




WHITEPAPER

AGILE METHODOLOGY

Implementing the Best Scaled
Agile Framework for Faster and
Better Business Results

Akhila Pant





Scaled Agile Framework - how it can be applied to any organization to get better business results

Implementing the Best Scaled Agile Framework for Faster and Better Business Results

Challenges, Comparison of Frameworks, Recommendations, Use Case Scenarios

Almost all IT organizations have adopted agile software development techniques in some manner. The results have been positive for the most part. However, few organizations find that agile works well for them when the teams are small and face relatively straightforward situations. They believe that their teams were struggling and spending significant effort to determine how to be agile in the situation that they face. It is not this hard. The good news is that organizations are applying agile techniques at scale and are succeeding in doing so. They find astounding business results, a substantial increase in quality and productivity.

This paper is about the Scaled Agile Framework and how it can be applied to any organization to get better business results. It attempts to provide answers to the following questions:

- > What challenges do the teams face when scaling agile?
- > What are the frameworks used to scale agile, and how do they compare?
- > What have you to do to scale agile delivery?
- > What is Disciplined Agile Delivery (DAD) and why?
- > How does the process-goal-driven strategy enable scaling?
- > How do disciplined agile teams work at scale?

Agile teams achieve better quality, higher levels of stakeholder satisfaction, quick time to delivery and better ROI.

Challenges of Agile and Scaling

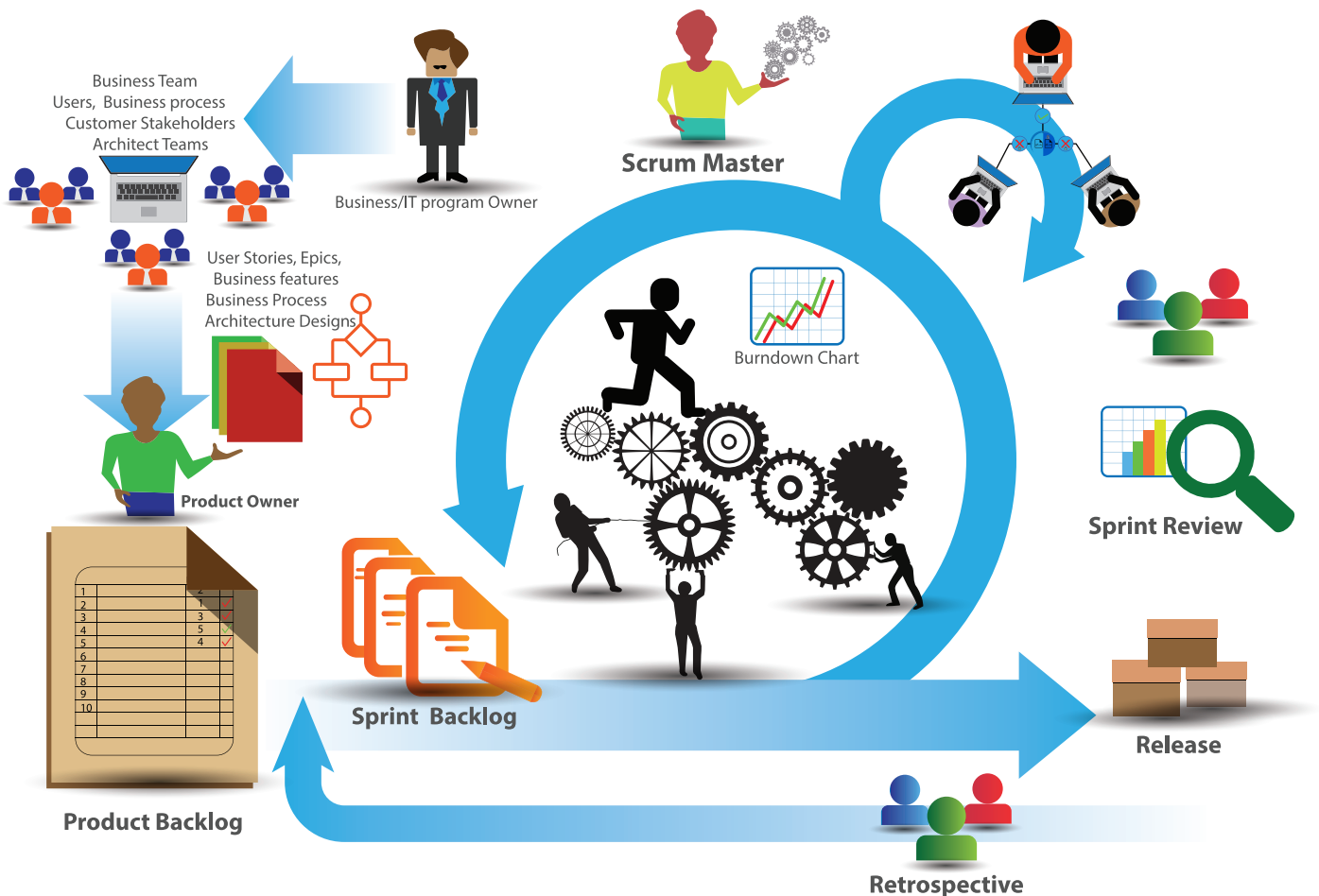
Majority of organizations have embraced or are in the process of adopting agile software development techniques. For many of them, this works out well. Agile teams achieve better quality, higher levels of stakeholder satisfaction, quick time to delivery and better ROI. Organizations experience challenges with applying agile strategies at scale. This happens because agile methods are:

