Don't you dare play it SAFe

Sub-title: Beware heavyweight software development frameworks that kowtow to status quo leadership practices

I remember my first introduction to the Rational Unified Process (RUP) as if it was yesterday. I had a tremendous amount of experience with a wide variety of traditional Waterfall methods. Each promised to solve nearly every challenge associated with software projects and each failed to do so. So I was eager to try something new with RUP and was hopeful that it would help improve my software project success.

And as part of RUP I'm including the Unified Modeling Language (UML) and Use Cases as the defacto requirement artifact. There were a lot of new approaches within the RUP. It was sold as a 'framework'; one with lots and lots of expert guidance. I think one of the overwhelming or inherent assumptions in RUP was that we (the customer of the process) lacked the ability to define an effective process.

So, RUP solved that problem. It provided templates, guidelines, phases, techniques, iterations, gates, hand-offs, planning guidance, testing guidance, models, workflow advice, etc. Literally thousands of pages of guidance for how to define requirements, plan, construct, test, manage, and deploy a software project. It was an amazing amount of "stuff".

But interestingly, RUP did very little at the team level. It put a tremendous amount of information "out there" for the organization to leverage in controlling software development. But then it assumed that the execution was relatively straightforward. I.e., we've provided all of this sound expert advice, now the simple part is just following our guidance and you (the developers, the project) will naturally receive "great results".

It always seemed to me that it was an iterative variant of Waterfall with some new requirement and modeling tools thrown in. Not bad really, but in my experience it didn't result in successful projects any more than Waterfall did. And it didn't seem to last that long either.

Agility and the Scrum Framework

Scrum is characterized as a *framework too*[1]. But it's much, much lighter weight than RUP was. The Scrum Guide, one of the definitions of Scrum, is a mere 16 pages long. I can deliver a well-rounded introduction to Scrum in about 90 minutes. I can get a team running their first sprint in about a day; the key requirement being whether they have a clear backlog of work to begin sprinting.

When I think of Scrum I think of a generic set of guidance (roles, artifacts, ceremonies, and practices) that surrounds how we build software. Yes, there are some rules and recommendations, but they are essentially in place to create and foster an environment of collaborative software development. Here are the 5 Core Scrum Values from the Agile Atlas:

1. Focus
2. Courage
3. Openness
4. Commitment
5. Respect

The Agile Atlas defines "Core Scrum" succinctly and is probably even shorter than the Scrum Guide.

The other reaction I have to Scrum is that it's not "for" the leadership or organization. What I mean by that is that the prime focus is team-ward first and organizational second. Let me use a concrete example. There are no PMO project status reports dictated by Scrum. No organizational level data gathering. No stop light (green-yellow-red) reports.

If a project manager or functional manager wants to know how a project is going, they need to engage the team. Attend the daily stand-up and the sprint review and find out what's going on, what the team has delivered, and what part they can play in helping the team be more successful. Asking themselves – how can they "serve" the team over how the team can "serve" them?

The Scrum framework provides "just enough" guidance and then asks or requires the team and organization to "fill in the blanks" by applying a customer value and lean mindset.

It's simple. It's clear. And it fosters collaboration around providing valuable "stuff" to our customers. Yes, I'm a bit biased towards agility and Scrum. But I've found that having a very lightweight framework and then engaging the team in extending it for their context and situation and domain and customers is a much better way to be successful.

Not let's shift the focus towards another, perhaps a bit beefier framework.

[1] In all the framework frenzy, we forget that Scrum is a framework as well—one that can be perfectly adequate to scale in organizations with reasonable and simple 'extensions' like the Agile Release Training and Scrum of Scrums. And that extension remains simple and truer to agile principles.

The Scaled Agile Framework – SAFe

The other evening I attended a talk at the ALN – Raleigh group presented by Dean Leffingwell. It was sponsored by Rally Software and held at their Raleigh office. Try saying the Rally Raleigh office three times quickly…but I digress.

I'd been reading about and studying Dean's Scaled Agile Framework or SAFe for quite some time. It's becoming a popular topic in the agile community as it slowly gains traction. Rally is a strong supporter of it. Part of that is their longer term relationship with Dean as an agile thought-leader and advisor to their company. Another part of it is clearly their products' evolution towards a 3-tiered Agile *ALM*[2] model and the alignment between their tools implementation and the SAFe framework.

Dean mentioned in the talk that an early adopter of SAFe was John Deere and that he and Rally deployed much of SAFe in the coaching and tooling implementation at Deere; so there is a bit of mutual business interest driving their collaboration as well.

