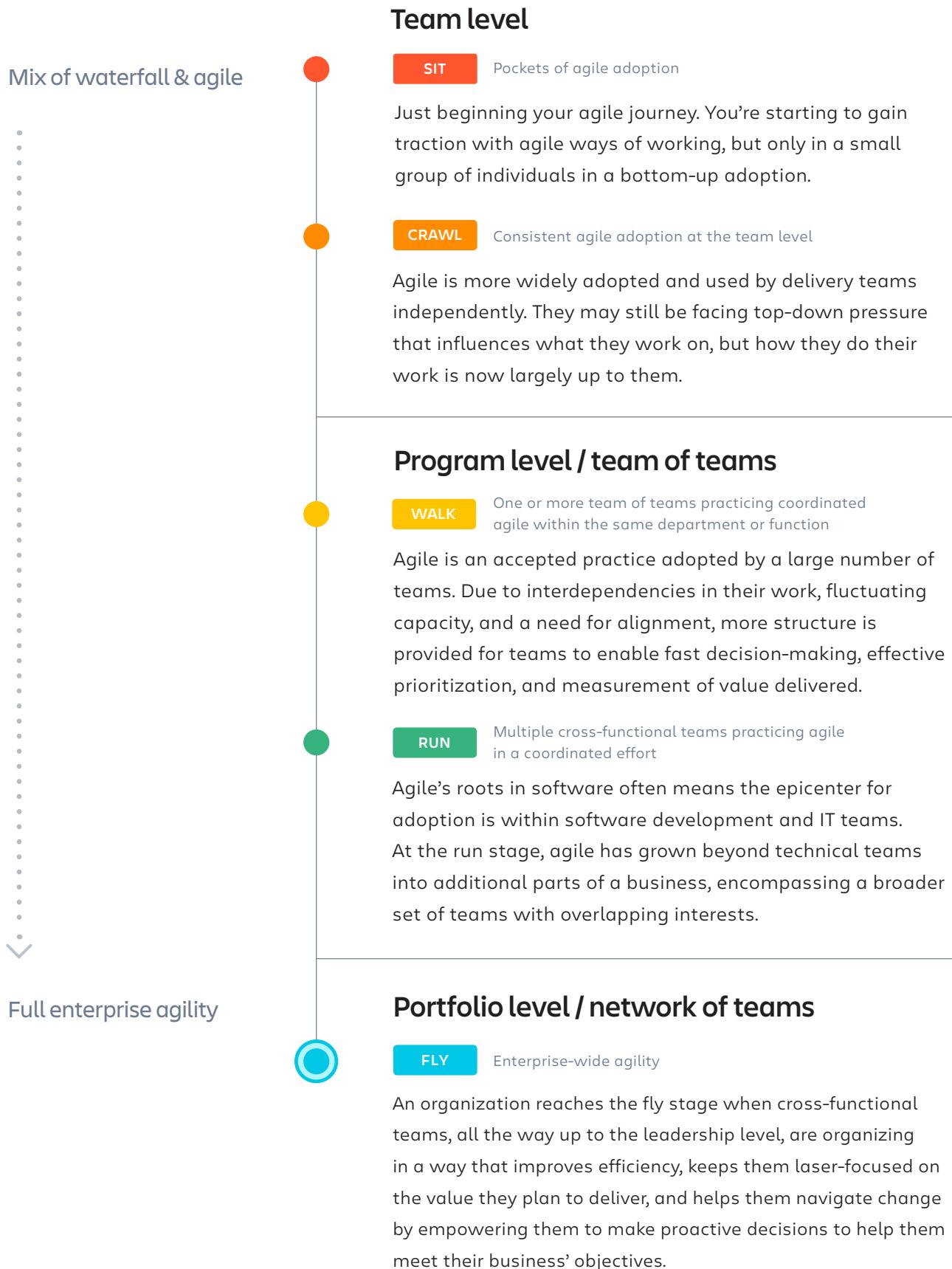
Ian Maple
Enterprise Agile Leader, Intuit

intuit

⚡ CHALLENGE Describe your goals without using the word ‘agile’.

Where are you on your agile at scale journey?

There are many ways to conceptualize your agile journey. We like to think of it in terms of how teams and individuals adopt agile behaviors. Here's a maturity model with five stages to consider:



Your scaling journey won't be linear

As you can see in the agile maturity model on the previous page, we've shown a progression of scaling agile—but it's not always a linear path. There's also no one right way to scale agile.

Agile may grow from pockets of one or more innovative teams to a broader software development organization. It can emerge as IT and software development begin to collaborate more closely and spread from there. And, in some cases, organizations will tackle aligning their entire company at once, with business and technology teams leading the charge together.

At the same time, respect and acknowledge that you should start where you are. You can't jump from a sit to a fly stage overnight—the work needs to be done at the team level first to ensure that you maintain agility as you scale.

Beware of the “big bang” agile transformation

There is an assumption that if a large number of teams are all sprinting on the same cadence that the organization has increased agile maturity. In reality, this is often completely false. We see lots of teams working together and possibly producing “stuff,” but if we look at outcomes, they still struggle to deliver value faster. Agile maturity should be about improving outcomes at scale, not about lots of teams and large scale coordination across teams.

Understanding the principles & frameworks for scaling agile

Is your organization set up for agility?

Whether or not to use a framework

With so many factors influencing modern software development and a large number of process, team, and cultural changes required to master agility at scale, frameworks like SAFe, LeSS, Spotify, and others have become popular tools for organizations on this journey. Scaled agile frameworks can provide helpful reference points for getting started, particularly if your organization is in the earlier stages of agile adoption.

But are frameworks anti-agile? Frameworks look process-heavy, and isn't the agile movement about "individuals and interactions over processes and tools"?

While it's true that a framework, when applied without thought or intent, can add unnecessary process, there are still benefits to codifying shared rituals, common roles, and guiding principles for scaling agile in your organization, especially when the organization is new to agile ways of working.

The frameworks provide guidance to help your teams get from A to Z—from team-level pockets of agile practices to complete enterprise-wide agility.

