CUTTER Business Technology Journal

Management, Innovation, Transformation

Vol. 33, No. 4, 2020

Disrupting Agile

Is Agile Ready?

Hillel Glazer
Guest Editor

CUTTER Business Technology Journal

As business models for creating value continue to shift, new business strategies are constantly emerging and digital innovation has become an ongoing imperative. *Cutter Business Technology Journal* delivers a comprehensive treatment of these strategies to help your organization address and capitalize on the opportunities of this digital age.

Cutter Business Technology Journal is unlike academic journals. Each monthly issue, led by an expert Guest Editor, includes five to seven substantial articles, case studies, research findings, and/or experience-based opinion pieces that provide innovative ideas and solutions to the challenges business technology professionals face right now — and prepares them for those they might face tomorrow. *Cutter Business Technology Journal* doesn't water down or delay its content with lengthy peer reviews. Written by internationally known thought leaders, academics, and practitioners — you can be certain you're getting the uncensored perspectives of global experts.

You'll benefit from strategic insight on how the latest movements in digital innovation and transformation, artificial intelligence/machine learning, Internet of Things, blockchain, analytics, and cloud, to name a few, are changing the business landscape for both new and established organizations and how cutting-edge approaches in technology leadership, enterprise agility, software engineering, and business architecture can help your organization optimize its performance and transition to these new business models.

As a subscriber, you'll also receive the *Cutter Business Technology Advisor* — a weekly bulletin featuring industry updates delivered straight to your inbox. Armed with expert insight, data, and advice, you'll be able to leverage the latest business management thinking to achieve your organization's goals.

No other journal brings together so many thought leaders or lets them speak so bluntly — bringing you frank, honest accounts of what works, what doesn't, and why. Subscribers have even referred to *Cutter Business Technology Journal* as a consultancy in print and likened each month's issue to the impassioned discussions they participate in at the end of a day at a conference!

Get the best in thought leadership and keep pace with the technologies and business models that will give you a competitive edge — subscribe to *Cutter Business Technology Journal* today!

Founding Editor: Ed Yourdon
Publisher: Karen Fine Coburn
Group Publisher: Christine Generali
Managing Editor: Cindy Swain
Copy Editors: Jennifer Flaxman, Tara K. Meads
Production Editor: Linda Dias
Client Services: service@cutter.com

***Cutter Business Technology Journal®** is published monthly by Cutter Consortium, an Arthur D. Little company, 37 Broadway, Suite 1, Arlington, MA 02474-5552, USA • Tel: +1 781 648 8700 • Fax: +1 781 648 8707 • Email: cbtjeditorial@cutter.com • Website: www.cutter.com • Twitter: @cuttertweets • Facebook: Cutter Consortium. ISSN: 2475-3718 (print); 2475-3742 (online).*

Subscription rates are US $485 a year in North America, US $585 elsewhere, payable to Cutter Consortium. Reprints, bulk purchases, past issues, and multiple subscription and site license rates are available on request.

☐ Start my print subscription to *Cutter Business Technology Journal* ($485/year; US $585 outside North America).

Name _____ Title _____

Company Address _____

City _____ State/Province _____ ZIP/Postal Code _____

Email (Be sure to include for weekly *Cutter Business Technology Advisor*) _____

Fax to +1 781 648 8707, call +1 781 648 8700, or send email to service@cutter.com.
Mail to Cutter Consortium, 37 Broadway, Suite 1, Arlington, MA 02474-5552, USA.

Request Online License Subscription Rates

For subscription rates for online licenses, email or call: **sales@cutter.com** or **+1 781 648 8700.**

CUTTER CONSORTIUM
Access to the Experts

Opening Statement

by Hillel Glazer, Guest Editor

The initial expression of Agile — which, for discussion purposes here, we can tie to the publication of the "Manifesto for Agile Software Development"[1] in 2001 — is arguably among the most disruptive events to affect the software world since the industrialization of software itself. What made it so disruptive? What were the conditions that allowed these ideas to be disruptive?

Let's suppose that effective disruption happens at the intersection of insufficient status quo and redefining fundamentals. Or in plain English, the current state isn't getting the job done, and it turns out that what we once thought was true (or good) isn't.

In software's case, by the 1980s the "productivity" of the overall software industry was abysmal. Costs were skyrocketing, quality was plummeting, and morale among software professionals was desperate. In that decade, a broad-scale study of software — later confirmed in The Standish Group's 1994 seminal work, "The CHAOS Report"[2] — found that the management and engineering of software indeed underperformed in terms of the needs of the market.

In response — and all with good intentions based on tried-and-true theories, best practices, and know-how — the market filled up with management methodologies, tools, and standards. Backed by defined fundamentals and validated by the US Department of Defense (at the time, the single biggest software customer by volume of code, not to mention cost), research universities, and many of the biggest names in software, these "solutions" codified a status quo.

Against this backdrop, the contributors to the 'Manifesto were each working (on their own, or in small cohorts) on ways of developing software and managing the software process. To some degree, each independent of the other came to a similar realization. "They" (as in, those just mentioned) had gotten the fundamentals wrong and the status quo was making things worse. The software product development industry needed a disruption and the writers of the 'Manifesto did just that. With — grammatically — three sentences.

In practical terms, three sentences with four values embedded within them weren't entirely sufficient to change the world. Evolving the development universe would require some help. The contributors to the 'Manifesto offered pragmatic practices to put the transition to a higher state of consciousness within reach. Some practices were more formalized than others, some more commercialized than others, and some came with a hefty warning label.

While their respective focuses were different, each one in its own way emulated the values in the original manifesto. Despite easily visible limitations and constraints, successes were swift and impressive. By all accounts, the manifesto got it right!

The introduction of Agile practices changed how product development was organized, managed, and executed with the objectives of enhancing customer satisfaction, improving customer/market responsiveness and developer morale, and cutting out impediments to delivery. Indeed, "Agile" has been clearly popularized in much of the product development world. But strong and steady signals are emerging that those objectives are not being met. Ugly, smelly things are crawling ashore covered in a sheen of vaguely Agile-esque meconium. What happened?

Proponents can justifiably counter that organizations not achieving the promised results of using Agile "are not doing it right" or "don't have the environmental fundamentals" to make it work. But what if the very characteristics that made Agile successful are no longer enough? Have the underlying assumptions about team size and organization, decision-making authority, product arrangement, and customer involvement run their course?

Moreover, what's needed to ensure that software teams can apply Agile practices in nonideal contexts, such as in larger and highly regulated organizations, and deliver results? Do the fundamentals introduced by Agile now need fundamental changes? What would it take to disrupt Agile?

It's the year 2020. The popularized notion of "Agile development" has now been around for the better part of at least two decades. That's a million astronomical units in Internet dog years and, to no one's surprise, the concept of Agile has sprung new trunks, branches, twigs, and leaves and is now an entire genus, sometimes making the proto-Agile ideas unrecognizable.

These new flavors, colors, and shapes can be described as evolutions of the original concepts. Or can they? Are they really evolutions of the concepts, or evolutions of one particularly dominant branch of the Agile genus? I argue that what's been evolving over these last couple of decades are not the concepts, but merely the practices. The concepts really haven't changed. Indeed, I would argue the concepts are solid. Fairly indestructible, actually.

Practices, however, are context-specific instantiations of the application of concepts. The "Manifesto for Agile Software Development" is about *values*. Not practices. I argue that too much (if not all) of the new trunks, branches, twigs, and leaves and the entire genus have been evolutions of *practices*. And, it seems Darwin has spoken: this isn't working.

This isn't a new thought, but lately I'd been pondering the following question: "Could popularized Agile be gently coaxed back into paying more attention to the fundamental aspects of Agile before 'Agile' became myopically focused on practices?" The unfortunate answer to that question, I concluded, is "no."

Well, if the answer is no, then what will it take to make good on the promise of Agile to solve broad issues in the software industry as the 'Manifesto initially set out to accomplish? If we can't ease the development world back toward its value-based roots, how can we make that happen? Clearly, making good on that promise needs something disruptive.

In my work and teaching, I'm noticing a number of trends. Among them are attempts to look "beyond Agile." Some are examining other disciplines and concepts such as psychology, society, and diversity, while other vectors point to product definition, budgeting, resource allocation, and priorities. And people are also looking in the opposite direction — a more granular, DNA-level working to "rebuild" the underpinnings of Agile from even more basic roots.

I'm not alone. Using math, economics, physics, detailed statistics, various manifestations of holistic engagement, or merely appealing to Maslow's hierarchy of needs and similar enlightened states of awareness, many of these other trajectories point to the same target. They indicate something along the lines of "practices aren't enough."

One of the problems addressed by the authors of the 'Manifesto was that of a fixation on processes. And by creating a shell of practices around the embryo of Agile, it seems they inadvertently let a pandemic escape from the lab that would eventually act to harden this shell. Essentially, the very practices that were supposed to usher in a new appreciation for people, product, relationships, and responsiveness ended up with the unintended consequence of shrouding the best parts of Agile so that their brave new species would continue to fight its way out. It turns out that the practices catered to the same non-systems thinking that the 'Manifesto was meant to smash through.

Which brings us to this issue of the *Cutter Business Technology Journal* (*CBTJ*). The status quo focus on practices is proving to be insufficient. If something drastic doesn't push Agile off its current path, evolution doesn't predict it will return to its initial promise on its own. I ran this idea by Cutter Consortium Senior Consultant and past *CBTJ* Guest Editor Alistair Cockburn, who said that he has long resisted the questions, "How do we disrupt Agile?" and "What comes after Agile?" He feels that these questions suggest that the manifesto was pointed in the wrong direction, when it seems to have stood well against the test of time. After considering how I framed the discussion, Cockburn said he liked it; it caused him to reconsider what "disruptive" might mean, how to detect it, and where to look for it. He told me he was happy to have his thoughts sent down a new direction.

This *CBTJ* issue takes us on an evolutionary journey of our own. In a rather normalized bell curve, we start

Upcoming Topics

Information Trustworthiness/Security
Claude Baudoin

Beyond Automation: AI, ML, RPA
San Murugesan

How Technology and Business Leaders Help in a Crisis
Steve Andriole

with fundamentals and progress through more advanced concepts. We then ease back with practical steps forward and wrap up with a cautionary send-off. The good news is that you're free to take the journey, or wait for the asteroid. Your call.

In This Issue

We start with a cohort of IBMers — Matt Ganis, Michael Ackerbauer, and Nicholas Cariello — who tee up our discussion directly from where our "What will it take?" question leaves off. They look at the challenges and missteps associated with Agile, beginning with adoption, which relies on expectation setting. And there's no expectation setting without education. Can it be that simple? Occam's razor says, "Probably."

Next, we move up the evolutionary scale with Eric Willeke's look at whether we've missed a turn somewhere on the path. Perhaps we need to gene-splice some deliberate characteristics into our next incarnation of Agile. Forget whether we're picking the right approach: Are we asking the right questions? Are we even asking questions? Do we know what we want to be? Are we even Agile for the right reasons?

Cutter Consortium Senior Consultant Masa K. Maeda then schools us in some hard-hitting, data-driven food for (evolutionary) thought. He helps us understand what we should be looking for when considering an agility path. It's no simple checklist or algorithm. Maeda's outline makes us take a holistic look at our environment; at our choice of primordial soup, as it were. The good news is that we are not completely afloat in the flotsam of the universe. We can choose how we evolve.

Our penultimate piece comes to us from Bob Galen, who picks up on our evolution theme and goes back to basics. When pursuing Agile, which comes first: the chicken or the egg? Clearly not making breakfast, Galen takes aim at whether teams or leadership "goes Agile" first. He gives us a taste for what it must look like to have teams come first and what seasonings to pepper leadership with so that leadership and teams can be "Agile-*y*" effective together.

Finally, Jeff Doolittle leaves us to set out on our own path to disruption. He suggests the most drastically disruptive action: don't do Agile. At the very least, don't do Agile the way too many others are doing Agile. Doolittle invokes the same line of thinking that started our thought experiment to begin with — what has Agile become? Has it grown in unintended ways? Have we lost what it is supposed to be? What else is there if not Agile? Should we completely abandon Agile? Wouldn't that be disruptive!

As we work through this issue of *CBTJ*, several themes coalesce. There does appear to be consensus among our contributors that practices are a stumbling block. To disrupt Agile, we need focusing mechanisms around a number of non-practice principles. We hope you see what we're seeing and are able to leverage this third-party validation to help get your Agile journey back on track or, even better, avoid being off track. Here's a thought: maybe Agile needs practices around not using practices. No, on second thought, let's not go there.

References

[1]Beck, Kent, et al. "Manifesto for Agile Software Development." Agilemanifesto.org, 2001.

[2]"The CHAOS Report (1994)." The Standish Group International, 1995. (I know for a fact there was such a study in the 1980s but have not been able to find, or remember, the source.)

Hillel Glazer is a Senior Consultant with Cutter Consortium's Business Agility & Software Engineering Excellence practice and CEO of Entinex, Inc. He is a career-long pathfinder who has been reframing how organizations perform. Mr. Glazer counsels executives in the technical, organizational, and operational integrations necessary to bring about the capability to dynamically respond to shifting demands. He is the consummate pragmatic systems thinker, combining his deep technical roots with broad business experience and expertise in operations, management, and organizational change. Mr. Glazer currently focuses on enterprise transformations, where the desired outcome is predictable results. In today's dynamic world, this frequently involves digital transformations to take advantage of the ever-increasing availability of, and demand for, data for real-time decision making. Having written the first full-length, peer-reviewed article on the intersection of Agile and CMM in 2001, he has become the foremost authority on blending Lean, Agile, and modern development techniques with CMMI, ISO, systems engineering, and other worldwide standards. Mr. Glazer is three-time Chair of the Lean Kanban North America *conference series, author of* High Performance Operations: Leverage Compliance to Lower Costs, Increase Profits, and Gain Competitive Advantage, *and spent six years as a Visiting Scientist with Carnegie Mellon University. When not working on transforming large enterprises, Mr. Glazer is a startup coach in Baltimore City's Emerging Technology Centers tech incubator. He holds a bachelor of science degree in aerospace engineering and a master of science degree in technology management. Mr. Glazer has guest lectured on entrepreneurship at Stevenson University and teaches a graduate and professional certification course on product development management at the University of Maryland, Baltimore County. He can be reached at hglazer@cutter.com.*

IN TEAMS WE TRUST

The Speed of Trust: Why Some Agile Teams Succeed and Others Do Not

by Matt Ganis, Michael Ackerbauer, and Nicholas Cariello

There are a variety of circumstances that can cause Agile methodologies to struggle to gain a firm foothold in many organizations. We believe the most prevalent issue surrounding adoption (or lack thereof) is the misunderstanding of many Agile practices and techniques, a key component of which is the lack of understanding of the underlying reason *why* it is necessary to perform certain practices.