Anyway, Dean was kind enough to go through SAFe in a talk he entitled: *Know the Way, Show the Way, Go the Way: Scaling Agile Development*. While the title sounds quite generic, the session was focused entirely on the Scaled Agile Framework.

What I like about SAFe

There seem to be quite a few 3-tiered models cropping up that seem to be useful in adopting agile at the Enterprise or larger-scale levels. The tiers often have Kanban practices in the upper two tiers and Scrum at the team or execution tier. That generative model seems to work quite nicely in funneling work from high level product roadmaps, through early feature validation, down towards team(s) implementing and delivering valuable releases for customers.

Mike Cottmeyer has been discussing these sorts of models, probably for longer than Dean has been discussing SAFe. There are a few connections to Mikes work in the references. And Alan Shalloway seems to have hitched his wagon to Dean and SAFe too for enterprise-level adoptions; and he's talked around similar models with an emphasis on Lean at the Enterprise in his NetObjectives webinars.

Point is – this sort of structure and funneling of work through a 3-tiered model can be a constructive way to attack agile within the Agile Enterprise. SAFe has to-date just done a better job of documenting and marketing it than any other *model*[3].

Another useful aspect is simply the terminology. For example, Dean introduced the notion of an Agile Release Train in his *two books*[4]. I've found that model to be an incredibly useful concept for deploying Scrum and other methods at-scale in organizations. It extends the basic, deliver at the end of each sprint model, for contexts where that is impossible for a variety of reasons. It introduces concepts like: sprint synchronization, hardening or stabilization sprints, and fosters release-level planning.

All of which I think are missing in many organizations implementing agility at-scale.

---

[2] Application Lifecycle Management
[3] See my note about DAD in the references
[4] Scaling Software Agility published in 2007 and Agile Software Requirements, published in 2011

So the terminology, 3-tiered model and even some of the smaller models within each layer are quite relevant and helpful in at-scale agile instances. So if everything is good, where's my concern?

But more importantly, what frightens me about SAFe?

In all of my research around SAFe and in the presentation the other evening, I get the overall impression that it's an exhaustive and detailed framework. It also seems to have a primary audience. It seems to be primarily "for":

- the Senior Leaders; Stakeholders
- the PMO; the Project Managers
- the Accountants
- the Managers
- the Architects and upfront designers
- the planners
- the formal process designers, ex: CMMi, ITIL
- the regulators; the quality checkers

The intent seems to be to wrap up this "agile team stuff" in a way that can be easily understood. In a way that can be managed, reported, and measured the way things have traditionally been managed, reported, and measured. In other words, SAFe seems to be a framework map that fosters very little organizational level change when moving to agile methods.

I can see why the above listed folks are so excited about SAFe...particularly in at-scale scenarios. But I worry that something is getting lost in SAFe; something important, something critical to the success of agility. What might that be you ask?

The People
The Creativity, Adaptability, Innovation, Empowerment, Engagement, Courage, Transparency, and the Energy
Surrounding the Team

It's about the Teams…stupid!

First, there seems to be a de-emphasis on the team and an emphasis on the organization. I'll explain it by sharing a story Dean told in our session.

He was referencing a 10 team instance of agile. The teams worked hard and scheduled ten sprint reviews at the end of their efforts. Stakeholders and managers dutifully attend the first round of ten meetings. Then they go away.

He asked – how many of them will return for the next of sprint reviews? The implication was that they clearly couldn't spend their valuable time iteratively reviewing all ten teams work. He implied that in the "real world" these folks simply didn't have the time for it and it wasn't sustainable.

Rather, the teams needed to roll up their results into a singular demonstration event much closer to or right before the release. Call it a release review. This way, the time of the stakeholders and leaders would be effectively honored. And the team would get the benefit of demonstrating more mature software and gaining broader feedback and visibility.

I say **phooey** to this notion or mentality that the leader's time is most valuable; that the team inherently needs to "serve" the leader. If the team is working on the highest business priority work on the planet, the premise of all agile teams, then as a leader I need to get out of my chair and go engage with the teams. I should passionately care about what they're doing, how they're doing it, and what it looks like.

Getting to a sprint review, reviewing the work we're delivering for our customers, should be my prime directive.

Now certainly, at-scale, the teams could and should consolidate their results so that the review is crisp, integrated, relevant, and time sensitive. BUT, organizational leaders should view it as a responsibility to attend, engage and learn.

