BEST AGILE ARTICLES 2019

Michael de la Maza, CEC • Cherie Silas, MCC

BEST AGILE ARTICLES

2019

Michael de la Maza, CEC • **Cherie Silas,** MCC

BEST AGILE ARTICLES 2019

Michael de la Maza, CEC • Cherie Silas, MCC
Copyright ©2020 Michael de la Maza. All rights reserved.
Printed in the United States of America

ISBN: 9798663128650

Cover Designer: Alex Kudinov
Layout Designer: Christine Murray

Third Edition: August 2020

Quarterly Online

Best Agile Articles Conference

Hosted by Publication Curators
Michael de la Maza, CEC & Cherie Silas, MCC

Attend Online Conference Live
https://baa.tco.ac/conf

Watch Past Conference Recordings
https://baa.tco.ac/vid

Table of Contents

Table of Contents (continued)

Table of Contents (continued)

Foreword

We are delighted to bring you the Best Agile Articles of 2019, our third edition of this publication. Our goal in publishing these yearly editions is to cull through the many articles that are published each year and provide you with a curated set of high-quality articles that capture the latest knowledge and experience of the agile community in one compact volume.

Our purpose is twofold. First, we understand that it can be hard to figure out where to go when looking for ideas and answers. There are thousands of blogs, videos, books and other resources available at the click of a mouse. But that can be a lot to sort through. So, we thought we could be of some assistance. Second, we wanted to bring some visibility to many people who are doing good work in this field and are providing helpful resources. Our hope is that this publication will help them connect to you, the ones they are writing for.

Our intention is that this publication is to be by the agile community as a service to the agile community and for the agile community. With that in mind, we pulled together a great group of volunteers to help get this work into your hands.
The articles in this volume were selected by:

- A diverse Volunteer Committee of sixteen people with expertise in a variety of areas related to agile.

- The agile community. A call for article nominations went out in early 2020 and several dozen 2019 articles were nominated by the community.

The articles themselves cover a wide variety of topics including organizational structure, culture, and agile leadership. There is something for almost everyone here.

All editions of the Best Agile Article publications are available on Amazon and free to download on the Best Agile Articles site.

We are thankful for the great participation by the agile community at large. If you would like to participate in delivering this publication in future years, we would welcome hearing from you.

Publication Curators

Michael de la Maza, CEC **Cherie Silas,** MCC

Best Agile Articles 2019

View Publication Online or Download for Free
https://baa.tco.ac/2019

When did we stop being OK with failing?

By Lyssa Adkins

At the beautiful Paraparaumu beach in New Zealand, I am watching a skim boarder. Reminds me of being a kid and watching my brother learn how to skim board. He would catch the board, ride for a while, fall down, roll over in the sand and PopUp ready go again. Over and over again. Ride, fall down, PopUp. I don't think it ever occured to him that those falls were actually failings. He certainly didn't seem bothered by them.

I think of all the things we learned as kids. Even the experience we all had learning how to walk. We fell down over and over and over again but it didn't seem to bother us. Precisely when did we start seeing those fallings as failings? What was the exact moment we started to put such a heavy toll on failing? I'd love to go back and change that very moment. I'd love to learn like a kid.

I have been learning a lot of new things lately and I notice the desire in me to "look good" and "get it right" the very first time. Even though I know that F.A.I.L. just means First Attempt In Learning, the deep impulse in me is to hide it or beat myself up when I fall down a bit. This happens over and over again as I'm learning something new.

When was the moment that I started to prize looks over learning? I timetravel to that precise moment and tell my young woman self that a different story can be written. That it is possible, and more joyful, to prize learning over looks.

About Lyssa Adkins

I am a coach, facilitator, teacher and inspirer.

My current focus is improving the performance of top leadership teams and Boards of Directors through insightful facilitation and organization systems coaching.

Making difficult decisions faster and with clear alignment, unknotting challenging, multi-department impediments, creating the conditions for organizational delivery…this is where I thrive and help thrive.

I am the author of Coaching Agile Teams, which is still a top-10 Agile book years after publication.

I like to explore facilitating intense conflict, societal change, organizational change, the benefits and costs of being human in the workplace, agile coaching, agile transformation, adult human development, human systems dynamics, the role of nature, and books of all sorts. I tend toward a balance of the provocative and practical.

To learn more visit

www.lyssaadkins.com
www.linkedin.com/in/lyssaadkins

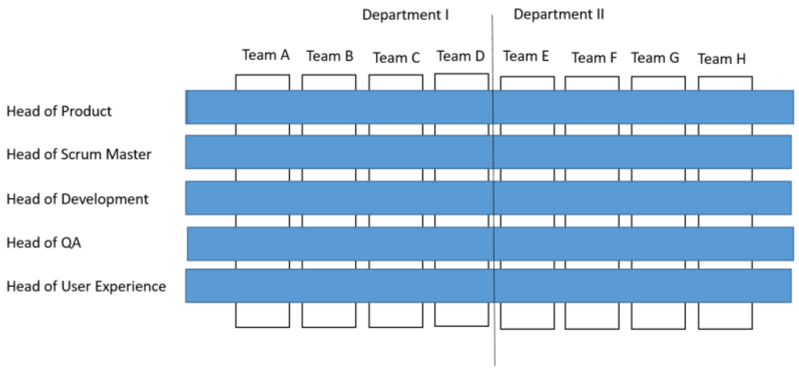
You want to adopt the “Spotify Model” ? I don’t think it means what you think it means!

Scrum and beyond…

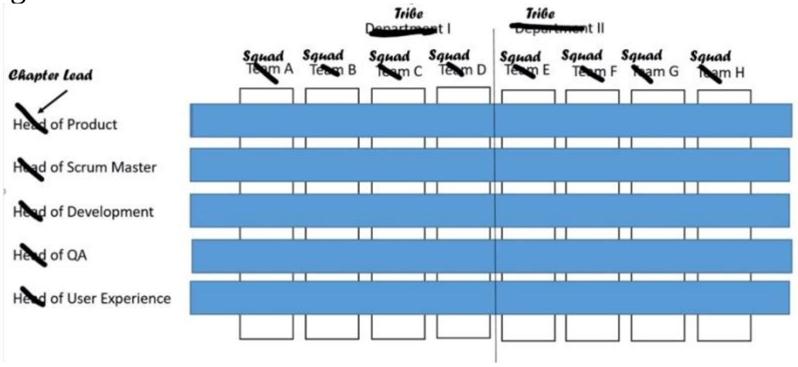
By Willem-Jan Ageling

So your company has a matrix organisation model? Then you saw the Spotify organisation model with Tribes and Chapters and you thought; “Hey, we can keep our good old matrix organisation and simply rename it to “The Spotify Model!” Did you? Do you think this makes you a front-runner? Well, think again.

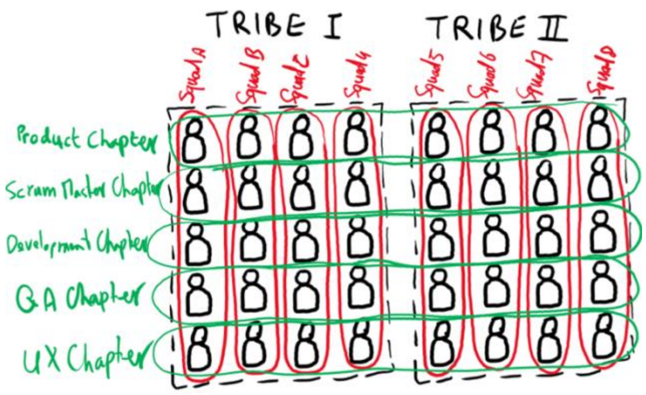
It doesn’t work to take this model:



Change some words:



And make it look more informal:



This misses the mark of the videos Spotify engineering culture part 1 and Spotify engineering culture part 2.There is so much more vital information that shouldn’t be ignored!

How did it miss the point of the Spotify engineering culture videos?
Yes, the organisation model in the video is a light matrix. But the chapter lead is a servant leader, focusing on coaching and mentoring the chapter members. This is a pivotal part of this organisation model. So, ask yourself the question: is your former head of development — now chapter lead — indeed a servant leader? Or is (s)he still a traditional manager?

When the narrator — Henrik Kniberg — explains the model there also is this pivotal remark: *“**It’s a pretty picture huh? Except it’s not really true […] in reality the most valuable communication happens in informal and unpredictable ways”.*** This is supported by having Guilds, lightweight communities of interests. Even more striking is the following statement: ***“Most organisational charts are an illusion.”***

Spotify puts community over structure. Is this what you had in mind when you copy-pasted the organisation model?

Here's another important message in the video: ***"If you need to exactly know who makes*** *the decisions you are* ***in the wrong place".***
Are you considering all of this when you map your matrix organisation to a "Spotify Model"? Do you understand the nature of the "Spotify Model" if you think it is similar to a classical matrix organisation?

Don't think that the "Spotify Model" gives you clarity and structure for years to come. Kniberg emphasises several times that ***the video shows a snapshot of a mix of the current situation combined with what they are aiming for.*** In the end, it is not about how the organisation is defined, but instead about how Spotify can adapt and change structure, practices, etc… towards what is needed at that time. Another important remark is that "healthy culture heals broken process."

What else is of often totally missed?

I now touched the topic of organisational structure. But there's more. ***These are videos about engineering culture, not about the organisational structure.*** That's only a small part of the two videos. Here are some bad practices that are often not changed when adopting a "Spotify Model":

- All squads have to use Scrum as a framework — Not according to Spotify! This was a Scrum organisation, but they made all Scrum practices optional. They found that some of the standard Scrum practices got in their way. They say "Rules are a good start, but break them when needed.” Spotify believes that Agile matters more than Scrum. You might not agree with this. Note that you then already deviate from how Spotify establishes self-organisation of squads.

- All Squads are in control of development. Deployment and operations are within another department — OK…. Spotify has teams with end to end responsibility for the stuff they build, including deployment and operations.
- All squads use Jira as the admin tool and the chapters determine what tools and techniques to use. Individuals can’t deviate from this. — Not at Spotify! They have very little standardisation and squads have a lot of autonomy. They value cross-pollination over standardisation and are experiment friendly. They love to see individuals try out tooling and find out what works best. Something that is working great then tends to be taken over by others: cross-pollination.
- You have part-time Scrum Masters — That’s not how Spotify does it. They have a bunch of full-time Agile coaches to help anyone to grow. By the way: Spotify’s Agile Coaches do the same work as Scrum Masters would do, but they allow the squads to find their preferred Agile way of software development.
- Squads and tribes work with roadmaps — Spotify values ***innovation over predictability and delivery over plan fulfilment.*** They work with a squad mission, product strategy and have short-term goals. It’s not that there’s no sense of direction, even without roadmaps.
- Squads focus on delivery (only) — sure, Spotify finds this important too, but they also leave room to experiment. Do you have something like hack time? Experimenting time? Spotify encourages people to spend 10% of their time on this.
- Your organisation wishes to avoid failure and as a result, is risk-averse. — At Spotify they welcome failure. “Fail fast →Learn fast → Improve fast.” Spotify recognises that when you wish to be faster than the competition that you need to experiment and

while doing so you will fail sometimes. But it’s the learning and improvement that results from this that is important. They ***value failure recovery over failure avoidance***.

Bottom Line

Don’t fool yourself and others. The Spotify engineering culture is NOT about their organisational structure. It is how people are allowed to determine what to do. It’s about autonomy. It’s about having a culture of safety. Among others. I advise you to revisit the videos so that you can experience it yourself.

There is a reason why the “Spotify Model” has become so popular. The videos are a great watch, fun and inspirational. But don’t translate them to practices and structure only.

The “Spotify Model” is victim of a bandwagon effect. The same happened with Agile and Scrum. Before you know it uninformed people create a Frankenstein monster out of it. And this is where it starts to derail.

What is also important to understand is that the videos are from 2014. Surely, they are inspirational. But know that Spotify already moved on.

“Organisations are complex, adaptive systems. Be inspired to lead genuine change in your organisation. Don’t put lipstick on a pig.” — Paddy Corry

Note: Scrum is mentioned a few times in the Spotify videos. The passages reflect Spotify’s view on Scrum then. These may or may not be in line with the Scrum Guide.

Read this article online: https://baa.tco.ac/1SwY

About Willem-Jan Ageling

Willem-Jan is co-founder, editor and writer of Serious Scrum, the fastest growing publication about Scrum and Agile on Medium and independent community for Scrum practitioners.

As a writer for Serious Scrum holding the PSM III certificate, Willem-Jan helps to raise understanding on Scrum, aiming to help people to use Scrum effectively.

At his daytime job for Ingenico, Willem-Jan is Agile Coach, facilitating the journey to improve value delivery continuously inspired by Spotify, Scrum and DevOps.

Retrospective Anti Patterns with Powerful questions

By Zeeshan Amjad

A Retrospective is one of the most important events in Scrum. It not only provides the opportunity to look back, but (as the 12th Agile principle said) tunes and adjusts the behavior to become more effective. However, still, it is one of the most misused Scrum Events after Daily Scrum. There have been a lot of anti-patterns observed and documented over time. Here we are going to discuss a few anti-patterns of a retrospective and how can we fix this with one of the most powerful coaching techniques: powerful questions.

Powerful Questions

Without asking questions, we end up with assumptions, judgments, and a lack of clarity. A powerful question is one of the most powerful tools one can have while coaching along with active listening. The most important thing about powerful questions are that they don’t have a simple answer, but they stimulate thinking and emotions. These types of questions help us to not just accept what others suggest but try to solve the problem. For example, “Do you have any other option?” will simply generate “yes” or “no” response; however, “What other options do you have?” will provoke more thinking.

It is also important to not ask leading questions. A Leading question hints towards some purposed solution or suggests the solution in the form of a question. For example, “Shouldn’t we get the approval from the management before even trying this?” or “Can we ignore the first mistake?”. This type of question does not incite thinking. Here are a few important properties of powerful questions.

- Open-ended
- Usually in the form of What/How
- Not information gathering
- Not leading to any solutions
- Lead to exploration
- Short, usually less than 7 words

Here are a few examples of powerful questions

- What about this is important to you?
- What will you do and when will you do?
- What else?
- What does fun mean to you?
- If you could do it again, what would you do differently?
- What if it works out exactly as you want it to be?
- What is stopping you?
- What works for you when you are successful at making changes?
- How do you deal with disappointment or failure?
- How important it is to get started?
- What is your purpose?
- Where do you want to be?
- What is important for you?
- What is getting in a way to achieve this?
- What do you need to do to get there?
- What can you use to remind yourself?

Anti-Patterns of Retrospective

Management attend the retrospective meeting

This is one of the most common anti-patterns where management decides to attend the Retrospective. A Retrospective is a fear-free zone for the scrum team, and when management is in the room, then they will start self-sensor themselves. The Scrum Master should have the "courage" (one of the Scrum values) to run the retrospective without any involvement of management. Here are a few powerful questions to management to fix this anti-pattern.

- What will this bring to you?
- What do you want to achieve by attending the retrospective?
- How else you can help the team other than attending the retrospective?
- How can you make the best use of your time to help the team?
- How can we give the best environment to the participants of the retrospective?
- What difference you are going to make by attending the retrospective?

There is no follow up of the last retrospective meetings

The main purpose of the retrospective is to improve the process, and to do this we not only identify the problem, but follow it up in the next retrospective. Without following up it means that we didn't validate our changes. Here are a few powerful questions for this anti-pattern.

- What action steps from the last time do we need to follow up?
- What have we accomplished since the last retrospective?
- What did we not able to accomplished since the last retrospective?

- What is our progress of each item we discussed since the last retrospective?
- What other options we can think of?
- What did we accomplish that should be celebrated?
- What do we want to do differently this time?

Discuss non-process related items in retrospective

How many times did we come across the situation where we discuss the non-process related items in the retrospective, such as resource limitation, some facility restriction, or even something even related to the cafeteria. Although these items are important, a retrospective is not a good platform for this type of discussion. In fact, we don’t have to wait for the retrospective to discuss this type of problem, rather it should be escalated as soon as possible. Here are a few questions for this type of anti-pattern.

- What do we need to focus on today?
- What do we want to achieve today?
- What changes do we want to make to improve the process?
- How would we make a long-lasting impact?
- What have you tried so far?
- What exactly do we want to achieve with retrospective?
- How does it relate to the process?

Trying to fix everything

Every environment has lots of room for improvement. There are lots of things that need to be fixed both at the process level and other levels. But with our limited resources, we cannot fix everything at once. If we are trying to fix everything, then, in the end, we may end up fixing nothing. It is important to fix a few which are most important and keep improving in every iteration. Here are a few powerful questions for this anti-pattern.

- What will we do by when?
- What obstacles do we have to accomplish this?
- What is our next step?
- What is the most important thing for us now?
- What is the implication of putting this on hold?
- What alternatives do we have?
- If you could fix only one thing, what would that be?

Too vague (non-actionable) items select at the end of the retrospective

This anti-pattern is sometimes even seen in mature Scrum teams where an excited team comes up with the retrospective item which is not clear enough or non-actionable. If the retrospective items are not actionable then we probably can’t do anything to improve and can’t follow up in the next Sprint. Here are a few powerful questions for this anti-pattern.

- What is the action plan?
- What are the possibilities?
- What about this is important to you?
- How will you know you have accomplished it?
- What do you mean by that?

- What other angles you can think of?
- What do you truly want by this?
- How do you explain this to yourself?

Read this article online: https://baa.tco.ac/1Syi

Examining Scrum Myths with Reframing

By Zeeshan Amjad

When I was a kid, I heard this story: Once upon a time, there was a king. One day the king called the astrologist to ask about his future. After doing the calculation the astrologist replied that all relatives of the king will die in front of his eyes. This made the king angry and he decided to punish the astrologist. The next day the king called another astrologist to do the same. However, after the calculation, the second one replied that the king will have the longest age among all his relatives. As a result, this time the king was pleased and awarded that astrologist. This is a perfect story for reframing. Although both astrologists articulated the same, they framed differently.

Reframing is all about changing the point of view of the situation. I can't find any other simple, yet interesting example to explain what reframing is. Though, the famous Chinese Taoist story "The old man and his horse" is also a perfect demonstration of reframing example.

Moreover, here is a renowned quote that illustrates the reframing."Don't ask kids what they want to be when they grow up, but what problems do they want to solve. This changes the conversation from who do I want to work for, to what do I need to learn to be able to do that." Jaime Casap.

Furthermore, in his famous book "Rich Dad, Poor Dad" Robert T. Kiyoska (1) discusses a series of reframing with respect to money management. The examples he presents in his book are "I can't afford it" to "How can I afford it?", "The love of money is the root of all evil" to "The lack of money is the root of all evil", "Don't take the risk" to "Learn to manage risk".

Likewise, one of my favorite facilitation books by Sam Kaner (2) discusses stimulating reframing in one chapter of his book, such as “It is a problem” to “It is an opportunity”, “We don’t have enough resource” to “We are wasting the resource we have”, “Our goal is unachievable” to “We don’t have our goal broken into realistic steps”, “We don’t have enough money” to “We haven’t figured out how to find new sources of funding”.

Reframing can be done in various ways, for example, the timeline, such as the present to future, the first person to second person, powerless to empowered, negative to positive, passive to active, part to complete, self to other, weakness to strength, individual to a community or vice versa. To summarize, here are few questions that incite the reframing.

- How would you manage a company if you will become a CEO?
- When did you achieve a goal in the past, what did you do to achieve that?
- In which context will this habit/action/response be useful?
- What will you do differently if you will be in his/her position?
- If you have all the power, what action will you perform?
- How do you prefer to see yourself in this situation?
- Where do you see yourself in a year from now?
- Where do you see your organization in a year from now?

Reframing with Respect to Scrum Myths

In any knowledge area, there are few myths disseminated around, and Scrum is no exception. Let’s inspect a few of them and come up with some questions that will help us reframe the statement from a different perspective.

- **Anyone can become a Scrum Master**
  What attributes do you want to see in the Scrum Master?

  If you will be the Scrum Master, what action will you execute to help the team?

- **In Scrum, we spend too much time in meetings**
  How can we use our meeting efficiently?

  What other ways can we do inspect and adapt if we don't have these meetings?
- **Scrum Master must resolve every problem**
  How do you want to grow your team?

  What is stopping the Development Team to not try to resolve their problems first?
- **Scrum Master must be present in the Daily Scrum**
  What value is he/she adding in the Daily Scrum?

  In what other ways can the Scrum Master help you other than being present in the Daily Scrum?
- **We get more value with higher velocity**
  If you are a customer, what does velocity mean to you?

  If you are a customer, how do you correlate velocity with value?
- **Sprint Backlog can't be changed during the Sprint**
  How do you inspect and adapt during the Sprint?

  If you discover something new that doesn't impact the Sprint goal, how do you make it transparent?
- **Only the Product Owner can write the Product Backlog Item**
  If you are still doing this, how would this impact you in the long term?
  If you are a customer, what benefits are you acquiring by doing this?
- **Unfinished Product Backlog Items (PBIs) will move to the next Sprint Backlog**
  If you have a more valuable PBI than this, what advantage will you acquire by doing this?

  Where will the PBI move if not moving into the next Sprint Backlog?

References

1. Rich Dad, Poor Dad by Robert T. Kiyosaki

2. Facilitator’s Guide to Participatory Decision-Making, 3rd edition by Sam Kaner

Read this article online: https://baa.tco.ac/1Syh

About Zeeshan Amjad

Zeeshan is a lifelong learner. With over 20 years of experience in Information Technology, he played several roles as Agile Coach, Agile Transformation Leader, Tech Evangelist, Development Manager, Project Manager, Enterprise Architect, Team Lead, Scrum Master, SAFe Agilist, Mentor, Speaker, and Author.

He enjoys Scrum to solve complex problems (not only in Software, but also in other domains) and Coaching to grow people by unlocking their own potentials.

He loves reading, writing, public speaking and have a healthy discussion about coaching and Scrum.

In his free time, he does landscape and macro photography.

To learn more visit

www.linkedin.com/in/thezeeshanamjad

The Value of an Agile Coach - or Football Coach

By Wendy Avery

As I sit here today, three days from Super Bowl LIII, I'm thinking of two things. First, will the Patriots win the Super Bowl this year? And second, after speaking with coaches who have worked with multiple companies, why do companies not understand the value of an agile coach?

It made me think about the Patriot's coach, Bill Belichick, and his job as coach to, arguably, one of the best all-time football teams. In comparison, an Agile coach for an organization that is shifting its culture. What are the parallels? What are the differences? Why do we call both of them a "coach"?

Bill Belichick doesn't actually play football. He's not on the field, catching passes or blocking the opponent. But, he's considered the foremost guide on how the team actually should play on the field. Why? Is it because he used to play football? Well, he actually did play football, during high school and college, but not professionally. He just "gets it", and is able to articulate that into guiding principles that the team can use to win games.

He doesn't tell the team what to do, he works with them on plays and strategies, and acknowledges that once they're on the field, they're going to have to think for themselves when things go sideways. He helps them remember the goals, strategies and principles of football so that they can use them as needed in those situations.

How difficult would it be to coach that team if he was also a player? Why can't the quarterback, who is the leader/coach during the play do both? I bet none of the NFL would ever consider that.

Let's consider the agile coach in the same light. She's not on the team. She doesn't necessarily know how to code but she's probably pretty familiar with the strategies that agile teams need to adopt to make their job work better. She "gets it" and is able to articulate that into guiding principles that the team can use to be successful.

She doesn’t tell the team what to do, she works with them on strategies and acknowledges that once they’re working, they’re going to have to think for themselves when things go sideways. She helps them remember the goals, strategies and principles so that they can use them as needed in those situations.

How difficult would it be to coach that team if she was also a coder and busy working instead of seeing the whole of the team and how they work together? Why can’t the scrum master, who is the leader/coach during the day to day operations of the team do both?

Hmm. I just realized that I repeated those two paragraphs, just about word for word, except for the type of coach.

I bet if we were to ask the NFL to quantify the value of their coaches in ROI and dollars, they’d be hard pressed to actually say how much BETTER the team is in comparison to if they had no coach at all. They’d just say it’s impossible to NOT have a coach, let alone employ one just for the beginning of the season. Early in the season, the Patriots were only winning 50% of the time. If you were to measure Belichick’s value 4 games into the season, Robert Kraft would certainly be justified in firing him. Who wants a coach with a 50% win-loss ratio? They would have fired possibly the coach of the next Super Bowl LIII Champions! (Or at least the second best depending on this weekend’s outcome).

All I know is that if Belichick were pulled from the Patriots right now, three days before the Super Bowl, the Patriots would lose. So why would you stop paying for agile coaches while your teams are working toward achieving the goals of your organization? Do you still expect to win? I wouldn’t bet on that game.

Read this article online: https://baa.tco.ac/1Syf

About Wendy Avery

Wendy is an Agile Change Facilitator.

Currently a SAFe Enterprise Agile Coach, Engagement Manager, and instructor for Cigna Healthcare with 30 years of experience in the Healthcare insurance industry.

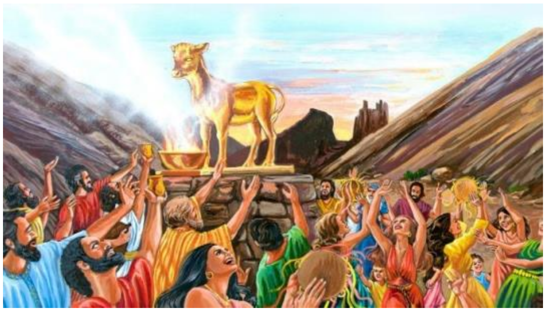
She is a leader in the development and improvement of Core processes, Governance, and Product Delivery.

To learn more visit

www.linkedin.com/in/wendy-avery

The Agile Community Embraces an Unworkable Fantasy

By Cliff Berg



I am an Agilist. The company that I co-founded in 1995, with almost 200 employees by the year 2000, adopted eXtreme Programming (XP) that year. My 2005 book High-Assurance Design interpreted secure and high reliability engineering practices in an Agile context, proposing a concrete answer to the question, “How can organizations build highly reliable and secure systems using Agile methods?” Since then I have helped more than ten large organizations to move to Agile and DevOps approaches.

But the Agile community is very off the rails. It teaches people an ideal model that does not work. This model is like a sacred calf: its elements are discussed by Agile coaches as if they are truisms. One challenges it at great peril: if one speaks against the model, one can be labeled an “Agile doubter”, or someone who is “not really Agile” - a mark of death in the Agile community. The Agile community has historically been very closed minded, and not able to debate Agile ideas in an open-minded and healthy way. My editor at Pearson described it as “insular”. It was DevOps that burst through the Agile doors and humbled the community into accepting more open debate.

Indeed, writing about this is always a personal risk for me: those who don’t agree will refer to me as “that anti-Agile” guy. I know because it has happened. They are wrong though: I was an early adopter of Agile, and am fully committed to the core ideas of Agile; and I have been immensely successful in implementing Agile - but not in the typical manner of adopting out-of-the-box standard Agile practices.

My approach is more of a DevOps approach: to always look at the uniqueness of each situation and consider it as a *system*: Consider what behavioral changes will achieve the business goal, whether it is shorter time to market and building the *right* things, or higher quality and building things that *can be trusted to work*. I always start from a business driver perspective and draw from Agile ideas, Lean ideas, behavioral therapy ideas, and many others, rather than trying to force-fit the sacred calf idealist model on every situation.

Let's look at what the Agile sacred calf model is.

If one takes a training course in Agile "basics", one will be taught the Agile Manifesto - a brilliant and timeless document - but then one will be taught the Scrum process - even though Kanban is arguably a much better fit for today's DevOps methods. In addition, one will be told the following falsehoods:

1. Self-organizing autonomous teams are the best way to organize programmers.
2. Managers are an "anti-pattern"; ideally there should be no hierarchy, and no one should have explicit authority. All authority and all leaders should emerge through self-organization.
3. No individual should be accountable: only entire teams should be accountable.
4. Failure is the best way to learn.
5. One should always trust the team, even if you have never worked with the team before.
6. All development teams should be feature teams that work across the entire set of technology stacks of the product.
7. Anyone should be able to work on anything.