While there is no one-size-fits-all solution, scaling agile requires structure, including:

-  Shared ways of engaging other teams
-  Shared and agreed-to language
-  Alignment between behaviors, practices & tools
-  Evolving customization of a framework

Although the different agile at scale frameworks address needs in different ways, they have common core requirements across each of them. You can either choose to adopt a framework, which can help provide structure for your transformation, or you can attempt it on your own by addressing the core requirements.

There are a number of popular frameworks that exist in the market today. To make it easier, we've consolidated the frameworks into this reference guide. When reviewing these frameworks, notice they are more similar than different when it comes to addressing the core requirements for scaling agile. The differences lie in exactly how they prescribe to achieve each of the areas of focus, or each row. For example, every framework takes into consideration long term planning and strategy, it's how they do it that is slightly different.

Understanding the difference between frameworks for scaling agile

	FRAMEWORKS				
	SAFe (Scaled Agile Framework)	LeSS (Large Scale Scrum) & LeSS Huge	Spotify	DA (Disciplined Agile)	Scrum @ Scale
Long term planning and strategy	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined with recommendations	Defined and prescribed
Multi agile teams	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined with recommendations	Defined and prescribed
Team of Teams	Agile Release Train (ART)	Area	Tribes	Defined with recommendations	Scrum of Scrums
PM/PO	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined and prescribed
Scrum Master/ Agile Coach	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined and prescribed
Release Engineer/ Group Manager	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined with recommendations	Defined and prescribed
Agile practice (scrum, kanban, etc.)	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined and prescribed
Demo	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined and prescribed
Retros	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined with recommendations	Defined and prescribed
Customer driven/ value focused	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined with recommendations	Defined and prescribed
Dependency management	Defined and prescribed	Defined with recommendations	Defined and prescribed	Defined with recommendations	Defined and prescribed
Strategy transparency	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined with recommendations	Defined and prescribed
Portfolio management	Defined and prescribed	Defined with recommendations	Defined and prescribed	Defined with recommendations	Defined and prescribed
Release on demand	Defined and prescribed	Defined and prescribed	Defined and prescribed	Defined with recommendations	Not clearly defined
Risk Management	Defined and prescribed	Not clearly defined	Defined with recommendations	Defined with recommendations	Defined with recommendations
DevOps	Defined and prescribed	Defined and prescribed	Defined with recommendations	Defined and prescribed	Defined and prescribed

Processes: Defined and prescribed Defined with recommendations Not clearly defined

7 essential principles for practicing agile at scale

As we examined the frameworks used to help companies like yours bring an agile at scale approach to software development, we recognized seven essential principles common across most of the frameworks. We encourage you to take a look at each of the frameworks and find the one that appeals to your company's culture the most. However, there is no framework that'll help your company scale agile unless you're ready to recognize and address these 7 essential principles.

As you read through this, we encourage you to really reflect and ask whether the box on the left (good agile at scale) or the box on the right (bad agile at scale) sounds most like your organization.

1. Defined roles and organizational structure changes

Asking someone to serve as both the software developer and their own Scrum Master or Product Owner is a lot like asking them to “be their own boss.” That may work for some, but it doesn’t work for most. Specialized roles help hold teams accountable, assist in removing process blockers, and improve the overall efficiency of the organization. It’s very difficult to need help while being asked to provide the help needed.



However you decided to organize your teams, you have some structure to the teams with defined roles (such as PM/PO/Scrum Master/Agile Coach) that helps accelerate your transformation.

You have someone who represents your scaled agile process (such as a Scrum Master or Agile Coach) as well as someone that represents the Customer, Product, and Business (such as a Product Owner). Whether this is one for each team, or a shared resource, is up to whatever works best for your organization.



Individuals are asked to continue doing their regular jobs, and take on additional responsibilities for scaling agile. You’ve asked an engineer on each team to serve as the team’s Scrum Master. This lack of division of responsibilities and labor has lead to burnout and a false sense of being “agile.” Your representation of the customer is a leader demanding new features and your backlog is the support ticket queue prioritized by which company is paying the most money.

2. Customer-centric organization and development

It's very easy to look at your organizational reporting structure and think: "That's the way our teams should be established." However, individuals can maintain the same reporting structures today while working on a completely different team. So while it may make sense to have the teams in the same reporting structure for your company, it doesn't have to be that way. Instead, we recommend looking at establishing teams as a collection of people, from multiple parts of the business, on a single team in order to deliver end to end value to your customers.



You established teams around the value you bring your customers. This may, or may not, be based on your traditional products or organizational structures. You found that one product provides two different values to your target market, and you built two separate teams around those values. Or conversely, you found that three of your products provide, together, a single value to your target market and you built your team around that value. Either way, you identified the value your customers are getting out of your product and then built your teams around that value.



You continue to organize the teams around managers and products, because that's the way it has always been done. Some individuals may be too nervous to switch to a new team or a new boss. You also continue to have entire teams established around developing features and products that are not based in any type of value to the customer. Most of your roadmap ideas start with "I think this is a good idea because..." or "We need to build this because our largest customer demands it..." There are a lot of cross team dependencies, and any one team has a hard time bringing a full product/feature to market alone.

3. Agile practices and cadence: sprints, retros, iterations, improvements, and transparency

Calling any development process agile, does not make it agile (and we see this all the time.) If you still have gated processes requiring extensive documentation, without room to pivot if the market changes, then you're not operating in an agile way. You'll be unable to scale your agile practices across the entire enterprise, if you're not first practicing agile at the team level.



Every team is implementing true agile practices, whether it's Scrum, Kanban, or some other process. Your teams have implemented true transparency across their work, as well as implemented ceremonies for continuous improvement. There is a regular cadence for conducting these evaluations, and the cadence is relatively short and consistent.



"We're doing agile" is your favorite phrase, but you're not actually seeing any improvements at the team level. There is zero transparency into what any of the teams are working on, or their status. This makes it nearly impossible to coordinate efforts between teams. "Inspect and Adapts" or "Retrospectives" are just a laundry list of things that have gone wrong that never get addressed and fixed.