Several other problems tend to arise due to downward pressure from leadership looking for a "status" or "quicker" delivery. Agile is frequently sold to leadership as a cheaper and faster method of accomplishing tasks, when the reality is often the opposite. In our experience, an Agile practice can do more harm than good when it results in a misunderstanding, specifically around cost or delivery. This misunderstanding ultimately creates friction and, more importantly, a lack of trust between leadership and Agile teams.

In practice, we have observed that Agile teams can misunderstand some practices they have been taught, resulting in an inefficient or even incorrect use of these methods. We believe teams need to ensure they have a consistent understanding of their Agile practices in order to become more effective. This understanding across a team reinforces *why* a practice is structured the way it is, and even why it is considered to be part of an overarching methodology.

As teams become more consistent in how they deliver successful releases or minimum viable products (MVPs), we believe they will begin to build deeper trust with their leadership teams. Then, as the "window of trust" between development teams and leadership grows larger, teams will become more amenable to experimentation and will be more likely to innovate while continuing to hone their understanding and implementation of Agile practices. Moreover, with deeper trust, leadership will feel more comfortable allowing teams to run with their Agile process and thus be more willing to accept an incremental release of an MVP with the confidence that the "best is yet to come."

In this article, we address an issue that we call the "circular trust problem," where teams need to establish trust with their leadership before leadership grants them more trust. To reach that state of trust, we believe there needs to be a better *understanding* of Agile practices. This understanding is neither about *what* to do, nor about *how* to do it; rather, it is about truly understanding *why* Agile is done.

The Agile Story

Most people who have been working with Agile teams know the story of how the "Manifesto for Agile Software Development" was conceived and documented. In a nutshell, back in 2001, several seminal players in the software industry met in the US town of Snowbird, Utah, to craft a public declaration of how they believed software should be crafted. This declaration significantly altered the way many have approached software development and is the foundation for all variants of Agile software development. The ideals of the manifesto are relatively simple, and certainly by now everyone in software development has read these words:

> We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
>
> **Individuals and interactions** over processes and tools
> **Working software** over comprehensive documentation
> **Customer collaboration** over contract negotiation
> **Responding to change** over following a plan
>
> That is, while there is value in the items on the right, we value the items on the left more.[1]

These ideals are meant to shape how software delivery teams work. The manifesto was based on 12 principles that assist in creating an environment of collaborative productivity. These principles are meant to be a "guiding light" for teams as they implement and execute with agility. To provide some structure, several Agile

methods (Scrum, XP, Lean, etc.) have gone on to define practices or techniques that emphasize these ideals.

Why would an organization decide to choose Agile methods over a traditional methodology? Some organizations may adopt an Agile method to help their speed to market while attempting to meet ever-changing customer demand, or perhaps they simply want to increase team productivity. To put it simply, organizations want to develop software better, faster, and cheaper.[2]

However, the rapid rise and commercialization of Agile methods has created what some might term a "degradation" of the original philosophy. One of the co-creators of the "Manifesto for Agile Software Development," Arie van Bennekum, has said "[today] I see people being an Agile coach absolutely not knowing what they're talking about," and Jon Kern, another co-creator, has opined, "You get a lot of people, just snake-oil salesmen — folks that say they're doing Agile when it's Agile in name only."[3]

But why would this be so? Organizations are legitimately attempting to better themselves and developers are honestly attempting to hone their craft, so why this disillusionment? Perhaps, as we believe, it is the media mantra of "Adopt Agile and deliver better software, cheaper and faster" without really understanding *why* some Agile methods prescribe certain steps or actions be taken. Perhaps it is because senior leadership sees Agile as a way to quickly lower bottom lines and, therefore, raise greater revenues. Perhaps it is that, in the rush to educate all our teams about Agile, we employ teachers and coaches who don't fully understand the philosophies that underlie Agile and who pass their misunderstandings on to their students, who then struggle to realize the value and promise of Agile methods.

Adoption Failures

So the overarching question becomes: why do so many teams have issues adopting and internalizing Agile methods?[4] A fair number of published papers discuss the issues teams have in moving to an Agile culture.[5] Many of these works agree that three of the most important factors leading to successful adoption of Agile methods are culture, people, and communication.[6] A supportive culture around a team is paramount to successful implementation of any Agile method. Developers need to be given the freedom to make decisions that will lead to frequent work releases of code and should not be doubted during a project. Along with a level of trust in the development team, Agile environments should foster rapid and frequent communication among team members, stakeholders, and senior leadership.[7] These statements may appear obvious and these are things that all Agile methods must take into account. After all, don't stand-up meetings provide for frequent and rapid communications? They do. But perhaps the type of conversations that are happening aren't really supporting the team's drive to deliver value at regular intervals.

The rapid rise and commercialization of Agile methods has created what some might term a "degradation" of the original philosophy.

For example, one of the strongest detriments to successful Agile adoption is the lack of trust (or perhaps *implied* lack of trust) that leadership inadvertently demonstrates to its teams. Such behavior can become a vicious circle. If a team is attempting to hone its skills while understanding the need to fail fast (i.e., make course corrections, understand their impact, and react accordingly) and leadership is second-guessing the team, or "calling it out" for being late and not delivering, the trust between leadership and the development team begins to erode. As that trust erodes, the team feels less secure in following Agile practices and more compelled just to deliver "something" to show progress. The less trust leadership imparts, the less effective the Agile team becomes. Alternately, the more leadership would/could grant the team leeway to continue becoming more effective at how it practices and gains skills with Agile methods, the more the team, knowing that leadership is showing patience and giving it time, will deliver using iterative methods and thus produce the desired results.

Another leading cause for failure to adopt comes from a misunderstanding of what teams should actually be doing. We know that in order to effectively execute work a team needs to be enabled with knowledge. In the case of adopting Agile, some teams simply lack accurate education about, and therefore knowledge of, Agile. According to a survey published in last year's "13th Annual State of Agile Report," 36% of teams have insufficient training and education, 35% have inconsistent processes and practices across teams,

and 40% have a lack of skills or experience with Agile methods.[8]

Teams are clearly not obtaining the knowledge they need to appropriately implement Agile practices. Why are these teams not getting the education they need? The answer is simple: when teams are making the shift to Agile, they often do so at a very quick pace (perhaps to please their leadership), in the process creating inconsistencies and missing the underlying philosophies of Agile. Instead of the shift to Agile enabling teams to work in an Agile manner and produce better work, they are now limited in their ability to work. This is the point where teams get caught up in the "how" of work, rather than actually producing good work. By rushing straight into a development project, teams are essentially "doing the best with what they've got." The lack of the pivotal component of education leads to frustration and dissatisfaction with Agile.

Leadership bears as much responsibility as teams do when it comes to misunderstanding Agile. When organizations decide to adopt Agile, they are making a commitment — to their clients, teams, and themselves — to change the way they work. However, according to the previously mentioned survey, 52% of teams still believe that there is an organizational culture at odds with Agile values, and 44% think there is inadequate leadership support and sponsorship.[9] With over half of teams confirming that leadership is not playing its part in the Agile adoption that organizations are driving, we can obviously understand why that adoption is failing.

Leadership, like teams, lacks the knowledge necessary to execute Agile. While the leadership may be excited to implement this "new" way of working that is supposedly cheaper, faster, and will produce better work, it is not making the necessary commitment to abide by the practices that would allow Agile to flourish. To enable teams to work in an Agile manner, both leadership and teams need to understand Agile from the top-down.

Agile Is Faster (Well, Sort of)

Usually in a top-down decision to adopt Agile methods, a driving force behind that decision is the "promise" that Agile methods enable teams to deliver faster. But does the use of Agile methods really cause a team to deliver faster? The answer is simple: yes (and no). This brings us back to the point we raised earlier: do teams (and, more importantly, their leadership) understand *why* an Agile project can deliver faster?

Let's look at the Agile process. When a team starts a project, the assumption is that the customers, or those asking for the work to be done, know they want something but are unclear exactly what that "something" is. Said another way, they have an idea, but perhaps are not ready to enumerate all the details (requirements or features) to the team. Working in two-week iterations, the Agile team commits to delivering something that works (working code), which moves the customers closer to their vision. Along the way, the customers are allowed to change the requirements, permitting the team to pivot in a different direction, or to add requirements to the queue as they gain understanding of the increased value those requirements may add. The team happily chooses the stories from its backlog with the highest value and continues working in two-week iterations.

What about speed in delivery? Based on the scenario above, work queues, called "backlogs," will continue to grow as the customers continue to refine, until finally all requirements are met, and the final product is delivered. But that process seems far from being "quicker" than the non-Agile process. And it isn't. An Agile project is bound to take much longer than a traditional development cycle because of all the churn Agile causes. The "speed to market" comes in having the customer see something delivered and declaring that the project, while not having fulfilled all the requirements, is "good enough," and further development on the release ceases.

As a simplistic example, look at Figure 1. Assume a project has 100 features believed to be needed and each feature takes one week to complete. Based on a simple estimate, the project needs 100 weeks to reach completion. But with an Agile method, using iterations and frequent reviews, if at any point in the project a customer is delivered a working version of the deliverable that incorporates only 60 of those features, and the customer declares that the remaining 40 features aren't needed to formally release it, then the project was completed in 60 weeks and, therefore, delivered faster.

So it's not that Agile ways of working are necessarily faster; it's that Agile provides a framework for allowing customers to decide that:

- A certain set of delivered functions is sufficient or better than what they had envisioned

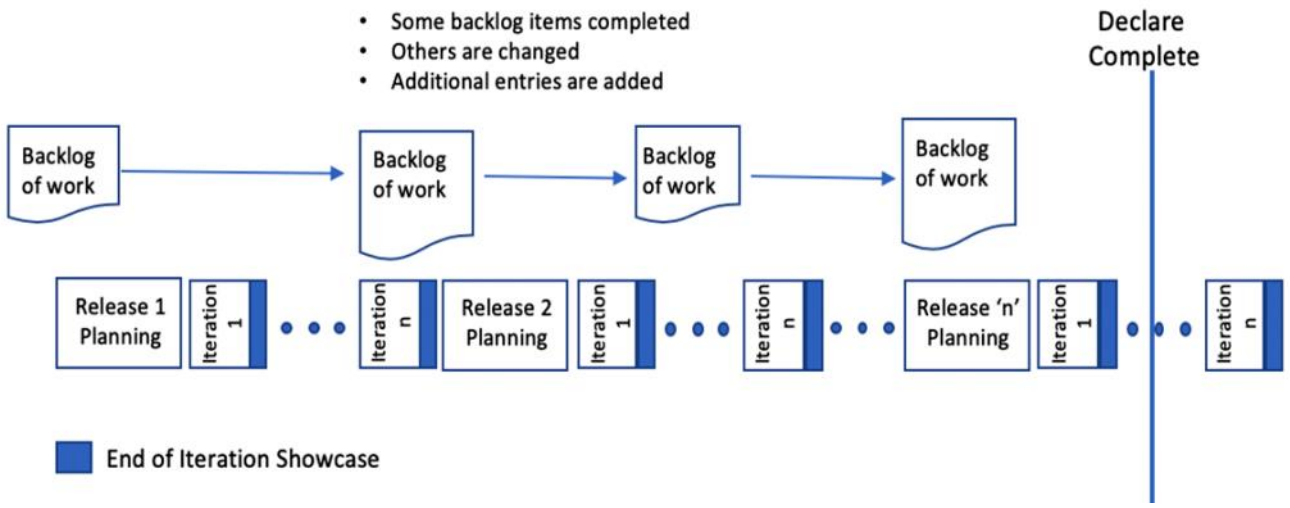

Figure 1 — Agile project example.

- A certain set of not-yet-delivered features aren't really needed

Incremental releases show tangible value that gives customers and stakeholders time to evaluate a working product. Regular releases of value heighten customer expectations for what might be next. The positive psychology of this is that an organization can potentially release enough incremental value over time that customers and leaders do not try to force-fit feature requests into a backlog that is at capacity.

"You Only Get What You Pay for"

Always on the mind of any senior manager is project cost, and rightly so. Obviously, the lure of a cheaper, faster solution, promoted by various trade journals and media hype, is also always going to pique the interest of business leaders. So is Agile really cheaper? Well again, yes (and no).

Referring again to the example in Figure 1, were we to implement all 100 features of the software in question using Agile methods, it would undoubtedly take longer and cost more than it would using a traditional development cycle. An Agile team does not focus only on development activities; it allocates time for planning (long term for releases and short term for their iterations) and for viewing the most recent build at the end of each iteration. In addition, Agile teams are continuously testing and, using various DevOps tools/techniques, are constantly deploying and debugging. These are all desirable activities — they are how we achieve a higher level of quality — but that quality comes at the "cost" of time.

How, then, is Agile cheaper? Recall that customers are able to declare the project complete when they see a working version that is "good enough" to meet their needs. By doing so, they've chosen to discontinue work on items now deemed unnecessary, which means that less money is spent on that project. So we might say Agile practices are more cost-effective, in that they allow customers to identify areas of improvement or to declare completion sooner than would occur with traditional development and delivery methodologies.

Communication with a Purpose

Often, teams look to Agile to help with their development process as well as the communication issues within their existing environments. After all, much of the marketing material around Agile espouses rapid communication, a characteristic that is also implicit in the various methodologies. One of the more well-known communication channels for the team is a *daily stand-up*.

A daily stand-up is a short organizational meeting, usually limited to 15 minutes or so. The purpose of the meeting is to allow all team members to quickly provide their current status and to understand the status of their fellow developers. When learning about stand-ups, teams learn about the "three questions" to address when they participate ("What did you do yesterday?" "What will you do today?" and "What's blocking you?"). These questions provide a wonderful framework for the stand-up, but often team members forget the reason for holding the stand-up in the first place.

The stand-up is *not* meant to be a place to solve problems; its purpose is to make other members of the team aware of progress toward the iteration's goals or potential blockers to those goals. The stand-up should provide the status of the *current iteration*, nothing more. Agile teams work in two-week "chunks" of time, with each chunk delivering value, in an attempt to incrementally deliver on the larger project. Any communication outside the scope of the current iteration could cause issues and a delay in delivering. Many teams find themselves simply "going through the motions" of stand-ups, providing information that isn't potentially useful, and thereby missing out on the value of a stand-up, which is to focus on their near-term goals.

Organizational "walls of work" help track the flow of work and where there may be bottlenecks. Walls of work are a physical representation of the organization work that is currently underway and provide a visible status for all to see. The more eyes on a wall of work, determining what is going well and what isn't, the more opportunity for organizations to consider improvements to their process.

> *The biggest challenge for leaders moving to an Agile environment is throttling the funnel of work that is currently in progress.*

Increased communication among team members leads to a culture of transparency. Transparency is about making shared assumptions explicit. The more transparent teams and stakeholders are with one another, the more the organization understands priorities, and the sooner critical feedback loops get closed. Teams can build trust by turning shadow projects into known work and updating leaders when circumstances change that help or hinder their delivery capability. Leaders can build trust by participating in playbacks and showcases, giving context to the customer landscape, and providing the rationale behind project prioritization decisions.