- > **Software-focused:** Agile promotes the philosophy of working software becoming the primary measure of success. Software alone should not be the only measure. Development teams are also dealing with hardware related issues, documentation, evolving business processes around the usage of the system, etc. Shouldn't we consider all of those questions and not just software?
- > **Construction-focused at the expense of delivery:** Scrum, one of the agile methods, focuses on the construction part of the software lifecycle, yet agile teams in practice still need to perform initiation and release activities. They are likely to forgo the benefits provided by a bit of up-front, strategic thinking.
- > **Oriented towards small teams in straightforward situations:** Most of the agile methods are oriented towards small group size, who are either co-located or near-located, who have access to the primary stakeholders and who are working on software. What about large teams?
- > **What about geographically distributed teams? What about teams working in complex situations? Can't we implement agile there too?**
- > **Prescriptive at the expense of flexibility:** Agile methods such as Scrum are rigidly prescriptive. For example, in Scrum, one way to address changing requirements is the product backlog, one way to coordinate activities within a team is through daily Scrum meeting, and so on. And the Scrum community is very clear that this is the way it has to be done. Teams work differently in different situations, and they need several options to choose from to achieve these goals. Indeed, advocates of lean techniques have argued that iteration-based approaches like Scrum are not always the best choice.
- > **Too narrowly defined:** Agile tend to focus on a portion of the overall delivery process. We need to figure out how to fit them together into a cohesive whole that is right for you.

Agile methods provide the starting point but need substantial effort to make them scalable.

> The team is focused at the expense of the overall enterprise: Scrum stresses the value of sheltering the team from external distractions. This is often used as an excuse for not collaborating with other delivery teams, which results in discrepancies and miscommunications. It leads the team to increase technical debt, miss opportunities for reuse, and even implement functionality that exists in other products.

Agile methods provide the starting point but need substantial effort to make them scalable.





Agile has various approaches and frameworks in place to make the transition painless

Comparison Matrix Of Scaled Agile Frameworks

Worshippers of Agile argue that scaling this framework is pretty straightforward. Just let your teams and units decide what's best for them, and trust them to improve their work. Agile has various approaches and frameworks in place to make the transition painless.

Table 1. Agile approaches and frameworks

Criteria	Large Scale Scrum (LeSS)	Scaled Agile Framework (SAFe)	Disciplined Agile Delivery (DAD) + Agility at Scale
Description	LeSS helps to implement Scrum in the organization. It is a minimal framework that offers a high degree of flexibility for implementation. It is non-prescriptive, and merely gives suggestions.	SAFe is a highly structured and prescriptive method that helps large enterprises to set out on the road to Agile. The Scaled Agile Framework is mainly implemented at three levels: Team, Program, and Portfolio.	Disciplined Agile Delivery (DAD) process decision framework is a people-first, learning-oriented hybrid agile approach to IT solution delivery. DAD's strength is in architecture, design, and DevOps. On another hand, its documentation lacks coherence.
Portfolio	Medium	Medium	Medium
Program Structure	Medium	High	High
Inter Team Coordination	High	High	High
Team Level	Medium	High	High
Tech Practices	Medium	Medium	High

Here DAD proves to be an appealing option.