4. Adoption maturity: take time to change

It's nearly impossible to scale any organization's agile practices overnight. Often, it takes months or even years to achieve good agile at scale adoption. It's important to take the time to change. Approach these changes step by step, one team at a time (if possible), and mature them.



In addition to your teams working in an agile fashion, your implementation strategy is agile. You have adopted a Work in Progress (WIP) process by starting with one team at a time and guiding them through the maturity process. For example, you have identified five phases of maturity/adoption for your agile at scale practice, ensuring that only one or two teams are in each phase at any one time. Limiting the number of teams in transition allowed you to focus better on each team, with as much attention as they need to get it right.



You called every developer into a meeting, announcing the adoption of a new framework and stating everything they had been doing up until that point was wrong and isn't working.

This is a company wide mandate, everyone is going to drop what they're doing and adopt this new practice. People are starting to question how your company became so successful if they were doing it wrong the entire time.

5. Dependency improvements

Regardless of what you do, you have (and will continue to have) dependencies across your development efforts. A lack of accounting for, and improving, dependencies can have a significant negative impact on your agile transformation.



You've made a conscious effort to take a look across all of your teams, and identify where all of the dependencies are for each team they rely on in order to deliver working software to market.

You've also taken steps to lessen the burden these dependencies will inevitably cause. Whether you map these dependencies and plan for them, or you move all of the dependencies into the same team, you have actively acknowledged their existence and taken steps to improve them. You've also prioritized the people, teams, processes, or systems that are the biggest bottlenecks for the most teams, and have worked to remove that bottleneck completely.



You know that all of your teams rely on a single team to get your software into market, whether this is a single ops team or maybe a cross-supporting platform engineering team.

Unfortunately, everyone knows they're a bottleneck, but they'll "figure it out." You instruct them to have more meetings, work out their calendars, and just get it done. Everyone knows there are dependencies, but nobody really knows what they are or what's involved in them. Delays often surface late into the development process because another team "isn't ready or available."

6. Bottom up & top down buy-in - actually change!

To have any hope of making an agile transformation stick, regardless of the framework you choose, you must get buy-in. Leadership buying into the benefits of the process will help you get the resources you need to make this transformation possible. Development teams buying in will ensure that you're able to see measurable changes in your company's overall velocity. If you do not have full buy-in from both the top and bottom of the organization, you'll be just putting another coat of paint on a rusty old machine.



You have buy-in across the entire organization on this new direction. Your development teams are excited about a new way of working (even if it's one team at a time), and your leadership fully supports the process to make this transformation.

They also speak not just with words but with the corporate budget, providing you with the resources you need to make this transformation a reality. Your leaders have also attended necessary training for this new way of working, and are in the trenches of understanding with you.



Leadership in the organization has mandated that your company will now be moving towards an agile at scale model, but nobody understands what that means or how it impacts them.

Your development organization has been reluctant to embrace this new change. They don't understand it and they seem to think it adds a lot of extra work for no benefit up front. Conversely, your teams want to adopt better agile at scale practices, but your executive team are still pushing down directions on what to build, when, and how.

7. People, lean & systems thinking

Last but not least, without a healthy culture to support your organization's agile at scale transformation, it will fail. Changing culture is difficult and it doesn't happen overnight. The culture at your organization is the result of years of practices and procedures that successfully got your company to where it is today.

To initiate your organization's cultural shift, start by changing the way you think about how you deliver software to market.



You take a true "Enterprise View" of your software development process going beyond the developers and the code they're working on. You account for and map the different aspects impacting your end-to-end software delivery. You look at the entire system as a whole, from coding and software to delivery, marketing, and sales motions. In addition, there are many great principles of Lean Management that help ensure you are able to continue moving your enterprise forward in a rapid manner. And at the heart of it all, you take a "people first" approach to your agile at scale transformation. Your mandate isn't based on what a framework told you to do, but instead is based on what's right for the people who come to work every day.



You've found a framework for scaling agile, and you're sticking to it no matter what. You've thrown out all of the things that made your company successful, and you're starting over from scratch. You've asked every single person to undergo a radical transformation in their day-to-day working lives, leaving them scratching their heads, frustrated, and potentially very burnt out. You've become so focused on getting software to market that you're tracking success by the number of lines of code delivered to production. Your idea of agile at scale is to just make sure all of the teams are agile, and add more agile teams to the development organization. You've directed change in development, with no change in management, strategy, or execution.

With a solid foundation of the overall journey and excitement to tackle the 7 principles we've outlined above (the what), now it's time to start figuring out the implementation process (the how). We'll help outline the best practices we've identified for implementing scaled agile. We find these principles, again, are quite common across any of the frameworks. To make a cooking analogy, think of these as cuisines and the frameworks as specific recipes.

An interview with Dom Price

Work Futurist at Atlassian



Tell us about your role as a Work Futurist for Atlassian. Why is agile at scale a popular topic that comes up as you meet with industry leaders?

DP: Virtually every organization in the world is going through a transformation, whether that's digital, cultural, or agile. Organizations understand that what got them to where they are today won't get them to where they want to be tomorrow. And they come to us wanting real-life examples from people on the ground of how to drive these transformations.

My team goes into organizations like ANZ Bank, where we work with them to help them understand the change that they're about to come up against, and the congruence between tools and practices at all levels of the organization.

As a Work Futurist, there are two components to my role:

- 1.** Help Atlassian understand how we continually scale and evolve, because our biggest existential threat is standing still when the world around us is changing.
- 2.** Under the banner of our core value 'Open Company, no B**S**', as we learn ourselves, we want to share our story with other organizations. When we share with other organizations how we've scaled, we hear back from others on what's worked exceptionally well and what's failed, which we bring back internally.

The reason agile at scale is such a hot topic right now is because it's complex. There's no single way of doing it that you can cut/copy/paste and achieve. By the very nature of it, it often consumes a large part of the organization and takes a long time, if not forever. While agile at scale can feel like quite a scary thing to embrace, it's incredibly valuable once you start on that journey.