Agile Allows for Flexibility

Agility is about being responsive to marketplace changes and customer needs. While there may be apprehension when adapting to a new reality, there is also opportunity for discovery. Inspection before and after an iteration allows teams not only to see what everyone is doing but to evaluate how projects are progressing. If the process isn't working, teams can either ask "What's wrong?" or move on to "What is a more optimal outcome?"

Frequently, teams are told that working within an Agile framework allows them flexibility in time, budget, and scope. This is often a false promise. Use of Agile methods does not automatically grant teams the flexibility to choose the work they do, but it does allow them to adapt and improve how they work. When unplanned priority shifts take place, Agile ways of working can enable teams to more effectively embrace the change and feedback they receive from stakeholders.

The biggest challenge for leaders moving to an Agile environment is throttling the funnel of work that is currently in progress. Effective governance means limiting the number of projects and reprioritizing them so that the teams can focus on one project at a time. This ensures teams can deliver more value sooner, from one iteration or sprint to the next. Although this may seem like an oxymoron, it is true. The secret sauce is allowing teams to be able to focus.

The leader's job is to ensure teams are doing the right work, while it is the teams' job to do the work right. Once a team is empowered to make process decisions, it will do the work right and find ways to improve. As teams develop confidence in their throughput capability, leaders grow confident in being able to manage the funnel of work to focus on a customer's most essential priorities.

A popular YouTube video shows the evolution of Formula 1 racing over a 60-year span of time.[10] The dramatic shift from 1960s' pitstops taking well over a minute to under three seconds in 2013 highlights the essence of continuous improvement. In an era of hypercompetition, such improvement may involve changing or redefining rules. Making small changes over time reduces team resistance and simultaneously builds organizational resilience. We believe each of these incremental outcomes enlarges the window of trust between development teams and leaders.

Conclusion

We began this article with the opinion that for teams to truly realize the value of an Agile methodology, they need not blindly follow a prescriptive set of steps but rather understand why those steps are useful and how

they affect an outcome. While there are many teams that have internalized these Agile values and practices, we believe there are many more that have not. We don't mean to imply that all those operating with Agile methods are mere lemmings following a specific method, "word for word," but we do believe there are teams that just don't understand the principles undergirding the practices.

Perhaps we need a better educational model. As we stated earlier, roughly a third of teams practicing Agile cite issues with education or the adherence to a common set of practices within their organization. (We are hopeful that perhaps the more persistent teams will be able to stay the course and realize the value of these Agile methods.) And 40% of teams attribute their issues with Agile adoption as stemming from lack of experience. This calls to mind something Mahatma Gandhi once said: "Knowledge gained through experience is far superior and many times more useful than bookish knowledge."

The potential for success with Agile exists. Teams know it, stakeholders know it, leaders know it. The easily removed "blockers," such as miscommunication, speed of execution, and adoption failure, directly contribute to why success is out of reach for some teams. Resolution is possible only when there is trust and communication among all those involved. Agile methodology at its very inception was designed to focus on the things that really matter. If organizations rally behind this idea — to focus on what really matters — projects following Agile will find success.

References

[1]Beck, Kent, et al. "Manifesto for Agile Software Development." Agilemanifesto.org, 2001.

[2]McDonald, Kent. "Agile Q&A: Why Do Organizations Adopt Agile?" Agile Alliance, 2020.

[3]Nyce, Caroline Mimbs. "The Winter Getaway That Turned the Software World Upside Down." *The Atlantic*, 8 December 2017.

[4]McAvoy, John, and Tom Butler. "A Failure to Learn in a Software Development Team: The Unsuccessful Introduction of an Agile Method." In *Information Systems Development*, edited by W. Gregory Wojtkowski, et al. Springer, 14 September 2008.

[5]Tanner, Maureen, and Ulrich von Willingh. "Factors Leading to the Success and Failure of Agile Projects Implemented in Traditionally Waterfall Environments." Proceedings from the *Management, Knowledge, and Learning International Conference (Human Capital Without Borders: Knowledge and Learning for Quality of Life)*, Portoroz, Slovenia, 24-27 June 2014.

[6]Samia, Abdalhamid, and Alok Mishra. "Factors in Agile Methods Adoption." *TEM Journal*, Vol. 6, No. 2, May 2017.

[7]Basili, Victor R., et al. "Empirical Findings in Agile Methods." *Proceedings of Extreme Programming and Agile Methods — XP/Agile Universe*, Springer 2002.

[8]"13th Annual State of Agile Report." CollabNet VersionOne, 7 May 2019.

[9]"13th Annual State of Agile Report" (see 8).

[10]CpatainCanuck. "Formula 1 Pit Stops 1950 & Today." YouTube, 2019.

Matthew Ganis is an IBM Senior Technical Staff Member in Armonk, NY. He is also an Adjunct Professor of Information Technology and Astronomy at Pace University, where he focuses on the implementation of the Internet of Things. Dr. Ganis is the coauthor of Social Media Analytics: Techniques and Insights for Extracting Business Value Out of Social Media. *He can be reached at mganis@pace.edu or via Twitter @mattganis.*

Michael Ackerbauer is an IBM Leadership and Innovation Consultant and ICAgile-certified coach and facilitator in Armonk, NY. He is also an Adjunct Professor of Information Systems at Marist College, where he focuses on strategy and policymaking, leadership, and creative problem solving. Dr. Ackerbauer earned a PhD in creative leadership for innovation and change. He can be reached at Michael.Ackerbauer@marist.edu or via Twitter @macker.

Nicholas Cariello is an IBM Cybersecurity Engineer and Software Developer in Armonk, New York. He earned a bachelor's degree and a master's degree in computer science, with concentrations in cybersecurity, from New York Tech. In his free time, he is a community theater Production Director. He can be reached at nicholas.cariello@ibm.com.

KNOW YOUR WHY

Fit-for-Purpose Agility: There Is No "One True Agile"

by Eric Willeke

Agile proponents need to recognize that agility itself is materially different when applied in different environments and that while the core practices and values are fairly portable, the way that they fit together ends up being remarkably different. The best agility is one that is "fit for purpose." That purpose is defined by the strategy of the business unit or company, or even by the behaviors of the market category into which the company is selling.

> *Positioning and posturing about which is more important or "the right thing" fails to serve the community at large and actively misleads companies attempting to improve.*

There is little doubt that agility and its various outcomes will be absolutely necessary for companies that want to be successful in the future. However, corporate leaders who are trying to lead their companies to the next level are finding the landscape of agility ever more fractured and unclear, and the burnout associated with failed adoptions continues to decrease people's desire to engage in future Agile-related changes. Four problems in how Agile is applied in most companies today are at the root of those challenges — yet are not explicitly addressed by the various frameworks and adoption approaches popular today. These four problems are:

1. Agility is splintered into many branded subdomains.
2. Agile is treated as a one-size-fits-all model, detached from context.
3. Agile is applied to value pockets rather than the full value chain.
4. Agile is applied to groups without a clear shared purpose.

(Note: this article refers to "team" and "group" as interchangeable concepts referring to small groups of people working together within a company. These could be Scrum teams, call center support cells, or executive leadership teams.)

Escape Splintered Agility

One of the critical things the various Agile-related communities struggle with is having failed to maintain a single comprehensive perspective on all the factors that influence agility. They have decided that Lean Startup, DevOps, Kanban, and others are somehow unique and different from Agile and, thus, different from the shared goal of "uncovering better ways of developing software by doing it and helping others do it"[1] while solving business problems. In practice, each of these approaches and methods addresses a completely different part of the system while being necessary for the effective functioning of the whole. Positioning and posturing about which is more important or "the right thing" fails to serve the community at large and actively misleads companies attempting to improve. If we look at a typical value chain, each step has a different framing around agility, as shown in Figure 1 and discussed below.

Starting at the front of the cycle, Lean Startup[2] looks at the iteration of business models and the rapid experimentation required to discover whether you have a business that is fit for purpose, or viable for scaling. If you are unable to rapidly test ideas, you can't iterate. If you cannot build a sustainable platform while iterating ideas, you may discover that you have a wonderfully fit-for-purpose business that others are able to copy quickly. You then lose in the fast-follower game to another player with strong execution capabilities because you did not build the habits needed for execution.

"Traditional" Agile, as seen in Scrum or other methodologies, tends to look at the execution aspects, as seen in the ability to fill a product backlog, execute against it, and iterate empirically based on what you have learned by the building and testing of that backlog. However, if you are unable to deploy to production, you cannot test

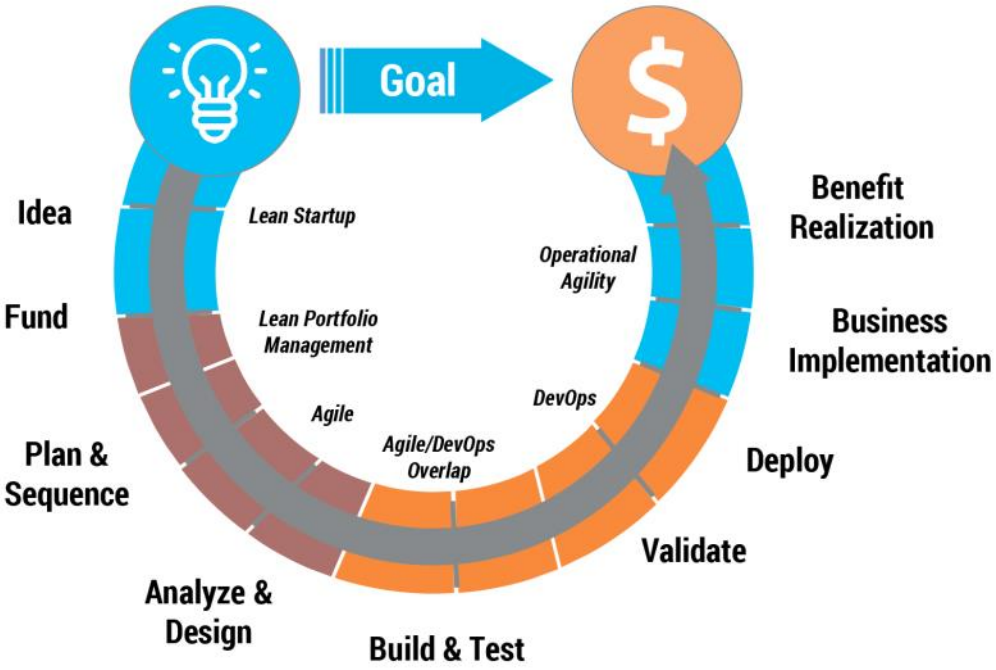

Figure 1 — Technology delivery value stream.

customer value meaningfully. If you have not built your pipeline to consider operations and sustainability, your cost of future features and operating the product will skyrocket, eliminating your ability to execute and test quickly in the market. Finally, if you are not applying the Lean practices inherent in Kanban, you can iterate all you want and yet your continuous improvement will stall out.

We can focus on resolving just one of those problems through approaches like DevOps, which centers on single-piece flow along the pathway from development through operations, including the cost of operating a system already in production. However, if it is not paired with effective ideation or a larger view of what needs to be built, you have built an engine that is going nowhere. In theory, we can solve the "problems" caused by the well-tuned engine of DevOps by putting something like Lean UX[3] on the front end of the process to determine what you are testing and how to experiment in the market. This works exceptionally well for single products but does not work in the same way for larger products or multiproduct families that require many collaborating teams.

The practices in the Large Solution[4] and Lean Portfolio Management[5] configurations of the Scaled Agile Framework (SAFe) can solve the challenge posed by larger products and collaborating teams, yet even then the results are not fully tied back to the run-the-business metrics and KPIs that the people who are chartering and paying for the technology work use. This disconnect leads to the need for Lean benefits realization management (BRM)[6] for value recognition and technology business management (TBM)[7] for costing approaches.

To summarize, the reality is that technology development needs to be a single, cohesive whole with a number of supporting practices that deliver real value when implemented together but which are insufficient when utilized separately. This is where we get the true concept-to-cash cycle of business agility, where the process starts with ideation and discovery practices and ends only when customers are receiving value from what has been produced, and the business is receiving some value in exchange. Achieving this level of end-to-end view requires a certain degree of alignment across the entire company and forces the erosion of the silos between technology and "the business." The connections between these various topics, the areas they cover, and the overall end-to-end cycle can be seen in the value stream wheel in Figure 1.

Addressing any single portion of this flow will indeed deliver value and likely pay for the investment of money and time needed to drive change. However, any one of these solutions will fail to sustain on its own; at best, each will locally optimize for a while, and, at worst, will actively damage neighboring sections of the value flow.

Note that the flow depicted is not a uniform approach that will be the same for every company but is one that must be adapted to your organization, your business model, and your operating environment. The attempt to deploy someone else's model as written will result in subpar results, at best, and could be catastrophic to your operating capabilities, at worst. These concerns arise, for example, with how the Spotify model has become popularized, along with many naive applications of SAFe. These individual frameworks and methods, including the value stream overview wheel in Figure 1, are each incredibly valuable as a collection of practices and as a checklist of things to consider and address. They are not, however, intended to serve as the final, complete answer to agility in your organization.

The manifestation of agility should be completely dependent upon the goals and desired outcomes of the team under consideration.

Realize That There Is No One-Size-Fits-All Model

The manifestation of agility should be completely dependent upon the goals and desired outcomes of the team under consideration. You need to understand what the mandate is for the team you are considering, and whether the "team" is a single development team, a business operations team, or an executive leadership team. Each type of team is responsible for delivering a different outcome, and, thus, how agility is applied looks very different for each of those types of teams.

You should look at three things to understand a team. First, you need to understand the team's purpose and mandate and how those fit with the overall strategy of the organization. Second, you need to understand the composition of the team and the capabilities and motivations of the current members of the team. Finally, you need to understand the nature of the work the team does and how that work interacts with other work items — and with the work of other teams. Generate the team charter based on the intersection of those three considerations. The charter, therefore, encompasses the purpose, funding, responsibilities, and general operating approach for that team and communicates those elements to other teams in the organization.

As one example of how this might work in practice, let's look at a typical Agile product development group. These "teams of Agile teams" generally have a product space in which they play, a market segment they are targeting, and the expectation of generating a certain set of financial results for a given level of funding. To generate those results, these teams often look at creating a product vision and a product roadmap and then planning in terms of increments that allow them to pursue their strategic experiments in a well-aligned manner, while freeing up individual teams within the group to pursue tactical experiments very rapidly. These types of organizations often benefit from applying a model such as SAFe or Large-Scale Scrum (LeSS)[8] because the model provides higher-level constructs at a product level while providing a useful container with a set of constraints within which teams operate mostly independently.

Conversely, consider the customer-facing support team associated with that same product. Its responsibility is to protect customers from disruptions to value achievement for the product while providing an adequate level of service to all customers at the lowest possible funding level. This team's work arrives sporadically and in an unexpected manner, often from unexpected sources. While some items can be put into a plan safely, most of the team's work needs to be handled as it arrives, in compliance with a known service-level agreement (SLA). As a result, these teams often are organized into cross-functional work cells, with an overall intake manager that determines priority and routing, and each team works its queue in priority order with no estimation or forward commitment other than to maintain the SLA. These teams tend to use a straight Kanban model, working from a single-level, ticket-based system rather than a multilevel hierarchical system. Something like Scrum is not at all a fit for these teams.