DAD inspires teams to be more flexible in your approach. It has been found that four of the twenty-two process goals seem to take about 80% of the tailoring impact

DAD's Life Cycle

Concept: This is the phase where we envision the idea.

Inception: This phase comprises the following activities.

- > Form initial team
- > Develop shared vision
- > Align with enterprise direction
- > Explore initial scope
- > Identify initial technical strategy
- > Develop initial release plan
- > Secure funding
- > Form work environment
- > Identify risks

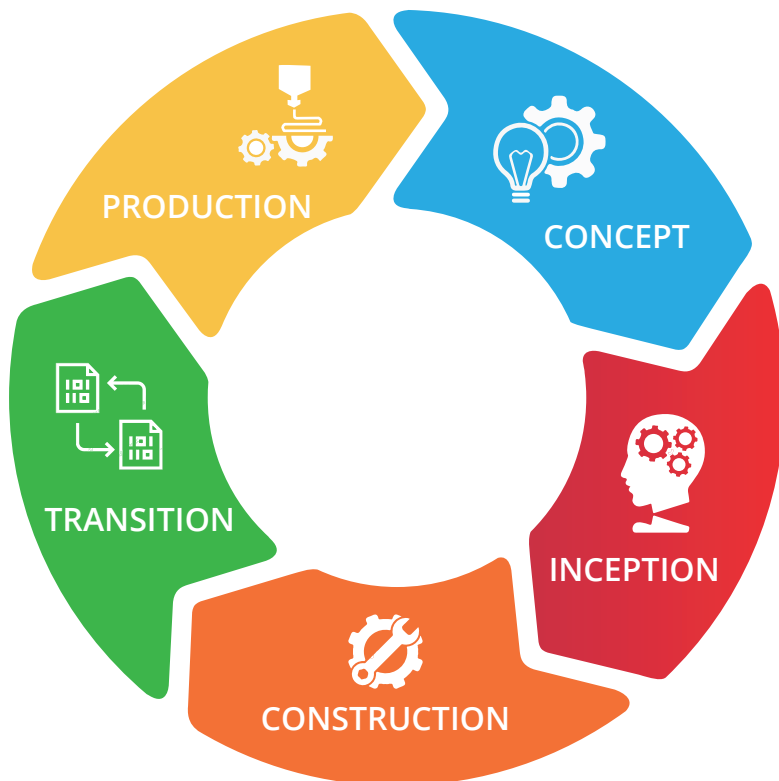
Construction: This phase defines how we produce a solution.

- > Provide a potentially convenient solution
- > Address changing stakeholder needs
- > Move closer to deployable release
- > Improve quality
- > Prove architecture early

Transition: This phase defines how we deploy.

- > Ensure solution is consumable
- > Deploy the solution

Production: This phase is a pure support phase.



DAD's Goals:

- > Explore initial scope
- > Identify initial technical strategy
- > Move closer to a deployable release
- > Coordinate activities

Scaling Factors Faced by Agile Team

What are the scaling factors that should be considered when tailoring our approach? The table below summarizes the six scaling factors of the Software Development Context Framework and indicates the range of each of its factors. Any given team will find itself somewhere close to the simple extreme.

Table 2. Scaling Factors of the Software Development Context Framework

Factors	Simple Extreme	Challenging Extreme
Team Size	2	1000
Geographic Distribution	Co-located	Global
Organizational Distribution	Single Division	Outsourcing
Compliance	None	Life Critical
Domain Complexity	Straightforward	Very Complex
Technical Complexity	Straightforward	Very Complex

DAD's Goal Driven Approach to Scaling

DAD inspires teams to be more flexible in your approach. It has been found that four of the twenty-two process goals seem to take about 80% of the tailoring impact.