8. An open team room is the best arrangement, so that people can collaborate most easily and will benefit from “osmotic” communication.
9. More collaboration is always better, and verbal communication is always best.

The Agile Manifesto does not say any of these things. Let me repeat that:

The Agile Manifesto does not say any of the nine things in the list.

While the Manifesto mentions elements of some of these things, it never says “do only this” or “always do that”. Nearly every statement in the Agile Manifesto is written to imply a tradeoff, so that one can decide what approach is best for a situation.

The above nine axioms form a model that is taken as truth by most of the Agile community at large, despite lots of evidence that the model is wrong. People who *really understand Agile* don’t make the above nine absolutist claims, but those people are a small percent, and they know to not openly criticize the sacred calf model, unless they are speaking privately with someone else who truly understands Agile.

Books debunking the above model abound, although they do so in a careful and politically correct way, tiptoeing around the model and claiming instead that one needs to “augment” typical Agile practices. A recent one is *Rethinking Agile: Why Agile Teams Have Nothing To Do With Business Agility,* by Klaus Leopold. In that book, Leopold expertly connects the dots on the leadership gaps that tend to exist in organizations that create autonomous self-organizing teams and then expect that they will be Agile as a result.

Despite the attention-getting title, the book does not directly criticize the nine-axiom Agile model, but instead shows how the adoption of that model by organizations leaves huge gaps, and Leopold explains how to fill those gaps.

Agilists (including me) are quick to point out that one cannot simply “insert” Scrum or self-organizing teams: one must “change the organization’s culture”. But what does that mean? The very same people who stand by the above nine prescriptions seldom have a good answer for what a culture change means. They say platitudes like, “One must get managers to embrace failure”, and “The organization needs to learn to experiment”.

Fine, but that kind of advice is too vague, and abstract advice that ends in a platitude tells me that someone does not really know what they are talking about - that they learned something in a book but have not actually solved that problem.

I have helped more than ten organizations “go Agile” or “go DevOps”, and I have seen every permutation of this. I have also seen common patterns that are usually present. These are,

1. Organizations keep their existing market research and product definition methods, which have a long lead time, and which “hand off” product requirements to “IT” to build.
2. Organizations dismantle old style program-level or product-level leadership, but don’t replace it with an Agile equivalent.
3. Organizations try to “roll out” Agile, treating it like a process change, when in fact it is mostly a learning journey.
4. Organizations fail to put sufficient emphasis on the technical side of Agile, which today is “DevOps”.

Many organizations try to “roll out” Agile by adopting SAFe, or some variation of that. This actually helps somewhat because it fills some gaps pertaining to item 2 above, and possibly even impacts 1 - although that is less common. But the huge Product Increment planning exercise that SAFe prescribes is unnecessary, in my opinion.

The biggest leadership gaps that I usually see pertain to *technical* leadership. Organizations might adopt a “Lean portfolio” process (which SAFe recommends) and a product “release train” structure, but they fail to address how the many components of their product integrate. They dismantle their traditional approach of having a separate QA team focus on “system integration testing” but they don’t replace it with an Agile and DevOps equivalent, which ideally is frequent, cadence-based on-demand automated integration testing, backed by “shift-left” local integration testing.

They often put a development manager in charge of the product teams, but the manager has heard the message that teams should be “self-organizing and autonomous”, and so he or she fails to step in and provide the leadership that is needed. The result is that the teams tend to operate in a highly independent manner, and fail to adequately address integration across teams. The sacred calf model is then the direct cause of the ensuing dysfunction.

Agile methods include having regular retrospectives, and most organizations do this at a team level because Scrum prescribes it, but they fail to institute it at a product or program level. Agilsts will point out that this is a symptom of “not having an Agile culture”, because an Agile culture would surely think to have retrospectives at every level. But when an organization is trying to adopt Agile methods, they cannot magically “become an Agile culture”. They need guidance on what practices to install. Eventually those practices will foster more Agile thinking, but that change in thinking will take a long time.

And this is where the Agile community, in general, is confused: the claim is that organizations must change their culture to be Agile. Yet the reality is that *to change an organization’s culture, one must first change behavior - not the other way around*. Culture change follows behavior change, and if the behavior change is maintained for long enough, the change becomes self-sustaining and becomes the new culture. (In the behavioral therapy community, this change lifecycle is described by the “transtheoretical model”.)

In other words, it is unreasonable to expect organizations to come up with Agile approaches just by learning the Agile Manifesto: they need help in defining new processes and practices, that over time will become the new normal; and in the course of those behavioral changes, they need Agilists to interpret the changes for them, to help them to see the behavioral changes through an Agile lens - that way, their thinking begins to change. But the behavioral changes must come first. People go from concrete to abstract, not abstract to concrete.

Thus, it ***is*** the Agile community's fault for popularizing the overly simple model of autonomous teams and not helping organizations to anticipate all of the new processes and practices that they will need so that those teams will work together. Telling them that they should have a collection of self organizing teams and that they should "start to think Agile" is not enough. The teams need explicit methods for organizing their collective behavior: they will not adequately self organize around a product or value stream, and most development managers will not know how to stimulate the required level of cross-team collaboration: they will stand back and watch, and expect the teams to magically start collaborating - because that is how they have been told Agile works - but it doesn't happen.

A failure to have retrospectives at a product or program level blocks the improvements that Agile might bring: product feature cycle time cannot be substantially reduced, nor can the rate of escaped product defects, unless one focuses on end-to-end improvement, spanning the many teams of a product or value stream.

Improving individual teams will not unlock the benefits of Agile. And Agile coaches, who tend to be non-technical, shy away from discussing technical issues during retrospectives, so don't rely on them to help you improve your end-to-end technical flow. They will tell you that you need to identify dependencies and reduce your "WIP", but they won't have any ideas about how to manage the dependencies so that integration happens frequently and tests at all levels of integration are sufficient. (See this article on dependency management, and this article series on defining a testing strategy.) Nor do most teams know the CI/CD practices that are needed for rapid integration - they don't know what they don't know: thus, the adage "trust the team" fails to account for the DevOps mentoring that teams need.

The fact is, someone needs to be accountable for the product level technical flow. Otherwise, there is no product level leadership around things like end-to-end testing. The leadership style should be servant leadership, so that it is collaborative and enabling - not autocratic and fear based. But someone needs to have their eye on that ball all the time: the integration level issues that come up are complex and numerous, and they will totally block substantial improvement if you don't address them.

A person who has accountability needs to have actual authority: accountability without authority is a terrible situation to be in. But a servant leader will not exercise their authority most of the time: they will save it for rare occasions when they need to make a decision about something that cannot be resolved collaboratively, such as the need to re-organize some teams. The rest of the time, they are watching for things that are falling through the crack, and making sure that issues are being discussed and resolved - mainly by asking questions and making suggestions, but sometimes - hopefully rarely - by insisting that something be addressed, and asking the team to trust them.

So much for the falsehood that one should have autonomous self-organizing teams! What teams need is servant leadership, at all levels: the team level, the product level, and the program or value stream level.

Then there is the claim by Agilists that failure is the best way to learn. *No one likes to fail.* A market facing failure is never a good thing. The idea of "fail early, fail often" pertains to controlled experiments, which might be market-facing if they need to be. SpaceX uses the "fail early, fail often" technique: they build lots of rocket engines and run them to failure, to learn how they fail. They even do that for whole rockets. But the last thing they want is a failure during a mission!

Instead of the very misleading "fail early, fail often", the maxim should be, "try things - experiment a-lot and expect many experiments to fail - that way you will learn, and you won't fail when it counts".

The other items in the list of nine Agile falsehoods can be summarized as extremes that contain good ideas but that usually don't work in their extreme form - the form that is usually explained during an "Agile Basics" course and which is advocated by a large percent of Agile coaches.

Self-organization? Sure - to a degree, but better to have a true servant leader who can help the team to organize and to have high quality dialectic discussions. Anyone can work on anything? Maybe - but sometimes business-critical core microservices need to be maintained by a single team in order to ensure their design cohesion. Open team room? No one actually seems to like that, but maybe the "coffeehouse" format is a good compromise - thus we come full circle to the old "caves and commons" idea. In-person communication is always best? For simple things yes, but for very complex issues, some people communicate better by writing their thoughts down first: that is a matter of personality.

And so on.

Extremes don't work, except in extreme circumstances. Most organizations are not in extreme circumstances. The Agile Manifesto was about balance and judgment - not about extremes. Those who teach the nine falsehoods do not really understand Agile: they are cargo-cult Agilists. All those nine ideas are extreme variants of good ideas, but the extremes don't make any sense most of the time.

Each of the nine maxims should be taught as an idea to be considered, and nothing more, with the caution that the right approach depends on the situation.

It is time to dismantle the false ideal model of extremes that is being taught in Agile Basics courses and that does not actually work, and bring judgment and reality back into Agile.

About Cliff Berg

Cliff has helped more than ten organizations adopt Agile and DevOps methods, working with leadership and teams to move the needle.

Cliff has experience with Agile and DevOps in a wide range of contexts, from large multi-product digital platforms to embedded systems.

Cliff works with senior leadership to understand goals and help to define transformation strategies. He also supervises coaches to provide a second pair of eyes and make sure that work is aligned and effective.

To learn more visit

www.linkedin.com/in/cliffberg

Warring Tribes into Winning Teams
Improving Teamwork in Your Data Organization

DataOps and Relational Coordination for Chief Data Officers

By Christopher Bergh and Eran Strod

If the groups in your data-analytics organization don't work together, it can impact analytics-cycle time, data quality, governance, employee retention and more. A variety of factors contribute to poor teamwork. Sometimes geographical, cultural and language barriers hinder communication and trust. Technology-driven companies face additional barriers related to tools, technology integrations and workflows which tend to drive people into isolated silos.

The Warring Tribes of the Typical Data Organization

The data organization shares a common objective; to create analytics for the (internal or external) customer. Execution of this mission requires the contribution of several groups shown in Figure 1. These groups might report to different management chains, compete for limited resources or reside in different locations. Sometimes they behave more like warring tribes than members of the same team.

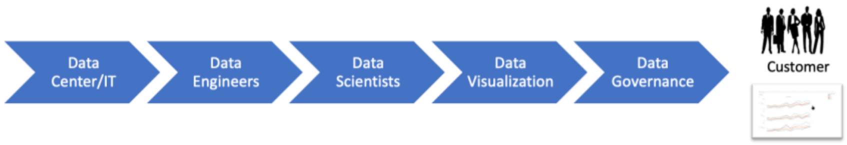


Figure 1: Delivery of analytics (the value chain) to customers requires contributions from several groups in the data organization

Let's explore some of the factors that isolate the tribes from one another. For starters, the groups are often set apart from each other by the tools that they use. Figure 2 is the same value chain as above but reconstructed from the perspective of tools.

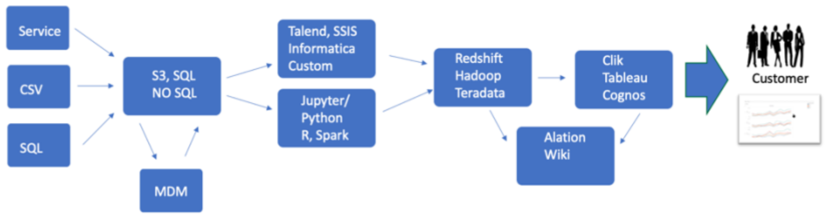


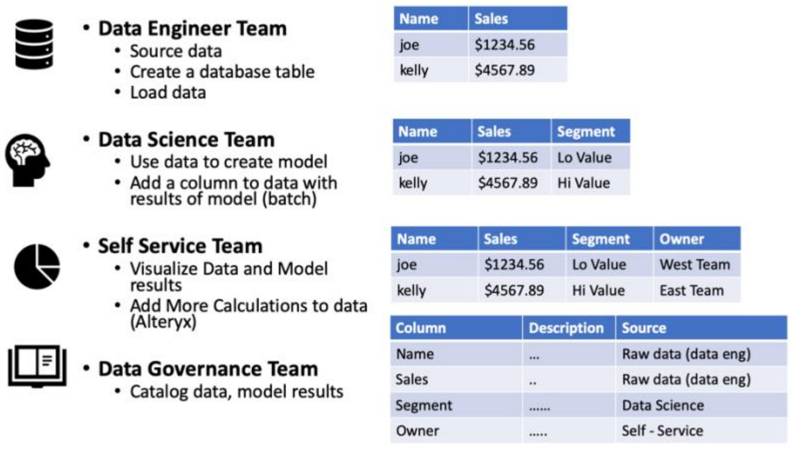
Figure 2: The value chain shown from a tools perspective

To be more specific, each of the roles mentioned above (figure 1) view the world through a preferred set of tools (figure 2):

- Data Center/IT — Servers, storage, software
- Data Science Workflow — Kubeflow, Python, R
- Data Engineering Workflow — Airflow, ETL
- Data visualization, Preparation — Self Service tools, Tableau, Alteryx
- Data Governance/Catalog (Metadata management) Workflow — Alation, Collibra, Wikis

The day-to-day existence of a data engineer working on a master data management (MDM) platform is quite different than a data analyst working in Tableau. Tools influence their optimal iteration cycle time, e.g., months/weeks/days. Tools determine their approach to solving problems. Tools affect their risk tolerance. In short, they view the world through the lens of the tools that they use.

The division of each function into a tools silo creates a sense of isolation which prevents the tribes from contemplating their role in the end-to-end data pipeline. The less they understand about each other, the less compelling the need to communicate about actions taken which impact others. Communication between teams (people in roles) is critical to the organization’s success. Most analytics requires contributions from all the teams. The work output of one team may be an input to another team. In the figure below, the data (and metadata) build as the work products compound through the value chain.



- **Data Engineer Team**
  - Source data
  - Create a database table
  - Load data

Name	Sales
joe	$1234.56
kelly	$4567.89

- **Data Science Team**
  - Use data to create model
  - Add a column to data with results of model (batch)

Name	Sales	Segment
joe	$1234.56	Lo Value
kelly	$4567.89	Hi Value

- **Self Service Team**
  - Visualize Data and Model results
  - Add More Calculations to data (Alteryx)

Name	Sales	Segment	Owner
joe	$1234.56	Lo Value	West Team
kelly	$4567.89	Hi Value	East Team

- **Data Governance Team**
  - Catalog data, model results

Column	Description	Source
Name	...	Raw data (data eng)
Sales	..	Raw data (data eng)
Segment	......	Data Science
Owner	.....	Self - Service

Figure 3: Each group adds unique value to analytics. In most cases, the work of one group is an input to the next group.

In many enterprises, there is a natural tendency for the groups to retreat into the complexity of their local workflow. In figure 4, we represent the local workflow of each tribe with a directed-acyclic graph (DAG).

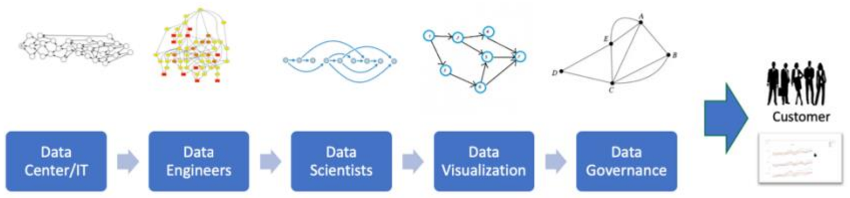


Figure 4: Work groups tend to focus on the complexity of their local workflow

It is too easy to overlook the fact that the shared purpose of these local workflows is to work together to publish analytics for end-customers.

Other Factors that Increase Group Isolation

Group isolation is also induced by platforms, release cadence and geographic locations. The example below shows a multi-cloud or multi-data center pipeline with integration challenges.

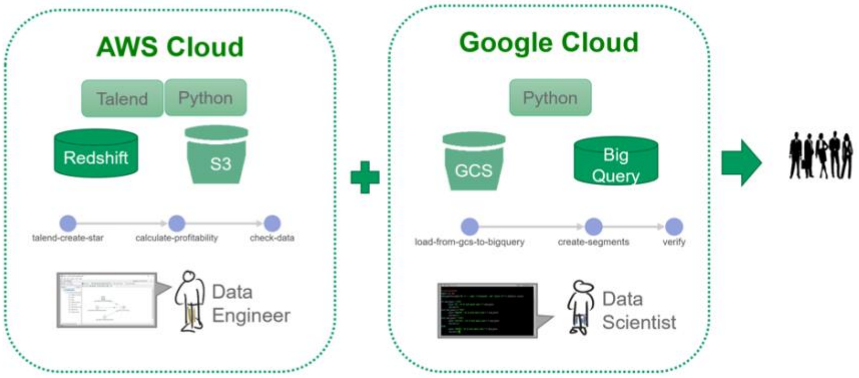


Figure 5: Multi-cloud or multi-data center pipelines with integration challenges

The two groups managing the two halves of the solution have difficulty maintaining quality, coordinating their processes and maintaining independence (modularity). Group one tests part one of the system (figure 6). Group two validates part two. Do the part one and two tests deliver a unified set of results (and alerts) to all stakeholders? Can tests one and two evolve independently without breaking each other? These issues repeatedly surface in data organizations.

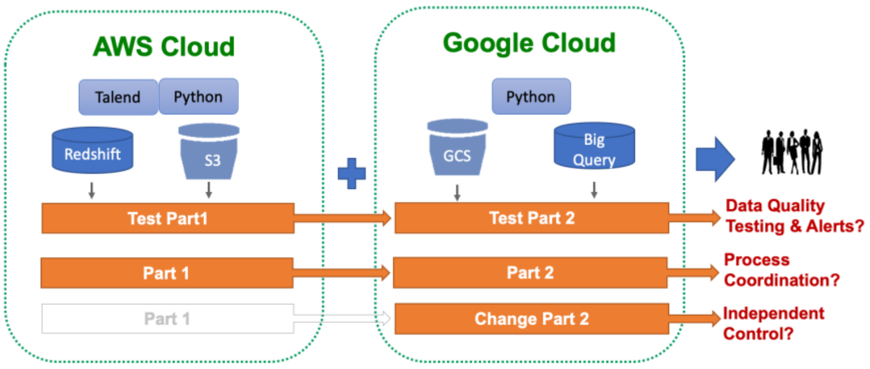


Figure 6: Integration challenges of multi-cloud or multi-data center solutions

In another example, assume that two groups are required to work together to deliver analytics to the VP of marketing. The home office in Boston handles data engineering and creates data marts. Their iteration period is weekly. The local team in New Jersey uses the data marts to create analytics for the VP of Marketing. Their iteration is daily (or hourly).

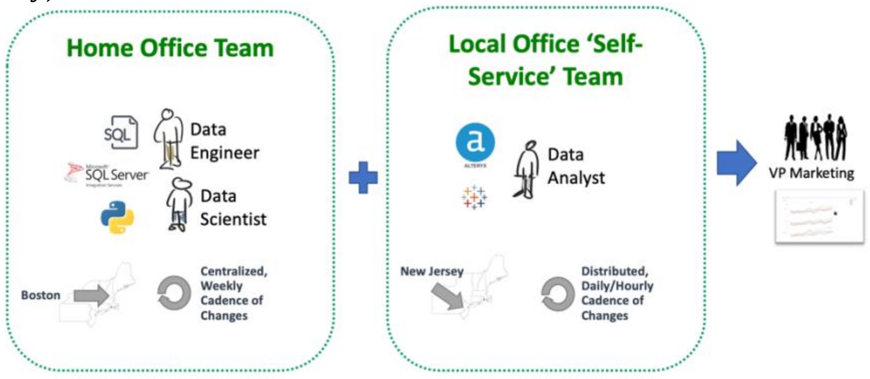

Home Office Team
SQL
SQL Server
Data Engineer
Data Scientist
Boston
Centralized, Weekly Cadence of Changes
Local Office 'Self-Service' Team
Data Analyst
New Jersey
Distributed, Daily/Hourly Cadence of Changes
VP Marketing

Figure 7: Issues related to multi-team workflows

One day, the VP of Marketing requests new analytics (deadline ASAP) from the data analysts for a meeting later that day. The analysts jump into action, but face obstacles when they try to add a new data set. They contact data engineering in Boston. Boston has its own pressures and priorities and their workflow, organized around a weekly cadence, can't respond to these requests on an *"ASAP"* basis.

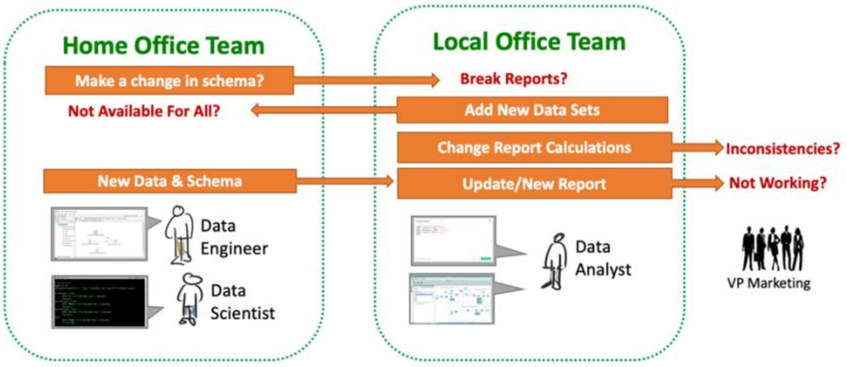


Figure 8: Challenges with multi-team coordination

The home office team in Boston finally makes the needed changes, but they inadvertently break other critical reports (figure 8). Meanwhile, out of desperation, the New Jersey team adds the required data sets and updates their analytics. The new data sets are only available to New Jersey, so other sites are now a revision behind. New Jersey's reports are inconsistent with everyone else's. Misunderstandings ensue. It's not hard to imagine why the relationship between these groups could be strained.

These challenges may seem specific to data organizations, but at a high level, everything that we have discussed boils down to poor communication and lack of coordination between individuals and groups. As such, we can turn to management science to better understand the problem and explore solutions.

Relational Coordination

Strip away the technological artifacts from the situations described above and you are left with an organization that cannot foster strong role relationships and communication between employees. These challenges are not unique to technology-driven organizations. Many enterprises across a wide variety of industries face similar issues.

For those who don't remember, the airline business in the 1980s and 1990s was brutally competitive, but during this same period, Southwest Airlines revolutionized air travel. By the early 2000s, they had experienced 31 straight years of profitability and had a market capitalization greater than all the other major US airlines combined. Brandeis management professor Jody Hoffer Gittell investigated the factors in Southwest Airlines' performance and, back in 2003, published a quantitative, data-driven analysis shedding light on Southwest's success.

Dr. Gittell surveyed the major players in the airline industry and found a correlation between key performance parameters (KPP) and something that she termed Relational Coordination (RC), the way that relationships influence task coordination, for better or worse. "Relational coordination is communicating and relating for the purpose of task integration — a powerful driver of performance when work is interdependent, uncertain and time constrained."

In her study, higher RC levels correlated with better performance on KPPs, even when comparing two sites within the same company. Since that time RC has been applied in industries ranging from healthcare to manufacturing across 22 countries.

One common misconception is that RC focuses on personal relationships. While personal relationships are important, RC is more concerned with the relationship of roles and workflows within the organization. RC studies how people interact and exchange information in executing their role-based relationships.

Relational Coordination can be expressed as characteristics of relationships and communication:

Dimensions of RC	Low RC	High RC
Relationships	Functional Goals Exclusive Knowledge Lack of Respect	Shared Goals Shared Knowledge Mutual Respect
Communication	Infrequent Delayed Inaccurate Finger-pointing	Frequent Timely Accurate Problem-solving

Members of the “Low-RC” organization express their goals solely in terms of their own function. They keep knowledge to themselves and there may be a tendency for one group to look down upon another group. Inter-group communication is inadequate, inaccurate and might be more concerned with finding blame than finding solutions. As expected, the “High-RC” organization embodies the exact opposite end of this spectrum.

“High-RC” team members understand the organization’s collective goal. They not only know what to do but why, based on a shared knowledge of the overall workflow. Everyone’s contribution is valued, and no one is taken for granted. There is constant communication, especially when a problem arises.

At this point you may be thinking: “OK fine, this is all *touchy-feely* stuff. I’ll try to smile more and I’ll organize a pizza party so everyone can get to know each other." Maybe you should (smiling will make you feel good and parties are fun after all), but our experience is that the good feeling wears off once the last cupcake is gone and the mission-critical analytics are offline.

How do you keep people working *independently* and *efficiently* when their work product is a *dependency* for another team? How can one team reuse the data or artifacts or code that another team produces?

For most enterprises, improving RC requires foundational change. You need to examine your end-to-end data operations and analytics-creation workflow. Is it building up or tearing down the communication and relationships that are critical to your mission? Instead of allowing technology to be a barrier to Relational Coordination, how about utilizing automation and designing processes to improve and facilitate communication and coordination between the groups? In other words, you need to restructure your data analytics pipelines as services (or microservices) that create a robust, transparent, efficient, repeatable analytics process that unifies all your workflows.

Building a High-RC Enterprise Using DataOps

DataOps is a new approach to data analytics that applies lean manufacturing, DevOps and Agile development methods to data analytics. DataOps unifies your data operations pipeline with the publication of new analytics under one orchestrated workflow.

- **Robust**– Statistical process control (lean manufacturing) calls for tests at the inputs and outputs of each stage of the data operations pipeline. Tests also vet analytics deployments, like an impact review board, so new analytics don't disrupt critical operations.
- **Transparent**–Dashboards display the status of new analytics development and the operational status of the data operations pipeline. Automated alerts communicate issues immediately to appropriate response teams. Team members can see a birds-eye-view of the end-to-end workflow as well as local workflows.

- **Efficient**– Automated orchestration of the end-to-end data pipeline (from data sources to published analytics) minimizes manual steps that tie up resources and introduce human error. Balance is maintained between centralization and decentralization; the need for fast-moving innovation, while supporting standardization of metrics, quality and governance.
- **Repeatable**– Revision control with built-in error detection and fault resilience is applied to the data operations pipeline.
- **Sharable and Chunkable**– Encourage reuse, by creating a services oriented architecture (SOA) for your team to use together.

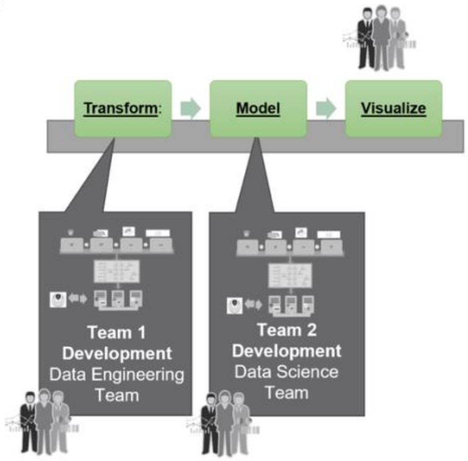


Figure 9: DataOps is a task coordination and communication framework that uses technology to break down the barriers between the groups in the data organization.

It may help to provide further concrete examples of a DataOps implementation and how it impacts productivity. Some of these points are further explained in our blog DataOps in Seven Steps:

- **Data Sharing**– data sources flow into a data lake which is used to create data warehouses and data marts. Bringing data under the control of the data organization decouples it from IT operations and enables it to be shared more easily.

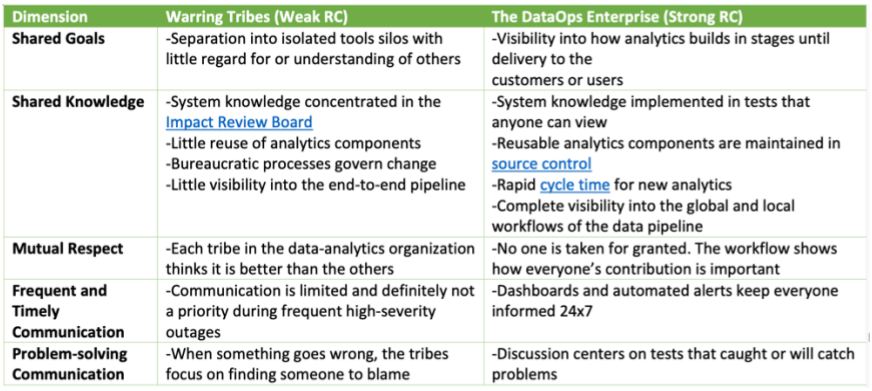
- **Deployment** of code into an existing system — continuous integration and continuous deployment of new analytics, leveraging on-demand IT resources and automated orchestration of integration, test and deployment.