Finally, let's look at the example of an entire development portfolio with thousands of contributors. The portfolio has a mandate to ensure that investments are applied in the most effective way pursuant to the strategic goals of the business and the need for sustainability across the portfolio. The funding may run into the hundreds of millions of dollars, and the work covers an incredibly wide variety of responsibilities. The executives leading this portfolio need to behave as a cross-functional team themselves, and their team's agility is not about doing the work but rather about shaping overall outcomes and direction and then ensuring that the organization is designed effectively to do the work repeatedly, adaptably, and in the shortest

sustainable lead time. The executives will do very little tactical planning work but instead will rely on having a clear set of strategic goals, well-understood objectives and key results (OKRs) and KPIs, and an effective visualization of how they have allocated funding and responsibilities across the organization and how those responsibilities are interdependent. They will focus on making fewer, more impactful decisions and communicating them clearly rather than maximizing the throughput of decisions. Agile for this group will look like a strong quarterly planning cadence, an incredibly effective capability to generate clarity across the organization, and a focus on nurturing the structure that allows information and impediments to flow to them for resolution as areas of low clarity and incorrect assumptions are identified. Meanwhile, they are decentralizing as many decisions as possible lower in the organization and encouraging the leaders that report to them to do the same. From squint distance, this could look like a leadership team applying Scrum, but this cross-functional executive team has a completely different shape and feel than would a team of individual contributors.

Consider the Full Flow of Value

Each of the above cases demonstrates a wildly different opportunity to apply Lean and Agile approaches depending on the type of team and where the team fits in the organization. In the first section, we identified that agility needs to address the entirety of the concept-to-cash cycle, starting from the moment that a need or opportunity is identified and not ending until the customer has received value and the business has received value in return. Even though that cycle could start and end with an internal customer, it needs to be end-to-end as part of the team's mandate.

A real danger arises when Agile techniques are applied in a team that does not have end-to-end responsibility for its mandate. When this occurs, the team lacks the ability to truly organize and optimize for delivery of value and can optimize only for how well it produces work. These work-optimized teams are among the most dangerous teams in your organization, especially when they are at their best! Success usually leads to expansions of responsibility, but in this case, responsibility becomes unmoored from purpose and detached from the original customer-centered intent. This failure is especially true for functional teams focused on producing a certain type of work because they become very good at doing that work while failing to question how much of that work should be done in the first place. A classic example might be a customer support team that becomes so effective at resolving issues quickly that it fails to notify the product organization of the recurring sources of dissatisfaction, thus preventing the continuous improvement of the product.

This struggle of maintaining an end-to-end mandate is the source of many of the design decisions in the various approaches that exist for scaling Agile beyond a single team. Some approaches focus on decreasing the span of the mandate (e.g., focusing on running a single microservice as if it's a product). Others focus on creating a larger team-of-teams construct to allow encompassing the entire value chain, while still preserving a degree of agility in how things are delivered. Still others focus on the overall flow in a pure Lean perspective, trusting that the intelligent leaders along the stream can adapt the team structure to fit smoothly. Each of these is quite effective in the right circumstances, but too often organizations fail to slow down to look at that end-to-end question as part of their initial Agile adoption.

A real danger arises when Agile techniques are applied in a team that does not have end-to-end responsibility for its mandate.

Regardless of the approach chosen, the group (team) needs to have collective responsibility for each aspect if it is going to maintain organizational health and retain flexibility in the face of changes in the business. When teams do have this mandate, however, the results are nothing short of astounding. At one extreme, it could be the US $100 million program that went from six months behind schedule on an 18-month plan to being on time and on budget; or, at the other, it could be the small end-user documentation team in Hyderabad, India, that partnered with the product development teams to form work cells and then released the highest-quality, highest-rated customer documentation that company had ever produced … at a lower cost, with a faster lead time, and without any manager involvement.

Generate Intense Clarity of Shared Purpose

Having clear, effective, end-to-end mandates captured as explicit team charters provides two critical anchors for a team. First, such charters make the team's mission and goals incredibly clear to everybody on the team, serving very powerfully to decrease internal misalignment and help people collaborate more effectively. This alone can be one of the more effective tools for triggering continuous improvement within a team because the team members have a shared understanding of what they are working to improve (beyond their ability to produce work).

> *We don't have a choice; we need to change, and the best we can do is avoid the mistakes of the past.*

Second, such charters are tools that communicate very clearly how a given team needs to work with the rest of the organization. When there is tension between teams or lack of agreement on priorities, very often the disagreement can be traced back to something in each team's mandate. An explicit charter allows for much more effective discussions and easier resolution through exploring tradeoffs or compromises, or even win-win opportunities.

One powerful example of clear, well-defined mandates comes from how organizational theorist Geoffrey Moore sets up the four zones in his book *Zone to Win*.[9] Each group of leaders becomes very clear on what their group is responsible for as a leadership team, while senior leadership can manage the system of priorities and mandates in a very effective way.

Finally, mandates make executive leadership's job significantly easier. Looking across the charters of a number of teams provides executive leadership with a far easier visualization from which to design and redesign the organization as needs change. Each team, based on the mandate its charter represents, can be funded to a given level, allowed to execute with minimal day-to-day instruction, and evaluated continuously based on its results against the goals captured in the charter. Additionally, once teams have clear charters, the combined leadership of the various teams can much more effectively design and execute large change initiatives based on which charters and mandates the change is going to affect. Best of all, the overall structure of the organization is easier to see and manage, dramatically simplifying the information landscape confronting senior leaders and allowing far more effective decision making.

Time for Action

We are a decade past the point where we should be making the case for Agile. We should be figuring out how quickly and effectively we can achieve the benefits of improvement in every aspect of our business. However, failure to address the four mistakes outlined in this article routinely results in Agile deployments that achieve only a sliver of their total potential, often becoming ostracizing to one cohort or another, whether operations, "the business," executive leadership, or some other group. This leads to situations where people are resistant to, or do not believe in, the power of focused improvement because they have been burned or ostracized in the past. Most large companies are on their third, fourth, or even fifth attempt at enterprise-wide agility, and it is entirely understandable that people are hesitant to try yet again. But we don't have a choice; we need to change, and the best we can do is avoid the mistakes of the past.

In summary, there are four structural considerations to attend to for each and every Agile team, regardless of your model for scaling or at what level in the organization the team sits:

1. **Don't splinter your transformation.** Treat all aspects as part of one holistic effort and focus on outcomes, not "Agile" or "DevOps."

2. **Fit agility to the team.** Have a common language for how you work, while encouraging team-specific practices that accelerate that team's value.

3. **Align around the full value.** Ensure that each team can own the end-to-end delivery of value, especially if that end-to-end mandate doesn't fit your existing organizational boundaries.

4. **Define mandates and write charters.** Create deep and precise clarity on expectations and outcomes and keep them updated and fresh.

These elements do not stand on their own, of course, but they are absolutely crucial and often are not addressed by today's common adoption and scaling frameworks. I sincerely hope that this article is not seen as advice to ignore the experience of the world and completely roll your own approach. Rather, it is very much advice to start with the outcomes and what each team needs, and then identify how your selected approach will best serve those teams.

References

[1]Beck, Kent, et al. "Manifesto for Agile Software Development." Agilemanifesto.org, 2001.

[2]Ries, Eric. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Penguin Books, 2001.

[3]Gothelf, Jeff, and Josh Seiden. *Lean UX: Applying Lean Principles to Improve User Experience*. O'Reilly Media, 2013.

[4]"Large Solution SAFe." Scaled Agile, Inc., 2020.

[5]"Lean Portfolio Management." Scaled Agile, Inc., 2020.

[6]"Benefits Realization Management Framework." Project Management Institute (PMI), November 2016.

[7]Tucker, Todd. *Technology Business Management: The Four Value Conversations CIOs Must Have With Their Businesses*. TBM Council, 2016.

[8]Larman, Craig, and Bas Vodde. *Large-Scale Scrum: More with Less*. Addison-Wesley Professional, 2016.

[9]Moore, Geoffrey A. *Zone to Win: Organizing to Compete in an Age of Disruption*. Diversion Books, 2015.

Eric Willeke is cofounder and Principal of Elevate.to. He partners with leaders at all levels in applying Lean-Agile techniques to get better results through improved change management, better steering and alignment, and a deeper focus on the outcomes that matter. Mr. Willeke engages across many industries in some of the world's largest and most complex companies, as well as the smallest, high-growth startups. He helps by instilling Agile business management techniques, accelerating the deployment of critical initiatives, and mentoring executives leading large-scale Agile transformation efforts. Mr. Willeke is a principal contributor to the Scaled Agile Framework (SAFe), a SAFe Fellow, and a SAFe Program Consultant Trainer. He can be reached at eric@elevate.to.

IT'S BIOLOGY

Evolving Business Agility Through Directional Selection

by Masa K. Maeda

There appears to be a dichotomy in Agile adoption worldwide. After briefly walking through this dichotomy, this article explores how a directional selection approach encompassing mindset, the human aspect, collaboration frameworks, and methods can help organizations evolve.

A Dichotomy in Agile Adoption

Google Trends data suggests that, as of March 2020, the search popularity of the term "Agile" had grown from 28% to 73% over the prior 16 years (see Figure 1),[1] while the search popularity of "scaled Agile" had grown from 1% to 92% in just seven years (see Figure 2).[2] The impressive growth of searches relating to scaling Agile demand is an indicator of the strong need to improve processes and results at large organizations.

Last year, only 6% of survey respondents to the "13th Annual State of Agile Report"[3] reported that, when it comes to Agile maturity, "Agile practices are enabling greater adaptability to market conditions," and just 12% have experienced a "high level of competency with Agile practices across the organization." These low percentages show that, although Agile has entered the mainstream, its expected benefit has been falling short of expectations. Herein lies the dichotomy. Of the Agile methods developed prior to the Agile Manifesto's publication in 2001, only Scrum (the most popular of all) and Extreme Programming (XP) are widely used, and, to a lesser extent, Crystal. A significant minority of organizations use feature-driven development (FDD) and the dynamic systems development method (DSDM), based on my research. Kanban, which surfaced in 2009, has gained some popularity as well. However, most organizations practice it rather poorly, which is unfortunate given how powerful it is (my partners and I have consistently achieved better results with Kanban than with Scrum). Still, none of these methods is sufficient to satisfy the demands of large organizations or large projects and programs.

That insufficiency motivated the creation of Agile at scale frameworks, such as the Scaled Agile Framework (SAFe), Large Scaled Scrum (LeSS), and Nexus. The compromise involved in adopting those frameworks is that the Agile principle of simplicity fades away as the frameworks, to some extent, force a shift toward mimicking the complexity of large organizations, which makes them attractive to mid- to high-level decision makers due to the similarity to structures with which those decision makers are already familiar.

A question then emerges: what alternatives, if any, exist to successfully increase adaptability to market conditions and increase competency with Agile practices at a large scale? That's where directional selection comes into play.

Directional Selection

That the first Agile methods work well for small organizations and projects but not as well for large, complex organizations and projects puts pressure on those methods to evolve. That pressure is similar to what happens with directional selection in biological evolution.

Directional selection (one of three types of evolution) occurs when the environment pressures a species to display a more extreme form of a trait. Imagine, for example, one insect of a certain species that blends in better with its surroundings and is better camouflaged than another insect of the same species, whose color contrasts with its environment. Birds will much more easily see, catch, and eat the contrasting individuals compared to the camouflaged ones. Directional selection will favor the coloration that offers superior camouflage, and it is that greater fitness that will prevail.

In this article, I am adapting the term "directional selection" to a business use to indicate the natural shift an organization goes through when one way of achieving a business goal has a better economic outcome, or directional selection, than any other alternative. Indeed, directional selection in complex, large organizations is the force behind the many

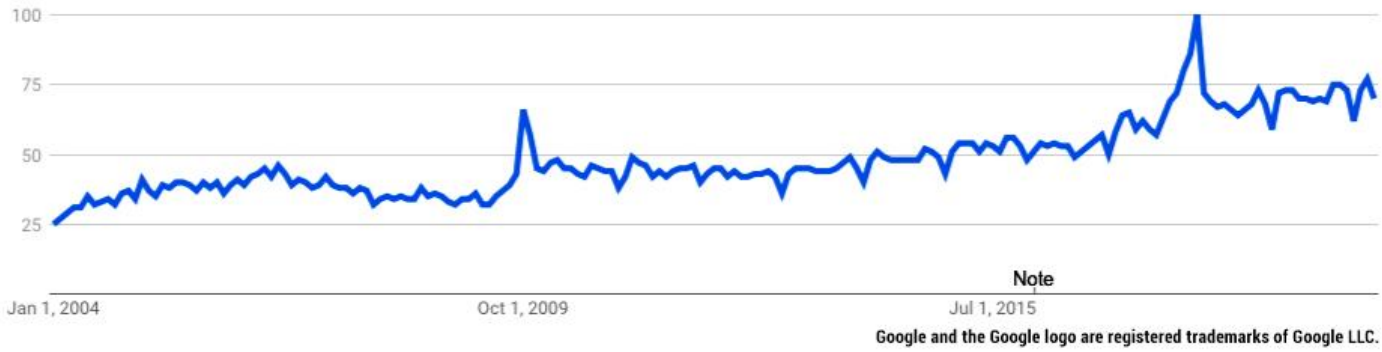

Figure 1 — Popularity growth of "Agile" in Google search (January 2004-March 2020). (Source: Google Trends.)

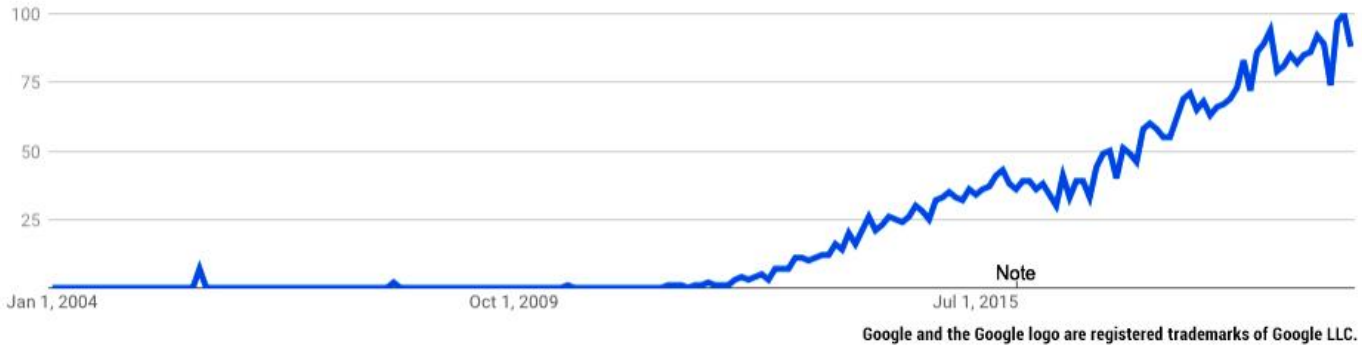

Figure 2 — Popularity growth of "scaled Agile" in Google search (January 2004-March 2020). (Source: Google Trends.)

changes required to increase an organization's fitness. Seeking successful change by focusing only on methods isn't enough. Agile is both a mindset and a method for taking action, but, in unfortunate reality, most Agile implementations stick to applying the mechanics of the method only.