Why is agile at scale a thing in the first place?

DP: Let's start at a meta level with the sheer rate of change in business:

There's more competition than ever before, with lower barriers to entry and competition from non-traditional businesses. Consumers no longer compare your business to your competitors - they compare you to the best experience they've had recently. At the same time, there's a war for talent to hire the best people.

What we're seeing is large, incumbent enterprises getting threats more than in the past. Banking, telco, and insurance are being challenged and threatened by startups offering similar services. These large organizations need to achieve the same nimbleness, adaptability, and agility. And that requires a new way of working.

Ultimately, you need to be innovative, operate with agility, and adapt to the ever-changing environment that you're in. One way of doing that is turning large, monolithic pieces of work into manageable bite-sized chunks so you can experiment and explore.

The whole rationale of agile at scale is inherent uncertainty. If you have inherent uncertainty in what you're doing, how you're doing it, and who you're doing it for, agile at scale lets you experiment and learn a lot more quickly.

What are the top things people get wrong when scaling agile?

DP: 1. The training course is the answer.

I've seen it too many times from senior leaders who say, "But I sent 1,000 people to an agile course and nothing changed." Exactly, I agree. Remember that the people and the environment need to be congruent. If you send people to a training course on how to be agile, but don't create an environment to be agile, you've probably disengaged them more and made things worse instead of better.

2. Senior leaders think agile is something other people do. The best results I've seen come from senior leaders taking on the mantle of modeling agility.

3. The agile compliance regime. When people decide that agile is the answer, they create a checklist to "make everyone agile." While it looks agile on the surface, the agile spirit, philosophy, and intent are not understood. They measure the rituals (i.e. did you do a standup, a sprint planning exercise, etc.), but not the intent of the rituals.



What advice would you give to people who are trying to scale agile in their organizations?

DP: A few things come to mind...

1. Realize that agile at scale is a never-ending path. It's not a certificate that you achieve and then you're done. There's no finish line, there's just a lot of milestones along the way.

2. Work out who you're solving for, and ideally make that the customer. Agile at scale is about working across the organization to delight your customers with the best thing in the quickest way possible.

3. Work with other teams to be effective. If all you do is optimize how your team works, then the whole attempt hasn't been successful. You may have to compromise to benefit the whole system. Find a way to work laterally across the organization with shared goals, shared outcomes, and shared backlogs, finding ways to experiment and see success.

What is holding organizations back when they try to scale agile?

DP: Unlearning the old ways of working. Most organizations want to buy and implement the answer. If they don't give themselves the capacity to unlearn old habits and rituals, they won't have the time and freedom to practice new ones. You're going to struggle if you expect to get it right the first time.

That doesn't mean stop, it means persevere. Give people the time and freedom to turn agile at scale into a real muscle for the organization, then you'll be successful.

What value does Atlassian offer organizations looking to scale agile?

DP: We're scaling agile ourselves. We're living and breathing it every day, and we're happy to share our experiences about what's working and what isn't.

We're also listening to hundreds of thousands of organizations going through this experience right now and sharing what's working out there.

We not only think about our tools but our practices together. Not only will we provide you with the tools to enable you to think about agile at scale, but also the human to human ways of working to amplify the power of those tools. ●

Best practices for implementing scaled agile

How do I get started?

Now that you know what scaling agile is all about and have a bit more insight into the stages of an agile journey, you may be wondering: *what's next? How do I get started?*

First, we'll start by acknowledging that scaling agile is not easy, and won't happen overnight! There are changes at stake for people's roles and responsibilities, the processes they follow and how teams work together in new and sometimes uncomfortable ways. But that isn't to say it's not a worthwhile undertaking—it is. And it starts with motivated change agents like you.

Whenever you're considering making a major change to teams, processes, or even the tools that support the former two, it's important to begin with the end in mind.

To the right are some best practices to consider when getting started with scaling agile. Remember, each journey is unique—so feel free to move these plays around, skip some, or add others. There isn't a “right” way to scale.

Understand the process.
In this next section, you'll learn how to:

1. Assess where you are in your agile journey
2. Define the scope for your MVP
3. Make the pitch to leadership ...and the teams
4. Decide on areas of alignment
5. Measure your success
6. Just get started!



Assess where you are in your agile journey

Ground yourself in some data to get started.

- How many agile teams do you have?
- What team-level agile metrics are you measuring across your organization?
- Do you have stable teams who are able to operate autonomously and make their own decisions?
- How predictable and high-quality are your releases?
- Do your teams know why their work matters in the bigger picture of the company strategy?

Sometimes, it's helpful to take an agile maturity assessment to establish a baseline.

There are a number of existing resources you can consider, and many other examples of companies who have defined their own agile maturity model internally. Check out the [Sit, Crawl, Walk, Run, Fly](#) agility maturity model on page 5 for a starting point.



Start at the team level

You can't effectively scale if your agile fundamentals aren't in place. It's perfectly okay (and recommended) to start by focusing your efforts on team-level agile planning and DevOps practices before scaling up to teams of teams, programs, or portfolios.

Define the scope for your MVP

If you've been through an agile transformation before, you might already know that the first unofficial rule of scaling agile is: "Don't." That's why a good practice is creating a minimum viable product (MVP) for scaling agile in your organization.

Your biggest risk is over-scaling, taking on too many changes or new processes unnecessarily, or applying scaled agile practices to a team that doesn't require them, which adds unwanted overhead.

When scaling agile across an entire organization, focus on "just enough." Successful agile development at scale should mirror agile development at the team level, which means preserving transparency, responsiveness to change, and a focus on integrated, working software at frequent intervals.

Whether your organization starts with Scrum@Scale, SAFe, a custom process, or another established methodology, remember that the process itself should be agile. Keep testing new ideas and making incremental improvements.



Find common problems to solve that exist across the organization

To increase your likelihood of getting buy-in and proving success, find something that everybody can agree is a problem, and then focus on that as you define your scope and goals for an MVP.