These goals are:

- > Explore initial scope
- > Identify initial technical strategy
- > Move closer to a deployable release
- > Coordinate activities

We have mentioned considerations for each objective below.

The greater the domain complexity, the team will need to invest more in modeling its initial requirements.

Explore Initial Scope

When a project release begins, one of the process goals which team needs to address is to explore initial scope.

- > The team needs to have at least a high-level understanding of what they're trying to achieve.

- > Teams should be able to answer few of the fundamental questions like what you are attempting to make, the duration and the overall cost of it.
- > DAD recommends starting with usage modeling, domain modeling and non-functional requirements.
- > The team needs to think even about addressing its non-functional requirements like availability, security and performance.
- > The greater the domain complexity, the team will need to invest more in modeling its initial requirements.
- > Geographical distribution of the team may require you to spend more effort in capturing and communicating the needs or requirements.

Identify Initial Technical Strategy

- > Another significant phase which comprises of initial architecture envisioning or simply initial architecture modeling.
- > First, the team should think up-front about their architecture so as to identify a viable strategy to fulfill that.
- > Second, the team will have to think about leveraging their existing frameworks, web services, etc. to reduce the overall time and cost. This is an aspect of DAD's philosophy of working in an enterprise sensible manner.
- > The team will have to evolve a strategy to build a solution from scratch.
- > The greater the technical complexity, the more the team's need to invest in its initial architecture modeling.
- > Work for the geographically distributed team by enabling them to work in greater isolation and thereby reducing collaboration overhead.

The majority of projects face testing and quality assurance early in the life cycle so that they can reduce the cost of fixing defects.

Move Closer to a Deployable Release

- > Moving closer to a deployable release is another important process goal.
- > First, it deals mainly with the packaging aspects.
- > Second, it also deals with deployment planning
- > Third, this goal covers critical verification and validation

procedures. The majority of projects face testing and quality assurance early in the lifecycle so that they can reduce the cost of fixing defects.

- > This process goal is primarily affected by the scaling factors of domain complexity, technical complexity, and compliance.

Coordinate Activities

- > Coordination plays another major phase here.
- > First, one of the major issues found in the projects is the artifact ownership and coordination of the team.
- > Second, as DAD's teams are enterprise-aware, it describes the strategies to coordinate with an external group.
- > Third, the team needs to address issues related to scaling.
- > Also, the team will have to devise strategies to communicate within the team based on its size.
- > A technically sophisticated project team would prefer to adopt the architecture owner and the ones having domain complexity would have product owner as a part of the team.
- > A team where there is vendor involvement may choose to adopt a disparate ownership strategy for some project artifacts.

Use Case Scenarios

Scenario A: Large Development Team

This scenario describes a large software development initiative to build a large number of features in parallel to meet an aggressive timeline.

A technically sophisticated project team would prefer to adopt the architecture owner and the ones having domain complexity would have product owner as a part of the team.

Table 3. A large software development initiative

Scaling Factor	Situation Faced
Team Size	120 total team members available for this initiative
Geographic Distribution	Distributed team across four offices within two time zones
Compliance	Internal compliance
Organizational Distribution	Stakeholders are willing to relocate near the team
Domain Complexity	Complex
Technical Complexity	Medium complexity

Table 4. Process decision for each of the key goals (large software development initiative)

DAD Goal	Strategy
Explore Initial Scope	<ul style="list-style-type: none"> > Chief product owner to facilitate the creation of vision. > Product ownership team will meet with other stakeholders to obtain a common understanding of the initial scope. > A program backlog of features and work item lists are created for each team. > Architecture and product owners will meet to agree upon system-wide non-functional requirements for all delivered features.
Align with Enterprise Direction	<ul style="list-style-type: none"> > Architecture owners outline standard guidelines and come with the mandatory guidelines. > A User Experience (UX) expert will collaborate with all the teams to ensure consistency and optimize the user experience. > Enterprise Architecture (EA) and Product Management teams to have regular meetings to ensure coherence and manage dependencies across teams.