Showing the Agile Manifesto to teams, managers, and stakeholders and discussing it in a two-hour Agile introduction session serves no practical purpose. We need instead to center mindset-based actions on the human aspect in order to achieve sustainability and generate better, high-quality value with a positive economic outcome for customers and the business. That healthy balance comes naturally from directional selection, which merges mindset, the human aspect, collaboration frameworks, and methods together.

Mindset considers — moving from a low to a high level — Agile thinking, first- and second-generation Lean thinking, systems thinking, and ecosystems thinking. The human aspect includes behavioral economics and organizational behavior. Collaboration frameworks are tools that motivate high participation in the activities needed to bring products from idea to market. Methods include XSCALE, Spotify's approach, Disciplined Agile, Scrum, XP, Kanban, SAFe, LeSS, and Nexus.

The Ecosystems Thinking Mindset

The most important element of directional selection as applied to business is the mindset. The mindset is what supports the alignment of all members of an organization in a cultural shift to the creation of an ecosystem. In an ecosystem, all of an organization's products and services, its market, and its customers interconnect in such a way that all derive benefit. (Apple and Amazon are good ecosystem examples.)

Let's first consider the lower levels of mindset. There is, of course, the well-known and easily accessible Agile Manifesto. Next, the first-generation Lean mindset is the adaptation of many aspects of the Lean manufacturing mindset onto knowledge work. (The work of Mary and Tom Poppendieck is a great introduction.[4])

Note that not all elements of Lean manufacturing apply to knowledge work. Consider, for example, Six Sigma, which focuses on eliminating variation and favors the stability of a settled process over exploration, innovation, and creativity; hardly what anyone would consider desirable for knowledge work. Donald Reinertsen, however, introduced second-generation Lean, which brings a 180-degree shift to

product development management.[5] This version of Lean fosters:

- An economic view of value flow that includes efficiency over productivity.
- Embracing positive variability. Not all variation is negative, just as not all risk is negative.
- Proper management of small batches and queues over large batches and queues by limiting the amount of work in progress (WIP). If you think about just-in-time, then you will have an idea of what this entails.
- The economic balance of centralized and decentralized decision making over centralized decision making. Consider, as an example, how efficient telecommunication networks function.

Reinertsen's seminal work consists of 175 principles, and I strongly recommend that every decision maker apply them.

A system is a set of interrelated and independent parts. Systems thinking is about considering those parts and the consequences of their interrelationships. In knowledge work, systems thinking isn't about software or computer systems but rather about the organization, its components, and its contexts. As a systems thinker, you consider:

- Interconnectedness over disconnection
- Circularity over linearity, which allows feedback to occur
- Emergence over silos, to tackle unpredictability and to enable influence and collaboration
- Wholes over parts, because everything is interconnected
- Synthesis over analysis, to integrate instead of separate
- Relations over isolation, to generate correct systems

Some of the most relevant figures in systems thinking are W. Edwards Deming[6] and Gerald Weinberg.[7]

We can bring ecosystems thinking to our customers as formalized in XSCALE[8] — the most robust approach to Agile de-scalability — because it is a superset of the other mindsets. De-scaling the complexity of an organization or project toward the simplicity of Agile is the opposite of scaling Agile methods to match the complexity of the organization or project. An organization as an ecosystem is a network of mutually balanced interconnectedness. In an ecosystem, you think *breadth-first* instead of *depth-first*, meaning that you consider not only the issue or need at hand but also how it relates to its context and how that context interrelates with its contexts. Ecosystems thinking has 12 principles:[9]

1. "Observe and interact: business, design, and technical stakeholders must work together continuously face to face as peers to design solutions that fit each others' constraints."
2. "Capture and store learning in small, self-organizing, cross-functional teams: as these work to satisfy current constraints they become capable of breaking through future bottlenecks."
3. "Obtain a yield: each team must measure its contribution to top-line business throughput as the result of the work it is doing."
4. "Align teams into self-managing value streams that continuously adapt their work priorities to changing market feedback."
5. "Share resources, services, and learnings across teams and streams: motivate mutual benefit to reduce silos, dependencies, duplicated efforts, and missed opportunities."
6. "Mercilessly refactor value streams: reuse, recycle, or reduce all the resources a stream produces until all provide value and none go to waste."
7. "Design breadth-first. Step back to see patterns in and between markets and value streams. These patterns form skeletons of designs that we can detail as we learn more."
8. "Collaborate rather than delegate: the right people are in the right relationships when conversations develop between them so they learn together and support each other's work."
9. "Take the time to simplify and automate solutions: simple, automated systems cost less to maintain than big, manual ones, doing less work to make more useful outcomes."
10. "Use and value experimentation: experiment to reduce risk and adapt each product to the changing constraints of your organization and its markets."

11. "Enrich interfaces and serve under-served markets: spaces between market segments are where most opportunities for innovation and productivity occur."

12. "Transform to embrace change: change the patterns of your organization to open its bottlenecks and generate opportunities for new markets."

XSCALE depends on tools (collaboration frameworks, discussed later) to practically apply some of these principles. XSCALE is a natural way to transition from silo thinking to ecosystems thinking.

In my colleagues' and my work, we bring ecosystems thinking to organizations through a combination of discussion workshops and continuous reference to these 12 principles during hands-on work. For example, when WIP limits need to be determined while implementing Kanban or Scrumban, SAFe, Scrum, and poor Kanban implementations have teams figure out the WIPs for their boards in a self-organized fashion without ensuring that the teams have an in-depth understanding of the effects of limiting WIP. That is dangerous because, without a good understanding of the economics of the process visualized by those boards, the situation is more likely to get worse than better. What we do instead is discuss with teams the context of the project and the process while referring to the two U-shaped curves shown in Figure 3.

The total cost (TC) curve is, as Reinertsen proposes in his work of second-generation Lean,[10] the sum of the holding cost (HC) and the transaction cost (TnC). TC determines a good economic batch size range, or WIP. HC is the cost associated with idle batch items (those waiting for attention). TnC is the cost of actively completing a batch item. The new total cost (NTC) curve is my proposal to consider the human aspect by including the cost associated with people's level of stress; the NTC curve determines the best economic WIP. Considering the stress cost (SC) increases sustainability.

Teams that apply the quantitative management approach shown in Figure 3 mature more rapidly and improve their economic efficiency better than teams that arbitrarily determine WIPs. Teams with lower levels of stress produce better-quality products and are healthier; healthy people spend fewer days sick and, therefore,

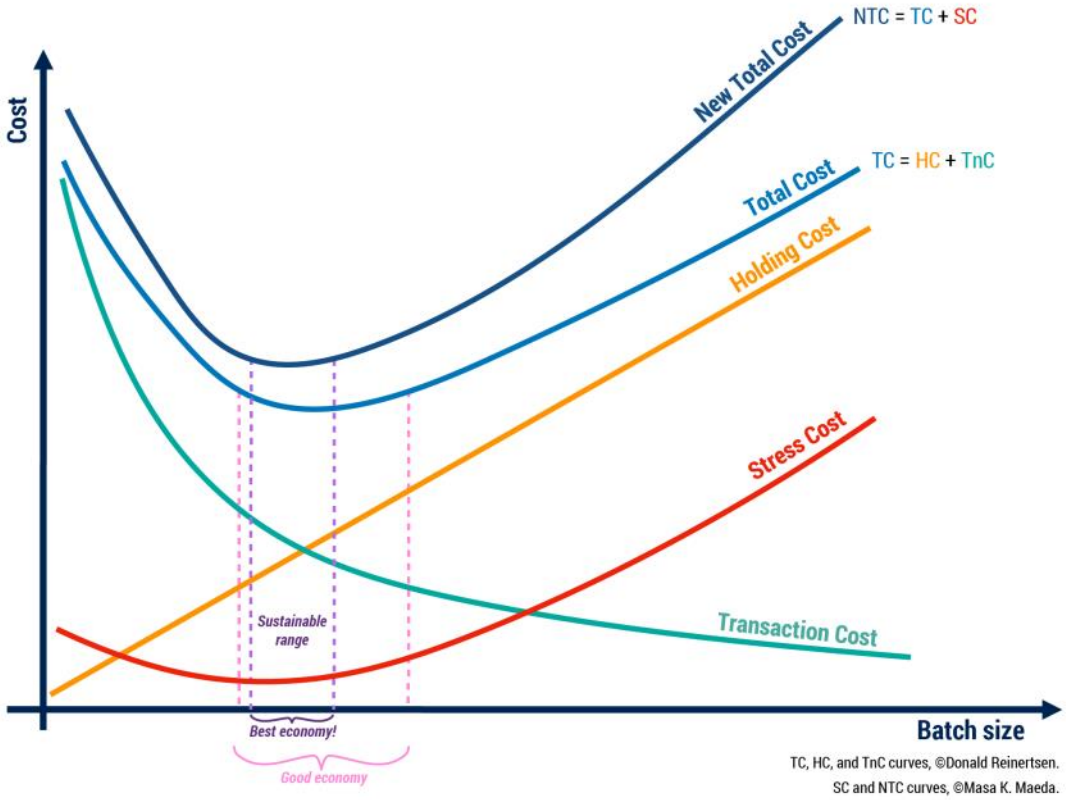

Figure 3 — U-shaped optimization curves to reach the best economic sustainability in a system. (Source: Reinertsen, with additions from Maeda.)

more days working. Overall, organizations that apply ecosystems thinking in a practical way have a competitive advantage that is difficult to match in any other way.

Human Aspect

The human aspect of directional selection, as it relates to organizations, has three elements: people's needs as human beings, organizational behavior, and behavioral economics.

People's Needs as Human Beings

The number one complaint I hear from teams is that their managers treat them as though they were robots or objects, not people. Decision makers are constantly under pressure to turn ideas into products and to bring those products to market; to achieve their goals, they may feel they have no choice but to ignore the adverse effects on people's mental state and health.

Stress in the workplace has disastrous consequences for both the business and its people. Last year, the American Institute of Stress reported that:[11]

- 83% of US workers suffer from work-related stress.
- US businesses lose up to US $300 billion yearly as a result of workplace stress.
- Stress causes around 1 million workers to miss work every day.
- Only 43% of US employees think their employers care about their work-life balance.
- Depression leads to $51 billion in costs due to absenteeism and $26 billion in treatment costs.
- Work-related stress causes 120,000 deaths and results in $190 billion in healthcare costs yearly.

When my colleagues and I collaborate with decision makers to adopt Agile methods and collaboration frameworks, we advise them to always consider the human aspect when making decisions. A vital element of our work is to guide decision makers through self-discovery.

As an example, let's say that Michael is responsible for a hard-deadline project that is now at high risk of not making the deadline. Michael wants people in the organization to work two extra hours on weekdays, seven hours on Saturdays, and four hours on Sundays over the next two months. He is aware this schedule will burn people out, but he sees no alternative. His response is to promise his teams that he will grant them one half-day off per week once the teams meet the deadline.

Guiding Michael through questions focused on empathy and increased awareness of economic impact lead him to take alternative courses of action. In this scenario, my colleagues and I would work with Michael to figure out what modifications at the systems level would maximize the chances of meeting the deadline while at the same time minimizing the negative impact on his people. The critical point is to get decision makers to focus more on managing methods, processes, and tools to increase value flow instead of focusing on managing through changes that negatively impact people.

Organizational Behavior

Organizational behavior is about how people act in groups. Organizations need to be aware of the benefits of applying knowledge gained from organizational behavior to increase efficiency. Agile methods and collaboration frameworks are highly effective at improving the efficiency and effectiveness of people interacting in groups.

Behavioral Economics

Behavioral economics helps us find two disciplined ways to make decisions with better economic outcomes. One way is through understanding how some aspects of Agile methods eliminate or reduce the negative impact of biases of intuition that influence the decisions we make and have an economic impact. Let's consider, for example, ego depletion. This is a mental fatigue state we reach later in the day or toward the end of the week as our brains get tired from mental activity. We have a natural tendency to make better decisions earlier in the day (or, for night owls, later in the day) and even earlier in the week, when we are mentally sharper.

The Kanban method is one way to stay objective and avoid poor decisions due to ego depletion. Figure 4 lists, on the left, circumstances that increase ego depletion and, on the right, which aspects of the Kanban method help reduce ego depletion and its adverse effects. The Kanban method helps maintain cognitive ease (being mentally relaxed), which results in lower stress levels.

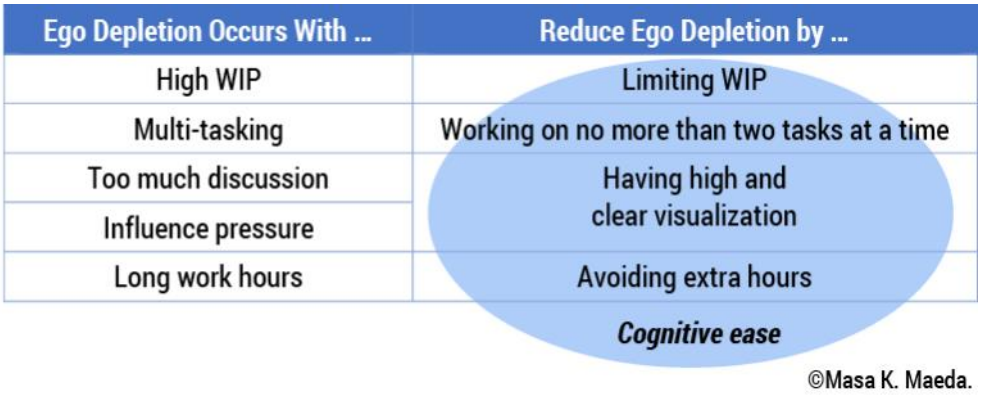

Figure 4 — How the Kanban method helps minimize the adverse effects of ego depletion.

A second way to stay objective and avoid the negative effects of ego depletion is by understanding the impact of habitual ways of thinking (biases), and either not falling back on our biases or using them to gain economic advantage. For example, risk aversion and risk seeking are biases that make most people more sensitive to losing than they are to winning, and this bias can cloud judgment. Let's say that your business can work on only one project at a time, and you have two potential contracts. One of them assures you earnings of $9 million, and the other one is conditional, with a 90% chance of earning $10 million. Which of the two contracts would you choose? Most decision makers, affected by the bias of risk aversion, would choose the first option.

Let's now say that at some other point in time, your business confronts a situation in which you will either certainly lose $9 million or have a 90% chance of losing $10 million. Which alternative would you prefer? Most decision makers, affected by the bias of risk seeking, would choose the second option. In both examples, the options are the same from the point of view of logic and statistics. However, the biases of risk aversion and risk seeking cause people not to be as logical and rational when making decisions, as is often presumed. When decision makers are aware of biases like these, they can improve their economic decisions and the economic results of their organizations.