Dependencies are a good problem to consider first: all teams have them, and they become more challenging to manage as your teams and programs become larger.

Make the pitch to leadership... and the teams

Is your organization in the middle of an agile transformation from a top-down mandate driven by leadership, or a bottom-up movement starting with agile teams? We've seen agile at scale unfold both ways.

But either one of these approaches, in isolation, is destined to fail. A command-and-control agile transformation will see team resistance. In fact, a command-and-control agile transformation is actually a contradiction, since agile is about team-level autonomy wherever possible. The key is to focus on alignment and shared concepts without stifling or overloading your teams.

“This [new process] just adds work” is a common objection. Teams don’t want to input additional data around an initiative. It feels like a burden and the benefits to the team aren’t clear. You wind up with limited buy-in from the team, and from the middle of the organization, so the changes don’t take hold.

When agile is adopted from the team level, there can be some fear that creeps in at the middle and executive levels of the organization. There are concerns about loss of control, or not being able to easily access data to understand if the investments that you’ve agreed to as an organization are actually being executed. From an executive standpoint, they may find themselves wondering “Why am I making this investment? The teams are already agile, why do I need to invest in something that’s going to scale?”

Instead of thinking about your adoption of scaled agile as command-and-control or just grassroots, think of it a cycle where strategy, goals, and priorities are set at the top level of the organization, and flow down, and information from teams and customers cycles back up. Teams are empowered and in turn, deliver predictability, information on spend, and achievement against outcomes.



To stay competitive and move faster, ANZ Bank embarked on a scaled agile transformation, embracing iterative processes and constant feedback. Their initial decision to change course, with its proper authority, the one that sets the direction and puts people in action, came from the top.

“This change absolutely started with our CEO Shayne,” says Darren Pratt, Technology Lead Customer Engagement. “I want to be clear on that. It did come from the top. Shayne was upfront and clear. This was his vision. We had very clear leadership.”

[Learn more about ANZ’s story.](#)

Decide on areas of alignment

There will be a time within your journey to agile at scale where you need to decide whether or not you need to push forward and insist on a common approach to organizing teams' work or whether it's okay for your teams to locally optimize. When do you have to optimize for the system and get teams to agree, and when can teams do what they want?

Here are three areas to consider:
Terminology, Cadence and Estimation.

Terminology

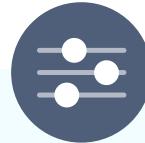
Do you insist that teams align on a common semantic understanding of the work that's being done across the organization? Or can teams call things what they want?

What do you call the work item that sits above a story? We've seen organizations end up in two-week debates about whether it will be an Epic, Initiative, or Feature. The passion that goes into these kinds of debates is somewhat unbelievable but understandable. But is this a debate that you need to have? Is it a hold'em or fold'em moment?



Align the teams

There's plenty of ways to approach this. But ultimately, we recommend standardizing your terminology as agile scales within your organization. You want to be able to compare apples to apples versus apples to oranges. If you're not speaking the same language across the organization, collaboration gets very confusing.

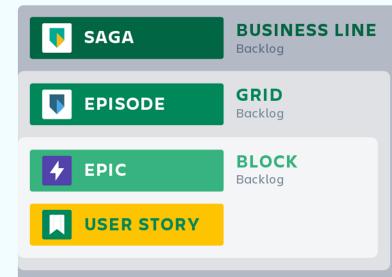


Terminology: what you name work matters

Naming your work hierarchy can be unique to your organization, even as you align the terms

ABN Amro decided to create story-themed items in their hierarchy above the Epic.

[Learn more about their story.](#)



Cadence

These are the increments of time that your agile teams will be working in. The sprint is the most common increment of time for agile teams. But the challenge is: do all teams need to work in sprints?

Some teams will want a one-week sprint, others two weeks, some three. Teams will have different preferences for which day of the week their sprint starts on. Does it matter?

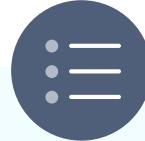
As you scale, you end up with a desire from some stakeholder to align on a common cadence for sprints across the organization, or at least within a program or a portfolio. At the same time, other parts of the organization want to maintain a sense of independence so that teams are empowered to choose the length of their sprint and when they want to work.



Leave to teams

In our experience, common cadence doesn't matter too much, as long as each team is driving towards some level of predictability and the sprints can roll up and there's some sense of how those teams work together.

In general, the specific days that sprints start or the length of a sprint can be aggregated and normalized at scale. You can still use the agile metrics to roll up what teams are doing from an investment perspective across the organization.



Common cadences in SAFe

Using SAFe? When you're in a common program increment (PI), the teams need to align more closely to use something like a program board. In this scenario, it becomes pretty important that locally, within that program, teams are aligned on sprints within a PI.

Estimation

When other business stakeholders are looking at what the teams are doing, they may be curious why teams estimate similar pieces of work differently. Teams have a different sense of what a point is to them. But how do you get every team within the organization to normalize on a common understanding of what a story point is, so we can roll information up?

For example: Over the course of a quarter, you'll summarize the total number of story points for a program, which may be 15 teams that are working together. You can then look at the velocity of the program, and as teams contribute to that, it's fine that there's some differences between how each individual team does their estimation.



[Leave to teams](#)

In general, what we've seen is that the law of large numbers tends to normalize variance in scoping across lots of teams. Once you've scaled agile and are looking at metrics like velocity across more than 2 or 3 teams, you're in the realm of enterprise agility and the ability to expand up to dozens and hundreds of teams across thousands and tens of thousands of developers, the law of large numbers washes out the differences.



Comparing team productivity is a fast track to losing trust

Velocity may be one of your goals for scaling agile. But that doesn't mean it's okay to compare two teams with different story point estimations and conclude that one is performing better than the other. Once you start comparing teams based on story point throughput, all you'll succeed in doing is getting teams to game their metrics.