- **Environment** startup, shutdown — With computing and storage on-demand from cloud services (infrastructure as code), large data sets and applications (test environments) can be quickly and inexpensively copied or provisioned to reduce conflicts and dependencies.

- **Testing** of data and other artifacts — Testing of inputs, outputs, and business logic are applied at each stage of the data analytics pipeline. Tests catch potential errors and warnings before they are released so the quality remains high. Test alerts immediately inform team members of errors. Dashboards show the status of tests across the data pipeline. Manual testing is time-consuming and laborious so it can't be done in a timely way. A robust, automated test suite is a key element in continuous delivery.

- **Reuse** of a set of steps across multiple pipelines — Analytics reuse is a vast topic, but the basic idea is to componentize functionalities as services in ways that can be shared. Complex functions, with lots of individual parts, can be containerized using a container technology (like Docker).

We have seen marked improvements in analytics cycle time and quality with DataOps. It unlocks an organization's creativity by forging trust and close working relationships between data engineers, scientists, analysts and most importantly, users. DataOps is a task coordination and communication framework that uses technology to break down the barriers between the groups in the data organization. Let's look at the DataOps enterprise from the perspective of Relational Coordination.



Dimension	Warring Tribes (Weak RC)	The DataOps Enterprise (Strong RC)
Shared Goals	-Separation into isolated tools silos with little regard for or understanding of others	-Visibility into how analytics builds in stages until delivery to the customers or users
Shared Knowledge	-System knowledge concentrated in the Impact Review Board -Little reuse of analytics components -Bureaucratic processes govern change -Little visibility into the end-to-end pipeline	-System knowledge implemented in tests that anyone can view -Reusable analytics components are maintained in source control -Rapid cycle time for new analytics -Complete visibility into the global and local workflows of the data pipeline
Mutual Respect	-Each tribe in the data-analytics organization thinks it is better than the others	-No one is taken for granted. The workflow shows how everyone's contribution is important
Frequent and Timely Communication	-Communication is limited and definitely not a priority during frequent high-severity outages	-Dashboards and automated alerts keep everyone informed 24x7
Problem-solving Communication	-When something goes wrong, the tribes focus on finding someone to blame	-Discussion centers on tests that caught or will catch problems

Conclusion

Technology companies face unique challenges in fostering positive interaction and communication due to tools and workflows which tend to promote isolation. This natural distance and differentiation can lead the groups in a data organization to act more like warring tribes than partners. These challenges can be understood through the lens of Relational Coordination; a management theory that has helped explain how some organizations achieve extraordinary levels of performance as measured by KPPs. DataOps is a tools and methodological approach to data analytics which raises the Relational Coordination between teams. It breaks down the barriers between the warring tribes of data organizations. With faster cycle time, automated orchestration, higher quality and better end-to-end data pipeline visibility, DataOps enables data analytics groups to better communicate and coordinate their activities, transforming warring tribes into winning teams.

You can read more about DataOps by downloading the DataOps Cookbook, our free book explaining DataOps in detail and how you can get started immediately.

Read this article online: https://baa.tco.ac/1Sxj

About the Co-Authors

Christopher Bergh is a Founder and Head Chef at DataKitchen where, among other activities, he is leading DataKitchen's DataOps initiative.

Chris has more than 25 years of research, engineering, analytics, and executive management experience.

Previously, Chris was Regional Vice President in the Revenue Management Intelligence group in Model N. Before Model N, Chris was COO of LeapFrogRx, a descriptive and predictive analytics software and service provider. Chris led the acquisition of LeapFrogRx by Model N in January 2012.

Twitter: @ChrisBergh

Eran Strod works in marketing at DataKitchen where he contributes to the DataOps blog and the DataKitchen blog. Eran previously served as Director of Marketing for Atrenne Integrated Solutions (now Celestica) and held product marketing and systems engineering roles for Curtiss-Wright, Black Duck Software (now Synopsys), Mercury Systems, Motorola Computer Group (now Artesyn), and Freescale Semiconductor (now NXP), where he was a contributing author to the book "Network Processor Design, Issues and Practices."

Twitter: @EranStrod

Sign the DataOps Manifesto
Download the DataOps Cookbook and Visit DataOps Blog

Flowing through time: the need for a certain slowness

By Sonja Blignaut

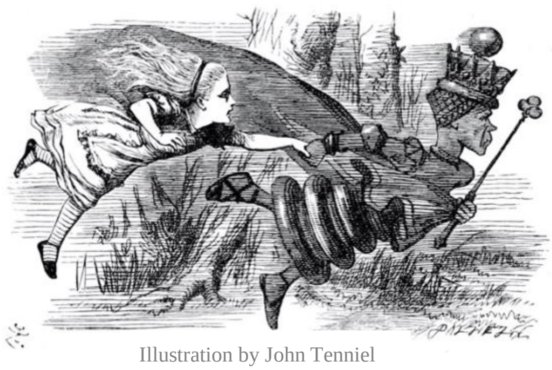


Illustration by John Tenniel

It is exactly when one would think that being fast is what is required that slowness proves its worth." — Paul Cilliers

We don't often think of our lives as temporal and spatial flows. However, as we transition from birth to death, we age, we learn, we flow in and out of relationships, we move from place to place: we flow through time. Life is one great flow. It is curious then that we seem so pre-occupied with speed. We want everything to be not only fast but better yet instant. We want everything "on-demand".

In business, in the so-called VUCA world, the mantra is also "go faster". When I ask senior leaders why they've embarked on Agile Transformation processes, the answer I hear most often is "we need to be faster". Agile teams (mostly Scrum) are under continuous pressure to improve their "velocity", i.e. their ability to increase the amount of work they can deliver in the same amount of time.

As strategic agility is becoming the goal for most organisations, and Agile practices or New Ways of Working (NWOW) are spilling into the rest of our organisations, questioning this pre-occupation with speed becomes even more critical. In one organisation I work with, the need for speed is driving very disfunctional behaviours. Management has mandated that special teams need to deliver high-value products every 12 weeks. In a culture where failure is not an option and managers rule by fear, team members cannot raise concerns about unrealistic expectations and timelines. So they send in "the cannon fodder", contractors who are expendible. This organisation has an extremely high churn in contractors, and many refuse to work their anymore. This need for speed has made an already dysfunctional culture, toxic.

While the efficient flow of value through our teams and organisations is necessary, the implied association of speed with efficiency has become problematic. We can be highly efficient and very fast, but if we never take time to slow down to reflect and learn, we will simply bring about our end faster.

Chronos vs Kairos time

At the inaugural Complexity in Human Systems symposium earlier this year in DC, Alicia Juarerro reminded us of the importance of time in complex systems. She highlighted the difference between Chronos (chronological or sequential) and Kairos (opportune) time. Chronological time is agnostic of context as it inexorably moves along. Kairos time, defined as right, critical, or opportune moments, is inherently linked to context. For each unique context, there will be unique Kairos moments. This begs the question: Are we missing critical Kairos moments as we seek to get more and more done, and speed up our flow through Chronos time?

How fast we flow through time then, should also be contextual, i.e. sometimes we do need to go fast, but sometimes we need “a certain slowness”.

I borrow that phrase from Paul Cilliers, whose article, on the importance of a certain slowness, has long been a favourite of mine. I think it is useful to reflect on some of his writing here (in italics):

“In his novel Slowness, Milan Kundera (1996) uses the metaphor of somebody riding on a motorcycle as being constantly in the present. Speed and the demands of the machine reduce his horizon to something immediate. Someone walking, however, is moving at a pace that allows for a much wider horizon. The stroll unfolds in time in a way that opens

up reflection about where we are coming from and where we are going to as we walk. This theme of both the past and the future being present in a meaningful experience of the present ... the argument for a meaningful temporality — that is, something slower — will be made here from the perspective of the dynamics of complex systems. "

"It should be stated up front that ***there is no argument against an appropriate fastness****. A stew should simmer slowly, but a good steak should be grilled intensely and briefly.* ***The argument is against unreflective speed, speed at all cost, or, more precisely, against speed as a virtue in itself: against the alignment of "speed" with notions like efficiency, success, quality, and importance."***

If we fall into the trap of unreflective speed, we may end up being very efficient at delivering products that offer no value. Also, making speed a primary measure of performance can have unintended consequences. Take the idea of team velocity again: Strathern's variation of Goodhart's law states that *when a measure becomes a target, it ceases to be a good measure.* When speed becomes our target, quality and learning suffers.

In call centres, value statements like "customer first" are plastered all over the walls. But typically, call center agents are measured on efficiency and speed — how many calls they can handle in the shortest possible time. In a bank I work with, agents are given a performance target of limiting customer calls to 2,5 minutes. So if you're the agent who gets the problematic customer, when it comes to 2 minutes 15 seconds, what will matter more: customer service or your performance bonus? Again, the focus on speed and efficiency leads to ineffectiveness and unintended consequences.

While innovation is often linked to speed and being disruptive and "first to market", innovation and delivering sustainable business benefit is often short-circuited by the pressure of unrealistic timelines. I recently spent time with the Executive leadership of a new unit, created (or incubated) within a large corporate with an innovation mandate. They were expected to completely disrupt the very established and highly competitive short term insurance market and given a very short time horizon in which they had to show a profit. This pressure completely paralysed them. Instead of taking the necessary risk, the unrealistic targets made them overly careful. When they did take a chance, time pressure often led to them abandoning their ideas prematurely before they had time to prove their value. As any farmer knows, seeds take time to sprout, grow and bear fruit. The same is true for business ideas. This short-termism, driven by the need to show quarterly results undermine innovation and sustainable growth in many (if not all) public companies.

The tyranny of the moment

"The way in which contemporary society lives in an eternal present, or what Eriksen (2001) calls the "tyranny of the moment," is made possible, and augmented, by the surge in technology, especially computer and telecommunication technology. We are instantaneously in contact with everybody, everywhere. Not only has the distinction between home and the workplace collapsed, but also the distinction between work time and private or leisure time. It is expected of many of us to be available, always and everywhere. This state of affairs may have been less detrimental if it did not also demand instant response. The very reason for mobile phones and email lies in the fact that immediate response is possible. ***It is in this "immediate" that the main problem lies. There is less and less time for reflection. Reflection involves delay, and in a cult of speed, delay is unacceptable."***

As part of the culture work I do in organisations, I gather and make sense of hundreds of anecdotes of people's experiences. In virtually all my projects, there are emergent themes linked to burn-out, lack of work-life balance and inhumane work environments. An acquaintance who works in a clinic that is situated near the head offices of three of our large banks told me that they deal with so many burn-out cases that they've named these symptoms after the originating banks ("BankA-itis" looks slightly different than BankB-itis"). And this happens in organisations whose number one espoused value is typically something along the lines of "we care about our people".

Another friend is a business coach who works with very senior C-suite executives. Recently she was told by two clients, who don't know each other, how disturbing they found it that they had to "pay for empathy". What kind of dehumanised organisations are we creating in our when efficiency is all that matters?

As a response to increasing work pressures, many have turned to mindfulness and meditative practices. Predictably though, these also need to fit into our speed-obsessed culture: if a meditation or exercise isn't short enough to fit into our busy schedules, we're not interested. The same with learning, if something can't be done in a one-day (or even better half-day) workshop, most companies aren't interested.

This reminds me of the famous marshmallow experiment where children's capacity for delayed gratification was tested: they could have one marshmallow immediately, or two later if they were able to wait. Nowadays I think most of us would fail that test: we consume fast food; we have movies- on-demand and binge-watch our favourite series because we simply cannot wait a week between episodes. It seems we're all living in the tyranny of the moment while ironically becoming less and less present in it.

On memory and complex systems

"An important aspect of complex systems, one that certainly complicates our understanding and modeling of such systems, is their temporal nature. Complex systems unfold in time, they have a history that co-determines present behavior and they anticipate the future. Moreover, as we know at least since the work of Prigogine, the behavior of complex systems is not symmetrical in time. They have a past and a future that are not interchangeable. This being "situated in time" does not always receive adequate attention in our analysis of complexity.

The central notion at stake when we talk of time and complexity is that of "memory." Memory is the persistence of certain states of the system, of carrying something from the past over into the future. It is not merely

the remembering of something in the past as if belonging to that past, it is the past being active in the present.
If one characterizes memory as the past being carried over into the future, it follows that the future can only be anticipated in terms of the memory of the system. Anticipation is not, or at least should not be, simply an extrapolation of the present. It is a complex, non-linear process that tries to find some trajectory, some way of "vaulting" from that which has already been experienced to that which has to be coped with. The quality of the anticipation is a function of the quality of the memory. A more varied, richer, deeper, and better-integrated memory will open up more sophisticated anticipatory capabilities.

The point is that a system that has carefully accumulated the relevant memories and experiences over time will be in a better position to react quickly than one that is perpetually jumping from one state to the other."

Like Lewis Carroll's Red Queen's Race (illustrated in the headline image by John Tenniel), many organisations seem stuck in a cycle of churn without ever achieving any real change.

"Well, in our country," said Alice, still panting a little, "you'd generally get to somewhere else — if you run very fast for a long time, as we've been doing."

"A slow sort of country!" said the Queen. "Now, here, you see, it takes all the running you can do, to keep in the same place. If you want to get somewhere else, you must run at least twice as fast as that!"

— *Carroll, Lewis: Through the Looking-Glass and What Alice Found There, Chapter 2*

I often ask workshop participants how many restructurings they've been through in the last year. Most of their answers are something along the lines of: "I've been through 2 in the last 6 months, never mind a year!" In organisations where a habit has formed of continuous change (seemingly) for the sake of change, people become severely "change fatigued". Also, the changes that are made a never given time to make a real impact. Before we can see if a restructuring had the desired impact, we restructure again. If we take the gardening analogy again, the absurdity becomes apparent. No gardener would plant seedlings and rip them to plant new ones before they have time to flower or bear fruit! This constant churn, reminiscent of Carroll's Red Queen sacrifices not only forward momentum but also the accumulation of deep memory that facilitates change resilience in our organisations. It is also extremely demoralising. I believe this is a big contributor to the shockingly high statistics of staff turnover, absenteeism and disengagement in our organisations.

Distinguishing signal from noise

"Memory is information from the environment that has been filtered, it is that which has been interpreted — by the memory already sedimented in the system — as significant. The identity of the system is, in some sense, its collection of dynamic memories. The implication is that the system cannot reflect, or act on, everything that is going on in the environment at a given moment. If that were the case, the system would always be merely a reflection of its environment and would have no identity of its own. In order for it to be a system at all, a system that has its own identity, that can react to the environment and not merely mirror it, a certain hysteresis is required. The system must be slower than its environment.

The system has to hang on to some aspects with a certain tenacity: not let go of them too quickly. There is risk involved in this, of course. The system has to invest resources in this process. It cannot maintain everything; it has to select. If too many of the wrong things are carried over it will impair the system's performance. However, if not enough is carried over, it will also fail.

To put it in slightly different terms: The system has to find a way to discriminate between information and noise. If it follows every trend in its environment, it will also be following noise. If it reacts too slowly it will only follow the low-frequency trends, which may also be just *noise.**The system must be stable enough not to be buffeted around by every fluctuation, and it must be flexible enough to be able to adapt when necessary. “***

Jack Welsh famously said: *“If the rate of change on the outside exceeds the rate of change on the inside, the end is near.* We often use quotes like this to justify the need for agility (I must admit I’ve used this particular one myself). But if it is true that a system can only survive and maintain its identity through constant reflection and wise selection of what to respond to and what not to; changing as fast or faster than the environment is not a sustainable strategy. If we never take time out to think deeply about where we’ve come from, where we are now and where we are going, we run the risk of losing the very DNA that is our competitive advantage. Also, if leaders in organisations have no time to read, learn and cultivate diverse networks, how will they know what to pay attention to? John Boyd, a brilliant military strategist and the creator of the OODA loop stated that it is not only speed that matters, but sometimes “accurate perception” is just as important, if not more so. This means that advantage comes not only from speed, but the ability to Observe and Orient to the current context in a way that is most coherent with reality. While speed ultimately does matter, especially in disrupting our opponent’s ability to orient, acting faster based on incoherent orientation will lead to our own irrelevance or destruction faster.

*“It must be stressed again that the argument for a certain slowness is not a conservative argument. A certain amount of conservation is a pre-requisite for a system to maintain itself, of course. The important point, to which we shall return, is that a “slow” strategy is not a backward-looking one. **If a somewhat slower tempo allows a system to develop a richer and more reflective memory, it will allow the system to deal with surprises in its environment in a better way. The argument of slowness is actually an argument for APPROPRIATE SPEED.** There is no objective or immediate rule for what that speed is. If anything, it is a matter of experience, and experience (as Aristotle urged) has to be gained, it cannot be “given” in an immediate way. It is experience that determines which piece of meat should be fried quickly and which should simmer slowly in the stew. She who fries everything quickly will*

only have something nice to eat now and then, and then purely by chance.

We need to cultivate a healthy respect for the Restraint/Action paradox

Leaders and decision-makers in complex systems continuously need to navigate the tension between the risks associated both with practising restraint and taking action. On the one hand, if the context requires it, one may need to consciously practise restraint and create space that allows for the emergence of ideas, trust, opportunity, and even epiphany to loosen tangled problem knots. Like sowing seeds and waiting for the seedlings to sprout and grow, it takes time to allow emergence to unfold. On the other hand, one needs the courage to take action in a mist of uncertainty because, in a complex system, the consequences of our actions are never entirely predictable. No matter how good our knowledge, there is never an objective “right” decision. Being conscious of (and comfortable with) this paradox is critical to successfully fostering and practising adaptive leadership.

References:

Cilliers, F. P. 2006. On the importance of a certain slowness. Emergence: Complexity and Understanding 8:106–113.

Rogers, K. H., R. Luton, H. Biggs, R. Biggs, S. Blignaut, A. G. Choles, C. G. Palmer, and P. Tangwe. 2013. Fostering complexity thinking in action research for change in social–ecological systems. Ecology and Society **18**(2): 31. http://dx.doi.org/10.5751/ES-05330-180231

About Sonja Blignaut

Sonja is a thinking partner to decision-makers and leaders who need to find their way through complexity and uncertainty.

She has been working in the field of complexity for almost two decades and counts many blue chip companies as clients.

Sonja works across industries and disciplines and is a sought after speaker and consultant.

To learn more visit

www.morebeyond.co.za
medium.com/@sonjablignaut

Does this need to be said?

By Kevin Brinley

I love finding great coaching lessons in unexpected places. A few years ago, I watched a comedy special by Craig Ferguson where he posed these 3 questions one should ask others before speaking.

Does this need to be said?
As a coach, not everything we think of needs to be addressed. Perhaps the team didn't discuss a story during the standup. Given the current development of the team, the day of the sprint, and the importance, consider if the omission is important enough to call out.

Does this need to be said by me?
A coach doesn't have to be the one to point out an omission or problem. The best agile coaches are teaching there team to work together, and this includes holding each other to high standards and ensuring everything gets done.

If a team is far enough along in their growth, it may be detrimental for the coach to call out very basic advice or poiut out obvious problems. But, if the coach waits long enough, someone else may speak up. My best coaching moments have come from when others are applying the lessons I've taught them and calling out the problems I've taught them to look for.

Does this need to be said by me now?
The final question ponders the timing of coaching intervention. Is now the right time to act? Or, would it be better to save the conversation for another time, such as the sprint retrospective?

Conclusion
Coaching is based on soft skills, which can be learned from anywhere. Keep your eyes and ears open, and you may find some great tips from the unlikeliest of sources.

Read this article online: https://baa.tco.ac/1Sy3

About Kevin Brinley

Kevin is a Servant Leader experienced in leading teams on their journey to "being agile".

He is a relentless driver of improvement, for himself and others.

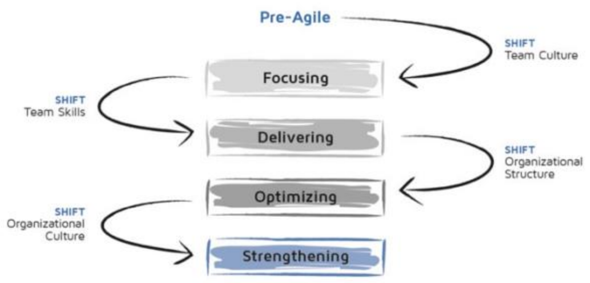
Through teaching, coaching, and mentoring, he aims to make the world a better, wiser place, 1 person at a time.

To learn more visit

www.linkedin.com/in/kevin-brinley-48564012

Yes, Cross-functional Teams, but Real Ones!

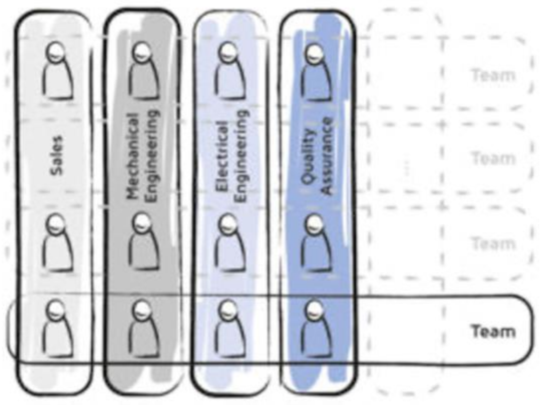
By Jutta Eckstein and John Buck



If you start with Agile, one of the first things you typically do is come up with a team. And yes of course, the team will be cross-functional. But what's actually meant with cross-functionality? At first people (in software) understand this as different kind of developers (e.g. backend & front end experts) and also testers are working together on the same features / user stories. And, there is even some business knowledge in the team through the product owner. That is fine, but this is only a start. In fact, if you check with the Agile Fluency™ Model this is the first shift for your agile journey and is called the 'Team Culture Shift'.

Yet, if a company decides an agile team isn't enough, it will invest in a different shift. And some of those other shifts require implementing real cross-functional teams. Meaning, the whole team has the full business expertise, knows the market, can even disrupt the market and isn't waiting for some person (e.g. the Product Owner) to decide on priorities. It also means the team fully understands the company's business and has a holistic view of it, knowing its contribution to the company's value stream. Thus, a cross-functional team overcomes the limitations of the classic stovepipes in organizations.

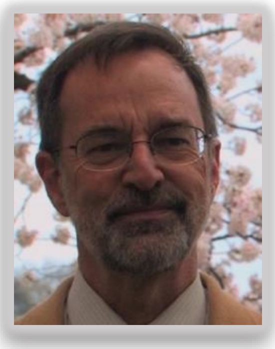
Like a lot of other companies, one of our clients, an insurance company, is currently facing the challenge of digitalization. They now understand that digitalization means software is their product and no longer insurance policies. In this company, the teams in software are using Scrum. Next to software there are the business experts and the mathematicians who have the domain knowledge about insurance. The next leap is bringing these different units together (not only through the interface of the Product Owner) to form teams that are actually as knowledgeable about business, contracts, mathematics, sales, marketing, and everything else that is relevant for the company’s value stream as they are in software. The leap means for the Scrum teams that they can’t “hide” behind their backlog or the Product Owner, but need to explore and learn about the market, their customers, and the company’s value stream themselves. Basically it means that now the agile teams’ focus is beyond software.



Other examples: a large charity, a university, a construction company. What would cross-functional teams look like in one of those organizations? First, the organization might have classic stovepipes. Schools of a university would, for example, focus on music, or English, or economics, or agriculture, and so forth. They would publish in different professional journals and be invited to attend totally different professional conferences. Thus, a cross-functional team at a university might include professors from widely different schools as well as representatives from the administration. But, what would they focus on? Their customers of course! Their customers would include students, the various professions and industries they serve, and also the other members of the university. They would produce insights gained by combining their studies. (Note: this output would be something other than the classic survey course for freshmen that might look at a topic from the standpoint of various disciplines but rather a real synthesis of those disciplines.)

As can be learned from the Agile Fluency Model, it is fine to start with cross-functional teams that span the different software expertise and support these teams with some business know-how. Yet, if you are thinking of implementing company-wide agility, the expertise of real cross-functional teams spans beyond software and comprehends the whole value stream. In order to do so, ask, does the team really mirror our overall organization in its assignment and authorized scope of action? Are additional skills or more empowerment needed to make the team mirror the whole?

About the Co-Authors



John Buck is president of GovernanceAlive LLC in Washington, DC, USA, showing companies how to survive and thrive disruption by helping them create conditions for self-organization. He is collaborating with Fujitsu's advanced software lab to develop Weaver, an app that helps meetings go better – in-person, online, and asynchronous.

His leadership experience includes managing large IT projects. His diverse clientele span the globe and include plastics manufacturers, schools, colleges and universities, long-term care facilities, co-housing groups, NGOs, food producers, and software companies.

www.linkedin.com/in/john-buck

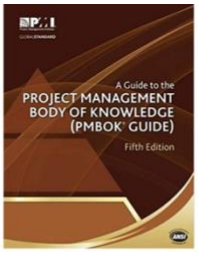
Jutta Eckstein works as an independent coach, consultant, trainer, author and speaker. She has helped many teams and organizations worldwide to make agile transitions. She is experienced in applying agile processes within medium-sized to large, distributed mission-critical projects and has written about her experiences. She holds a M.A. Business Coaching & Change Management, a Dipl.Eng. Product-Engineering, and a B.A. in Education.

She is a member of the Agile Alliance (having served on the board of directors from 2003-2007) and a member of the program committee of many different American, Asian and European conferences, where she has also presented her work.

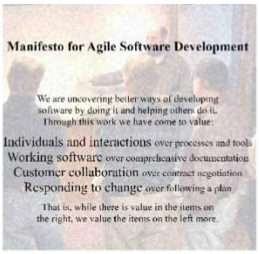
www.linkedin.com/in/juttaeckstein

PMI: Please Get Out of Scrum

By Stacey Christiansen



≠



Manifesto for Agile Software Development

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.

As it stands today, the PMBOK is in direct conflict with the foundational principles of the Agile Manifesto.

The Project Management Institute is a 50-year-old organization dedicated to the practice of project, program, and portfolio management. Their mission as stated on their website is to "… work on your behalf to advance the project management profession worldwide."

The group has also authored the PMBOK (Project Management Body of Knowledge). It's the bible for project management. Seriously, that sucker is 756 pages long!

As the authority on project management, the PMI offers 8 different certifications. One of these credentials is the PMI-ACP (Agile Certified Practitioner).

Full Disclosure: I have the fancy credentials. Along with being a CSM, CSPO, and CSP, I am also a PMP and PMI-ACP certified.

I have nothing against the PMI as an authority and promoter of project management best practices. The group provides invaluable information to and for project/program/portfolio managers. The certification programs are very well done in terms of preparation materials, testing relevant skills, and experience requirements. The PMI is very, very good at project management.

PMI Entered the Agile Space for the Wrong Reasons

The PMI-ACP is a money play. Pure and simple. The PMI saw the ever-increasing adoption of agile frameworks, particularly Scrum, and wanted to get in on the action. More certifications, more training, more members equal more money. Yes, even non-profits need to make money.
The absence of a project manager role in Scrum also causes concern for the PMI. As Scrum continues to be adapted over traditional waterfall methods, what are all the PMP's going to do as project manager roles are replaced with Scrum roles? The risk to the PMP revenue stream is huge.

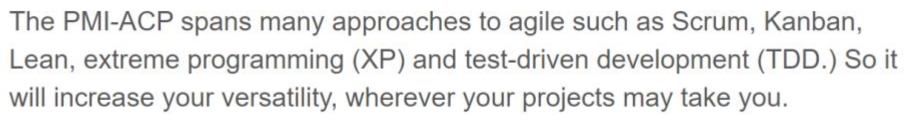
According to the 2018 PMI Annual Report, $239 million USD were collected worldwide for membership fees, certifications, and book sales. The report also boasts more than 6 million copies of the PMBOK in circulation. With the PMP being their flagship product, the PMI should be worried.