DAD Goal	Strategy
Identify Initial Technical Strategy	<ul style="list-style-type: none">> Architecture owners design the architecture at a high level.> Technology diagrams will be created using a few UML diagram, and UI prototypes will be hand-drawn. All models will be built and evolved in informal modeling sessions. An existing reference architecture will be used, if applicable.> Attempt will be made to reuse services from existing systems.
Move Closer to Deployable Release	<ul style="list-style-type: none">> Team to follow two weeks Sprint. The work would be integrated and published externally to stakeholders via a transition phase.> The product owner to come up with user documentation.> The architecture owner to update the architectural handbook.> Those teams that are not yet ready to adopt test-driven development would follow test-after programming approach.> Build-processes and regression-test-execution to be automated.> Informal code reviews are done as needed.
Coordinate Activities	<ul style="list-style-type: none">> Information sharing to be primarily via direct conversations.> Team to have time blocked for the coordination meetings.> All artifacts will be visible across teams.> All teams to deliver work in two-week iterations, with a common start and end dates.> Architecture owners and product owners to have regular scheduled meetings to share information about work details, priorities, dependencies, and issues.> After every four iterations one week of transition would be planned to deploy the combined work of all the teams.> Teams gather at a common location for two days of release planning at the beginning of each release.

The requirements of the application should be captured via a collection of epics and user stories in JIRA. Stories have to be prioritized by the PO with advice from the Architecture Owner about technical risk considerations.

Scenario B: Geographically and Organizationally Distributed Team

Consider an organization with geographically and organizationally distributed team with considerable time-zone difference.

Table 5. Geographically and organizationally distributed team

Scaling Factor	Situation Faced
Team Size	12 total team members available for this initiative
Geographic Distribution	Team members are distributed having considerable time-zone difference
Compliance	Internal compliance only
Organizational Distribution	Business stakeholders are not willing to relocate to the team work areas
Domain Complexity	Complex
Technical Complexity	Complex

Table 6. Process decision for each of the key goals (geographically and organizationally distributed team)

DAD Goal	Strategy
Explore initial scope	<ul style="list-style-type: none"> > Chief product owner to facilitate the creation of vision. > Proxy PO, the team lead, and two developers fly to client's location for a two-week workshop. This workshop would be performed in a large dedicated modeling room. > The requirements of the application captured via a collection of epics and user stories in JIRA. Stories to be prioritized by the PO with advice from the Architecture Owner about technical risk considerations.



DAD Goal	Strategy
Align with Enterprise Direction	<ul style="list-style-type: none">> Architecture Owners outline standard guidelines and agree on mandatory guidelines.> For the collaboration, the Enterprise Architecture (EA) and Product Management teams decide to formalize regular coordination meetings.
Identify Initial Technical Strategy	<ul style="list-style-type: none">> Architectural owners and developers to work closely.> Most of the modeling are performed via whiteboards and Visio.
Move Closer to Deployable Release	<ul style="list-style-type: none">> The team can take a developer-level test-driven development (TDD) approach to programming.> The continuous integration (CI) strategy includes code analysis tools that provide metrics.> Deployment planning is discussed and agreed by the team leads. The application is deployed at the end of iteration into a testing environment where it can get tested.> Create deliverable documentation and deliver it to testers so that testers can start their testing.
Coordinate Activities	<ul style="list-style-type: none">> Each team to hold its daily coordination meetings .> Project dashboard can display quality and progress metrics.> The team can have sixty-minute teleconference every day with the proxy PO first thing in the morning to answer any requirement related questions.> Every four months the PO and Team Leads meet for a week to do long range release planning and general coordination between the locations.



Requirements are captured with informal modeling sessions in their team room and interviews.

Scenario C: Medium-Sized Team, Domain Complexity, and Regulatory

This is a software development team of eighteen people working out of a single location. Each person has their cubicle, and there is also a permanently reserved team room where they've installed whiteboards and a corkboard.