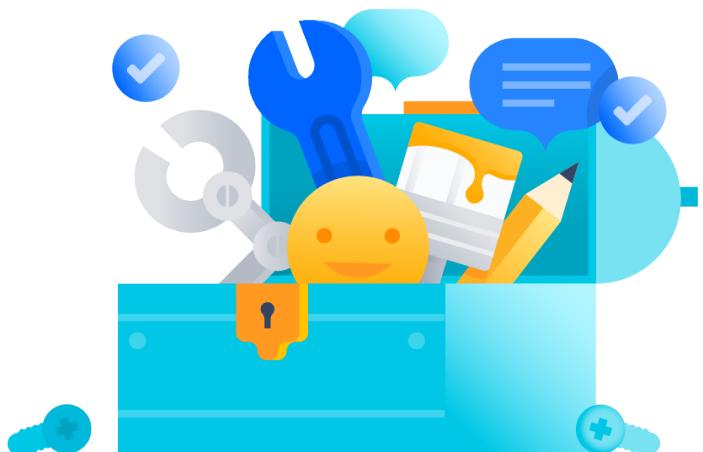
Measure your success

It's important to have objective metrics laid out that will help you know whether or not you're making progress towards your goals of agility. Whether you use KPIs, OKRs, or a different set of agile metrics, define these, establish a baseline, and use them as guideposts to refer to over time.

Just get started!

Finally, it's important to just get started. Aim for progress over perfection, while understanding what that progress looks like. This is part of why defining your internal MVP for scaling agile is so important—it's exceptionally difficult to transform your entire organization's ways of working at once.

Stay focused on demonstrating successes in the context of your MVP; once you have a successful trial run of scaling agile, advocating for leadership and team buy-in will become easier.



How to use Atlassian tools for agile at scale

Putting the tools in the context of people, principles and practices

Once your organization has decided to adopt agile at scale, you'll need to figure out how to visualize work and track dependencies. This can feel like an overwhelming task, especially if you need to provide updates to your leadership team. After all, you want to ensure information is clearly presented and easily digestible for executive stakeholders. If your team is co-located in one office, you might already be using physical whiteboards, post-its and index cards to visualize work.

Or, maybe you're using spreadsheets to manually track work. You might even be exporting reports from another tool and formatting them into a spreadsheet. Physical representations of work and spreadsheets are great, but oftentimes, updating your work becomes a chore and can be time-consuming, manual, and prone to human error. This is where tooling can help!



Tools can help organizations amplify existing practices and realize the full benefits of those practices. Scaling your tools is also a way to support scaling your practices. There are a few benefits to using agile at scale tools:

Transparency

When your team's work is tracked in a tool, other teams (including your leadership team) have visibility into the work being done. Tools can serve as a single source of truth to keep everyone on the same page, even if they're not based in the same office as you. Having the right tools can help your organization ensure it's delivering a positive impact on customers, and that teams are working together to meet organizational objectives.

Consistency

When individual product or program teams track work in their own ways, a lack of consistency can make it hard to visualize work across multiple teams. Tools are only as accurate as the data stored inside them, and they can help standardize the way teams share work and highlight dependencies. At scale, tools can also help you track organizational investments versus value delivered.

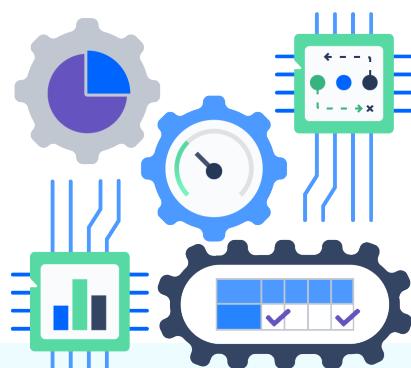
Predictability

Tools can help calculate velocity, capacity, and other important measurements to help you determine when your team's work will be on time (or not).

"A transparent work environment leads to greater trust and alignment and for a global organization, that's paramount to our success."

Katie Burke
HubSpot, Chief People Officer

Having the right tools can support major change to teams and processes, but they're not a panacea. Even with the right tools, teams need to understand their roles and responsibilities in order to work better together.



Let's take a look at how you can use Atlassian tools for agile at scale.

Jira Software Data Center

A foundation for your transformation

The screenshot shows a Jira Scrum board titled "Android Sprint - 09.01 - 09.15". The board is divided into three columns: TO DO, IN PROGRESS, and DONE. The TO DO column contains tasks like "ADR-11" and "ADR-9". The IN PROGRESS column contains tasks like "ADR-6" and "ADR-7". The DONE column contains tasks like "ADR-2", "ADR-3", and "ADR-14". A sidebar on the left provides navigation and filtering options.

Jira is the one place you can go to understand the work that is happening across your organization. Jira Software Data Center is designed to support the greatest number of users, issues, custom fields, and workflows. Support transformations of any size, from enabling teams to practice DevOps or agile in the ways they want to, to running agile at scale across your enterprise.

Roles / Practices

Developer

Uses Jira to create user stories and issues, plan sprints, and distribute tasks. Their team uses Jira to prioritize and discuss work before every release.

Atlassian Admin

Responsible for maintaining a healthy Jira instance while focusing on implementing best practices and governance for the next wave of users.

“Once we onboarded teams onto Jira, they started to see the benefit of giving more visibility to colleagues. Instead of managing projects by having tasks and tickets spread across email and spreadsheets, they could manage projects more efficiently using scrum boards.”

Denis Boisvert

Atlassian Tools Product Owner
National Bank of Canada

Confluence Data Center

Bring people and ideas together to create shared context

The screenshot shows a Confluence page titled "Android Team Strategy". The page header includes navigation links like "Confluence", "Spaces", "People", "Questions", "Calendars", "Create", and a search bar. Below the header is a sidebar with icons for dashboard, teams, and other features. The main content area has a title "Android Team Strategy" and a subtitle "Team vision" with a lightbulb icon. A sub-section "Goals" lists three items: "Ship Android app version 2.0", "Achieve 95% customer satisfaction across all versions of the app", and "Ensure Android and iOS apps have feature parity". Another section "Measures" contains a table comparing Baseline and Target metrics for app downloads, monthly active users, and returning users. A final section "Milestones" is partially visible.