PMI-ACP — A Unicorn is Born

Just to make sure we're all on the same page, **there is no project manager role in Scrum.** The "agile project manager" role that emerged from the PMI-ACP certification is a unicorn.

In an organization practicing Scrum, the functions of a traditional PM are spread between the Scrum Master, the Product Owner, and the Development Team. And some traditional PM functions are simply no longer necessary.

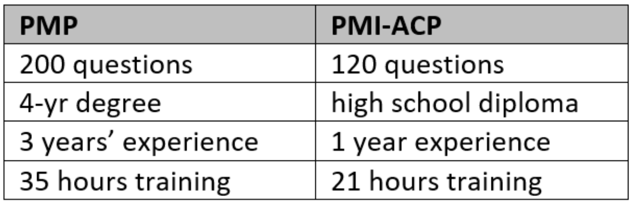
An agile project manager role causes confusion and conflict within organizations trying to transition to a Scrum framework. No one is quite clear on who is responsible for what.


The PMI-ACP spans many approaches to agile such as Scrum, Kanban, Lean, extreme programming (XP) and test-driven development (TDD.) So it will increase your versatility, wherever your projects may take you.

Wow! That's some certification!

Really. I'm mean seriously. One exam and the newly certified person is supposed to have a solid understanding of five different agile frameworks? Enough said.

If that doesn't make you cry agile tears, take a look at the requirements to sit for the PMP vs the PMI-ACP:



PMP	PMI-ACP
200 questions	120 questions
4-yr degree	high school diploma
3 years' experience	1 year experience
35 hours training	21 hours training

Pre-requisites to sit for each exam.

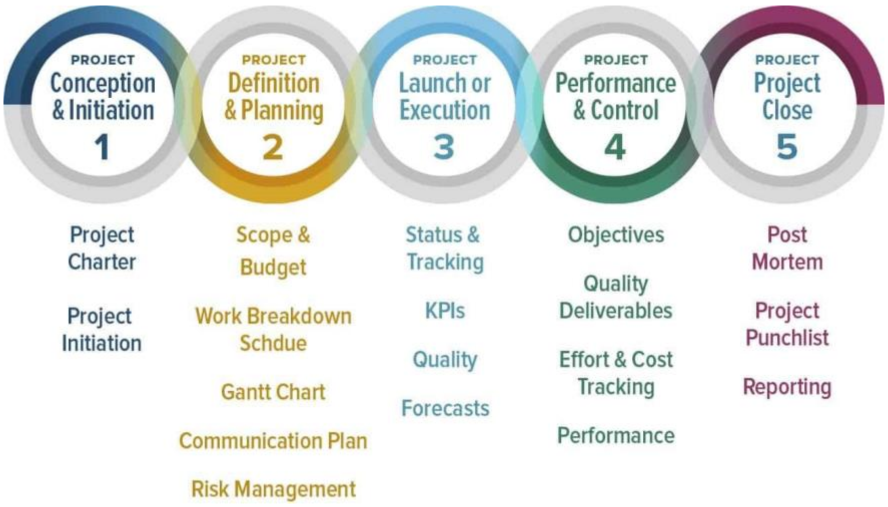
I have three Scrum certifications, been in formal classroom training, attended multi-day Scrum or Agile conferences, and have spent countless hours studying Scrum. I was part of the leadership team responsible for a successful transition from waterfall to Scrum. I have 20 years' experience in software product development, 10 years in real-world Scrum environments. And I have so much more to learn. After all that, I'm a Scrum neophyte.

Scrum requires experienced professionals. The Scrum Master and Product Owner roles are not entry-level roles to be filled by high school graduates with little experience.

The qualifications to become PMI-ACP certified show that even the PMI isn't taking the certification seriously. The PMI-ACP was an ill-conceived reaction to a sudden shift in market demand.

Foundational Conflicts Between the PMI and Scrum

The conflicts between project management principles and Scrum go far beyond process, roles, and semantics. The principles taught and evangelized by the PMI explicitly undermine the very principles that make Scrum effective. Straddling the fence between the two worlds causes much of the failure of Scrum implementations in organizations today.



PMI Stages of Project Management

Let’s look at the first two stages of PMI project management and see where we find clashing principles.

1 Conception & Initiation

Step one in PMI-style project management is to create a project charter. Often, the executive sponsor of the project physically or digitally signs this document. This is an executed contract intended to lock in the objectives on day one of the project.

Conflict: This violates two of the four values in the Agile Manifesto:

Customer collaboration over contract negotiation.
Just the psychology of signing the project charter immediately creates a division between stakeholders and the project execution team. It's us against them.

Responding to change over following a plan.
Keep in mind, many projects span months and years. To think the project objectives would not evolve during this time is naïve and/or irresponsible. Particularly in the field of software products, responding to competitors and new technologies in a timely manner is critical.

2 Definition & Planning

Or, as I call it "How to Set Your Team Up for Failure from the Beginning." Please note the vocabulary in the name of the field "software development."*Development* is the creation of new things; it's inherently exploratory.

Conflict: Requiring full project definition up front violates the Scrum principle of empirical process control. It defies the value of responding to change, which will also likely lead to a breach of the principle of establishing a sustainable development pace. The team will inevitably fall behind schedule and be asked to donate their nights and weekends for the last month of the project to "catch up."

Asking a development team to estimate the effort in something they've never done just doesn't make sense. Add to that the timing of this effort — not one line of code has been written but developers are expected to understand every nuance of the project and accurately estimate it. The benefits of iterative development will never be recognized.

I could continue going through stages 3, 4, and 5, but you get the point. The very beginning of a traditional project management approach obliterates the basic foundation on which Scrum teams are built. How can you possibly continue on from that and expect to be successful?

Where Does a PMO Fit in an Agile Organization?

I really wish I had a great answer. I don't. I've often wondered if an SMO (Scrum Management Organization) wouldn't be a better fit. While I hate that name (it's also taken: Service Management Organization), I do like the idea of a single organization within a company evangelizing Scrum and the values and principles outlined in the Agile Manifesto, ensuring proper training is made available and helping perfect the execution of Scrum practices.

Ideally, the PMI and a Scrum authority organization would come together and figure out how these two bodies can align in a spirit of support rather than conflict. I think there's room for both.

Regardless, the current proliferation of PMI-ACP certifications and "agile project managers" put us on a terrible path for future Scrum teams. The best we can do right now is to educate company leaders on the critical differences between the PMBOK, Scrum practices, and the Agile Manifesto. And, to those hiring for or applying to "agile project manager" positions — buyer beware.

Read this article online: https://baa.tco.ac/1Sya

About Stacey Christiansen

Stacey grew up professionally in a startup, which explains her diverse background in software engineering, product management, digital marketing, and business process improvement.

Stacey is passionate about breaking down silos, building cross-functional bridges, and driving "soft sand" out of business process to help teams run faster and jump higher.

As a writer and editor for the Serious Scrum publication, Stacey enjoys sharing her experiences to help others improve, and working hands-on to get new Scrum teams started off right.

Outside of work, Stacey is an avid equestrian and is always searching for the next great book.

To learn more visit

www.medium.com/@staceychristiansen

Ask Different Questions: Building Your Agile Leadership Skills

By Lorena Connolly

The Situation

It’s Monday morning. You’ve just come off a long weekend of work. You were up all night Saturday deploying the product and then had multiple configuration issues to address on Sunday.

Now, it’s Monday and you need to be in the office for an 8 AM meeting to hear what the new Agile Consultants have come up with for your organization. You stagger over to the kitchen thinking there isn’t enough coffee in the world to get you through this presentation.

You’re probably thinking, *“They can’t possibly have anything meaningful to present. They don’t know what it’s like here.”*

One of your employees texts you letting you know there seems to be a performance issue with the new release. You wish you could skip over this two-hour presentation, get to your desk, and do some real work.

The consultants launch into their presentation, talking about improving the business by organizing into small cross-functional teams and working in small batch sizes.

What makes them think that will work here? If they really want to improve quality, why don’t they get the Sales people to stop making commitments before the technical team has had a chance to say what is feasible?

Now, they are going on about empowerment and leaving decisions to the team.

What makes them think that the team is equipped to make those decisions? These are decisions that you normally make! And they want you to change the way you manage, too? After all the literal blood, sweat, and tears of frustration you've shed to get the deliveries out on time, all the late nights, weekends, all of a sudden, you're the problem?

Suddenly, through this whirlwind of thoughts, your boss's boss asks you, "You're critical to this being a success, what do you think?"

You look around and see all the smiling faces and expectant looks. Through a dry throat and clenched jaw, you say, "Sounds great, I will get behind it."

Silently, you wonder if it's time to update your resume.

Sound familiar?

When an organization has made the decision to adopt an agile way of working, they often only acknowledge the impact to teams. However, there is also a significant impact on leaders and middle management. This shift requires them to change the way they have been managing and delivering past success to the organization–often without a seemingly tangible explanation as to why.

As focus is placed on the teams and helping them work in new ways, managers are often overlooked. This is unfortunate, as these people are often influential, well-respected, and valued leaders.

It's no wonder their initial reaction is one of fear or skepticism. So, as mid-level managers, how do we combat our reactions to change and take advantage of the situation we're in?

How To Start Building Your Agile Leadership Skills

As you saw in the situation above, the questions this leader asked themselves could only lead that poor soul to one conclusion: “This is bad, I’m the victim, and I’m out.” Even though this leader is experiencing the pain of long cycle times, tough production deployments, and production issues, they remain skeptical and on the defense. This is because human nature leads us to tolerate the pain we know and reject/avoid the pain we don’t know.

Don’t get me wrong. An organization should acknowledge a transitioning leader’s emotions as valid and provide the support, path, and plan for this new leadership style. However, at the same time, the transitioning leader needs to work through these emotions and concerns and *accept* the support, path, and plan. But, how?

Change the way you approach the problem.

Ask Different Questions

Much easier said than done, I know. I’ve been there–more than once. However, learning to ask questions that open up new possibilities is key to unlocking leadership potential and successfully building your Agile leadership skills.

In the situation above, we saw an example of how a leader initially thought about the change of an Agile transformation and the perceived threat to their job. In the table below, I provide some examples of how to re-frame these internal questions in order to take advantage of and benefit from the situation

What makes them think this will work here?	What company-specific research or assessment was done to come up with this approach? What methodology was used? What data backs up their findings?
Why don't they get the Sales team to stop pre-committing?	How can we provide more accurate information to Sales to help them understand the time needed for similar efforts? How can an Agile environment help us partner more effectively with the Sales team? Why does Sales feel the need to pre-commit? Is there an underlying trust issue with our predictability?
What makes them think that the team is equipped to make those decisions?	How do we plan to equip the team to make these decisions? How can we handle design decisions? How do I need to prepare/support my team?
Why do I have to change the way I manage?	If I change the way I manage, what will I be losing? What will I be gaining? Is the gain worth the pain? What do I specifically need to change? Who can help me change? Who will keep me honest?
Do I leave the details to the team?	How can I equip/support the team? What methods can I use to stay sufficiently engaged to coach the team without directing them?

I’m the problem?	What am I not seeing that the executives are seeing? Who can give me open feedback on the business challenges we have? How is my team contributing to those challenges? Have I gotten so used to “the way that we do things” that I’m missing something? Are late nights and weekends really how I want to lead my team? How can I show that is not sustainable? How and who can I work with to set proper expectations? What do I own here?
How do they come up with this stuff?	What are they seeing that I am not? Have they helped other organizations with similar issues? What were the results? Is there a way out of this pain?
Am I no longer critical to the success of the organization?	In this new way of working, in what new ways am I expected to contribute? Does that interest me? Does it open other opportunities? Am I ready to contribute in that way? If not, who can help me get ready? Is there an opportunity to create a new/additional role for myself?
What do they mean by you focusing on “higher-value” items?	Are there additional skills, tools, challenges in which I can obtain a level of mastery? Are there higher-value problems I can solve for the organization? How might that help my career/job satisfaction?

Conclusion

The truth is, this change will require you to call on leadership skills you may not have yet–or that you didn't know you had. You will have to think differently than you have in the past to navigate this very real and challenging situation. Additionally, your team will be watching you very closely to see how you respond and will take their lead from you.

As a leader in your company, you understand the value of being responsive to the business needs, improving business outcomes, and improving the work life of your team. Use these re-framed questions and new Agile Leadership skills to help you work through your concerns and map a path to meet the needs of the business. Once you have your own answers, use this technique to help a struggling team member or colleague work through the transition.

Read this article online: https://baa.tco.ac/1Sy9

Packing for the Yellow Agile Brick Road
5 Tips for A Successful Transition to Agile

By Lorena Connolly

Happy days. Your organization has made the commitment to experiment with Agile or to go whole hog and do a "rip-the-band-aid-off" transformation. Now, you're learning about stand-ups, backlogs, and self-organizing teams. During this transition to Agile, you may be wondering, "What are we really in for?"

Indeed, you may feel like Dorothy in the Wizard of Oz. You've been spun around in a tornado of terminology and change. Now, you've landed in an unfamiliar place that bears little resemblance to the place you've been working at over the past few years. You really aren't in Kansas anymore.

It is important that you learn the principles, values, and practices of an Agile way of working. Even after you complete your first training class, keep doing that. That is your Yellow Brick Road.

This article is about the more elusive, yet essential things, you'll need to take with you on your Agile journey. The help you need to battle the Wicked Witch of The Way We've Always Done Things and the Flying Monkeys of unplanned work, stories that carry over, and big batch work.

Let's get back home (though it may look different) and to a place where you feel comfortable and confident again. Here are 5 tips for a successful transition to Agile:

1. The Lion: Courage

The first thing you will need is **courage**. I just finished an engagement with a client, and as I said my good-byes, I wanted to leave them with a final shot in the arm to sustain them in the days ahead. We sent them medal stickers with the words “Change Takes Courage” emblazoned on them. A reminder that courage is foundational to work in an Agile way.

You need courage to:

- Ask the questions that break you away from the way we’ve always done things
- Not succumb to the “fire of the day”
- Say “No, that’s not going in the backlog”
- Tell your boss to take his or her request for work to the Product Owner
- Have healthy conflict and end with a better solution
- Speak up and give your opinion, no matter who is in the room
- Give and receive needed feedback
- Be self-reflective
- Let go of long-held management, process, and project control beliefs and work differently
- Change and transition to Agile

Courage is at the very core of all Agile transformations. I see so many organizations where courage has been all but snuffed out by intentions both well-meaning and not-so-well-meaning.

Whoa! If I only had some courage! The good news is that you do. It’s there within you. It may only be a little spark, but fan it with small acts of courage and, in no time, you'll be marching into the Wicked Witch of The Way We've Always Done Things castle to vanquish her.

Agile transformation will require courage from leaders, stakeholders, and teams. Be courageous and celebrate courage when you see it. When you see someone holding back you can use a permission-giving phrase to encourage that person to take the leap. Perhaps by saying "Put 'em up! Put 'em up!," in your best Cowardly Lion's voice!

2. The Scarecrow: Mindset

In Agile circles, we talk a lot about how Agile is not a process, methodology, or lifecycle. We don't get the benefits Agile has to offer if we "do Agile." We get the benefits when we ARE agile. That means shifting the way we **think** about how to work, how to interact with one another, and how to lead.

More than memorizing the Agile Manifesto, principles, and values we need to think and act in concert with them. Only then can we transform and realize the business outcomes that Agile helps us achieve.

In practice, this mindset shift looks like this:

- Turning away from the tyranny of the urgent and moving toward sustainability and predictability
- Resisting going back to familiar (comfortable) processes and methods to forge new or repair relationships to have productive interactions
- Moving from managing through status and reports to encouraging and supporting teams to meet their commitments, allowing working software to be the most important measure of progress

- Enabling our teams by reaching out to our peers to solve long-standing systemic issues that have been a source of slowness all along
- As a leader, resisting the urge to solve a problem you know how to solve and allowing the team to wrestle with it for a bit and solve it on their own
- Instead of business and technology working separate from one another, because the other "just doesn't get it," both teams are being transparent with themselves and each other
- Delivering value instead of delivering "deliverables"

Whoa! If I only had an Agile brain! The good news is that you do. It's in you, it just may need a little rewiring.

Be intentional in your thinking and decision-making. Ask yourself and others, "Does what we are doing align with Agile principles and values?" Truly explore the worst case scenario of not doing something, rather than responding to the urgent and often inaccurate, "we just have to do it."

As a leader, before responding, ask yourself, "Will my response empower my team and encourage ownership?"

3. The Tin Man: Compassion

The transition to Agile is challenging from a tactical perspective. It's even more challenging from an interpersonal perspective. In my opinion, we don't talk about or equip people sufficiently to deal with what we each experience during an Agile transformation. We teach how to write a user story–relatively easy. But teaching someone how to remain confident when their job, who they work with, and how they interact is significantly changing? That's much more difficult.

As leaders and teammates, we need to have heart and give grace when people and teams:

- Mess up
- React emotionally
- Try and fail
- Need support in their new roles/jobs
- Need to adjust

Whoa! If I only had a heart! You know you do. You just need to have the courage to let it show.

Be willing to have conversations about “humanness” during your Agile transformation. By showing and accepting a little vulnerability, you will be amazed at the trust you can build even in this time of change.

4. Toto: The Friend

Toto was Dorothy’s stalwart companion. He would not abandon her and jumped off a moving bicycle to get back to her. Likewise, Dorothy would not abandon Toto and ran away from home to protect him.

On this journey, you’ll be tested. You will doubt yourself. Expect it. You are trying new things! Find yourself a Toto to be your sounding board, your cheerleader, your sympathetic ear, and your voice of reason.

Return the favor for your friend–and don’t let your friend get carried off by a flying monkey!

5. The Wizard: A Coach

The Wizard was ultimately able to help the Lion, the Scarecrow, the Tin Man, and Dorothy realize they already had what they needed within them. They had the power to change their circumstances, they just needed some coaching. The Wizard could see what was inside them, even when they were not yet able to. He didn’t fix their predicaments, he showed them how to fix them on their own.

As you trek down your Agile Yellow Brick Road, you will need a Wizard. Someone to see what you can not yet see. Someone who has traveled this road many times and will help you solve your predicaments and challenges on your own. Someone to guide you away from the poppy fields and flying monkeys to keep you on course.

The Wicked Witch of the Way We’ve Always Done Things is extremely powerful and hard to resist. You need someone who has battled her before and won.

Conclusion

I hope these 5 tips help your Agile transformation go a little smoother, and I wish you a successful journey.

If you are looking for a Wizard, give Agile Velocity a call. Or you can learn more about our approach to building lasting business agility on our Transformation Services page

Read this article online: https://baa.tco.ac/1Sy4

About Lorena Connolly

Lorena is an experienced Agile and leadership coach who helps organizations improve performance and employees gain more enjoyment out of their work.

With many years spent driving toward deadlines and seeing the organizational and human cost, advocating for an agile way of working and leading became the most sensible and humane thing for her to do.

Her combination of real-world delivery, senior leadership, and agile experience helps Lorena empathize with coaching clients as they are often in the difficult position of maintaining delivery while transforming.

As an agile coach, certified CSM, SAFE SPC 5, certified Path To Agility®facilitator, holder of multiple ITIL 4 certifications, and certified trainer for Managing Transitions©model, Lorena pulls from multiple sources to pragmatically help her coaching clients.

Lorena's permission-based approach to coaching meets clients where they are and is based on patience, consideration, and kindness. She strongly believes that the kindest act is to be honest and direct about impediments to organizational or personal improved performance.

A life-long learner, Lorena is a coach, teacher, blogger, speaker, questioner, listener, supporter, and cheerleader.

Zombie Agility and 3 Antidotes to Eradicate It In Your Organization - Part 1

By Erik Cottrell

Preface:
The author assumes the reader has a working understanding of zombies and is open to the possibility they are real.

(Hint: They are.)

Disclaimer:
“Every analogy breaks down eventually.” – Marc Escobosa,1998.
“Zombie Agility is here. I seen it.” – Erik Cottrell, 2019.

Zombie Agility takes over when the goal of an organization is simply to “go Agile.” Whenever Agile becomes the goal, expect a zombie epidemic. Surprise! It doesn’t just happen at the start of an Agile transformation.

You know your organization has Zombie Agility when…
- Your Agile teams are half-animated, joyless, and listless
- You’re Agile! But you still aren’t seeing desired business results
- Your colleagues observe lackluster results and disengaged people

Thriving Agility is good. You achieve this when the business goal is to deliver more value, with more predictably, and better learning. In other words, the goal is continuous improvement for customers and employees.

I’ve been a part of Zombie Agility. In fact, I have since learned I contributed to it. It was soul-crushing. It was physically taxing for many of my teams. It nearly broke us. We were not vibrant, growing, or thriving. We often felt like we were sleepwalking through the motions.

Good news? Zombie Agility has proven antidotes. I hope these antidotes will spark conversations among your teams about how to eradicate the Zombie Agility threat inside your organization.

In today’s article, we’ll start with the first of 3 antidotes–Stay tuned for my articles on the other 2.

Antidote #1: Inoculate your teams against Zombie Agility’s mindlessness by the regular and generous application of compelling Business Outcomes

Mindlessness incubates Zombie Agility. With no clarity of purpose, no compelling results in sight, and no clear objectives, teams start to wander aimlessly. The zombie contagion starts when “doing agile things” becomes the focus, without a clear business result as a goal. Beware of Agile transformations that lack a crisp, mind-sharpening focus on business results.

The goal of a transformation is not to “go Agile.” You can start to inoculate your organization by banishing that term forever. Instead, the goal for adopting Agile is obtaining better outcomes you can’t get with the way you work now.

Here are some examples of better Business Outcomes:

- Market Responsiveness
- Customer Satisfaction
- Employee Engagement

Business Outcomes are the goalposts for high-performance teams that use Agile to deliver those valuable outcomes. Maybe you can see how teams that aren’t inoculated can be infected. As soon as the company decides to “go Agile” without clear Business Outcomes, the minds of team members turn into zombie mush. Like real zombies focusing on “Brains…Brains…Brains,” they’re focusing on “Scrum… Stand ups… Burn downs…Retrospectives…Mmmmmm. ” Their valuable brainpower turns to mindless process compliance. As a result, energy is not invested in better customer experiences or Business Outcomes.

To treat this, regularly track and communicate progress towards the organization’s desired Business Outcomes.

These Business Outcomes, delivered with agility, provide:

- Crucial direction for the organization
- Clarity to inform better decision making
- Collaboration towards shared goals that make work more engaging

Progressing together towards these Business Outcomes instills pride and reinforces teamwork as people strive together for improvement. No room for mindlessness there.

Conclusion

So, to inoculate against mindless Zombie Agility, make the Business Outcomes you seek credible and inspirational. Talk about why they matter as often as you can. You literally cannot under-communicate compelling outcomes.

“When you’re tired of saying it, people are starting to hear it to hear it.”
– Jeff Weiner, Measure What Matters by John Doerr

The next article in series, I’ll share Antidote #2: Making sure your Agile change agents aren’t infected.

About Erik Cottrell

Erik is the Senior Vice President of Client Success & Marketing at Agile Velocity.

With over 20 years of experience in Product Management and leading strategic growth, Erik now leads his team in developing new products and initiatives with the goal of delighting customers.

Erik believes the path to lasting high performance goes beyond process change and champions that a uniquely agile mindset informs culture, technology experiences, cross-department collaboration, and employee engagement.

Applied holistically and pragmatically, agility becomes a key competitive advantage.

Meaningful Agile Metrics

By Denise Jarvie and Jessica Crowley

"Tell me how you will measure me, and then I will tell you how I will behave. If you measure me in an illogical way, don't complain about illogical behavior." - Eli Goldratt

Many organizations implement measurements that unwittingly encourage or elicit poor practices and behaviors. As they become more Agile, the pillars of Empiricism allow for increased transparency which can easily be abused. This transparency can provide a wealth of information on how scrum teams are performing and allows for identification of improvement areas. Metrics that are put into place without consideration of the desired outcome result in a change in behavior and practice which may not lead to Agility. Metrics that elicit and encourage appropriate practices and values such as: Commitment, Courage, Focus, Openness and Respect will propel teams to higher performing states.

It's just as important to limit the number of metrics that you measure as it is to measure the right things. Leadership should keep the desired end state in mind when determining which metrics to use.

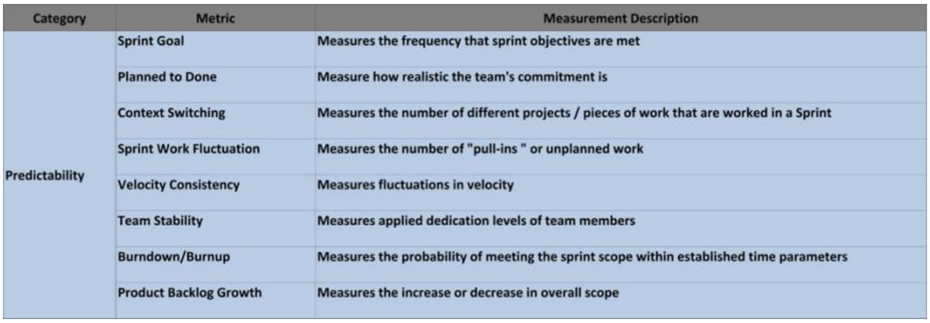
"If you don't collect any metrics, you're flying blind. If you collect and focus on too many, they may be obstructing your field of view." - Scott M. Graffius

Customers ultimately need insight into when work will be completed. Organizations in a beginning phase of Agility often look to simplistic metrics such as velocity and burn-down charts to calculate development duration. Unfortunately these indicators don't contain enough information to be used independent of more comprehensive metrics. They are meant to stimulate and guide deeper discussion, not provide a "one-stop-shop" for insight into development performance.

The duration and completion date is highly affected by the predictability, stability, improvement, efficiency and quality of the team's work. For this reason we have selected a number of Agile metrics that can help expose development issues.

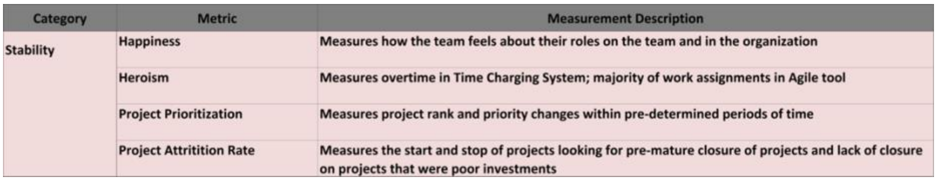
The intent of this article isn't to encourage the implementation of all the below metrics; instead it's meant to help organizational leaders select meaningful metrics that encourage behaviors in alignment with the company's quest for Agility. A best practice is to limit the number of metrics selected from each category. Periodically review the effectiveness of the metric and determine if it is more appropriate to focus on alternate metrics.

Predictability measurements encourage consistent execution of best practices. This type of metric also provides insight into how likely teams are to meet their commitments.



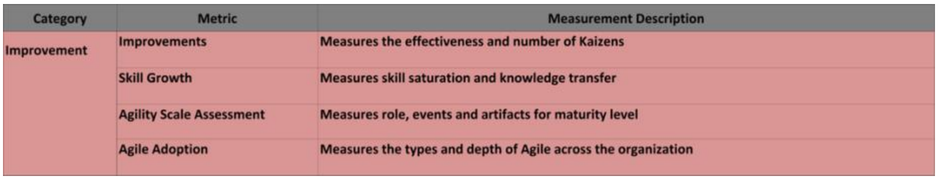
Category	Metric	Measurement Description
Predictability	Sprint Goal	Measures the frequency that sprint objectives are met
	Planned to Done	Measure how realistic the team's commitment is
	Context Switching	Measures the number of different projects / pieces of work that are worked in a Sprint
	Sprint Work Fluctuation	Measures the number of "pull-ins " or unplanned work
	Velocity Consistency	Measures fluctuations in velocity
	Team Stability	Measures applied dedication levels of team members
	Burndown/Burnup	Measures the probability of meeting the sprint scope within established time parameters
	Product Backlog Growth	Measures the increase or decrease in overall scope

Stability measurements encourage a sustainable pace of development. This type of metric provides insight into the impact of rapidly changing priorities.