Collaboration Frameworks

Collaboration frameworks contribute to directional selection as tools that make collaboration and communication more efficient by reducing the complexity of those activities at large, complex organizations or on large, complex projects.

Agile methods are applicable only to a subset of the value chain (the series of activities) that brings a product from idea to market. What about those activities not covered by Agile methods, or what about enhancing some activities that exist in some Agile methods? That's where collaboration frameworks — tools that improve activities commonly conducted at organizations, such as customer understanding, business modeling, strategic roadmapping, portfolio prioritization, prototyping, requirements generation, retrospectives, and so on — come in handy.

Organizational alignment at all levels of the enterprise has always been a challenge, so let's consider an example. Tug of War (TOW) is a Lean collaboration framework I developed to visualize alignment and build strategies to improve it. Figure 5 illustrates the general format of a TOW.

TOW has six sections. In this case, we'll say that the area generating the TOW is software development. The upper-left section of the artifact indicates what aspects upstream (e.g., architecture, business analysis, UX, marketing) benefit software development work, such as requirements, commitment, and leadership — illustrated as three smilies in Figure 5. The lower-left section shows various negative influences from the upstream that tamper with software development, such as poor market understanding and bad UX, in general. The upper- and lower-central sections are about what software development does to itself that helps (e.g., behavior-driven development) or harms (e.g., poor modularity). Finally, the right-hand sections indicate what both the upstream and software development do that benefits, or harms, their downstream. In the example used in Figure 5, the downstream isn't getting any benefit.

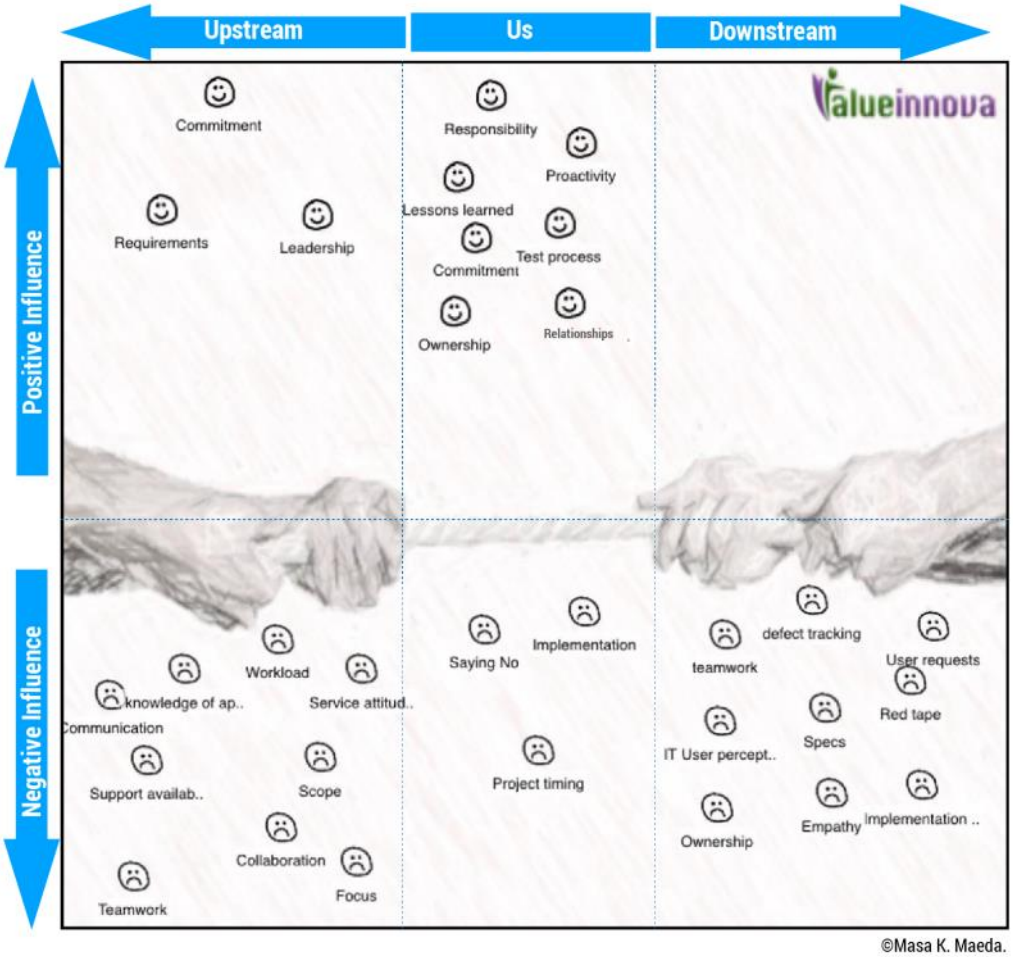

Figure 5 — A Tug of War collaboration framework to visualize and understand alignment.

Applying TOW to all the steps of the value stream provides valuable data from the perspective of each step in the value stream. To determine the degree of alignment along the different steps of the value chain all we have to do is observe how compatible the upstream and downstream influencers of one step of the value chain are with respect to their upstream and downstream accordingly. As illustrated in Figure 6, the more upstream influencers from software development's TOW are in agreement with the downstream influencers of UX's TOW and architecture's TOW, the higher their alignment. The same criteria apply for the alignment between software development's TOW downstream and the TOW upstream of operations.

TOW is applicable to organizations of any kind and size for strategic and tactical decisions. A larger number of items in the positive sections of the artifact and a lower number of items in the negative sections indicate a higher level of alignment.

Another collaboration framework that has had an outstanding impact is Luke Hohmann's budget games.[12] I had the privilege to be one of his collaborators when budget games were used, for eight years in a row, to prioritize the annual budget for the city of San Jose, California, in the heart of Silicon Valley.[13] San Jose had a budget surplus of almost $1 billion by year seven. I applied this framework to prioritize the annual project portfolio at one of the four largest telecoms globally with resulting savings in project execution of around 40%.

(See "Recommended Reading" at the end of this article for links to additional collaboration frameworks.)

Multiple Methods Are the Right Approach

A method is the baseline way of working the value stream, be it end-to-end or a subset. Some Agile methods are a better fit than others at large scale.

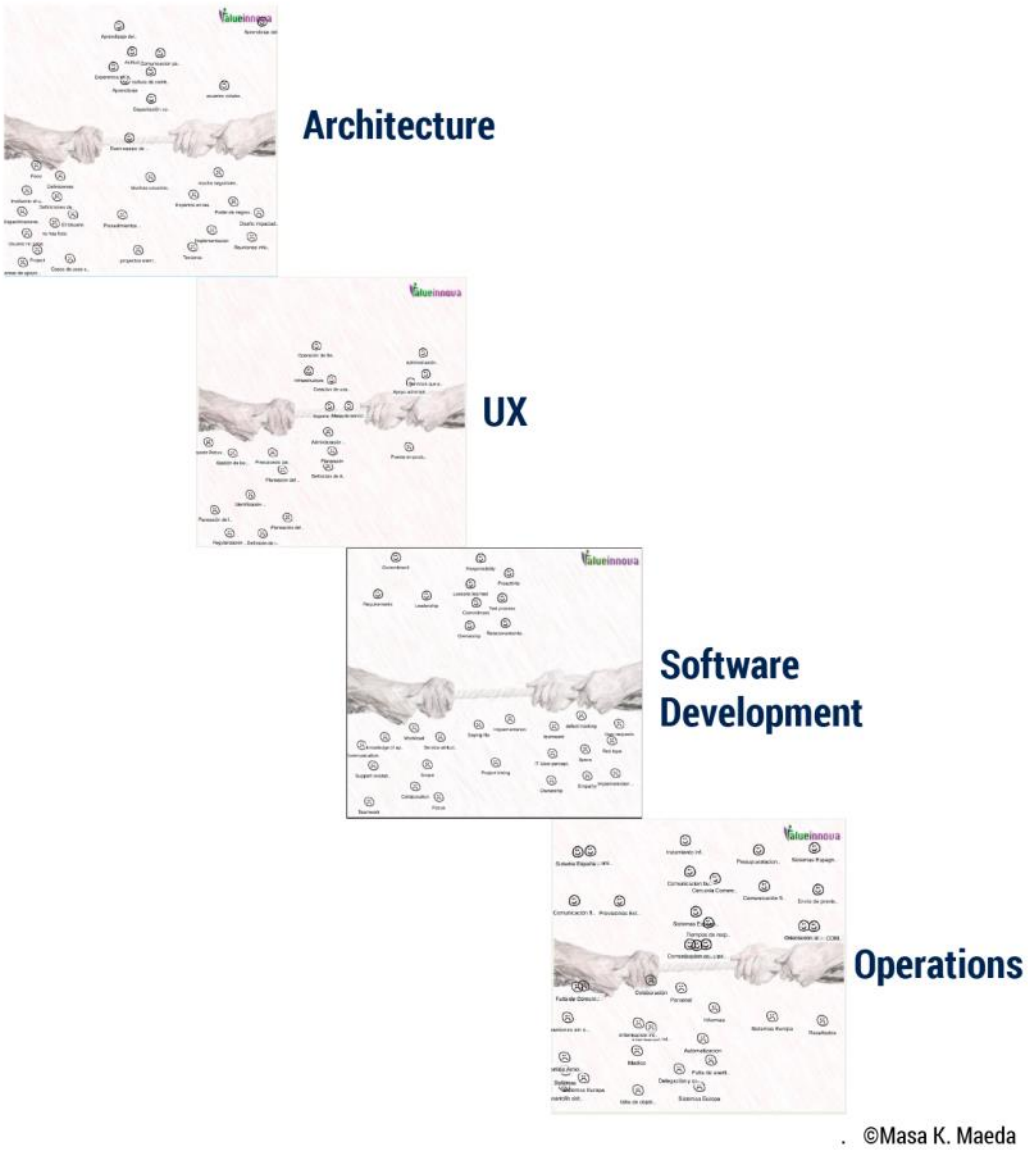

Figure 6 — Chain of TOWs to determine alignment at an organization.

The ones discussed here are those I consider good for directional selection.

Literature on each Agile method exists elsewhere, with great detail. Within the scope of this article, XSCALE, Disciplined Agile, and the Spotify approach to agility are the only ones I know of that have a fractal structure and are fit for directional selection, with XSCALE being the most suitable. These methods' base structure is applicable at small and large scale without increasing complexity, and thus make it possible to de-scale — in other words, simplify — the complexity of the organization or project.

Ultimately, a digital and Agile transformation based on only one method will achieve marginal success due simply to the nature and limitations of each method. The right approach is to apply multiple methods and, often, combinations of methods in each area of the organization and to mature those methods as the organization itself matures. What aspects of which methods to apply to an organization or project depends on the state and kind of organization, its culture, and its processes.

My Main Advice

Your organization has a higher chance of succeeding with its Agile or digital transformation through a directional selection strategy because it brings a rather natural and more complete approach that considers the whole organization and offers a broader set of aspects and activities than are available by merely adopting Agile methods. This gentle disruption also reduces resistance to change because it helps people realize the personal benefit they receive (like stress reduction, as shown in my discussion of the U-shaped optimization curve), in addition to the benefits to the organization.

References

[1]"Search Term: Agile." Google Trends, 2020.

[2]"Search Term: Scaled Agile." Google Trends, 2020.

[3]"13th Annual State of Agile Report." CollabNet VersionOne, 7 May 2019.

[4]Poppendieck, Mary, and Tom Poppendieck. *The Lean Mindset: Ask the Right Questions*. Addison-Wesley Professional, 2013.

[5]Reinertsen, Donald G. *The Principles of Product Development Flow: Second Generation Lean Product Development*. Celeritas Publishing, 2012.

[6]Deming, W. Edwards. *Out of the Crisis*. MIT Press, 2018.

[7]Weinberg, Gerald M. *An Introduction to General Systems Thinking*. Dorset House, 2001.

[8]"XSCALE." XSCALE Alliance, 2020.

[9]Merel, Peter. "12 Principles of Ecosystems Thinking." LinkedIn, 29 August 2018.

[10]Reinertsen (see 5).

[11]"42 Worrying Workplace Stress Statistics." The American Institute of Stress, 25 September 2019.

[12] Hohmann, Luke. *How to Prioritize Your Project Portfolio Using Conteneo Collaboration Games*. Conteneo, 2014.

[13]"San Jose, CA, Community Leaders Budget Games Results." InnovationGames, 15 February 2011.

Recommended Reading

Ariely, Dan. *Predictably Irrational: The Hidden Forces That Shape Our Decisions*. Harper Collins, 2009.

Beck, Kent. *Extreme Programming Explained*. Addison-Wesley, 2004.

Beck, Kent, et al. "Manifesto for Agile Software Development." Agilemanifesto.org, 2001.

Beehr, Terry A. *Psychological Stress in the Workplace*. Routledge, 2014.

Beehr, Terry A., and Rabi S. Bhagat. *Human Stress and Cognition in Organizations: An Integrated Perspective*. Wiley-Interscience, 1985.

Burrows, Mike. *Kanban from the Inside: Understand the Kanban Method, Connect It to What You Already Know, Introduce It with Impact*. Blue Hole Press, 2014.

"Foundation of Disciplined Agile." Disciplined Agile, 2020.

Hohmann, Luke. *Innovation Games: Creating Breakthrough Products Through Collaborative Play*. Addison-Wesley Professional, 2006.

Johnson, C. Merle, and Terry A. Beehr (eds). *Integrating Organizational Behavior Management with Industrial and Organizational Psychology*. Routledge, 2012.

Khaneman, Daniel. *Thinking, Fast and Slow*. Farrar, Straus, and Giroux, 2011.

Kimsey-House, Henry, and Karen Kimsey-House. *Co-Active Coaching: Changing Business, Transforming Lives*. Nicholas Brealey, 2011.

Maeda, Masa. "Some Collaboration Frameworks." LinkedIn, 24 March 2020.

Maeda, Masa. "XSCALE's Ecosystems Thinking." LinkedIn, 24 March 2020.

Osterwalder, Alex. *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*. John Wiley and Sons, 2010.

Rubin, S. Kenneth. *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley Professional, 2012.

"SAFe for Lean Enterprises 5.0." Scaled Agile, Inc., 2020.

Thaler, Richard H. *Misbehaving: The Making of Behavioral Economics*. W.W. Norton & Company, 2016.