Confluence brings people and ideas together, providing a layer of connectivity throughout your organization. Teams rely on Confluence Data Center to collaborate around the clock and around the globe. Create requirements and documentation or use Confluence to communicate and create shared context throughout your enterprise.

Roles / Practices

Developer

Uses Confluence to create documentation for their work. Their team uses Confluence to share context around each release.

Atlassian Admin

Responsible for maintaining a healthy Confluence instance while focusing on implementing best practices and governance for the next wave of users.

Release Manager

Uses Confluence to manage the release process, including process checklists.

Product Manager

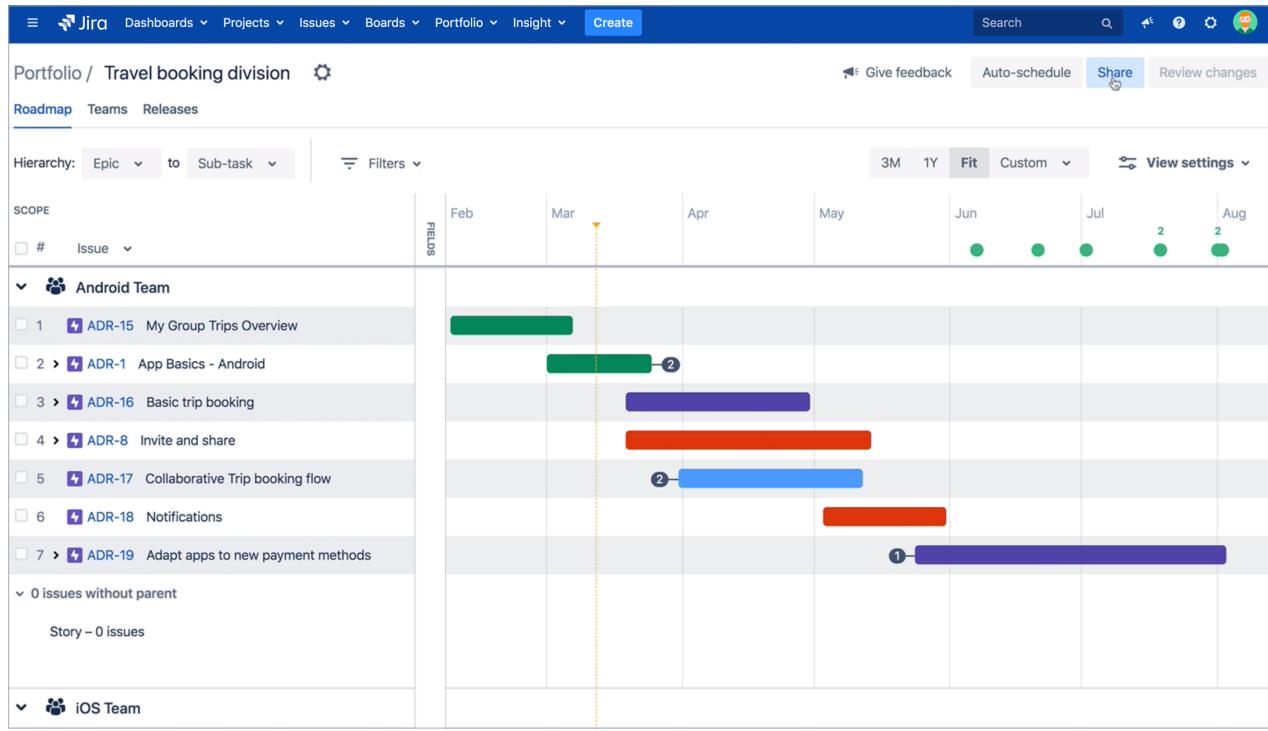
Creates product requirements and seeks feedback in Confluence pages. Uses Confluence to keep others in the organization updated on progress.

All other roles

Use Confluence pages as flexible, collaborative containers for information. They may keep pages organized in spaces, and share relevant pages with other teams. Powerful search capabilities allow individuals to find the information they need, when they need it.

Portfolio for Jira

Visualize work for teams or a single program



Portfolio for Jira helps you visualize the work happening in Jira Software. It's a great fit for teams of agile teams, up to the program level. It helps you create a realistic plan, manage dependencies, manage capacity and share a visual roadmap of future releases.

Roles / Practices

Release Manager

Uses Portfolio for Jira to evaluate the scope of future releases, whether work is at risk, and what can be done to make sure teams deliver on time.

Product Manager

Portfolio for Jira helps them visualize teams' progress, status, and any red flags. They use Portfolio for Jira to understand how work is progressing across one or more projects/teams, and how to ensure teams deliver on time.

Program Manager

Portfolio for Jira helps them visualize teams' progress, status, and any red flags. They use Portfolio for Jira to understand how work is progressing across the program.

“Portfolio for Jira helped us scale agile across our organization. It enables teams to align on shared objectives and gives us insight into our portfolio that we didn't have before.”

Rik de Valk
Product Owner at ABN-AMRO

Jira Align

Connect strategy to execution and outcomes

The screenshot shows the Jira Align Program Board interface. At the top, it displays 'Portfolio: Digital Services' and 'Program Increment: PI-5'. The main area is titled 'Program Board' with a sub-header 'Are we on-track based on our Program Increment com...'. It shows a timeline from 'June 2019 To September 2019' with sprints S23 through S28. The board is organized by 'Program: Mobile' and includes sections for 'Unplanned Sprint' and 'Team and Program Objectives'. Two teams are visible: 'Baltimore' and 'Cowboys'. Each team has a list of work items, each with a unique ID, a brief description, and a progress bar indicating its status. The Baltimore team has 11 work items, and the Cowboys team has 10. The interface also includes various navigation and configuration buttons at the top right.

Jira Align helps solve business agility challenges by connecting strategy to work execution and outcomes. It helps visualize teams' progress, status, and any red flags. It's built to improve value delivery at every level - across programs, solutions, and portfolios. Jira Align is the only solution that can support your choice of framework, including custom or hybrid versions.