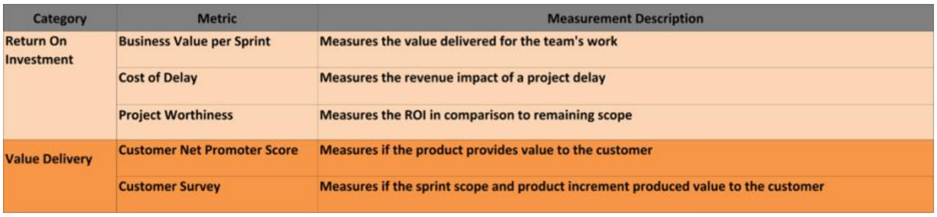
Category	Metric	Measurement Description
Stability	Happiness	Measures how the team feels about their roles on the team and in the organization
	Heroism	Measures overtime in Time Charging System; majority of work assignments in Agile tool
	Project Prioritization	Measures project rank and priority changes within pre-determined periods of time
	Project Attritition Rate	Measures the start and stop of projects looking for pre-mature closure of projects and lack of closure on projects that were poor investments

Improvement metrics help focus the organization on spreading best practices, knowledge transfer and continuous improvement.



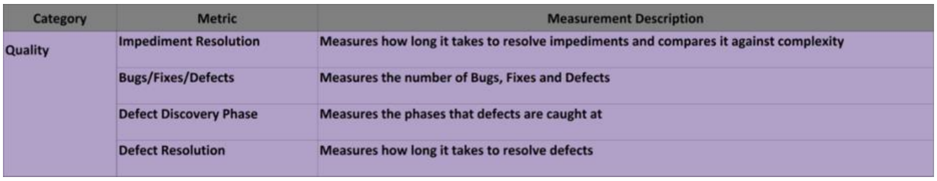
Category	Metric	Measurement Description
Improvement	Improvements	Measures the effectiveness and number of Kaizens
	Skill Growth	Measures skill saturation and knowledge transfer
	Agility Scale Assessment	Measures role, events and artifacts for maturity level
	Agile Adoption	Measures the types and depth of Agile across the organization

Return on Investment and value delivery metrics help focus the organization on the customer's needs and the profitability of the work that's being completed.



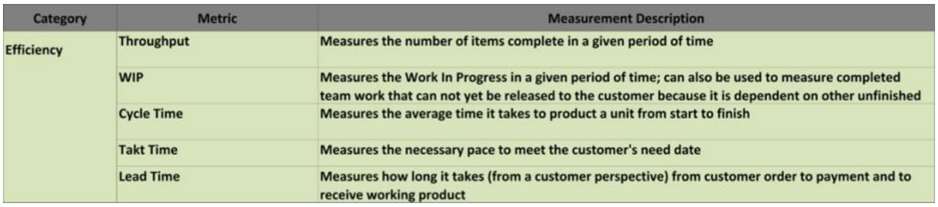
Category	Metric	Measurement Description
Return On Investment	Business Value per Sprint	Measures the value delivered for the team's work
	Cost of Delay	Measures the revenue impact of a project delay
	Project Worthiness	Measures the ROI in comparison to remaining scope
Value Delivery	Customer Net Promoter Score	Measures if the product provides value to the customer
	Customer Survey	Measures if the sprint scope and product increment produced value to the customer

Quality measurements help encourage teams to reduce technical debt and find bugs earlier in the development cycle by incrementally testing functionality and using techniques and best practices such as peer reviews.



Category	Metric	Measurement Description
Quality	Impediment Resolution	Measures how long it takes to resolve impediments and compares it against complexity
	Bugs/Fixes/Defects	Measures the number of Bugs, Fixes and Defects
	Defect Discovery Phase	Measures the phases that defects are caught at
	Defect Resolution	Measures how long it takes to resolve defects

Efficiency measurements help focus the organization on optimizing the flow of work, identifying bottle necks and improving the value stream to produce product by the customer's need date.



Category	Metric	Measurement Description
Efficiency	Throughput	Measures the number of items complete in a given period of time
	WIP	Measures the Work In Progress in a given period of time; can also be used to measure completed team work that can not yet be released to the customer because it is dependent on other unfinished
	Cycle Time	Measures the average time it takes to product a unit from start to finish
	Takt Time	Measures the necessary pace to meet the customer's need date
	Lead Time	Measures how long it takes (from a customer perspective) from customer order to payment and to receive working product

By carefully selecting and using metrics, organizations can encourage appropriate behaviors and support best practices that result in the outcomes they desire. Never be afraid to pivot from previously chosen metrics! More than likely you will outgrow the first ones you choose.

Read this article online: https://baa.tco.ac/1SyA

About the Co-Authors

Denise Jarvie is a Licensed Scrum Trainer, Certified SAFe 5 Program Consultant, Certified Scrum Master, and Certified Product Owner that can train and certify in both the Scrum@Scale and Scaled Agile Frameworks.

Denise is an experienced Enterprise Agile Coach with a strong background in the Government Sector. She is passionate about developing capability for the warfighter faster using Agile development approaches. Denise co-created a branch of Humphreys & Associates called Radically Better Agile and she promotes the adoption of Agile in the Department of Defense.

www.linkedin.com/in/denisejarvie

Jessica Crowley is a Licensed Scrum Trainer, Certified SAFe 5 Program Consultant, Certified Scrum Master, and Certified Product Owner that can train and certify in both the Scrum@Scale and Scaled Agile Frameworks.

She is an experienced Enterprise Agile Coach that has helped create a branch of Humphreys & Associates called Radically Better Agile.

She organically journeyed through an Agile Transformation in the commercial sector from inception to successful expansion.

She guides some of the largest US Department of Defense contractors, subcontractors, and customers through Agile Transformations providing training, certifications, and coaching support.

www.linkedin.com/in/jessicajcrowley

H/R - Related LESS Experiments Deciphered

By Gene Gendel

Large Scale Scrum has a history of more than a decade. The *first book* about LeSS was published by C. Larman and B. Vodde (the co-creators of LeSS) in 2008. There were two more books on LeSS, subsequently written in *2010* and *2016.* There is no surprise, why the collection of LeSS experiments from the field is so valuable: the authors have documented many (more than 600) experiments, based on their personal experience with LeSS adoptions, as well as feedback and information collected from other organizational design consultants, coaches and early adopters of LeSS, around the globe.

Today, references to *LeSS Guides and Experiments* can be found in various places on the internet and intranet of many companies that have decided to experiment with LeSS.

This writing is about a small sub-set of LeSS experiments that are specifically related to ***HR norms, policies and practices.*** They are all listed in the guide (referenced above), under the section "Organization" and this implies that they are directly related to an organizational design – the first-order factor in success of LeSS adoptions and agile transformations, at scale.

Experiments with Performance Appraisals:

"Avoid… Performance appraisals – p. 273" — There is a lot of research and evidence, supporting that individual performance evaluations and individual appraisals that are linked to monetary rewards, are not an effective way to make individuals become more efficient and productive. When a manager appraises an employee, there is usually only one opinion that matters: a manager's. Feedback that is delivered once or twice a year is outdated and therefore is hardly actionable by an employee, thus for the most part is useless. Neither an individual that delivers an appraisal, not an individual that receives it – like the process. The process, is also pretty expensive, as it uses a lot of company's resources: it involves lots of documentation, coordination and men-hours spent by many people, from first-line management to HR.

It is worth noting that there is an indirect relationship between conventional Budgeting process and conventional Performance Management process – both of which harmfully feeding off of one another. This is described in the book "*Implementing beyond Budgeting: Unlocking the Performance Potential*", by Bjarte Bogsnes. In his work, Bjarte refers to performance appraisals as "*legal trail for a rainy day"*. **"Avoid… ScrumMasters do performance appraisals – p. 275" —**Just like performance appraisals done by agile coaches could lead to serious dysfunctions (*page. 130*), performance appraisals done by ScrumMasters are also harmful. Drafting ScrumMaster into the role of an appraiser may create a serious conflict of interest and will hinder ScrumMaster's ability to influence natural growth and evolution of learning among team members. Impartiality and neutrality of ScrumMaster is highly important; becoming an appraiser – takes away this advantage. Only by remaining neutral and non-authoritative (performance appraisal is exhibition of authority) will ScrumMaster be able to help a team to self-discover, self-improve, and become autonomous in their journey to success.

"Try… De-emphasize incentives – p270." - "Avoid… Putting incentives on productivity measures – p. 271." If achieving a higher productivity (output, velocity) is coupled with monetary incentives/perks or other political gains (typical of many companies that overuse scorecards, metrics, KPIs, RAGs), there is will be always attempts by individuals/teams to claim successes/achievements by 'playing the system', in pursuit of recognition and a prize. For example, in pursuit of 'higher productivity' teams may start inflating estimates, to claim higher velocity or deliver work that is low in priority but simple to deliver – just to create an illusion delivering value (output != outcome). Incentivizing 'higher velocity' is an invitation to moving from "low Fibonacci numbers to high Fibonacci numbers" during estimation. (Also, see *Addressing Problems, Caused by AMMS*)

"Try… Team incentives instead of individual incentives – p. 272" — The process of individual performance reviews loses its original meaning when people work on a same team, where swarming (working together on the same task) and collective ownership is encouraged. Offering individual incentives to people would just polarize them and move in opposite directions, towards becoming self-centered, individual performers and super-heroes. In cases such as these, people may be easily drafted into unhealthy competition with each other over claims of success, trying to privatize what should be owned and worked on collectively. Companies that continue incentivizing individual performance with monetary perks, just continue widening the gap between "*what science knows and business does*" (quote from Daniel Pink).

"Try… Team-based targets without rewards – p. 273" — Clearly, team-level behavior, is an extension of an individual's behavior. Just like individuals could be inclined to 'game a system', so could entire teams, if they are put under bad conditions. Just like individuals, whole teams might be drafted in unethical conspiracies to game numbers, in pursuit of meeting targets, or beating other teams (e.g. producing 'higher velocity'), whenever monetary rewards are offered. If it is mandatory to set targets for individual teams that work on par with one another, for the same organization, it would be best to decouple team targets from team rewards. The latter could be handled through, some sort of *profit sharing formula*, based on a company's financial success, achieved because of great team work.

Experiments with Job Titles:

"Avoid… Job titles – p. 276 | Try… Create only one job title. Try… Let people make their own titles – p. 277 | encourage funny titles" – p. 277 —In pursuit of job titles, individuals may also seek gaining authority and an "upper hand" over their peers and colleagues. This may lead to artificial organizational complexity and hierarchy, as well as a casting system. Individual job titles can also polarize people and drive them in opposite directions, away from shared ownership. It is for this reason that on agile teams (e.g. Scrum), there is only one title – Developer. This approach encourages people to think of each other as an equal colleague and be willing to grow into T-shaped, multi-skilled, cross-functional, willing-to-swarm workers. In situations, where some distinction between individual jobs is absolutely necessary, funny job titles are recommended. For example, instead of calling someone QA Tester, a person could be called "Bug Finder and Exterminator"

"Try… (if all else fails) Generic title with levels – p. 277" — If it is necessary to have title distinction (e.g. to signify different levels of seniority/expertise of individuals), try using a leveling system. For example, Developer level 1(junior), Developer level 2 (mid-level), Developer level 3 (senior).

Experiments with Jobs:

"Try… Simple general job descriptions – p. 278" – Do not over-complicate job descriptions. Precision in a description may lead to contractual perception of what a person should and should not do, in a workplace. This may also limit a person's willingness to step out of his comfort zone and learn other areas of work, other skills and becoming multi-faceted. It may then further lead to "managing by objectives" that are based on detailed job descriptions, and subsequently, bring about problems of performance appraisals, described above. Complex job descriptions also have a tendency attracting under-qualified external candidates, whose resumes are excessively long, as they are 'tailored to closely match complex job descriptions'. (Relevantly, attracting bad agile coaches, by creating inappropriate job descriptions is a known problem).

"Try… Job rotation – p. 279 | Try… Start people with job rotation – p. 280" — Give individuals opportunities to learn new domains, technologies, lines of business. This may reduce the risk of a person becoming uninterested/bored with his current job. Further, by rotating from one job to another, a person may discover where he fits best and delivers most value. By having this opportunity, a person will also have a higher chance of merging the gap between "having to do a job" and "wanting to do a job". This is especially important with newly hired people that have a limited industry experience (e.g. recent college graduates).

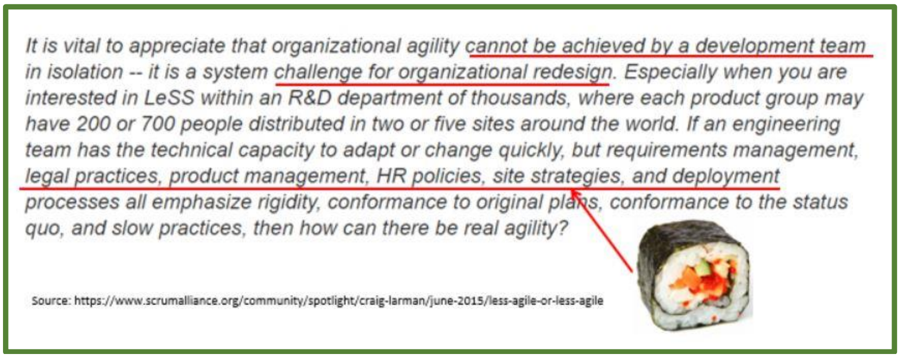
Experiments with Hiring:

"Try… Hire the best – p. 280 Avoid… Hiring when you cannot find the best – p. 281" — Do not settle for less *than "best people your money can buy".* It is better to rely on fewer great people that you already have on-staff than bring on more under-qualified people, to speed up work, especially at the end of a project that is already late (*Brook's Law*). From a *system thinking* perspective if you are trying to increase velocity (output) by a scrum team and decide to do so by adding more developers that you procured on low cost (low pay will most likely buy you low-skilled developers), you will most likely reduce velocity, by having low-skilled developers introducing more bugs into a system. Please, *see why*.

"Try… Team does the hiring – p. 281" — If you plan on hiring an individual to join a team, please make sure that a team does most of interviewing and vetting. Through that, not only a person's skills and experience will be examined but it will become more apparent if a person can organically jell with a team: if there is compatibility, chemistry and synergy with other team members. Panel interviews by whole teams are usually much more effective, since they include practical tests, real-life simulations and hands-on exercises. It also allows some people to observe, while others ask questions, and then rotate. Try to reduce the level of influence that HR personnel and first-line management have on the process as much as legally possible. This will reduce the amount of subjective, administrative, frequently bias and error-prone screening (refer to top of page 17).

Conclusion:

As a summary, please consider the following quote that describes sushi-roll-like organizational design in Large Scale Scrum (LeSS), by C. Larman (also, explained in detail in *Agile Organization, as a Sushi Roll):*



It is vital to appreciate that organizational agility cannot be achieved by a development team in isolation -- it is a system challenge for organizational redesign. Especially when you are interested in LeSS within an R&D department of thousands, where each product group may have 200 or 700 people distributed in two or five sites around the world. If an engineering team has the technical capacity to adapt or change quickly, but requirements management, legal practices, product management, HR policies, site strategies, and deployment processes all emphasize rigidity, conformance to original plans, conformance to the status quo, and slow practices, then how can there be real agility?

Source: https://www.scrumalliance.org/community/spotlight/craig-larman/june-2015/less-agile-or-less-agile

In it, HR policies is listed as one of the vital elements of overall organizational agility.

Read this article online: https://baa.tco.ac/1Sxv

Guidelines to Hiring a Professional Coach

By Gene Gendel

Let's face it, today, finding an experienced and credible agile coach, is not easy. If you disagree with this statement, you are either very lucky and have special access to some great talent (e.g. referrals or networking) OR …..your perception of the role may need to change.

There is no need to be ashamed of not being able to find a good coach. You are not alone, many companies face the same challenge. Truth be told, unfortunately, the industry has changed significantly over the last few years and it became the source of many problems (some very classic problems are described *here*). Today, the term "Senior Agile Coach" has been grossly diluted.

But *fortunately*, there are still great standards and guidelines you can follow, when looking for an agile coach, irrespective of the industry's trend-down. Please, consider the dimensions below, when looking for a professional agile coach, for your organization. The original sources of these requirements are listed at the bottom of this page and you are encouraged to explore them for additional details.

Please, do not reduce, simplify or trivialize some of the key expectations of a professional agile coach. Because, if you do, the following two problems will follow:

- Industry coaching quality (average) will be further decreased,… and even if you don't care about this fact as much…you will care about the next fact:

- Quality of service to your own organization will be also low

…with that….

"Must-Have" for Professional Agile Coach

Quantitative Assets:

- Has significant hands-on experience in at least one of the roles on Scrum Team
- Has coached multiple organizations, departments, or programs
- Has, at least, 1000 hours of experience coaching at the enterprise/organizational level or a combination of enterprise and multi-team level coaching
- Has diversity of coaching experience that can be demonstrated, using different client engagement examples that include experience at an organizational level

Demonstration of deep knowledge:

- Has formal and informal education about coaching and strong mentor relationships
- Has good working knowledge of Agile and Lean values, principles, and practices.
- Has helped individuals, teams, and leadership to understand and apply Agile and Lean values, principles, and practices effectively
- Understands dynamics, patterns, and development of multi-level teams and how they interact at an organizational level
- Knows the difference between consulting and coaching and knows when to take each stance

Ability to clearly articulate and substantiate one's own:

- Coaching Career Overview (coaching, agile history and how a person got where he/she is today. Include key milestone years)
- Coaching Focus (summary of a person's professional self today, including a coaching approach and/or philosophy to coaching)

- Coaching Goals (personal development goals in coaching)
- Formal Coaching Education (formal education activities which have contributed significantly to your coaching journey. This includes a wide range of courses on topics, including facilitation, leadership, consulting, coaching, process, tools, techniques, frameworks, and other related activities which have influenced our coaching practice)
- Formal Mentor-ship Education (coach mentor-ship and significant collaboration activities where a person has DEVELOPED a skill or technique or RECEIVED guidance to his/her coaching approach and mindset.)
- Informal Coaching Learning (significant topics you have studied outside of the Scrum literature which has impacted his/her coaching approach or coaching philosophy)
- Agile Community Participation (agile community events, such as user groups, gatherings, retreats, camps, conferences, etc. in which a coach has participated)
- Agile Community Leadership (leadership contribution to agile communities (e.g. writing, publishing, presenting, facilitating, organizing, training and other activities) through events, publications, courses, blogs and forums)
- Agile Community Collaborative Mentoring & Advisory (significant collaborative agile mentoring, advisory activities, where a person was mentoring, advising other individuals to increase their competency or in development of a specific goal)
- Coaching Tools, Techniques or Frameworks known (coaching tools, techniques or frameworks which you have implemented, customized, co-developed or developed in one or more client engagements)

Skills, Tools & Techniques:

- Has contributed to significant improvements in organizations or departments through coaching techniques

- Has helped organizations and teams beyond the basics of Scrum theory and practice
- Has enabled organizations to find their own solutions to business problems through application of Agile principles
- Is familiar with, promotes and embodies the mindset of Servant Leadership
- Uses a rich set of facilitation, training and coaching tools, and models

Personal Qualities:

- Coaching Mindset Coaching skills/practices and frameworks
- Evidence that a coach has taken both their Experience and Learning and synthesized these into definitive practices, frameworks, approaches, and strategies)
- Self-awareness: Able to reflect on their own contribution to the coaching by virtue of their own ‘being’
- Constant Learning: Has and continues to acquire Coaching oriented learning through multiple dimensions
- Diversity of Experience with different types & sizes of organizations
- Participation in the Agile community

Note: Your company needs to have internal expertise to validate a person, based on the above.

Why is LESS Authentic?
Why Should Leadership Not Exempt Itself From Learning LESS ?

By Gene Gendel

Large Scale Scrum (LeSS) is an agile framework that has a history of implementations, trials & errors, experiments and experience reports collected and documented throughout more than a decade.
LeSS is Scrum, performed by multiple teams (2-8) that work on the same widely defined product, for the same Product Owner.
LeSS stresses the importance of organizational descaling (a.k.a. removal of overhead/waste) that needs to happen before agility (Scrum) can be scaled. The first LeSS book (of three published so far) was written in 2008 and it had incorporated the ideas of its two authors, C. Larman and B. Vodde, by mainly including their own experiences of initial LeSS adoptions.

Overall, LeSS journey has begun many years before Large Scale Scrum has been officially presented to the world and recognized, as a framework, and this is important to acknowledge. **But why?**

Because LeSS, unlike some other very popular and commercially successful frameworks, that are very easy to 'unwrap and install', was not invented re-actively, as a "quick fix/hot patch", in response to growing market trends and business needs (commercial driver).

LeSS is authentic. LeSS took its time to mature and cultivate, as a philosophy and a way of thinking, not as a revenue-generating utility. LeSS did it at its own pace, without a rush, while incorporating learning of many coaches and companies that went through LeSS adoptions, over years.   LeSS has naturally "aged", in a good sense of this word.

Important Point: Whereas, deep learning of system dynamics and organizational design is equally available to everyone who attends LeSS training, not everyone can equally impact-fully apply this learning, when they go back to work**. But why?**

Lots of LeSS learning (through system modelling, using causal loop diagrams) touches upon organizational elements, such as HR norms and policies, reporting structures, career paths and promotions, location/site strategies, budgeting/finance processes, etc. – things that are considered to be "untouchable" for an average person (employee).

Of course, it does not mean that an average person is not able to start seeing things differently (they definitely do!) after studying LeSS but it is just that he/she may not have enough power/influence to make necessary organizational changes that are required by LeSS. In fact, for many people, this newly gained knowledge which is no longer possible to "unlearn" (e.g. ability think systemically), comes with realization of one's own powerlessness – and this could be pretty frustrating.

Things are different for people that occupy higher organizational positions. A senior manager is able to combine the decision-making power that is given to him by his organization and the power of newly obtained knowledge, coming from LeSS training. These two powers, if united, can have an amplified effect.

Notably, a senior manager who wants to apply LeSS learning to improve his organization must have something else that is very special, in addition to just having general curiosity of the subject and desire to experiment: it is called a 'sense of urgency'. The best examples of senior managers that have learned LeSS and then applied learning to reality, came from situations, where there was an urgent need to change and risks/costs of a failure were high. Then, if the above was true, the formula of LeSS adoption success becomes:

(Organizational Power + Power of Knowledge) x Sense of Urgency = Success of LeSS adoption

Important Point: It is strongly **not advisable** for senior managers to delegate LeSS learning to people that are below them organizationally, as the ladder are not empowered to make organizational changes. Granted, individuals at *all* organizational levels will be benefited from learning LeSS (it is a great eye opener). But senior managers – people that are empowered to make significant organizational changes, must attend LeSS training in person and not delegate attendance to their subordinates. Senior management should not exempt itself from learning.

In fact, and ideally, senior management should attend LeSS training, accompanied by their respective organizational verticals, so that everyone goes through the same learning journey together. Having HR and finance people, alongside with C-level executives and staff members of lower organizational levels – is a **HUGE BONUS.**

About Gene Gendel

Gene is an organizational design specialist, agile/lean coach and trainer, consultant and advisor to senior leadership with 20+ years of experience.

Gene is dedicated to working with companies of various sizes and lines of business, trying to help them improve internal dynamics, organizational structure and becoming a better place to work.

Gene engages at all organizational levels: senior- and mid-level management, teams and individuals. In his work, Gene uses various methods, tools and techniques to amplify learning by others and to ensure that people gain autonomy after Gene "coaches himself out of the job".

Gene significantly contributes to global and local agile communities, where he influences people by running workshops, webinars, seminars, panel discussions, lunch & learn, coaching retreats, professional gatherings and other events.

To learn more visit

www.keystepstosuccess.com/about-me

What Is Your Product?

By Ellen Gottesdiener

“So, what is your product?” That was the key question I posed in my Agile Cincinnati keynote recently.

In my product coaching work, I have come to realize that many organizations don’t have a clear and consistent answer to this fundamental question. This has serious consequences. A poorly defined product impacts your ability to respond to changing customer and market needs. It results in less than satisfying product outcomes. It causes organizational and communication woes. It thwarts organizations efforts to scale agile product development.

Everyone in your product development ecosystem should have a shared, consistent, and coherent answer to the fundamental question, “What is your product?”

You need a shared, consistent, and coherent answer to “What is Your Product?” Image by EBG Consulting

The Five Principles

There are five principles that can help you to define your product:

1. Adopt outside-in thinking.

Many organizations erroneously define their products by taking a technology perspective. They consider the product’s underlying technologies, components, and tools and call a logical grouping of them a "product." This approach defines the product from an engineering perspective and not from a customer perspective.

Alan Cooper's prescient book, *The Inmates are Running the Asylum*, was a wake-up call back in 2004. Its title comes from the bizarre 1989 cult film by the same name involving a plot whereby an experiment mind fluid exchanges in an insane asylum that goes array. Consequently, the doctor becomes the patient and the patient becoming the doctor. Cooper (the father of Visual Basic and originator of personas) makes the point that technology products are horribly designed and make users feel stupid

Define your product outside-in (not inside-out).
Image by Ellen Gottesdiener, EBG Consulting

This represents inside-out product thinking and it simply doesn't work. Instead, define your product from the perspective of customers, both users and choosers. Users are those people who directly interact with the product to obtain some goal or solve a problem. Choosers are those who making buying choices. Outside-in thinking is both humbling and enlightening. It allows you to approach your product work with empathy and curiosity.

2. Take the long view

Some seven years ago, I gave a presentation about *product roadmapping* with the title "Products, Not Projects." Even before I started I noticed that some attendees, many of whom were professional project managers, were not happy with the title. Projects come and go while successful products live a long and healthy life. A healthy product evolves and morphs through its lifetime, benefiting from continuous discovery and delivery.

A healthy product has a long lifecycle, from birth (introduction and launch) through decline.
Image by Ellen Gottesdiener, EBG Consulting

When organizations fall into the trap of viewing technology innovations as products in-and-of-themselves, they usually end up with multiple products that are serving the same customer or market. Some with the same core capabilities. This complicates organizational processes and confuses customers.

The better approach is to take the long view of your product. New technology variations and developments are part of ongoing improvements to your product's value proposition. They can improve your product's business viability, customer experience, and improve product quality.

A product organization taking the long view operates as a single virtual product community despite having distinct product management and product development organizations. Developers and engineers actively participate in discovery to validate technology innovations, parity imperatives, and new product capabilities. The product community invests in fast, efficient experiments before making costly changes to the product

A clear product definition should help you shift-left***er***. Shift-left is all about **using tests to guide product development** to reduce risk and build shared understanding early in development. Great product teams migrate even more to the left through validation to determine if they are building the right thing. Product options should be continually validated to ensure a long and healthy product lifecycle.

3. Define your product as broadly as practical.

Many organizations I work with define their products narrowly. This results in numerous kinds of organizational maladies including:

- Products with multiple overlapping and conflicting backlogs
- Functionality duplicated across multiple products
- Bottlenecks caused waiting for technology specialist teams
- Excessive coordination and product roles

Narrowly defined products lead to negative customer experience. Image by Ellen Gottesdiener, EBG Consulting

Most importantly, narrowly defined products can result in a bifurcated user experience. Completing an end-to-end value stream using your product is clunky, laborious, and abrasive. Customers needing to communicate with the organization to solve problems or answer questions are faced with a confusing array of disparate support teams in different departments. Problems and questions are slow to resolve as the customer gets passed from department to department.

A narrowly defined product has another unfortunate side effect. It promotes a natural bias to optimize outcomes that serve only that particular product's narrow goals. This phenomenon is known *as local optimization* and results in decisions, actions, and organization structures that impede the overall, and more global organizational goals.