The exception I take from Dean's story is that it clearly supported the mindset that the team was serving the leadership. And that their time was more valuable focused elsewhere. Yes, their time is valuable, but there is nothing more valuable than them providing vision, interaction, engagement, and INTEREST in their teams' efforts to delight their customers.

It seems to me that Dean and SAFe are potentially reinforcing traditional, command-and-control, hierarchical management and leadership. Or at least that's my overall impression AND my primary concern.

Sidebar

I thought it might make sense to provide an example of the level or prescriptiveness that SAFe provides at the lowest possible levels. Under the PSI/Release Objectives Abstract advice, SAFe recommends the use of "Stretch Objectives".

To the best of my knowledge, nobody in the agile community is recommending this notion. Now in my coaching, I personally like mentioning the notion of "stretching" in meeting sprint goals. Heck, I even talk about planning sprint stories as part of sprint planning. However, that is situational in nature and under my direct coaching guidance – so that teams understand the nuance and don't over commit to too much work.

However, I think it quite dangerous for SAFe to make this recommendation. It's not that it's inherently bad. It's just that there is an incredible amount of room for organizations to misinterpret and wrongly implement this sort of advice. In this example, requiring agile teams to "stretch" under all circumstances because SAFe supports the notion.

It begs the question as to whether Safe is truly "safe" without expert and situational coaching at a team, program, and organizational level?

I for one don't think so…

Wrapping Up

You may be asking yourself why I brought RUP up in the beginning of the article. First, it is a useful anti-pattern example for heavier weight frameworks. But that wasn't the primary driver. I see many similarities between RUP and SAFe. It actually starts at the roots of them both. Dean was an influential figure within the historical RUP community. At one point he created a company that developed a heavyweight requirements management tool – Requisite Pro. That company was then acquired by Rational; yes, the "Rational" in the Rational Unified Process. From 1997 – 2000 he was a Sr. VP at Rational Software Corporation; right up to and post the acquisition by IBM.

So, I think he has a tendency to think in terms of these all-encompassing, detailed, prescriptive processes. That's not necessarily bad. But you have to ask yourself is SAFe driven from a true understanding of agile principles and practices upward OR is it a modification of RUP thinking *over top of* the agile methods. I actually think it's the latter. Dressed up with a lot of lean terminology to make things confusing and perhaps more palatable.

Conversely, I think Mike Cottmeyer came to a 3-tiered model from the bottom up. That is, having a full empathy for and understanding of the team, before connecting agility to the organization by developing aspects of a 3-tiered framework.

So, is SAFe inherently bad? No, I believe it has the potential to be an effective model. But what I want to see in the forefront of SAFe's focus are these tenants:

1. A focus on the agile team
2. A focus on organizational transformation
3. A focus on leadership transformation
4. Less prescription of detailed practices and much lighter weight guidance, which honors context-based practices
5. An acknowledgement that it's about the organization serving the team rather than the team serving the organization

Oh and BTW, let's stop the certification insanity that surrounds SAFe. If Dean is going to put it into the public domain, then do so. But don't then turn around and wrap a 3-tiered certification model around it to create control of the implementations and drive a business model. I don't think you can have it both ways.

As an aside, I wish more folks would adopt the example that XP provided. While it's one of the agile methods, there's never been much prescriptiveness around the definition nor efforts to control it by certification. I admire Kent and the XP community from this perspective.

So as the sergeant in one of my all-time favorite TV shows (Hill Street Blues) often said: Let's be careful out there…in our use of SAFe!

Thanks for listening,
Bob.

References

1. Here are several of Mike Cottmeyer presentations surrounding at-scale recommendations for agile organizations:
    a. http://www.slideshare.net/mcottmeyer/pmi-atlanta-agile-lig-enterprise-agile
    b. http://www.slideshare.net/mcottmeyer/exploring-agile-transformation-and-scaling-patterns
    c. http://www.slideshare.net/mcottmeyer/scrum-and-kanban-in-the-enterprise-webinar
2. Disciplined Agile Delivery (DAD) is another SAFe-like framework being put forth by Scott Ambler. It's somewhat less prescriptive than SAFe and worth a look here: http://disciplinedagiledelivery.com/
3. While I didn't mention them in the article, Bas Vodde and Craig Larman have written two books that are very much aligned with many of the SAFe and DAD oriented principles. They focused on Agile at-scale. More info here: http://www.craiglarman.com
4. And another book jumping on the SAFe, more robust framework bandwagon: http://scaledagileframework.com/new-agile-book-with-safe-valpak-case-study/