Roles / Practices

Program Manager/Release Train Engineer

They use Jira Align to understand and prioritize the scope of work and conduct long-term planning. They also use it to see how work is progressing across multiple program or ARTs.

Product Manager

They use Jira Align to understand how work is progressing across projects/teams, and how to ensure teams deliver on time.

Portfolio Manager

They use Jira Align to understand how work is progressing across one or more projects/teams.

Director of Agile Solutions / IT Solutions

They use Jira Align to understand how work is progressing across one or more projects/teams. They choose tooling to support their business' day-to-day operations, reaching certain objectives, and aligning with agile practices.

Executive (VP of Product, CTO)

Jira Align provides executives and portfolio managers a better sense of the capacity of the organization, their ability to deliver against strategic objectives, and insight into which strategic bets are generating value for customers and the business.

Putting this into practice, here's an example of how you can take the outcomes of an in-person planning session and translate it into Atlassian tools.

In-person planning sessions are great opportunities to meet with your team, discuss strategy, assess resources and build a plan. It's important to remember that throughout this process, tools are meant to support major change to teams and processes and empower your organization to collaborate better together.

1. Before the session

In each tool, verify any upcoming dates, including sprint dates, company-wide holidays, and the date of your next in-person planning session. Make sure teams have been defined with proper allocations.

Check that all business and technology stakeholders are aligned on dates and objectives for the next planning cycle (e.g., program increment). This can be done by walking stakeholders through your tools on a prep call before your planning session.

Have program managers create physical cards (an index card or post-it is fine) to prioritize features and stories according to your program objective. Then, make sure you make a digital version of each card in your tool of choice. Or, if you're using Jira Align, you can create digital cards first and then print out pre-defined features and stories for your planning session.

Regardless of which version—digital or physical cards—comes first, you'll also need to prioritize features and stories on a physical board. That way, you can bring sprint boards and program boards to the planning session.

Review any outstanding work items from your current planning cycle. Make sure these are addressed (e.g., reassigned or updated) before closing out the cycle.

2. During the session

Start off by recapping your last planning session and planning cycle. Many leadership teams use presentation decks for this, but you can also encourage them to share the Jira Align views they visit most regularly. Sharing their views will help everyone in the room review the last cycle and celebrate the milestones your teams have accomplished.

3. After the session

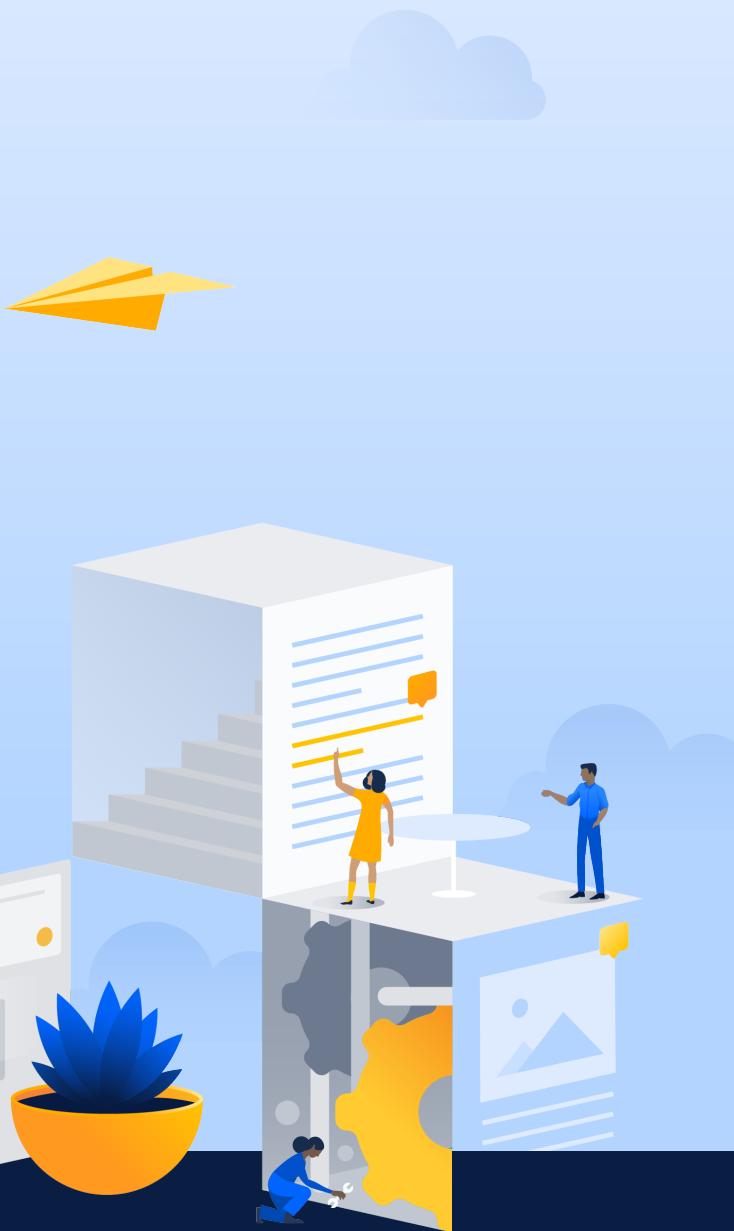
Capture the outcomes of your program board in your tool. Assign someone on the team to take pictures and perform a bit of data entry. Remember, your program board should serve as a one-stop shop for a quick scan of progress.

Now that you understand the new rules of agile at scale, how to understand your organization's agile maturity, and how to get started, what are you waiting for?

We know your agile journey will always be a work in progress and we're here to support you at each step.

Find more information about Atlassian's agile at scale solution on our website:

atlassian.com/software/jira/agile-at-scale



Additional resources

- [Agile at Scale](#)
- [Scaling Agile to the Enterprise](#)
- [Gartner's 2019 Magic Quadrant](#)