On the other hand, when you define your product as broadly as possible, you "see the whole." This affords you a view of a wider terrain of product options, simplifies communication, clarifies roles, and forces prioritization. Broadly defined products allow your organization to optimize people and resources for the entire organizational system. This is called systems optimization. Systems optimization through broader product definition facilitates your organization's ability to meet your higher-level goals. It also eases and simplifies:

- Strategic planning and roadmapping
- Backlog management
- Organizational and customer communications
- Prioritization and decision making
- Role definition

Another benefit with systems optimization is simplified organizational design. This is discussed in the next section.

4. Structure must follow product.

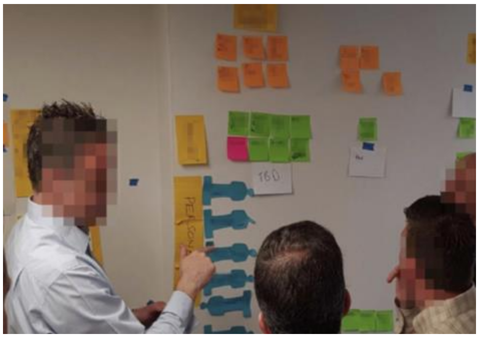
Some organizations fall into the trap of defining products according to their organization chart. Parts of products supported by a particular functional group or reporting line is declared to be a product. Chaos and confusion ensue each time there is a reorganization. Over time, the product architecture emulates a hairball of intertwined interfaces, technology knots, Wild West of cowboy code, and cruft. The product and product team suffers from the unfortunate effect of Conway's Law where a product's design is "a copy of the organization's communication structure."

When a product is defined outside-in, your organization benefits with a more product-centric design. Feature teams and end-to-end value stream teams are formed based on customer outcomes. You have one backlog per product, which helps ease prioritization and avoids the trap of local optimization. You can learn more about local optimization pitfalls from Michael James' excellent video about "team output owners."

Finally, Craig Larman's' brilliant Laws of Organizational Behavior states that "culture follows structure". When you have defined products in a customer-centric manner, with a long view in mind, and as broadly as possible, structure follows product.

5. Defining your product requires collaboration.

To obtain a shared definition of your product, employ team collaboration. In my keynote, I shared some of the experiments from product thinking facilitated workshops in which we defined a product through a series of progressive, interactive activities. These activities include clarifying the work of product management and ownership, using the Product Canvas, decision making rules, customer research, and value proposition canvas.



People Collaborating to Define Product
Photo from Product Definition Workshop by Ellen Gottesdiener, EBG Consulting

The Importance of Knowing Your Product

Agile product development is about continual discovery and delivery of products that are desirable to customers, viable for the business, feasible for technology people to build, and supportable. How you define your product guides drives your discovery and delivery.

The first thing to do when you are starting or reinvigorating your product organization is to answer the question, “What is your product”?

Many thanks to Andy Repton, Michael James (MJ), Cesario Ramos, and Tim Ottinger for their helpful feedback on this blog.

A Russian version of this blog is now available at Agilix Consulting. Thank you, Illia Pavlichenko!

Read this article online: https://baa.tco.ac/1Sxs

About Ellen Gottesdiener

Ellen is a Product Coach, and CEO of EBG Consulting focused on helping product management and development communities produce valuable outcomes through product agility.

Ellen is known in the agile and product communities as an instigator and innovator for collaborative practices for agile product discovery and using skilled facilitation to enable healthy teamwork and strong organizations.

She is the author of three books on product discovery and requirements, frequent speaker, and works with clients globally.

In her spare time Ellen is Producer of Boston's Agile Product Open community and serves as Director of Agile Alliance's Agile Product Management Initiative.

Foundation before Scaling: Lessons from Karate

By Gary Hansen

I learned everything I know in a karate school, well almost everything. Now that my son is in karate, I get to relearn much of what I learned, or at least remember where I learned it.

The picture is of my son Griffin. He holds the rank of a blue belt. In the picture he is practicing a weapon called a bo, it is also known as a staff. Calling it a bo staff is common, yet redundant and wrong.

All weapons are considered an extension of our bodies; this is even true of guns. These extensions are used to help us maintain distance from an attacker. It could also be used to attack from a distance, but martial artist use might for the right reasons.

Karate students usually begin learning weapons after a year or more of training. Why wait so long? Learning weapons is fun and cool, at least cool to karate nerds like me. Additionally, they are effective at maintaining a distance between yourself and an attacker. It is difficult to defend against an attacker with a weapon. Blocking a bo that is thrust at your head or blocking a sword or any other weapon with your bare arm is not fun, to say the least. Although painful, it is likely better to block the weapon than getting hit in more vital places.

Why wait a year to learn weapons? New karate students don’t know the basic aspects of controlling distance, movement, proper stances, and other foundational aspects of self-defense. And they don’t know what to do and why to do it. Every strike or block performed with a weapon is based on strikes and blocks performed without weapons. First, learn to punch without a weapon and then use a similar movement with a weapon; this is why we say weapons are an extension of a person's body.

Build the basics and then add on- this is a foundational aspect of learning anything. Being effective at the basics takes more than know-how, being proficient requires practice. We all know this, and perhaps we all wish it weren't true for us.

Wishing is a bad strategy, maybe the worst. We wish that we can transform entire companies without putting in learning and practice. We want to scale quickly yet often scale bad habits that are built on weak foundations. We have to be reportedly agile before we understand what it means and have practiced enough to be agile. We all try to earn black belts without having put in the years of diligent and purposeful practice.

In martial arts, we learn only a few concepts and techniques at a time. This provides us an opportunity to focus on trying, learning, and trying again. We need to have "clean" technique (no waste, no anti-patterns) before we try more challenging techniques. Martial artists have used Deming's PDCA (plan, do, check/study, act) for thousands of years before it existed. The only way we can learn is to limit learning-in-process (LIP, is this a new acronym? maybe instead of WIP...). Yes, I am not just a karate nerd; I am also an agile nerd; or just a nerd?

We need to start where we are, not where we want to be, and have clarity on where we are headed. Learn to do a front stance before learning to do the bo.

About Gary Hansen

Gary is a self-proclaimed "learning instigator."

He has worked with agile approaches since 2009 and his major engagements have been with United Health Group, Target, 3M, Best Buy, Thrivent Financial, & TCF Bank.

He works with all levels of an organization to ensure that transformation efforts are aligned. Gary is a Scrum Alliance Certified Team Coach (CTC), International Coach Federation Associate Certified Coach (ACC), and has 13 additional certifications. Gary has an M.Ed. in Human Resource Development from the University of Minnesota.

Email Gary at

hansengaryw@gmail.com

Business Outcomes: The Need For Speed & Fast Product Delivery

(Part 1)

By David Hawks

When we talk to leaders about what they want from Agile, most of the time, the top answer is **Speed**. Businesses need fast product delivery in order to satisfy customers and grow revenue.

Often the delivery team is seen as the bottleneck, and if the organization could just remove what's blocking them, they would definitely get more value out the door… right?

Turns out there's more to improving delivery speed than that. Read on to learn 6 organizational capabilities needed for achieving fast product delivery.

How To Measure Speed

Before we get started, let's define speed as the time it takes to deliver an idea into the market. Speed could be measured by metrics like cycle time, lead time, deployment frequency, mean time to restore (MTTR), etc.

In order to accelerate value delivery speed, there are 6 key capabilities companies need to acquire at every level of the organization. Let's dig in.

6 Organizational Capabilities Needed For Improving Speed

1. Ability to Focus
Level: Leadership
When teams are interrupted with new priorities, they get bogged down

by task-switching costs. In order for teams to go faster, leaders need to be clear about priorities and refrain from shifting directions constantly. The new leadership mantra should be "Stop Starting, Start Finishing." It will take discipline for leaders to make hard decisions, sequence the work based on priority, and allow teams to pull in work as they have capacity.

This is opposed to setting deadlines and then pushing all the work onto teams. Which just jams up the system and doesn't allow us to get things done. While everybody looks busy, value is not being delivered.

2. Predictable Delivery Cadence
Level: Team
In order to increase product delivery speed, teams have to start working in smaller batches or increments of work. In Scrum, we talk about working in 2-week sprints. While team members might think this is arbitrary at first, this is a forcing function to encourage teams to break work down so they can deliver an increment of value at the end of that 2-week period.

That means they need to break the scope of work into small, demonstrable increments of value, and teams need to learn how to collaboratively swarm to get work done. Similar to the first capability where leaders limit work-in-progress (WIP), teams need to learn the discipline of limiting WIP in their sprints and daily work. Instead of everyone working on their own thing, teams need to work on as few items as possible at a time in order to get things completely finished. This allows teams to shorten time-to-value. Even if it is incremental, it is better than waiting months to see any value.

3. Cycle Time
Level: Team
Once teams get good at breaking work down and getting small increments of value out the door, they can start focusing on shortening their cycle time. Cycle time is measured as the time from when they start the work to the time it gets delivered.

Teams that are good in this area foster a true spirit of shared ownership of the work, greatly reducing skill boundaries and gaps. By investing in more knowledge transfer and skill flexibility, teams are able to adapt to whatever type of work is of the highest priority. In addition, teams need to invest in reducing waste in their processes, such as quality issues that cause frequent interruptions. For software teams, this would be things like reducing the amount of technical debt in the product enabling teams to make code changes faster.

4. Release Continuously
Level: Team
It is not good enough to only shorten the time it takes to get something done in a sprint. What we also need to focus on is getting value out the door. How long does it take to get the finished product into the hands of the customer? How can we improve this?

Focus on ways to shorten deployment time through tools and automation. For software teams, this is where DevOps practices come front and center. They allow teams to get code into the hands of the customer faster, which means removing manual tasks as much as possible and using automation so that we can operate with speed *and* quality.

5. Faster Time to Value
Level: System
While an individual team could be delivering fast, the overall flow of work through the system or value stream could be compromised by a bottleneck outside the team. This is when leaders across departments need to work together to ensure that optimizations are being made to improve the overall system, not just one team or one department.

To help, leaders should explore systems thinking and try employing techniques like the Theory of Constraints.

6. Decision Agility
Level: Leadership
Part of speed is how long it takes to make a decision. In traditional hierarchical companies, it takes a long time for decisions to go up the

chain and come back down. And the decisions are being made by people too far removed to fully understand the problem.

In order to increase speed and deliver faster, we need to empower the people closest to the problem to make decisions without having to wait. However, we want these decisions to be aligned with the goals and direction of the company. So in order to empower the teams, leaders need to clearly communicate goals, objectives, and direction (The Why) and leave the plan and details (The How) up to teams. Leaders should be clear about the "goal posts" by defining what success looks like, not "how to achieve success."

As you can see, gaining speed and achieving fast product delivery is a lot of work and requires a lot of discipline. Many leaders approach speed the wrong way, trying to push more work into the system and hoping it will come out the other side faster.

As Henrik Kniberg said, "you can't just shove more paper in the printer trying to get it to go faster." The printer will jam…and this is what is happening to most teams today. They are jammed and going slower and slower as more work is piled on them.

You have to go slow to go fast. Reduce WIP, get small increments done, optimize the whole, focus with clear goals, and distribute decision making closer to the problems.

Don't forget to check out the next article in this series, **Achieving Predictability in Business**.

Learn more about how your organization can gain momentum and deliver products faster on our Path to Agility® page.

About David Hawks

Founder and CEO of Agile Velocity, David is a Certified Enterprise Coach and Certified Scrum Trainer who is passionate about helping organizations achieve true agility beyond the basic implementation of Agile practices.

David's primary focus is to guide leaders through their Agile transformation by helping to create successful transformation strategies and effectively manage organizational change with a focus on achieving real business results.

He received his Bachelor of Business Administration in Management Information Systems degree from the University of Texas at Austin.

When not helping organizations build lasting success, he can be found swimming, running, biking, spending time with his wife and kids, or supporting his Texas Longhorns. David not only bleeds burnt orange but is known for a crazy, 18-hour tailgate experience, arriving at the lot first and leaving last.

Building Personal Bonds with Your Team

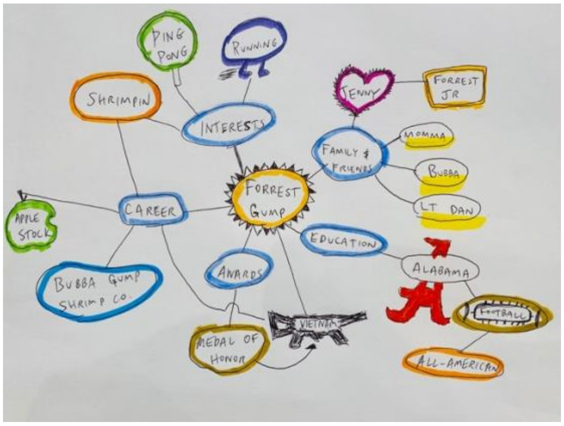
By Ben Kopel

Building Personal Bonds with Your Team

For me, there are few joys in life as satisfying as being part of a team of people that have each other's back, support one another and have a common, shared purpose. Whether that team is made up of family, co-workers or passionate strangers working to combat <fill in your political passion>, collaborating, building, learning and experimenting with like-minded people gets my juices flowing.

But not all teams have the same level of effectiveness. Thanks to the Google research on Project Aristotle, we know that psychological safety is critical when it comes to creating and sustaining effective teams. People need an environment where they are free to take risks, encouraged to speak their mind and not ridiculed for asking what may appear to be a stupid question. In my experience, creating this kind of environment comes in large part from trust within the team. To build trust, you can host a team happy hour, or participate in team exercises like axe throwing and trust falls. When people start to let their guard down and open up a bit more, I believe that it accelerates team and trust building. Here are some innovative techniques I have used.

Personal Maps



Jurgen Appelo’s Personal Maps are very simple, yet powerful. The map starts with your name in the middle and branches off to what is important to you. The categories are items such as Family, Work, Education and Hobbies. These main categories continue to branch out throughout the Personal Map. For example —Work might branch off to the various jobs you’ve had, and those might branch off to related skills or career growth.

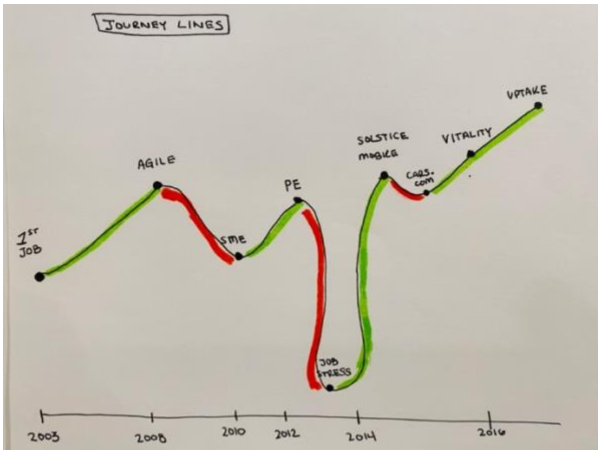
When I’ve used Personal Maps, we’ve either taken a few days to create them (“Let’s have this done for our Thursday meeting”), or we might take fifteen minutes in the beginning of a meeting to create them. Then, individuals will present theirs to the team, who follow up with questions to learn more about each other.

Even with extremely close teams, Personal Maps help to learn more about each other. Personal Maps are a simple and fun way to get to know each other at a deeper level.

Journey Lines

I was exposed to Journey Lines years ago by a mentor, Jeff Steinberg, and found it to be very effective and have used it with many teams. With Journey Lines, you create a timeline across the bottom of the page and add an additional parallel line half-way up the page.

Next you identify different events on your timeline. Events are often work related, but I like to include personal events as well in order to build deeper relationships. Events like marriage, a new baby, a cancer diagnosis or the death of a loved one would be common. Starting from left to right you chronologically identify the events along the timeline. Place positive experiences above the horizontal line and negative experiences below. Label them and connect all the lines.



When doing Journey Lines, we might take a few days to create them or simply provide time in the beginning of the meeting. When everyone has theirs ready, team members walk through their Journey Lines and a discussion ensues. You will surely learn something new about your teammates, and it will be fun and engaging.

For both Personal Maps and Journey Lines, I would encourage teams to hang them up in the team pods or areas where the teams meet. A few teams I worked with hung them up where they met for the Daily Scrum and when people would arrive early, they would look them over and start up new conversations. I'd also recommend that when new people join the team, they create an artifact and share with the team. And it probably wouldn't hurt if the team shared theirs with their new teammate.

If You Really Knew Me…

Recently, at the Agile Coach Camp in Washington D.C., I attended a session with Olaf Lewitz and Josh Magro that used an exercise that is part of Josh's P3 workshop. In small groups, we sat in a circle and went around one-by-one finishing a series of statements.

The statements started at a high level, beginning with something along the lines of, "If you really knew me, you'd know that ______", and each person recited the statement and filled in the blank with their choice of words. This allowed people to respond with a level of candidness with which they felt safe. I considered saying something about my love of tacos but opted for a deeper level and mentioned being an ambivert.

After each individual completed the first statement, a new fill in the blank statement was offered. Each new statement took the responses to a deeper level. I don't recall the specific statements, but the progression was something like…

- If you really knew me, you'd know that ______
- As a child, what I really needed was ______
- The part of me that I miss most is ______
- When I really open up, ______
- What I really struggle with is ______
- What I fear you will see is ______

It became emotional pretty quickly and was interesting to see how far people let you inside some private areas of their life. In the debrief, it was brought up that it almost became a challenge to "out-vulnerable" the previous person. It was a great exercise to build bonds between people. I have never tried this with a team, but from my brief experience, I imagine this would be an excellent exercise to bring people closer together as long as trust and safety is incorporated from the beginning.

I have truly enjoyed incorporating these three team building exercises to build trust and form bonds. Which exercises have you found most effective for building personal bonds within your team?

Read this article online: https://baa.tco.ac/1SxQ

About Ben Kopel

Ben is the Director of Agile Coaching at Project Brilliant. He is a curious learner focused on coaching teams and Leaders, as well as enabling adaptive organizations through awareness, collaboration and continuous improvement.

He has been working with Agile teams for over ten years as a developer, Product Owner, Scrum Master, Agile Coach, trainer and manager.

Ben is a Certified Enterprise Coach (CEC), Certified Team Coach (CTC) and Scrum Foundations instructor with the Scrum Alliance, as well as an Agile Leadership Journey Guide.

Ben serves on the CEC Review Team and the Guides Advisory Team, for the Scrum Alliance.

Blog: www.projectbrilliant.com/author/ben-kopel

LinkedIn: www.linkedin.com/in/benkopel

Twitter: @bckopel

Scrum Has A Messaging Problem

By Alex Kudinov

"You are not doing Scrum." How many times have you heard that? Scrum Police are a legion. "If you are not doing `insert your missing part of the framework here OR (even better) a complementary practice`, you are not doing Scrum."

Whether or not you should be doing Scrum, whether it applies to your environment and context, whether your organizational capacity for change is capable of absorbing the shock that the Scrum framework will inevitably introduce (and that shock, in this case, may not be necessarily a bad thing): these things are not the focus of this post.

I believe that the crux of the problem is twofold: a rigid and orthodox reading of the Scrum Guide and a less than desirable level of practitioners implementing and teaching Scrum these days.

The Guide does have the immutability clause, that reads, "Scrum's roles, events, artifacts, and rules are immutable, and although implementing only parts of Scrum is possible, the result is not Scrum." It does introduce some sense of a binary Scrum / Not-Scrum state of affairs.

However, life is not all that black and white. Any Agile framework or method introduction (or implementation) is about embarking on a never-ending journey, a continuous and virtuous improvement cycle, rather than a remote and coveted destination. Unfortunately, the Scrum/Not-Scrum binary statement from the Scrum Guide could lead some people to believe that Scrum is some coveted state, some target, that one can reach, and when there, the travelers will gain some riches, and all their labors will be behind them.

As an aside, I think that "because the Scrum Guide says so" explanation belongs to the same circle of Hell where the "because we have always done it this way" kind of arguments dwell.

"You fail because you are not doing Scrum." "Scrum cannot fail; you are doing it wrong, try harder." Attitudes like these embraced by (some) practitioners and consultants are unhelpful and lead people away from agility instead of towards it.

So You Aren't Doing Kanban Either?

The Kanban Method has a different message. "Whatever you decide to do now, it's all good if it takes you in the right direction. Start where you are; improve collaboratively, evolve experimentally; resistance to the change is like a rock, be like water, go around the rock." This is the mantra of the Kanban Method.

The Kanban Method uses a set of practices that comprise the "Standard Kanban," as opposed to the "Proto Kanban." Some of the practices are visualization, WIP limits (WIPL) throughout the system, active management of flow, explicit policies, feedback loops, and continuous improvement. Proto Kanban might miss some of these practices, which does not make it bad Kanban. It just can be improved, like everything else. Kanban Maturity Model (KMM) gives an interesting perspective at the evolutionary changes a Kanban introduction and adoption can go through, and the benefits each step can bring about. We can have a long argument whether maturity models are good or evil, in my mind, again, it's all about the journey, not the destination: the KMM charts that journey.

Some will say that, not unlike in Scrum, some Kanban practices such as limiting work-in-progress are nothing short of revolutionary and that all this evolutionary talk is just a façade for the hard and fast rules and rigid principles. Those arguments miss the point. While WIP limits are a must for attacking waste (both muri and mura – overburdening and unevenness), they are not required by Kanban Method from day one; they can take various forms that will allow for easing the transition and making it a true evolution of the process.

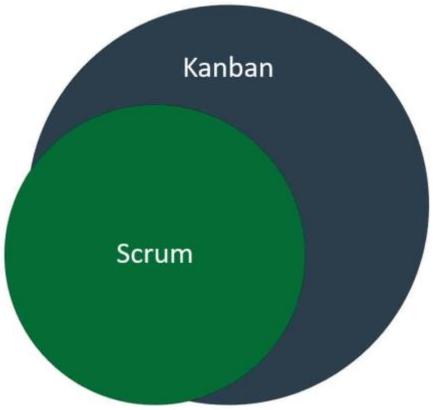
It is worth noting that in the absence of any or some of those practices, the chances are that you will not have Standard Kanban. It is also possible that your Kanban will devolve into a form of a Proto Kanban. We don't equate Proto Kanban, with something lesser than Kanban, or bad Kanban. It is just a transitory stage on the journey of continuous improvement.

What you will not hear from competent Kanban practitioners, is something like, "You are not limiting your WIP (or your policies are not explicit), so you are not doing Kanban, try harder." What they are more likely to point out is how existing practices improved your work, how they provided relief from overburdening and made the flow less uneven. What they will suggest are the evolutionary practices you can try to enhance your present state further. A lot of Scrum practitioners could benefit from a similar mindset and adjust their messaging accordingly.

Scrum and Kanban: More Similarities

Time and time again, "It's not Scrums fault, it's you," sounds to me as a defense of the framework. Please know that the Scrum framework does not need your protection. It works, it's holistic, and, applied correctly in the right environment and to the proper context, serves its purpose beautifully. What Scrum needs badly, is a lot of better-educated practitioners, who are adept at exercising their growth mindset, who can differentiate a journey from a destination, and who have a good command of sound change management principles and practices. Unfortunately, we pay insufficient attention to the latter.

Now, I am not arguing that Kanban is better than Scrum or vice versa. They both can shine given the proper environment and context. They both work beautifully together under the right circumstances. My understanding is that Kanban applies to some challenges where Scrum is inappropriate. For example, we know that Scrum is a framework to solve complex and adaptive problems. That's only one habitat from the Cynefin framework. In most cases, Scrum will be inappropriate in 3 other parts of Cynefin.



Kanban can strive in almost all of those. Kanban techniques of visualization, applying WIP limits, managing work in progress, continuously improving the workflow are a great help to Scrum teams in the Complex habitat. The framework will not work if you don't have stable, fairly cross-functional, and empowered self-organizing teams. Some organizations might not even need those (though, reading McCrystal's Team of Teams would lead you to believe that we entered the age of Teams and anything short of a team would be something of lesser quality and value). Groups of individuals coordinate their work; team members rally around a common goal and purpose. Scrum will not work for the former. Kanban will.

If I were to attempt to create a Venn diagram showing problems Scrum and Kanban can successfully tackle, it would have probably looked like this. I welcome a dialog about that diagram, since, right now, I have a hard time imagining which problems Scrum could tackle and not benefit from some complementary practices rooted in Kanban.

Scrum and Kanban are not foes, not antagonists, and not even competitors unless you talk to sellers of competing courses and certifications. We should stop conversations "Scrum vs. Kanban" or "Scrum or Kanban" right now. One is a framework to develop products while solving complex and adaptive problems. Another, with its laser focus on evolutionary change, is first and foremost, a change management method.

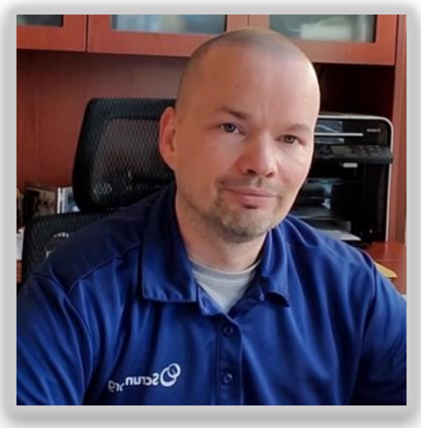
We can use some solid change management ideas and principles while introducing and working with Scrum. We can use some softer language while guiding teams on the journey to agility.

Common language and messaging are important. They underpin Transparency. Paraphrasing the Scrum Guide, "Transparency requires significant aspects of the process to be visible to those responsible for the outcome and be defined by a common standard, so observers share a common understanding of what is being seen." Moreover, "common language shared by all the participants" is specifically called out as one example of transparency.

I call on the Scrum and the larger Agile community to correct our messaging while we walk the continual path of tackling hard problems. Be less dogmatic. Think Agile. Be more like water and Scrum On!

Read this article online: https://baa.tco.ac/1SxE

About Alex Kudinov



Alex is a Technologist, Professional Coach, and Trainer. He delivers business competitive advantage to his customers by growing and nurturing great agile development organizations.

At Tandem Coaching Academy (https://tandemcoaching.academy) where we keep Agile non-denominational, Alex works with Agilists of all walks, bringing agile and professional coaching closer together.

Alex holds the International Coach Federation Professional Certified Coach (PCC) accreditation. He is also a Professional Scrum Trainer (PST) with Scrum.org, Scrum Alliance Certified Enterprise Coach (CEC), and an Accredited Kanban Trainer with Kanban University.

He holds an MBA degree from the University of Texas at Austin.

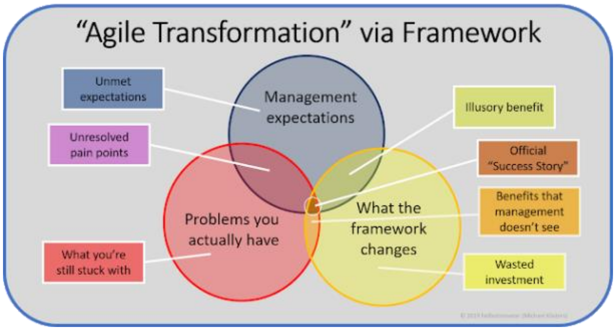
Connect with Alex at

https://lnk.tco.ac/ak

The problem with Agile Transformation Programs

By Michael Küsters

Many organizations want to "become Agile", then browse through the catalog of common frameworks, pick their favorite - and run a Transformation Program. While all of these are officially communicated as a massive success, I'd like to cast a bit of light on what is actually going on.


"Agile Transformation" via Framework
Unmet expectations
Unresolved pain points
What you're still stuck with
Management expectations
Problems you actually have
What the framework changes
Illusory benefit
Official "Success Story"
Benefits that management doesn't see
Wasted investment

Transformation Input

There are some "givens" that affect a framework-based Agile Transformation program before it has even been conceptualized: expectations, reality and the determined future state as described by the framework.
These are the constraints to the success of the transformation, and depending on how well they overlap, this success can be bigger or smaller. Worst case, this intersect is empty from the beginning - in which case, the transformation program is doomed.