Table 7. Medium-sized team

Scaling Factor	Situation Faced
Team Size	18 total team members available for this initiative
Geographic Distribution	Team working from single location
Compliance	Internal compliance only
Domain Complexity	Medium complex
Technical Complexity	Medium complex

Table 8. Process decision for each of the key goals (medium-sized team)

DAD Goal	Strategy
Explore Initial Scope	<ul style="list-style-type: none"> > They team to create use-cases and screen mockups for the primary screens and whiteboard sketches of the overall business process. > Requirements are captured with informal modeling sessions in their team room and interviews. > The requirements are captured in JIRA. > The team chose a lean work item pool strategy to manage work items using a JIRA virtual Kanban board.



DAD Goal	Strategy
Align with Enterprise Direction	<ul style="list-style-type: none">> Architecture owners outline standard guidelines and agree on mandatory guidelines.> For the collaboration, the Enterprise Architecture (EA) and Product Management teams decide to formalize regular coordination meetings.
Identify Initial Technical Strategy	<ul style="list-style-type: none">> The team takes an informal approach to modeling, as they did with their scoping efforts.> Initial requirements and architecture modeling takes about one month for the first.> Team to come up with the analysis if we have scope to reuse the existing system or build from scratch.> The architectural decisions are captured using a Wiki with the reasoning.
Move Closer to Deployable Release	<ul style="list-style-type: none">> The software development team to adopt a deployment cadence that reflects the deployment schedules of the devices.> Team to embrace a strict configuration management (CM) strategy.> Informal code reviews done as needed.> The technical writer works with developers to help them write the supporting documentation for the software.
Coordinate Activities	<ul style="list-style-type: none">> The team meets for ten minutes on a daily basis around their JIRA Kanban board.> Everyone is expected to update JIRA before the coordination meeting.> Architecture owner to meet with the architects and the teams on a weekly basis to coordinate technical issues.> Team leads meet once weekly to discuss any issues. The PO meets with her counterparts on a daily basis to coordinate requirements-related issues.

Scaling across your organization requires you to help individuals, teams, and departments to adopt an active mindset and agile ways of working together.

Parting Thoughts

This paper started by describing lines of thought when it comes to agility at scale: first, how to scale agile delivery, challenges, comparison of various frameworks and recommendation - the focus of this paper, and the use case scenarios. Scaling across your organization requires you to help individuals, teams, and departments to adopt an active mindset and agile ways of working together. The suggestion is first to scale agile delivery before you can think about scaling Agile across the organization. Only then can your organization operate as an agile enterprise.

References

Ambler, S.W. and Lines, M. (2012). *Full Agile Delivery Lifecycles*. <http://disciplinedagiledelivery.com/lifecycle>

Ambler, S.W. and Lines, M. (2013). *Coordinating Activities on Agile Delivery Teams*. <http://disciplinedagiledelivery.com/2013/07/12/coordinating-activities/>

Ambler, S.W. (2013). *The Software Development Context Framework*. <http://disciplinedagiledelivery.com/2013/03/15/sdcf/>

Ambler, S.W. & Lines, M. (2013). *Going Beyond Scrum: Disciplined Agile Delivery*. <http://disciplinedagileconsortium.org/Resources/Documents/BeyondScrum.pdf>

Ambler, S.W. & Lines, M. (2014). *Scaling Agile Software development*. <http://disciplinedagileconsortium.org/Resources/Documents/ScalingAgileSoftwareDevelopment.pdf>

Scaled Agile Framework (SAFe) home page. <http://scaledagileframework.com/>



About the Author

Akhila Pant is working as a Project Manager at Tavant Technologies. She has been working on Agile project management for over six years. This white paper has evolved based on her extensive experience executing scaled Agile projects across various companies.

About Tavant

Headquartered in Santa Clara, California, [Tavant](#) is a digital products and platforms company that provides impactful results to its customers across North America, Europe, and Asia-Pacific. Founded in 2000, the company employs over 2500 people and is a recognized top employer. Tavant is creating an AI-powered intelligent enterprise by reimagining customer experiences, driving operational efficiencies, and improving collaboration.



3965 Freedom Circle, Suite 750, Santa Clara CA 95054, United States

Tel: +1-866-9-TAVANT | Fax: +1-408-519-5401 | hello@tavant.com

Santa Clara | Dallas | New Jersey | London | Bangalore | Hyderabad | Noida |
Sydney | Tokyo | Colombia