Masa K. Maeda is a Senior Consultant with Cutter Consortium's Business Agility & Software Engineering Excellence practice. Dr. Maeda, a long-standing Agilist and member of the leading community on high collaboration frameworks, is a pioneer of XSCALE and Kanban for knowledge work. He is also founder of Valueinnova LLC. Dr. Maeda has 26 years' international experience providing consulting, coaching, and training services to companies from Fortune *500 to startups in four continents. He is a current member of the steering committees for the Ecosystems Alliance, Agile Testing Alliance, DevOps++ Alliance, and Kanban for Knowledge Work Alliance. Dr. Maeda was also on the team that founded Lean Kanban University. He is a founding member of four Silicon Valley startups that pioneered online socialization, online entertainment, and genomics/proteomics. In addition, Dr. Maeda performed R&D at Apple. He has over a dozen Agile-related certifications and holds advanced degrees in artificial intelligence. He can be reached at mmaeda@cutter.com.*

TELL ME WHY

The Chicken or the Egg … Who Goes FIRST in Agility?

by Bob Galen

For the past 20 years, the typical strategy for an Agile adoption — transformation, business agility shift, DevOps initiative, and the like — has been:

- Approximately two hours of leadership training
- One to two days of Scrum or Kanban training for teams
- Training and recruitment of coaches, scrum masters, and product owners
- Perhaps a reorganization into "cross-functional" teams

And then … *Bang*! The gun goes off, and we're "Agile." We start sprinting and flowing toward improved efficiency and increased results, right?

No, this is the completely wrong way to approach Agile adoption. I liken this to a *team-first strategy*, and in my 20 years of cleaning up other people's Agile messes, I've seen it time and time again.

Sure, it has some distinct advantages. First, you start moving quickly. You move from whatever you were doing to a so-called Agile approach. In some cases, this occurs with a few teams, and in others, with tens to hundreds. Often the transition happens quickly and is measured based on how many teams you are spinning up per week or sprint.

Another advantage is that it's easy and comfortable. From a leadership and sponsorship perspective, the leaders have kicked things off, and their job is mainly done. It moves from strategic to tactical execution, which is a management and team challenge. The leaders then can move onto another strategic initiative. Or a product-oriented pivot. Or ongoing budgeting, planning, and forecasting. Or an M&A activity. There's just one problem with this approach: it's wrong.

According to last year's "13th Annual State of Agile Report"[1] the top three challenges or barriers facing Agile adoption or transformation initiatives are:

1. Organizational culture at odds with Agile values (52% of respondents)
2. General organization resistance to change (48%)
3. Inadequate management support and sponsorship (44%)

> *A critical factor for the failure of Agile initiatives is that leaders are disengaged from their strategies.*

Based on these results and my own experiences, I contend that a critical factor for the failure of these Agile initiatives is that leaders are disengaged from their strategies. They're not taking responsibility for the deep learning, mindset shift, personal role shift, and cultural shifts required of them in a transformation of this magnitude.

The failure is not specifically about Agile. Indeed, any significant change initiative needs this sort of leadership engagement, where the leaders must go first in leading, or showing the way. But what does it look like to reconsider our Agile perspective?

Patterns of a Leadership-First Strategy

Six patterns are central to a *leadership-first strategy*:

1. Finding your compelling why
2. Engaging leadership training and coaching
3. Forming aligned first team
4. Creating Agile transformation team
5. Embracing change as the constant
6. Becoming active culture-shaper

Next, we explore each pattern more fully as fundamental to your leadership-first efforts.

Finding Your Compelling Why

The very first thing a go-first leader needs to consider is the *why* behind the intent to adopt Agile. Think of this why as being multifaceted, in that it's not merely about the business dynamics. And it should start as an inside-out job.

The first-level why is: why are *you* pivoting to agility personally as a leader? What's in it for you, and how do you and your abilities, skills, and mindset align with Agile principles? What compels and energizes you about agility?

> *The very first thing a go-first leader needs to consider is the* why *behind the intent to adopt Agile.*

The second-level why is at a *team* level: what's in it for your teams? How much of a change is it for them, and what will incent or motivate them to embark on this changing path? Look back at their history and challenges to map these to some of the success factors that agility will bring to your teams. This why should connect to compelling drivers for your teams to change.

Finally, the third-level why is at an *organization* level. As was done at the team level, look back broadly at the challenges and issues you have faced organizationally; innovation, quality challenges, predictability, and delivering customer value might come to mind. Then connect the dots from those challenges to your Agile strategy. This why needs to be compelling at an organizational level and lead everyone to a much better place.

Not only do you need to have a solid grasp of the three levels of your why, but you'll need to develop compelling stories that you can share to communicate your vision adequately. These whys become the heartbeat of your adoption strategy and efforts.

Engaging Leadership Training and Coaching

The second step in your leadership-first strategy focuses on in-depth training coupled with initial coaching. The strategy here is toward gaining an Agile mindset as a leader to more fully understand the pivot you'll be making in your leadership approaches.

I often see leaders in transformations attending a short, one- to two-hour overview of Agile and expecting to receive a deep understanding from it. At most, they might attend a day-long class. The rationale for any pushback is that they don't have time for a "class."

From my perspective, this isn't an acceptable learning investment for this size and scope of organizational change. Instead, leaders should start with a stance of curiosity to deeply understand Lean and Agile approaches to product delivery and organizational dynamics. To gain sufficient depth, you should attend multiday training classes. The topics should include coverage of organizational modeling and culture for Agile contexts. Most importantly, topics should explore the leadership shifts from a management mindset to a leadership-centric mindset.

An essential part of this shift toward a leadership-centric mindset is engaging an enterprise-level Agile coach or coaches to help guide you in your journey. Coaches serve as guides and sounding boards as you develop your transformation strategies. You don't want someone who tells you what you want to hear, but rather a coach who dares to challenge you and tell you the truth — no matter your reaction or the consequences.

The focus here is less on tools and tactics and more on the soft skills necessary to lead your organization. For example, practical communication skills are vital in the early stages of your Agile adoption, as is fostering trust, safety, and empowerment. And the importance of these elements never really ends.

Forming Aligned First Team

Often in organizations, there is quasi-strategic alignment at a high level. That is, most leaders say they're aligned horizontally toward business goals, but often their priority and a good portion of their focus is skewed vertically toward their own organizations.

It turns out that the cross-functional team alignment aspect of Agile applies not only to the delivery teams but to the leadership team as well. What an Agile transformation often uncovers is that the senior leadership team in an organization isn't a team at all. This lack of alignment can "seep into" the teams and negatively impact their teamwork and performance.

In his landmark book, *The Five Dysfunctions of a Team*, Patrick Lencioni described the notion of a "first team."[2] A first team emphasizes horizontal, peer-to-peer loyalty, teamwork, and accountability first. This is atypical in most organizations, as leaders' first loyalty is skewed vertically toward their own teams. A first team of this kind amplifies the horizontal connections that are crucial in an aligned Agile transformation.

Go-first leaders need to form a first team that focuses on the following:

- Finding and solidifying a common *why*
- Creating a *shared vision*, along with goals and outcomes
- Establishing deep-rooted, cross-functional *alignment*
- Implementing continuous learning and creating a *shared mindset*
- Fostering peer-to-peer *accountability*
- Helping one another with *peer-to-peer coaching*

While such a team can be a challenge to create and sustain, it's of fundamental importance if you want your transformation to be sustainable, consistent, balanced, and productive.

Creating Agile Transformation Team

Part of your leadership-first strategy is forming what I call an "Agile transformation team," or ATT. The ATT is a group whose purpose is to instantiate agility across the organization. This team will guide the strategy, culture, and tactical backlog of what gets done and in what order.

The ATT operates as a first team, no matter the functional responsibility of each member. A side effect of forming an ATT is that it operates entirely as an Agile team, leveraging either Scrum or Kanban. The members essentially show the organization and, most importantly, their teams, that they're willing to "eat their own dog food."

Similar to the ATT, a core element of the Scrum@Scale[3] Agile scaling framework is the notion of an executive action team (EAT). To the best of my knowledge, Scrum@Scale is the only scaling framework that intentionally engages leadership. The EAT fulfills the scrum master role for the entire organization. This leadership team creates an Agile ecosystem that allows the reference model to function optimally by supporting Scrum values, roles, events, and artifacts. It also serves to integrate the Agile parts of the organization with the non-Agile ones.

Using an EAT means that leaders are supporting the ecosystem and Scrum values in the implementation. In this manner, leadership is not just a checkbook function, but a fully engaged part of the Agile transformation. The EAT forms before any Scrum (or Agile) teams form. In other words, the leaders are going first in establishing that ecosystem.

Embracing Change as the Constant

Most leaders drastically underestimate the impact change can have on an organization. In fact, most organizations I encounter in Agile transformations are suffering from severe to chronic change fatigue. These changes are broad and deep, and include:

- Skill shifting
- Role migration
- Organizational structure change
- Expectation shifts
- Drastic cultural change

One concrete example of the sometimes excessive amount of change is the tendency of leaders to try and solve execution and cultural challenges by reorganizing. In many firms, there seems at times to be a monthly or quarterly tempo for continuous reorganizations. Of course, these leaders are trying to achieve results by reconnecting the organization, but for the most part, constant reorganizations only add to the change fatigue and role confusion.

That said, humans have an immense capacity for change, and go-first leaders do realize that change is the only constant in a volatile, uncertain, complex, and ambiguous (VUCA) world. But that doesn't mean you should simply throw the changes in and hope for the best.

When you go first as a leader, it implies that you internalize and experience the change before your teams do. You gain empathy and realization of the

scope of the change, allowing you to become more of a change guide or Sherpa for your organization — staying out ahead of your teams and leading them through each change.

One of the more useful change tools I've found is Dr. John Kotter's "8-Step Process for Leading Change."[4] One of the important things to remember about Kotter's model — or any change model, for that matter — is to take time to allow a set of changes to "seep in and settle" organizationally. Somewhere between Kotter's sustain and institute steps, for example, you'll want to pause and realize the benefit and impact of that set of changes. Take a breath, rest, and then look to create a sense of urgency around your next set of changes.

Becoming Active Culture-Shaper

I've saved the best and most important pattern for last. It's the role that go-first leaders play in creating, setting, and shaping their cultures. Even beyond the role being their *responsibility*, I'd expand it to be an imperative in Agile transformations. I like to call it a "culture-shaping role," as agility, at its essence, is a cultural and mindset shift for the organization, from the C-level down to each individual.

If you really look at the culture within any organization, you can see the behavior of the leadership team directly influencing the cultural landscape or ecosystem. This is revealed in the leaders' words, actions, expectations, commitments, behaviors, body language, and business goals. It shows up in what they choose to amplify and not amplify as important. And, before you discount it as something too big, hairy, or soft, I contend that culture-shaping is not a complex activity, nor does it have to be a considerable or protracted effort.

I like to think of culture-shaping as a baby-step or micro-step effort. That is, you shape the culture one action at a time, then rinse and repeat, over and over again. The practice begins at the individual leader level, crosses into the first team, and eventually infects every leader in the organization.

Culture-shaping amplifies, in everyday interactions, core Agile principles and values, including:

- Trust
- Safety
- Eating our own dog food
- Invitation
- Transparency

Here are some examples of culture-shaping micro-steps that you might plan for tomorrow:

- Surprise one of your teams by joining them at the daily scrum, merely to listen to and appreciate their efforts.
- When one of your teams is struggling, step back and let them figure things out on their own. Don't interfere unless they ask for help.
- Schedule a mentoring/coaching meeting with one or more of your leaders and help them trust their teams more. Stop pushing so hard for speed over quality.
- Take a "walkabout" (or Gemba walk) around your office to acknowledge your teams' great efforts and look for every opportunity to share your personal appreciation.
- Spend the entire day telling stories individually, in teams, and in groups. Stories that illustrate why folks are focused on what they are and why it matters to your business and customers.

You get the idea. One act at a time. One conversation at a time. One example at a time. One step at a time. Over and over, consistently and patiently. Keep this up, and you will inevitably reshape your culture.

There you have it: six patterns to consider, internalize, and implement in your leadership-first strategy. There might be more, but I believe this small set is a great baseline to begin your culture-shaping and Agile transformation journey.

Measuring Success

Beyond the patterns, go-first leaders need to pay attention to measuring the success of their efforts. But the focus is different from traditional, execution-based metrics. For example, you'll want to focus on sustainability; that is, the stickiness of your transformation.

One way to define that is whether your Agile transformation continues to grow and prosper without you. If it does, then you have successfully created an ecosystem that grows other change agents and culture-shapers, where you become unnecessary in generating momentum and focus.

The second measurement focus isn't a singular metric, but more of an aggregate that considers your level of balance. Specifically, are you balanced across these four quadrants of interest?

1. Predictability
2. Quality (product)
3. Customer value
4. Team health

When I present this quadrant model, it usually generates a lot of discussion and debate, often centered around which quadrant is the most important. I typically don't select one, but rather explore the tension between them, which is the balance to which I'm referring.

For example, if you focus too much on delivering as many features as possible (customer value), you may affect quality and team health. Your Agile metrics should focus on maintaining a healthy equilibrium across your newly culture-shaped Agile ecosystem.

The final adjustment is a realization that the metrics have a different audience. That is, instead of focusing the metrics to feed leadership's need for management steering, the metrics are actually for your teams. The metrics should focus primarily on areas and information teams can leverage in their continuous improvement strategies and efforts.

This metrics pivot from leadership-centric to team-focused is a crucial empowerment step for your teams. You'll need to trust that your teams are paying attention and capable of making balanced and appropriate adjustments.

Wrapping Up

For far too long, we've looked at Agile adoptions/transformations/initiatives as something that teams did. They went first, they owned the execution dynamics, and they had to deliver the results. In a word, there wasn't a *partnership* in the transformation. Leaders led (told), and the teams delivered (did). When you view it that way, it's not the right strategy for a change initiative/transformation as pervasive as the one agility entails.

For effective, "sticky" Agile transformations that thrive and deliver on the promises of agility, leaders need to:

- Become a partner
- Deeply engage
- Seek to understand agility
- Walk their talk
- Go first

In a word, in our next generation of Agile adoptions, we need our leaders to *lead*.

References

[1]"13th Annual State of Agile Report." CollabNet VersionOne, 2019.

[2]Lencioni, Patrick. *The Five Dysfunctions of a Team: A Leadership Fable*. Jossey-Bass, 2002.

[3]"The Scrum At Scale® Guide: The Definitive Guide to the Scrum@Scale Framework." Jeff Sutherland and Scrum Inc., March 2020.

[4]Kotter, John P. "8-Step Process for Leading Change." Kotter, 2020.

Recommended Reading

Cohn, Mike. *Succeeding with Agile: Software Development Using Scrum*. Addison-Wesley Professional, 2009.

Denning, Stephen. *The Age of Agile: How Smart Companies Are Transforming the Way Work Gets Done*. Amacom, 2018.

Lencioni, Patrick. *The Advantage: Why Organizational Health Trumps Everything Else in Business*. Jossey-Bass, 2012.

Marquet, L. David. *Turn the Ship Around! A True Story of Turning Followers into Leaders*. Portfolio, 2013.

Sinek, Simon. *Leaders Eat Last: Why Some Teams Pull Together and Others Don't*. Portfolio, 2014.

Sinek, Simon. *Start with Why: How Great Leaders Inspire Everyone to Take Action*. Portfolio, 2009.