Management Expectation

Typical management expectations from an Agile Transformation include, without being limited to:

- Faster time-to-market
- Lower development costs

- Higher Quality (fewer defects)
- Improved customer satisfaction
- Happier employees

The choice of framework often falls to that which promises most of these benefits.
"Good" transformation programs then set targets based on improvement seen on these metrics.

Unfortunately, to scope a proper project and/or program, the real work is oftentimes measures in amount of departments/projects/employees using the chosen Agile Framework "successfully" (whatever that means).

Real Problems

Usually less visible to management is the entire quagmire of problems the organization has accumulated over the years. Benefits are generated by getting rid of problems, they don't appear from thin air.
The more problems are known and the greater the pain to solve them, the easier it will be to actually get benefits out of a transformation.

Cultures averse to admitting or taking responsibility of problems will be struggling to gain actual benefits from any "Transformation Program", regardless of whether it's agile or not.

Framework Scope

Frameworks themselves have a very clear scope, mostly concerned with structure - roles and process, events and some artifacts. We can easily determine how much of this structure has been implemented, and that's how success is often measured.

What's significantly more challenging: determining how compatible people's mindset and behaviour is with the intent of the framework, and how significantly the "new ways of working" get impacted by "old ways of thinking and doing".

Transformation Output

To keep this article simple, let's not argue about how well any of the inputs was understood or change was actually realized, and keep to reality - "*we are where we are, and we go where we go.*"

This reality is:

- some expectations will be met, others don't.
- some aspects of the framework will be implemented perfectly, other won't.
- some problems will be solved, others won't.

Another reality is that at the point in time when the program is conceptualized:

- some expectations are based on known problems, others on unknown problems.
- some expectations are based on understanding the framework correctly, others on understanding them incorrectly.
- some program activity will be planned to do things that solve meaningful problems, others will focus on something that's not a root cause.
- some framework components will lead to beneficial change, others won't.

... and we can't know which is which, until we have done some experimentation.

For argument's sake, let's just assume that the program is sufficiently flexible to allow such experiments, and that everyone contributes to the best of their understanding and ability.

Programs are still time-bound, and it doesn't matter whether that timespan in 1 month or 5 years. Within this period, a finite amount of activity will happen, and this activity will lead us to wherever it does, and "not everything" will be perfect. And this is what the future reality will look like:

Outright failure

Some aspects of transformation will lead to success, others will fail to provide success - or even distract from success. Let's call things by their name: When you scope a transformation program and don't get something you planned to get, that's failure.

In this section, I want to highlight the failures your program will have.

Unmet expectations

There will be a number of management expectations that haven't been met (blue area outside intersects). Some may have been unrealistic from the outset, others "could have ... should have". Regardless, someone will be disappointed. Managers familiar with diplomacy and negotiation will stomach the ordeal, knowing they got something at least.

Just be careful that the higher your expectations are and the less aligned they are with the framework's actual capability, your organizational reality and the flexibility of the transformation program, the bigger the disappointment will be.

Wasted Investment

Frameworks are frameworks, and when shown an overview of everything that the framework has to offer, managers often decide that "we need all of that". Truth be told, you don't, because a lot of it provides a solution to problems you don't have (yellow area outside intersects). But you can't know *what* you need until you are in a situation where you *do* need it.

By deciding upfront to go full-scale into implementing everything a framework has to offer, you're going to load a massive amount of waste into your transformation program - and this waste costs time, money and opportunity.

Unsolved problems

Many of the problems in your organization won't get addressed at all (red area outside intersects) - because they're unkown, too complicated to resolve or simply not relevant enough.

The intent of an agile framework isn't to solve your problems, but to provide you the means of solving them - you still need the heart, the will and the power to actually do this.

Great transformations focus on tackling meaningful problems, thereby showing by action that resolution is possible and valuable - bad transformations avoid the mess of problem solving and focus on just covering the existing heap of problems under the blanket of a framework.

Unresolved pain points

Managers would prefer to have the perfect organization where everything is smooth and problems don't exist. But we don't live in cockaigne (purple intersect between blue and red on the left), and "Agile" won't create one, either. Problems are still real - and frameworks don't address them directly, they just provide means for addressing them.

The list of pain points is (near) endless, and seems to grow with increasing transparency. and we only have a finite amount of time. There will be un-addressed pain points. Even if the Agile Framework is perfectly implemented, many of the pain points will remain - most likely, more than imagined.

When a transformation program scopes more framework implementation than problem solving, don't be amazed if the outcome is more structural change than solved problems!

Partial Benefits

Transformation programs can and do provide benefits, in different categories:
What you see and feel, what you see but can't feel, what you feel but don't see - and what you neither see nor feel, although the latter is a difficult topic in and of itself.

Illusory benefits

Informed managers will expect the transformation program to implement certain framework elements that will indeed be implemented (full intersect between blue and yellow circle). This is great when these elements actually solve a problem the organization has - but there's no guarantee (greenish intersect between blue and yellow on the right). Oftentimes, we create "change for change sake" without getting any better results, because we changed something that's not a problem.
In some cases, the new status quo is even worse than the former state, but it looks better ... because it's new, and compatible with the Agile Framework.

These benefits are not real, they have cost money and kept people from doing the right thing!

Let me warn you right here: Unethical coaches/consultants will focus their efforts on the illusory benefits, to build management rapport and milk the cash cow as long as possible.
AVOID generating benefits in this category!

Hidden benefits

The framework may actually solve some problems that management isn't even aware of (orange intersect between red and yellow on the bottom), either because the benefits take a long time to become visible or because they do not affect anything of management relevance.

A typical example is the implementation of XP engineering practices - it may actually look like teams are getting slower when they write unit tests and create deployment automation, but the benefits will become visible in the future, as defect rates and stress levels decline. Example: Developers who have worked on Clean Code microservices with Continuous Deployment never want to go back to legacy processes or code, because it's so much easier and faster to work this way - but getting there could take years (and possibly be entirely unfeasible) on legacy systems.

Let me dive in with another note of caution: Ethical coaches/consultants will focus their efforts on solving real problems, many of which managers don't see. Managers must be curious to learn from their teams whether such hidden benefits are being generated, or whether the consultant is just trying to please management.

The success story

Every organization that has invested lots of effort and money on an agile transformation program will eventually produce a success story (brown intersect area of all circles in the center) based on what management expected, how the organization actually benefitted and how the Agile Framework has brought them there.

People are smart enough to not reveal publicly how many of their initial expectations weren't met, how much activity didn't lead the organization in a better direction and how big their problems still are. But depending on how well the Agile Transformation Program was defined and executed, this could easily be some 90% (or more) of the program.

Simply put, it's insane to start an Agile Transformation Program because you read someone else's success story, because the story doesn't tell you what went wrong, how much disappointment and frustration has accrued, how much time and money was wasted - and oftentimes, you don't even see the pointers of what actually made it a success.

Real success happens where the three items intersect, and the size of this intersect is determined by:

- How well do you focus on solving real problems?
- How flexible are your expectations?
- Are you using frameworks where they benefit, rather than making your organization framework-compliant?

Summary

An agile framework transformation program, conceptualized, planned and executed by people who do not exhibit an agile mindset and who do not practice agile organizational development - is going to produce a politically motivated, insignificant success story.

> Stop thinking frameworks, stop thinking programs.
> Start thinking agile, and embark the agile journey in an agile way.

Read this article online: https://baa.tco.ac/1SyM

About Michael Küsters

Michael works with organizations of all sizes as they embark upon their Agile Journey.

In the spirit of the Manifesto for Agile Software Development, he constantly strives to “find better ways and help others doing so.”

On this mission, he works with leaders, managers, teams and others alike to bring everyone together in creating a new, better working environment.

He records some of his experiences on his blog
www.failfastmoveon.blogspot.com

Has published various works on Leanpub under
www.leanpub.com/u/michaelkuesters

And has authored the “Scream Guide” and “Everything Wrong With Agile”, both collections of common antipatterns to inspire reflection.

To learn more visit

www.linkedin.com/in/michaelkuesters

Falling in Love with Agile (Again)

By Diana Larsen

Where did our love go?

In the immortal words of The Supremes,**“Baby, Baby, where did our love go?”**

(ICYMI, The Supremes were a very popular R&B group in the last century. I recommend them.)

If you’re like me, when I first met Agile, it went something like this.

Delivering real value to customers by nurturing collaboration and short feedback loops. More of that, please!

The idea of team members having genuine interactions appealed to me. I swooned over interactions that helped teams iterate toward constructive work processes.

I fell hard for low defect deliverables that would support the users as long as they needed it. And the business success that resulted–Dreamy!

In short, I fell in love.

Things Change

Somewhere along the line, Agile changed, as our loved ones sometimes do. Organizations began substituting tracking tools for team communication, and still called it Agile. Responding to squeaky wheels took precedence over fostering great relationships with customers. And they still called it Agile. The drive to produce faster and faster swept away long term maintainability. And they said it was Agile. Healthy team workspaces took a back seat to cube farms or vast open plans. An insistence on competitive individual performance reviews rolled over teams.

They vowed it was all Agile. People I respected began talking about the Agile Industrial Complex.

And who can love that?

I wondered, where did the things that attracted me to Agile go? What happened to the patterns of Agile that made me (us?) fall in love with it?

I want to reclaim the seductive blend that made my relationship with Agile so great. I yearn for the return to an emphasis on effective productivity, a humane standpoint, and sustainability–for products and people. We **can** find the Agile we fell in love with, then remember our affection, and embrace Agile again.

The Enticing Appeal of Productive, High Performing Teams

I'll admit it. I get moist-eyed over videos and stories about teams accomplishing something great together. The first time I learned about cross-functional, self-organizing, high performing teams, I was hooked. When I had the chance to work with such a team, I leaped at it and took the bait. (and that's where this fishy analogy ends)

Suffice it to say, I wanted more of those stories and experiences in my life. I felt an immediate kinship when I met the early Extreme Programming proponents. Then I introduced myself to Scrum, Crystal, DSDM, and the rest of the Agile family. For me, there's something enticing about the outcomes of great collaborations and team interactions.

For many Agile coaches, working with teams requires the ability to look at the team as a "whole entity." It's akin to Mr. Spock's, "the needs of the many." It's an art and a science. Practitioners move their focal point to a whole team. At the same time, they maintain a connection to individual team member's well-being. It's a tricky balance to achieve. And, well worth the effort.

In 2010-2012, James Shore and I researched and developed the Agile Fluency™Model. We realized competence with collaborative team skills formed an essential foundation for effective performance. No matter which of the Agile approaches teams adopted. In the model, our "Focusing" zone describes teams with these skills. Fluent "Focusing" teams make two additional shifts. First, from technical targets to business value focus. Second, from devoted attachment to a plan to the ability to divert from a plan toward a new course. When that new course promises greater value to the customers and the business.

When I work with a productive "Focusing" team, I'm captivated by their collaboration. I find reasons to hang out with them. Can you tell I get a crush on these teams? To paraphrase, I think highly of them, I greatly esteem them, I like them. (That's for the Jane Austen fans among the readers.)

The "Focusing" zone is the base zone on the path that forms our model of fluency. Even so, these teams have made the shift to an Agile mindset and outlook. They seek greatness as a team. Working with them, I fall in love with Agile all over again. (For more on Agile Fluency high performance coaching, visit us our Website.)

Read this article online: https://baa.tco.ac/1Sxr

Keeping the "Up" in Daily Standup Meetings

By Diana Larsen

Arlie's team has appreciated getting a full-time coach. Team members have had a first few rocky months of adopting agile practices for the first time. They thought a skilled coach could help them learn faster. The product and engineering manager have set a goal of Focusing zone fluency for the product development teams. What's more, they've made investments to support the team. One of those investments resulted in settling teams in dedicated team rooms. Another was hiring or assigning coaches to work with the teams. As a fairly new coach, Arlie has wondered whether they and the team are getting things right.

Arlie contacted me with a question about how to help team members stay engaged during daily Stand Up meetings. The team has begun to experience the kind of StandUp "downers" that happen on too many teams. No clear questions. Responses that are too long or too rambling or both. Inadequate understanding about the "why" behind the practice. Starting to wonder about having the meetings less often than daily. And answering the same three questions every day has begun to seem...well...uninspiring.

Are you having "Stand Downer" meetings too?

I reminded Arlie that StandUps offer the most benefit when they sustain communication about the state of the work. They work best when team members keep their responses succinct. Strive for brief (30-60 seconds per team member), frequent, and to the point. It helps when team members know the questions ahead of time and come prepared. I shared a suggestion I learned from James Shore; each person can write their answer on an index card to bring to the meeting.

What's more the “three questions" don’t need to always be the same. For fun and variety, and to keep it engaging, insert new questions from time to time. For example, switch it up every other week, every other iteration/sprint, monthly, or whenever the team feels the need for something new. As you try new questions, make sure they stay focused on the work.

Some possible variations:

Set A. (The Classic)

1. What story did you work on yesterday? Who worked on it with you?
2. What story do you plan to tackle today? Who will you work with on it?
3. What obstacles, if any, do you anticipate to finishing?

Set B. (Shared Learning)

1. In the work you did yesterday, what did you learn that could help the whole team?
2. What do you hope to learn today? How will you share it with all of us?
3. What gets in the way of your learning?

Set C. (Finding Help)

1. What helpful resources (e.g., websites, books, articles, repositories, team member expertise, etc.) did you access yesterday for your work?
2. Where will you look for help today?
3. When have you found it difficult to find helpful resources? What gets in the way?

Set D. (Achieving the Plan)

1. How did you help the team move toward achieving our iteration plan yesterday?
2. How will you help us move forward on the plan today?
3. What will impede your progress?
4. On a scale of 0 (no way) - 5 (super confident), how confident are you that we will complete all the work in our iteration plan?

Set E. (Continuous Integration)

1. What did you commit yesterday?
2. What do you hope to commit today?
3. What hinders your ability to continuously integrate your work today?

You and your team can brainstorm new sets of the three questions too. Ask one question that takes a quick look at immediate past experience of something that supports the team's work together. Ask the second question about the intentions for the day for that supporting behavior. Choose a third question that makes it easy to surface things that are getting in the way. You can see this pattern in the Question Sets above.

Note Bene 1: The path to removing any obstacles, impediments, hindrances, or problems should be discussed separately, by relevant parties only, outside of the standup meeting. This can happen just afterwards or later, but it's not part of the daily StandUp.

Note Bene 2: In addition, if your StandUps are suffering from any of the dysfunctions listed above, the answer is to make them even shorter and more frequent. I worked with a high-functioning team of 8 people that held two StandUps a day. Each meeting was no more than 5 minutes.

This way we all stayed updated on what the whole team was working on and our rate of progress toward our outcome. We were each able to jump in to help when our particular skills were most needed. It added to our effectiveness and our efficiency. When we saw that our progress was slowing, we increased to three daily meetings: early morning, after lunch, and end of day, so we could even better stay in synch. Still each under 5 minutes.

You can do it! Arlie's team has.

Download your own copy of the new Agile Fluency Model eBook.

About Diana Larsen

Diana is a co-founder, and principal mentor at the Agile Fluency® Project.

Diana co-authored the books Agile Retrospectives: Making Good Teams Great; Liftoff: Start and Sustain Successful Agile Teams; Five Rules for Accelerated Learning.

She co-originated the Agile Fluency® model and eBook, The Agile Fluency Model: A Brief Guide to Success with Agile.

For 20+ years, she guided Agile transitions for FutureWorks Consulting.

Through Agile Fluency Project's programs for training, mentoring, and supporting agile coaches and consultants, Diana shares the wisdom she's gained in over 30 years of working with leaders, teams, and organizations.

What Are the Limits of the Scrum Framework?

By Mark Levison

Frequently in workshops, I get asked, "Where shouldn't we use Scrum?" The short answer is there are lots of instances where the Scrum framework doesn't fit. However, to give a more complete and effective answer to this question, first we need to have an idea of why and when Scrum does work and what the key conditions are for success. We can then show examples of where it isn't a good fit.

Where is Scrum Applicable?

Scrum is a tool for building autonomous, self-organizing, high-performing teams and organizations which can successfully respond to changing business circumstances. People often choose to use Scrum because they want higher quality or greater speed, not understanding that these are outcomes of high-performing teams and not of Scrum itself.

Scrum has been used effectively with teams in a diverse array of industries, including Software Development (where it grew up), Hardware Development, Manufacturing[1], Marketing[2], HR… even Fighter Planes[3] and Gas Plant Design[4]! What these very different industries have in common is they rely on a form of knowledge work, which can be understood as work that primarily involves non-routine problem-solving that often needs creative thinking. Knowledge work benefits from collaboration, which is the primary focus of Scrum Teams, so it's no surprise that Scrum is well-suited for these industries.

Since teams are the core work unit of Scrum, many of the limits of Scrum come from the focus on how an organization's teams are structured.

Key Conditions for Scrum to Work Well

Understanding now where Scrum is effective, we can consider what structural foundations are needed for it to function well in a given work environment.

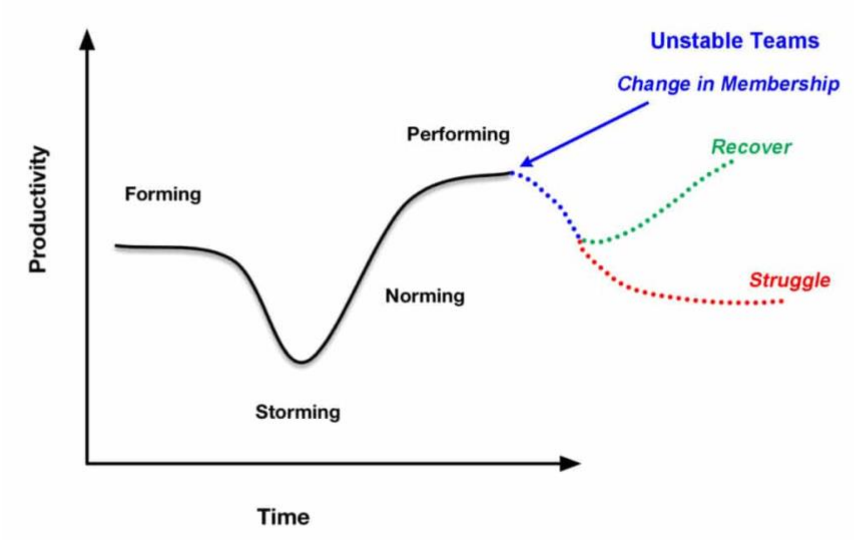
Knowledge Work – It may seem obvious but it's worth stating: organizations (including most retail and service industries) based around the performance of routine tasks that do not require complex problem-solving or creative thinking will not benefit from using the Scrum framework.

Common Goal – A group of people only becomes a "team" when there is a common goal or target that they're attempting to achieve. In Software Development, the common goal comes from the Product Vision and Strategy. In a Marketing team, this might come in the form of a brand strategy or marketing campaign plan. Whatever the industry, **the goal must unite the work of all team members to something greater than their individual contributions.** Without a common goal, there isn't really a *cohesive* team, and cohesion is key.

Sufficient Challenge – In tandem with a Common Goal, the problem must be challenging enough that people can't get the job done if they work alone. If people can work without interacting with others and still achieve their goal, they will often choose to do that. The difficulty of the problem must warrant the use of teams, otherwise Scrum is just unnecessary overhead.

Dedicated Team Membership – The costs of multi-tasking have been documented on numerous occasions and it's no longer considered a desirable skill like it once was thought to be.[5] Assigning people to work on more than one team forces them to multi-task, making them less productive. Any form of multi-tasking causes people to lose capacity. Assigning them to multiple teams is just the worst case. Their capacity is reduced due to the delays and costs to focus as they switch gears from one context to another and the teams suffer bottlenecks as they wait for their turn with that person. Ultimately, the number of errors will increase, in part because these individuals switching context will forget key details. Evidence shows that dedicating people to one team, and only one team, doubles the throughput of all teams involved.[6] Without Dedicated Team Membership, all groups are destined to a lower level of productivity and true teams – certainly not high-performing ones – never form. Scrum would be significantly shackled in this environment.

Stable Team Membership – Effective teams take time to form. It takes 6-8 months before a new team develops the cohesion necessary to be truly effective. Furthermore, every time there is a change in membership the team takes a temporary drop in productivity. After a few months, they may regain strength, and may even eventually improve, however some teams never recover. In situations where there is frequent changeover in team members, the team will always be stuck at a lower level of performance.



Tuckman's stages of team formation (copyright: Agile Pain Relief Consulting 2014-18)

Low Cost of Change – Agile came of age as the cost of making changes in software was being drastically reduced. Much of the work in the years since has been focused on further reducing the cost of making change – from Continuous Integration and Test Driven Development, to DevOps and Behaviour Driven Development. The cost of change in modern software development work isn't zero, but it is considerably lower than the green screens[7] and mainframes. For any flavour of Agile to succeed in a new environment, reducing the cost of change (even late in the game) needs to be a priority. In work that incurs a significant cost if changes are made, Agile/Scrum isn't going to be as practical.

Plannable – Scrum Teams normally work in two-week Sprints, so they need to be able to plan their work at least that far in advance and allow for accommodating small changes. For example, a software development team gives itself enough flexibility that it can fix a production support issue mid-sprint without derailing the main work of the Sprint. A marketing team could adapt their campaign in response to new data it received about customer behaviour. The practical limit is that at least half of the teams' work needs to be plannable. Companies whose entire business model is to respond to unpredictable client needs won't benefit from using Scrum to organize future work. *Caveat: outside the repair industry there are few businesses that survive in the long run on a purely reactionary basis.*

Empowered – Teams can only form when team members feel that they have autonomy. Scrum makes this explicit by establishing the team as self-organizing. Hopefully, this is never a key condition that is lacking but, if it is, trying to apply Scrum wouldn't help the team become self-organizing and efficient, but it may expose the problem, so it can be resolved before continuing further.

Cross-Skilling is Possible – Scrum (and Kanban teams) eliminate bottlenecks by sharing skills until delays can always be addressed by multiple team members. Bottlenecks are such a fundamentally important barrier to high performance that organizations eventually must address them. Toyota's approach is to pay people more to be able to handle multiple stations. In healthcare, there are some jurisdictions starting to address the issue by allowing some work previously only done by doctors to now be done by nurses or nurse practitioners. At the same time, there are some work environments where, due to regulation, law, or radically different skill areas (e.g. artist and metal worker), cross-skilling may be limited in its applicability or value, limiting the value of Scrum as well.

Early Delivery and Testing – In Scrum, we don't assume our expectation of the User needs are correct. Instead, we prefer to deliver products early and gather feedback. We operate in a mode of Product discovery rather than creation. In an environment where we fail to deliver a working product at the end of every Sprint, we are unable to gather feedback. The solution is to find something to show and gain feedback on at the end of every Sprint.

Co-location – Having all team members in the same room continues to be the best choice. Humans have evolved over millions of years for face-to-face interaction, so this is still the best way to build collaborative teams. While remote work and distributed teams are currently trendy in many businesses, the challenges these create are significant and result in high-performance taking longer to grow in the Team. If distributed teams are absolutely unavoidable, applying Scrum practices (e.g. daily stand-up) will be more challenging and require adaptations. It's still possible to practice Scrum effectively in distributed teams – it's just much harder.

Examples of Where Scrum Isn't Ideal

So now that we understand why Scrum works and its conditions for success, these are in order from teams where Scrum is the least likely to help, to where it will be challenging but may still offer some benefits.

C**onstruction –** When a team is tasked with pouring concrete or paving a road, the cost of change is effectively the cost of the work. Agile approaches, in general, can still work, but it comes with an increased cost. Consider Lean Construction and both the Empire State Building[8] and the London Shard[9] of examples of this approach.
Help Desk and Repair Services – When people in an organization provide digital or phone-based support services, their work does involve the **Knowledge Work** key condition, but it's entirely driven by interruptions. They can start the day working on one issue, but when a

call comes in with a more important issue then they must switch. This pattern can repeat itself several times during the day. This violates the **Plannable** condition, so Scrum won't be effective in this context. Consider a different tool that improves the flow through any system, such as Kanban. This may also apply to other organization and services – essentially anywhere that knowledge is the primary requirement, but the work is not plannable.

Back-Office Operations – Many organizations have groups that do all the background work such as finance and other related departments. Most of this work – invoices, expense tracking, and other bookkeeping – is done effectively by individuals working on their own. While the work is knowledge-based, it is often repetitive and so wouldn't be **Challenging.** These groups often lack a coherent vision or **Common Goal.** Consider Kanban instead of Scrum again, as a tool to better understand the flow of work inside these groups.

Infrastructure and Technology – Most organizations also have people who configure laptops, servers, security, networks, firewalls, and other technical infrastructure. This knowledge work is less repetitive and more **Challenging** than back-office work, and it benefits from collaboration because problems often require more than one skillset to solve. These groups also typically have a **Common Goal** (e.g. keeping internal users productive and safe). But in many cases, their unplanned work exceeds their **Plannable** work. However, Scrum might help by bringing into focus the volume of the unplanned work for the team, helping them put effort into reducing it.

COTS Apps[10] – Organizations often outsource a lot of their backend software (think: Gmail, QuickBooks, Expensify) to cloud vendors. This is fantastic, but eventually, there are enough applications each that requires occasional changes (new user, new accounting rule, etc.). None of these applications requires one person maintaining them as a full-time job, so you might end up with 6-7 people maintaining 10-15 individual applications. Lacking a clear **Common Goal**, this group is unlikely to become a Team. Scrum could work but the value might be limited. *The challenge for both the infrastructure and teams is that their knowledge* *is likely to stay fragmented since there is little reason for people to learn another team member's area. Whether the team chooses Scrum, Kanban, or some other framework, this will likely remain an issue. Consider asking if the group could be reorganized to create a space where a Common Goal is possible. An alternative option is the team could work to create a goal that is possible in their circumstances. These notes are inspired by a conversation with Petri Heiramo:*

"Given the ”product” they work on is clearly not sufficient to rally them, the discussion would need to be shifted to something greater than the work they are doing. Would they want to become the best support team ever? Would they want to do whatever they are doing in half the time? Would they want to NOT do their work and try to automate as much of it as possible? Do they know whose lives they are making better and could they possibly derive some worthy goal from that end?

One possibility would be to ask them how happy they are at work. If they're not happy, the next question could be are they willing to put effort into making themselves happy. After all, their three main choices are 1) keep doing what they are doing and stay unhappy, 2) do something to make themselves happier and proud of their work, or 3) leave the company for something else. Obviously, 3) is not desirable and not a great starting point, so the choice should really be between 1 and 2. Then the next step could be to ask them what makes them happy and proud at work, and/or what makes them unhappy and ashamed. This could help them establish a shared goal of becoming happier and discover a starting point for corrective action. There could be a discussion of how to measure happiness and pride (i.e. how to know they're making progress toward their goal). There could be an agreement of some self-reward (like beer on Friday) whenever they've achieved some concrete improvement in their work that they are proud of such as:

– removing the deepest information bottlenecks that prevent them from taking holidays without stressing about work.

– clearing up their worst technical problems.

– making a grumpy customer happy or even delighted with their team/service.– establishing some new practice to improve productivity, or reduce feedback cycles.

Individual Work – any business problem where there is no prospect for collaboration (company of one or everyone works in isolation), won't benefit from Scrum directly. After all, there is no team to grow. In that circumstance, consider Personal Agility and Personal Kanban.

Team Members **Matrixed** onto multiple teams and **no Stable Membership** – I can't imagine a business circumstance where this is desirable. If this is the scenario, Scrum will work quite effectively to highlight the organizational impediment, but not to help manage the problem. In every workshop, we discuss the fact that Scrum is a tool for finding problems. Scrum succeeds in the long run when the organization takes seriously the need to resolve issues like this that Scrum finds, not just manage them.

There are limits to what we can use Scrum for, but they're much wider than most would imagine.

Thanks to Petri Heiramo for suggestions and the visioning exercise for the COTS team.