Bob Galen is Principal Agile Coach at RGCG, LLC, where he helps guide companies and teams in their pragmatic adoption and organizational shift toward Scrum and other Agile methods. He regularly speaks at international conferences and professional groups on a broad range of topics related to Agile software development. Mr. Galen is also the author of Agile Reflections, Scrum Product Ownership, *and* Three Pillars of Agile Quality and Testing. *He can be reached at bob@rgalen.com.*

NO SILVER BULLET, NO MAGIC FORMULA

Don't Disrupt Agile. Drop It.

by Jeff Doolittle

Much has transpired in the 19 years since the original publication of the "Manifesto for Agile Software Development."[1] After nearly two decades, the word "Agile" now saturates conversations in the software industry and has even overflowed into the popular business vernacular. These days, nearly every company seeks recognition for its ability to "do Agile" in its workplace. Surely such efforts reflect noble intentions. Yet software engineers share an abundance of stories regarding the challenges and frustrations of enduring the real consequences of such intentions. Countless tales tell of processes and practices that have resulted in greatly reduced agility and ever-simmering frustration, contrary to stated goals.

As words enter the general vocabulary, their use and meaning morph over time. Sometimes we hardly notice the difference, even in the wake of significant impact from the change. While ignored by or hidden from many, such alterations of meaning have occurred in reference to the word "Agile."

Targeting agility as a goal for software teams has diminished over time, being replaced instead with efforts to "do Agile" by following prescribed, formulaic practices.

The title of the manifesto that catalyzed the movement employed "Agile" as an adjective. Over time, however, "Agile" has shifted into common use as a noun. This semantic shift has profoundly impacted the way software engineers, development managers, and the industry in general understand, experience, and influence the trajectory of the movement. Targeting agility as a goal for software teams has diminished over time, being replaced instead with efforts to "do Agile" by following prescribed, formulaic practices.

Successful businesses recognize the need to adapt in the face of market competition, industry complexity, and the relentless turbulence of the technology landscape. They do not measure success by conformity to a set of methodologies or practices, regardless of intriguing names or prestigious sounding certifications. The hallmarks of adaptability live and breathe within software teams that deliver real value to customers efficiently, effectively, frequently, and predictably. With that being said, the software industry should collectively pause and reflect on the good, the bad, and the ugly in the history of misguided attempts to "do Agile." The time is long overdue for an industry-wide reconsideration of Agile and what it hath wrought.

As a case in point, while recently surfing the Web, a deliciously insidious example of the bad and ugly side of Agile slapped me across the face. There before my eyes, in an impossible-to-ignore 52-pixel "Extrabold" font,[2] stood the tantalizing headline, "Become an Agile expert for just $39."[3] Expertise on the cheap? What luck! In fairness, the courses for sale appear to require actual study rather than providing a certificate to those who simply fork over the dollars. But "expert"? In what rational universe does mere completion of a course make one an expert, regardless of price paid or time spent in a class? Can anyone believe such snake oil can actually deliver on its false promises? I wish I could say no, but history tells the real story. These promises exist and persist for the simple reason that individuals and organizations want to believe them, and they affirm their status as true believers by forking over real time and money.

Perception of Reality

Even some of the original signatories of the "Manifesto for Agile Software Development" have distanced themselves from what Agile has become. For example, Dave Thomas penned an essay in March 2014 entitled "Agile is Dead (Long Live Agility)."[4] He lamented at that time that "the word 'Agile' has been subverted to the point where it is effectively meaningless." He also pointed out that employing the word "Agile" as a noun is "just plain wrong." While I believe I came to the same conclusion independently, the possibility exists that I came across Thomas's essay six years ago and the idea

burrowed into my subconscious, only to emerge at the opportune moment.

Ironically, even the progenitors of Agile jumped overboard years ago. In the meantime, the string ensemble plays "Nearer, My God, to Thee"[5] as the expansive sea of remaining passengers hum along while the ship sinks ever deeper. The meaning has changed, but the haunting melody lingers.

Changes in language surely affect our perception of reality. Notice the subtle shift in language when people speak of the Agile Manifesto. The original title, "Manifesto for Agile Software Development," emphasizes "Software Development," with "Agile" functioning as a modifier describing a way of approaching software development. Shortening the title has the effect of radically altering the meaning, shifting focus to the word "Agile" and employing it as a noun. Can this change alone account for what Agile has become? Hardly, but meaning matters and our perception of meaning becomes our reality.

Humans have an innate tendency to treat a tool as the tool. Given the tool at our disposal, we convince ourselves it must be the right one or, at least, the best we might conceive of at the moment.[6] Agile as a noun has filled that role marvelously, taking a prestigious (notorious?) position as the de facto goal toward which all software teams must strive. Contrarians observe such overwhelming consensus, unmoored from a sense of empirical rigor, and find themselves duty bound to poke, prod, and ask taboo, subversive questions.

In the 1960s, one of Warren Buffet's clients famously quipped, "No one gets fired for going with IBM."[7] Today's catchphrase would read, "No one gets fired for going with Agile," or whatever other Agile-type methodology you'd like to throw in there — Scrum[8] and Scaled Agile Framework (SAFe)[9] being particularly nefarious candidates.

If you have read this far, you may wonder about the value of spending so much time exploring what amounts to definitions and semantics. They matter greatly if you care about meaning. The grey wizard Gandalf spoke aptly regarding such things in words penned for him by linguistic luminary J.R.R. Tolkien:

> "Good Morning!" said Bilbo ...
>
> "Do you wish me a good morning, or mean that it is a good morning whether I want it or not; or that you feel good this morning; or that it is a morning to be good on?"[10]

Consider as well, as Phil Karlton famously cracked, "There are only two hard things in computer science: cache invalidation and naming things."[11] Difficulty in naming things must not prevent us from doing the hard work. So let's drive the point home further by exploring the meaning of the word itself. Yes, buckle up; we're in for a bit more definitions and semantics.

So What Does Agile Even Mean?

According to the *Oxford English Dictionary*, "agile" may be defined in the following ways:[12]

> 1. Able to move quickly and easily.
>
> 1.1 Able to think and understand quickly.
>
> 2. Relating to or denoting a method of project management, used especially for software development, that is characterized by the division of tasks into short phases of work and frequent reassessment and adaptation of plans.

The first definition describes attributes most software development teams would likely find desirable. Yet typically the necessary efforts for sustaining an ability to move quickly and easily, and to think and understand quickly, remain unconsidered.

Other important questions often remain unexplored as well: How can a team evaluate and measure such attributes in its environment? How exactly does a team achieve such characteristics and maintain them over a significant period of time? True skeptics and free thinkers may even add more to the list of crucial questions: How do team members define and understand the words "quickly," "easily," and "understand" in their context? Can we think of times when such characteristics should not take top priority? What risks, cost increases, or additional complexities might arise if these attributes take precedence above all others the majority of the time? An organization or team seeking agility should ask these and other probing questions about what it actually hopes to attain and how it would measure such attainment, or lack thereof.

Now for the second definition. Another irony reveals itself in that one may jump to the second definition while completely neglecting the first! Frequent reassessments and adaptations provide no actual guarantee of agility. In fact, such efforts may utterly fail to increase, or even maintain, agility. No doubt, value and benefit often come from breaking larger tasks into smaller phases. Once again, though, important questions often go unasked. What tools, techniques, or processes

provide helpful guidance for dividing tasks into short phases? Are they working as intended? What might go wrong if such a process of decomposition, contrary to stated desires, instead hampers agility? How would the organization or team detect or measure such an outcome?

Of course, words cannot bear the burden for their usage, the responsibility for which lies with the wielders of said words. Yet the second definition leaves the door wide open to closed mindedness, zealotry, and a lack of empirical measurement. Such a description adequately sums up the state of attempts to "do Agile" in the software industry. "Measuring agility" often degenerates into conformity to certain ceremonies and practices. Conversations peppered with frequent references to trainings, frameworks, methodologies, and manifestos substitute for observation, experimentation, adaptation, and critical thinking. For example, ask some random software engineers whether their team's Agile process serves them, or if, instead, they have ended up serving the process (and, by extension, serving the purposes of managers and "masters" maintaining the ceremony). All this frequently occurs as a substitute for what really matters: the ability of teams to reliably and predictably create value and respond efficiently and effectively to inevitable changes in requirements.

As a project manager or entrepreneur in the software industry, if forced to choose, which would you prefer? "Short phases of work with frequent reassessment and adaptation of plans"? Or the ability to "move quickly and easily" and "think and understand quickly"? It's a trick question, and, if I were a betting man, I'd wager you fell for it.

You Must Think (for Yourself)

Notable theoretical physicist Richard P. Feynman eloquently described the risks inherent in placing excessive trust in our intuitions and formulations:

> The first principle is that you must not fool yourself — and you are the easiest person to fool.[13]

Surely Mr. Feynman is not joking.[14] Humans excel at seeking shortcuts and quick solutions. From an evolutionary standpoint, this instinct has served us well. But, from a long-term planning and project management perspective, we follow such instincts at our ongoing and compounding peril. Promises of quick wins, easy training, cheap expertise, and magic processes appeal to our baser instincts. But we can sense what our elders have reminded us of all along: there's no such thing as a free lunch. Software architect Juval Löwy roots the basis of this wisdom in a fundamental principle of the universe: the first law of thermodynamics. In his book *Righting Software*, Löwy phrases it this way: "You cannot add value without sweating."[15]

We stand by and watch, or even participate arduously, as copious amounts of effort and sweat pour into various rituals and ceremonies prescribed by the high priests of Agile. Unfortunately, even monumental efforts expended on the wrong thing cannot provide value. The effort and sweat must flow toward productive ends for value to increase. Our industry has traveled long down the wrong road. Logical and forward-thinking action beckons us: the time has come to get on a different road.

As Thomas suggested, the word "Agile" had already outlived its utility years ago. Rather than disrupting Agile, we should move on, simply ignoring it until the fanfare dissipates. Of course, the snake oil peddlers can always discover new snake oil to foist on the unsuspecting or new ways to twist words and concepts to suit their profiteering purposes. In response, we can tie ourselves to the mast and avoid the siren songs promising cheap expertise and promoting dangerous shortcuts.

A final and exquisite irony exists to be discovered by mining the Internet domain where this morass finds its origin. If you look closely, you will find an understated hyperlink at agilemanifesto.org that reads "About the Manifesto." You will then discover on this page a 14-paragraph summary of the history behind the "Manifesto for Agile Software Development." Written by Cutter Consortium Fellow Emeritus Jim Highsmith, one of the original signatories, the written history terminates with an intriguing expression:

> We hope that our work together as the Agile Alliance helps others in our profession to think....[16]

At the end of an obscure page on a highly cited site, we find the only advice you really need. And you never needed permission from anyone for this advice to apply to you.

Final Thoughts

Drop the "stone tablets" written in manifesto form. Forget the alliance, the methodologies, the cottage industry, and the cargo cults. Ultimately, I have one basic and fundamental encouragement for you, dear reader, as you consider whether a need exists for a disruption of Agile.

Don't disrupt Agile. Drop it and think for yourself.

Why rely on a manifesto?[17] Why look to words written down by others based on their intuition and experience, which may or may not apply to your circumstances and situation? Why take the opinions and experiences of others at face value without a critical eye and an inquisitive mind?

I present no new manifesto. I offer no silver bullet methodology. I have no magic formula or perfect set of values, qualities, principles, and certifications to sell you. I could describe in great detail what matters most to me as a software architect and industry veteran: repeatability, reversibility, rigor, and the relentless, even ruthless containment of change whenever and wherever it reveals itself. It would please me greatly to explore such topics with you and your colleagues. But in the end, I repeat my one encouragement and exhortation to you.

Don't disrupt Agile. Drop it and think for yourself. Don't seek to uproot, transform, improve, reclaim, redeem, or "do Agile." Instead, disrupt your own processes in the relentless pursuit of continuous improvement. Think for yourself and invite others to join you in the endeavor. You may surprise yourself at the level of adaptability and value creation you can achieve.

References

[1]Beck, Kent, et al. "Manifesto for Agile Software Development." Agilemanifesto.org, 2001.

[2]Yes, I checked with Chrome DevTools.

[3]"Become an Agile Expert for Just $39." Android Authority, 3 March 2020.

[4]Thomas, Dave. "Agile is Dead (Long Live Agility)." The Coding Gnome, 4 March 2014.

[5]"Nearer, My God, to Thee." Wikipedia, 2020.

[6]"Law of the Instrument." Wikipedia, 2020.

[7]The Conservative Income Investor. "5 Investment Tips from the Legendary Peter Lynch." Seeking Alpha, 28 November 2011.

[8]"Welcome to the Home of Scrum." Scrum.org, 2020.

[9]"SAFe® for Lean Enterprises 5.0." Scaled Agile, Inc., 2020.

[10]Tolkien, J.R.R. *The Hobbit, or There and Back Again*. George Allen & Unwin, 1937.

[11]Fowler, Martin. "TwoHardThings." martinFowler.com, 14 July 2009.

[12]"Agile." Lexico, 2020.

[13]Feynman, Richard P. "Cargo Cult Science." *Caltech*, 1974.

[14]"*Surely You're Joking, Mr. Feynman!*" Wikipedia, 2020.

[15]Löwy, Juval. *Righting Software*. Addison-Wesley Professional, 2019.

[16]Highsmith, Jim. "History: The Agile Manifesto." Agilemanifesto.org, 2001.

[17]I must confess that I myself have previously bought into the value of manifestos, a sin for which I intend to spend the rest of my career paying penance.

Jeff Doolittle is a master architect with over 20 years of industry experience. He helps make good software engineers great. His passion, resolve, and can-do attitude are contagious. Mr. Doolittle began his career in a successful software consulting firm serving companies of all sizes, from small businesses to Fortune *100 companies; in just a few years, he took on the role of CTO. His experience as a consultant exposed him to a dizzying array of business problems for which he crafted valuable, customer-centric solutions. Mr. Doolittle's next challenge came in his role as cofounder and CTO of a SaaS startup, where he provided strong business, architectural, system design, and process leadership. Under his guidance, the company pioneered cloud automation processes and event-driven systems long before they became standard industry practice. Mr. Doolittle earned a master of arts degree in transformational leadership. He can be reached at jeff@jeffdoolittle.com.*

CUTTER Business Technology Journal

About Cutter Consortium

Cutter Consortium is a unique, global business technology advisory firm dedicated to helping organizations leverage emerging technologies and the latest business management thinking to achieve competitive advantage and mission success. Through its research, training, executive education, and consulting, Cutter Consortium enables digital transformation.

Cutter Consortium helps clients address the spectrum of challenges technology change brings — from disruption of business models and the sustainable innovation, change management, and leadership a new order demands, to the creation, implementation, and optimization of software and systems that power newly holistic enterprise and business unit strategies.

Cutter Consortium pushes the thinking in the field by fostering debate and collaboration among its global community of thought leaders. Coupled with its famously objective "no ties to vendors" policy, Cutter Consortium's *Access to the Experts* approach delivers cutting-edge, objective information and innovative solutions to its clients worldwide.

For more information, visit www.cutter.com or call us at +1 781 648 8700.