[1] Webinar – "Joe Justice on Agile Manufacturing”
[2] *Hacking Marketing* by Scott Brinker
[3] “Owning the Sky with Scrum” – Saab Designs Fighter Planes with Scrum
[4] “Designing gas plants with Scrum” by Simon Orell
[5] Multi-tasking and Interruptions resource links and“Multitasking Gets You There Later” by Roger Brown
[6] “The Impact of Agile Quantified” by Larry Maccherone
[7] Not familiar with green screens? See: https://www.quora.com/What-is-a-green-screen-application and https://en.wikipedia.org/wiki/IBM_3270
[8] “The Tyranny of ‘The Plan’”Presentation by Mary Poppendieck on the construction of Empire State Building
[9] “The Shard: Foot of the mountain” by Stephen Kennett
[10] https://en.wikipedia.org/wiki/Commercial_off-the-shelf

Image attribution: Agile Pain Relief Consulting

Scrum by Example – Team Friction Inspires Working Agreements

By Mark Levison

Working Agreements are a simple, powerful way of creating explicit guidelines for what kind of work culture you want for your Team. They are a reminder for everyone about how they can commit to respectful behaviour and communication. In this post we'll help you understand why these agreements are useful, and how you can help your Team create their own.

Dramatis Personae

- Steve – a ScrumMaster and the hero of our story
- Paula – the Product Owner of Steve's Team
- Kirby – one of the Team's software developers
- Tonia – the Team's Quality Assurance specialist
- Doug – another of the Team's software developers
- Ian – the Team's Business Logic programming specialist

In the past few Sprints, there have been moments of social friction noticed by both Steve and the rest of the Team. They included:

– Kirby was blunt when discussing a defect with both Tonia and Ian.
– Some discussions with the whole Team became so heated that the quieter team members stopped speaking up for several days.
– Doug was late for Daily Scrum on several days.
– On multiple occasions Team members needed to ask Paula some questions but she wasn't available by email and they often didn't know how to find her.

None of these individual issues was truly bad, however, taken together and left unchecked, they harm the Team's cohesion. When that happens, people opt to stay silent when there is an opportunity to share an opinion, and events like Daily Scrum becomes less useful when Team members signal their disrespect by frequently being late.

During the next Retrospective, Steve mentions a few of these issues and says that he wants to talk about making their environment better. Tonia shares that she knows another one of their peer teams has created their own Working Agreement to reduce some of the friction and improve respect.

What is a Working Agreement?

A Working Agreement is a short set of guidelines created by the Team, for the Team, that establishes what the expectations of the Team are for one another. A well-written Agreement should help establish, as well as reinforce, a clear, shared understanding between all Team members about what they acknowledge is good behaviour and communication. It is usually referred to as a single "Working Agreement", but in reality it's comprised of many individual agreements for each topic or issue.

Characteristics of an Effective Working Agreement

- **Public and Visible** – preferably written in a large font and posted in a prominent space
- **Collaborative** – created by all, not imposed by others
- **Short** – a small list of agreements that are easily remembered and lived up to trumps a big list that overwhelms people and gets forgotten.
- **Updated Frequently** – Taiichi Ohno once said: "If the kanbans do not change for one month you are salary thieves."
- **Consequential** – when the agreements are violated, Team members call out the violation.

A Working Agreement doesn't have a police force to ensure that it's followed. Its effectiveness is tied to its: Visibility (display it on a wall), Self-Discipline (awareness of our own behaviour), and Respectful Reminders to peers when trespassing occur

What Goes Into a Working Agreement

Your Team will create its own Working Agreement based on the context of its work. Elements of an Agreement *might* include:

- Core hours – times during the day that all Team members agree to be present and available for collaboration
- Daily Standup – the time that it happens and what to do if you realize that you can't make it that day
- Cell phones – if or when it is appropriate to use them during Team events
- Headphones – when it is fine to use headphones, versus when Team members will not use them so they're available to communicate and collaborate with others
- Product Owner – availability and contact information
- Definition of 'Done'
- How the Team limits Work In Progress (i.e. the Kanban Concept)
- How the Team ensures everyone has an equal voice and feels respected
- If you miss a meeting or event then you agree to support decisions made in your absence.
- If you can't attend a Team event you will let the Team know in advance.
- During each Sprint, at least one Team Member will work on growing a skill outside of their core strengths.
- In the case of software development, every Sprint the Team will make an effort to improve some part of their code base.

As part of coming up with this list, it is worth considering the **Scrum Values** as a source of guidance:

> **Focus:** Because we focus on only a few things at a time, we work well together and produce excellent work. We deliver valuable items sooner.
>
> **Courage:** Because we are not alone, we feel supported and have more resources at our disposal. This gives us the courage to undertake greater challenges.
>
> **Openness:** As we work together, we practice expressing how we're doing, and what's in our way. We learn that it is good to express concerns, so that they can be addressed.
>
> **Commitment:** Because we have great control over our own destiny, we become more committed to success.
>
> **Respect:** As we work together, sharing successes and failures, we come to respect each other, and to help each other become worthy of respect.

Teams normally create a Working Agreement before their first Sprint by setting aside an hour or two beforehand. In the case of our World's Smallest Online Bookstore (WSOB) Team, they haven't yet. No time like the present!

Building a Working Agreement – How Can a ScrumMaster Help?

Sensing the atmosphere of the Team, Steve starts by helping them learn about Working Agreements. He first ensures that the Team understands what Working Agreements are, and how they will personally benefit from having them.

Mark's note: Spending some time on a good setup will reduce cynicism among Team members. Giving people some categories or areas in which Working Agreements will help give them a framework. For instance, in our workshops, categories are cell phones, laptops, break times, and punctuality. In the context of a Team, you might use some of the areas mentioned above, and perhaps draw on the Scrum Values, but ultimately the Team chooses what works best for them.

Steve then facilitates a Retrospective to help the Team and Product Owner come together to sort out some of their problems. During the process, they discuss the friction they have felt over the previous few Sprints. They agree to create a Working Agreement and, since it will likely take over an hour, they elect to make it an action item for the next Sprint Retrospective.

There isn't an official or correct way to create Working Agreements, so Steve uses the approach that I share in my workshops. As usual for a ScrumMaster, good preparation pays dividends. Consider canvassing the Team beforehand about categories/areas for agreement.

When facilitating the creation of a Working Agreement:
Check-In. Start with a prompt question such as, "If we create Working Agreements, how will that help the Team work more effectively together?" or "What are the decisions that, if made now, could help us in the future when the pressure is on, or things haven't gone our way?"[1] Get everyone to answer, and challenge them to try one-word answers. Explain the format you will use and any ground rules that you have in place. For example:

- Voting rules. *Mark's note: Mine is typically: assume positive intent.*
- If you say 'no' to a proposed item, you're expected to try to make it better.
- Use the Decider Protocol:[2] Thumbs Up – I'm good; Sideways – I can live with it; Thumbs down – I will block this proposal.

Break groups larger than five people into sub-groups. In my experience, it's easier to get small group agreement first, then bring it back to the whole.
Invite the small groups to create a potential Working Agreement in each area or category. Invite them to create any additional Working Agreements they feel are needed to address concerns or conflicts.

At the end of 5-10 minutes, get the groups back together and review the proposed items.

After a couple of rounds of proposals, if there isn't any consensus on a particular item, move on —they can't establish agreement in that area for now. Consider revisiting the item the next time Working Agreements get deliberated.

Every few Sprints, the Working Agreement should be updated, often by checking it in Retrospective and asking a question like, "Are these still our working agreements? What would we like to update? What areas need new agreements?"

After talking to Team members, Steve selects "Daily Scrum Start Time," "Respect," "Phone usage in Team Events," and "How to give everyone equal voice for discussion." He explains that if there are other categories that they would like to consider including in the Working Agreement for, they're welcome to add them as they go.

Given the previous friction between some Team members, he opts for a 1-2-4 model[3] for discussing possible agreements. This model is designed to ensure that everyone has a voice in the process:

- Each Team member (1) takes a few minutes share to the whole group their ideas for potential agreements in each category.
- They then work in pairs (2) to record the best ideas from each person. As facilitator, Steve is paying attention to the

conversations and outcomes. It should be a red flag if one person is dominating their partner during this step.

- Next, they move to groups of four (4) to further consolidate their ideas and present them to the group for discussion. In the case of Steve's Team, there are only six people so they skip this step and move straight to final group consensus.

Steve starts asking for proposed agreements in their first area of focus: Daily Scrum Start Time. After each potential working agreement, he uses the Decider Protocol[2] to check for consensus rapidly. When there isn't immediate consensus, the person who said 'no' to an idea suggests what they think is a better one. If multiple people have an issue, then each is expected to offer a better idea. If too many people say no, then the proposer should consider withdrawing the proposal. In the case of Steve's Team, after 20 minutes, the team have their first set of Working Agreements:

– Respect – when another Team member is talking, don't interrupt. When they've finished, pause, reflect, and share back your understanding of their idea.[4]

– Make sure everyone has a voice – more vocal Team members volunteer to speak last to give quieter Team members an opportunity to speak up.

– Daily Scrum is 9:20 am, allowing all Team members to get their kids to school before coming in to work.

– Cellphones in Team Events – they stay outside the room.

– To improve the quality of all ideas we encourage respectful dissent. At least one Team member volunteers to be the dissenter. Even if they think the original idea is good, they're expected to find criticisms of the idea with the goal of making it stronger.[5]

– Focus – during the Sprint the Team is focused on the Sprint Goal. If something comes up that doesn't fit the goal, Team Members are expected to say no.

Notice that the last two items weren't among those previously suggested by Steve, instead they appeared *organically.*

Steve then offers a suggestion of his own: "If you miss a meeting/event it is expected that you will support the decisions made there." Kirby, who has previous experience facilitating, reminded Steve that if he truly is playing the role of facilitator as part of his ScrumMaster duties, then he is expected to remain neutral and not inject his own ideas.

The Team compiles all the individual agreements in the Working Agreement and posts it on the Team room wall. In the months afterwards, Team members slowly get used to the idea of reminding their peers of behaviours that don't honour the Agreement. Every few Sprints, Steve asks in a Retrospective "Is this still our Working Agreement? Is there anything you would like to change?" The list evolves as Team members find more areas where they see benefits. After six months, they find themselves much better able to deal with tense problems within the Team, or when the outside pressure increases on them.

Scrum by Example is a narrative-style blog series designed to help people new to Scrum, especially new ScrumMasters. If you are new to the series, we recommend you check out the introduction to learn more about the series and discover other helpful articles.

Want to learn how to put together really effective Working Agreements? If there is one idea at the heart of Scrum and Agile that is most important, it is respecting the people you work with. When you empower teams to make decisions about how they work, you use one the most powerful methods to help them be more effective and become high-performing.Working Agreements are a great step toward that improvement. In our Certified ScrumMaster workshops, attendees discover how to support and facilitate these kinds of agreements. If you are committed to helping create a respectful environment for your team, we invite you to join us to learn how and to gain practical, hands-on experience.

[1] https://nomad8.com/articles/creating-working-agreements-that-are-actually-useful
[2] https://www.mccarthyshow.com/online/
[3] https://www.liberatingstructures.com/1-1-2-4-all/
[4] https://www.agilelearninglabs.com/2008/03/active-listening-techniques/
[5] Ritual Dissent – a technique to formalize dissent: https://whatsthepont.com/2011/08/14/ritual-dissent-getting-better-proposals-and-dealing-with-saboteurs/
Image attribution: Agile Pain Relief Consulting

Read this article online: https://baa.tco.ac/1SyD

Choosing a Scrum Sprint Length – Shorter Beats Longer

By Mark Levison

How long should a Scrum Sprint be? A Scrum Sprint is a short period of time when the Scrum Team works, but there is no hard rule as to how long that should be – in this post, we cover the pros and cons of shorter and longer Sprints and how you can discover what works best for you.

Let’s start with the purpose of a Sprint: a fixed period of time for the Team to focus and develop a product with quality high enough that they could release it to the customer. A “good” Sprint Length then, has to be long enough to produce results, but short enough to limit risk.

The Scrum Guide says that:

> "The heart of Scrum is a Sprint, a time-box of one month or less during which a “Done,” useable, and potentially releasable Product Increment is created.

Sprints are limited to one calendar month. When a Sprint’s horizon is too long, the goal of what is being built may change, complexity will rise, and risk is higher – all of which contribute to increased costs and unpredictability. Sprints enable predictability by ensuring inspection and adaptation of progress toward a Sprint Goal at least every calendar month. As well, they limit risk to one calendar month of cost.

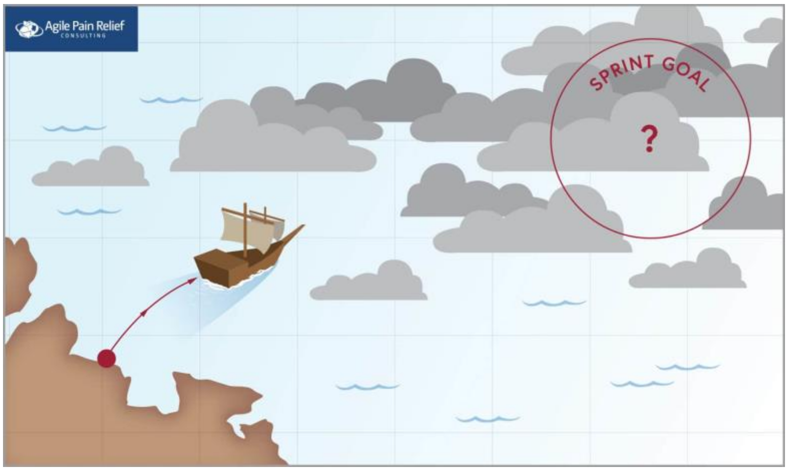
The Scrum Guide leaves it up to the Team to decide what Sprint length works best for them. When you’re first starting out in Scrum, it’s reasonable to experiment to find out what that ideal length is, but there is one caveat: shorter Sprints (1-2 weeks) help reveal problems and impediments faster. Sometimes this is uncomfortable, which results in Teams favouring longer Sprints to avoid dealing with these problems. That isn’t a Scrum-like approach; Scrum is intended to bring the problems we have to the fore so they can be addressed.

Before getting into the pros and cons of different Sprint lengths, something that you should consider is Sprint Cancellation. The Product Owner may elect to cancel a Sprint when the original Sprint Goal becomes irrelevant. Sprint Cancellation should be rare – in fact, most teams should never experience this. However, if you find good cause to cancel a Sprint more than once, you should be considering what circumstances make this happen and fixing the underlying problem.

A key consideration when deciding Sprint length is risk tolerance. Longer Sprints are riskier for predictability and cost.

Why Shorter Sprints are More Effective

To understand why you ideally want to lean toward a shorter Sprint, let’s look at some pros and cons of different Sprint lengths. I’ve also included some tips to help you decide what might work best for your Team. Remember: this isn’t a substitute for getting the Team to make their own decision based on their understanding of Scrum.



Longer Sprints (3-4 weeks)

Important: If your Sprint is longer than one calendar month, you're not doing Scrum anymore.

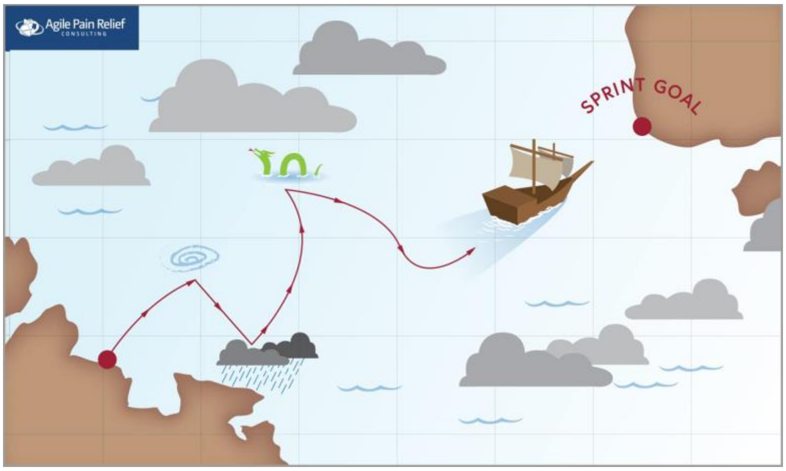
Pros

- It's easier to start practicing Scrum with longer Sprints because Teams often assume it will be easier to deliver a valuable chunk of work and get it to "Done" in one month rather than two weeks.

Cons

- It is difficult to plan well for a three to four-week Sprint during Sprint Planning. This tends to lead to more "dark work" [1] being done.
- Related to dark work – new features and needs tend to crop up more often mid-Sprint.
- The Product Owner will have a harder time not asking for change; e.g. new features or stories mid-Sprint.
- Fewer Sprint Retrospectives lead to fewer explicit opportunities to improve as a Team.
- Fewer Sprint Reviews give the Product Owner fewer opportunities to improve the product.

- Greater risk of Sprint Cancellation due to changes in the market or customer expectations.
- Greater risk of something going wrong that makes it impractical to deliver the original goal.
- Makes it easier to do "Mini-Waterfalls" within Scrum, i.e. Analysis -> Development -> Manual Test, with a certain number of days planned for each. This leads to the dark side and Scrum Theatre. Mini-Waterfalls also usually lead to the use of Hardening or Stabilization Sprints.
- Team and Organizational problems tend to be discovered and addressed more slowly.



Shorter Sprints (1-2 weeks)

Pros

- Since the Team has more, but shorter, retrospectives, they have more opportunities to make smaller changes. This also provides more opportunities to improve as a Team because….
- More frequent Sprint Reviews give the Product Owner more feedback and more frequent opportunities to update their thinking with respect to the Product Backlog. This should largely

eliminate the need for the Product Owner to ever ask for a change (e.g. new Story) during an in-progress Sprint.

- Impediments and slowdowns are highlighted more quickly since the Team is expected to get feature(s) to Done by the end of every Sprint. This forces the Team to come to terms with things that are slowing them down.
- Shorter cycles make planning easier, which increases focus and reduces the amount of "dark work."
- Forces Teams to do a better job of slicing stories or features into smaller chunks. This increases visibility and understanding of progress within a Sprint.

Cons

- It's harder to get to a finished product at the end of a one or two-week cycle. Caveat: this is true at first, however, most Teams are able to get the hang of it after three to four Sprints.
- Working in one-week Sprints can be more stressful at first.
- People say that Sprint meetings are too much overhead for a one-week Sprint. However, Sprint meetings should scale linearly with the length of a Sprint. So, a one-week Sprint will have two hours of Sprint Planning; a two-week Sprint will have four hours, and so on.

Today, most Teams new to Scrum pick two-week Sprints.[2] Some go as short as a week.

Note: On the rare occasion that I've seen Sprints shorter than one week, it seemed to reveal a much deeper dysfunction. In most cases, it's unlikely to be the best choice.

Important Tips

- Once a Sprint has started, don't change its duration (i.e. don't extend)
  - The end of Sprint is not a deadline, but an opportunity to pause, reflect, learn and improve.

    - If there are problems either in delivering the committed items or delivering the intended quality, you're better off pausing and learning from the problems than changing the Sprint duration.
- Consistency – Humans benefit from having some consistency, structure and predictability in their world. So, once the Sprint length is set, it's usually best to stick with it. The Scrum Guide even notes: "Sprints have consistent durations throughout a development effort."
- Experiment – As your team and organization evolve, what used to work might change. Run experiments – example: https://medium.com/akeneo-labs/experimenting-one-week-sprint-93dd04b42191 Note: I don't recommend growing a single team to 15 people.
- 2-day Sprints? In an extreme case, Mishkin Berteig coached one team to use 2-day Sprints using the short Sprint to reveal greater dysfunction in the organization.
- To do well with shorter Sprints, Scrum Development Teams need to work on some skills:
    - Story Splitting – so that they divide the work into small manageable chunks
    - Agile Engineering Practices – to ensure that increment delivered is of high quality
    - Learning to put Quality Assurance at the beginning and not the end of their development process. Usually this requires learning Behavior-Driven Development, also known as Example Driven Development.

Choose the Scrum Sprint Length

We've reviewed the purpose of a Sprint, and some pros and cons of Long versus Short. So now how do you decide what length of time to use? Consider everything by asking questions of the Team to prompt discussion. Some examples:

– How long are you comfortable with potentially creating the "wrong" thing? (i.e. what is your risk threshold?)

– How often do you want to get feedback from the PO/customer?

Can you think of more good prompt questions? Please share in the comments.

Whatever Sprint Length you set, honour it as the Team's choice. Then review, experiment, and adapt until you find the "sweet spot" that works best for your Development Team.

Additional References

"The Sprint Length" by Sjoerd Nijland
"What is the optimal sprint length in Scrum?" by Matthias Orgler

If that all seemed like too much…

The many different factors to consider can be a little overwhelming for people new to Scrum. That's okay, nobody has ever practiced Scrum perfectly. In our Certified ScrumMaster (CSM) workshops, you will learn through hands-on training on how to grapple with all that information and make uncomplicated decisions to help support your Scrum Team and organization.

Mark runs CSM workshops in Ottawa and across Canada throughout the year. To find out when he'll be near you, check out our ScrumMaster page.

[1] "Dark Work" is any work that wasn't intended to be done during the Sprint that is picked up and done without being surfaced to the team and placed on the board
[2] https://info.scrumalliance.org/State-of-Scrum-2017-18.html

Image attribution: Agile Pain Relief Consulting

Read this article online: https://baa.tco.ac/1SyC

About Mark Levison

Mark is a Certified Scrum Trainer and principal Agile Coach of Agile Pain Relief Consulting, learning and teaching Agile since 2001. Introducing Scrum, Lean, and Agile methods to many organizations, from government departments and banks to healthcare and software companies,

Mark has coached professionals throughout Canada.

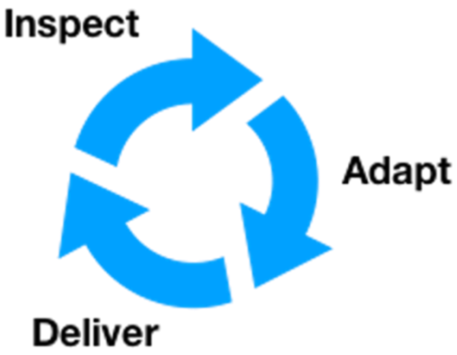
Mark's training benefits from his study and writing on the neuroscience of learning: Learning Best Approaches for Your Brain.

He has released one eBook, *Five Steps Towards Creating High-Performance Teams*, authors the blog "Notes from a Tool User", and is currently developing another eBook based on his online series, "Beyond Scrum."

How does LeSS optimize organizational ability to learn?

By Yi Lv

In my previous series of article "#backlogs and multi-learning", I wrote that "the agility is to deliver highest customer value in uncertain environment. With uncertainty, the ability to deliver is not sufficient, we need the ability to inspect and adapt, in order to deliver highest customer value". Inspect, adapt and deliver is the essential cycle.



To achieve higher agility, we need to optimize for inspectability, adaptability and deliverability. LeSS optimizes adaptability via one product backlog; LeSS optimizes deliverability in terms of end-to-end cycle time via feature team; then, how does LeSS optimize inspectability, i.e. organizational ability to learn?

I have been pondering on this question for a while. In the "fifth discipline" book, there are three disciplines closely related to learning - team learning, mental model and systems thinking. I try to look for insights from them.

Scrum/LeSS has built-in events for product learning and process learning in the sprint cycle. Therefore, let's look at product learning and process learning separately.

Product learning

The most related event to product learning is sprint review. In Scrum, during sprint review, team, PO and stakeholders including users and customers together inspect the current product increment and explore the ideas for improving the product further. In LeSS, with the whole-product focus, there is one sprint review for all teams. Review bazaar is the recommended practice for the joint sprint review.

The sprint review institutionalizes the discipline of team learning - for product learning. Furthermore, in order to deepen the learning, we also need the other two disciplines - mental model and systems thinking.

Lean startup popularizes the concept of validated learning. First make hypothesis, then design experiment to test core assumptions, then learn from the validation or invalidation. This reflects the discipline of mental model. We make our mental models explicit while making hypothesis, so as to examine and improve them both before and after running the experiments.

Impact mapping shows the logic connection between product goal and features. It explicitly asks to expand the view - who else and how else would impact the goal? This reflects the discipline of systems thinking - in the space dimension. Unfortunately, impact mapping does not highlight the time dimension, i.e. short-term vs. long-term impact.

Business seeks for growth and product needs growth engine. This is the reinforcing loop. The growth is inevitably limited by certain factors. This is the "limits to growth" system archetype. I could see the merits in applying systems thinking in the product domain, but I have rarely seen its comprehensive application - through the explicit use of systems thinking tools such as behavior-over-time, causal-loop diagram, stock&flow diagram, computer simulation - in the field. It is worth experimenting to leverage its potential.

Process learning

The most related event to process learning is sprint retrospective. In Scrum, during sprint retrospective, team and PO reflect on way of working and strive for improvement. In LeSS, besides team retrospective (i.e. team does the sprint retrospective on their own), team representatives, PO, SMs and managers will have an additional overall retrospective, which has the focus on improving systems.

The sprint retrospective - both team retrospective and overall retrospective - institutionalizes the discipline of team learning - for process learning. Same as in product learning, in order to deepen the learning, we also need the other two disciplines - mental model and systems thinking.

There is a LeSS guide called "improving the system". It explicitly recommends doing systems modeling to understand the system and have a conversation to reach shared understanding. As overall retrospective has the systemic focus, applying systems thinking there is a natural fit.

In fact, every improvement action is an experiment. We first model to make our logic behind the experiment explicit for critical thinking. While modeling, the whole team practices "balancing advocacy and inquiry", and exposes different mental models. We examine those underlying assumptions via "the ladder of inference", and improve them via reflection. Then, we return to our model and learn more after running the experiment. This is the combination of systems thinking, mental model and team learning.

Conclusion

I find that there are many similarities in product and process learning, when they are elevated into higher abstract level.

Every feature is an experiment; every improvement action is an experiment. Therefore, we make hypothesis (i.e. make the logic explicit) for critical thinking; we balance advocacy and inquiry with the whole team; we examine mental models and reflect with actual result to improve them.

In short, we aim for double-loop learning, rather than single-loop learning, for both product and process, via practicing systems thinking, mental model and team learning.

Read this article online: https://baa.tco.ac/1TUF

About Yi Lv

I am **Yi Lv**, living in Hangzhou, China.

From late 2005, while working in Nokia Networks, I started to get acquainted with agile software development, in particular, Scrum. That turned out to be my first experience of LeSS adoption.

During the year of 2008, I became the first Certified Scrum Trainer from China.

I left Nokia Networks in late 2010 and joined Odd-e till now.

As a coach at Odd-e, I worked in other industries such as Internet companies. My focus has been on large-scale product development, especially helping organizations benefit from LeSS and/or its adoption.

Turning Portfolio Management on its Head

By Mike Mallete

I summarise Agile leadership like this: "Stop giving solutions to people. Give them clear outcomes to aspire to. Give them enough space and support. They will figure it out."

Nothing Earth-shattering. It's no different to the military's age-old "Commander's Intent."

Most portfolio management schemes I've encountered are top-down management, coordination, and progress-tracking work. More mature ones have prioritisation, WIP-limits, and with strategic underpinnings in there as well. And a lot of them probably work well in their context.

However, they are still primarily solutions decided from up top, pushed down into the system. You will have portfolio managers deciding on "business and enabler epics." Some use "Kanban boards" to track them. Push them down to the program and development teams. Maybe deliver as a train, maybe as a tribe. Track the solution delivery, or increment, or deliverable, etc. And maybe some KPIs here and there, preferably results-focused rather than delivery-centric.

Management and monitoring of the work could also be top-down (or perhaps by external parties). Release/Solutions Train Engineers, Scrum-of-Scrum/Meta-Scrum Masters, etc. With the need for elaborate tracking and coordination tools (ready your Jira-fu!)

And again, they could be what is needed.

Separation of Concerns: Outcome and Solutions

Let me provide you with an alternative though. Allow me to push things a little more towards how I've defined Agile leadership.

There are two predominant layers here: the target outcomes and the solutions (or to use Jeff Patton's word, output).

Statoil's Ambition-to-Action (see: Beyond Budgeting) comes to mind. Organisational ambitions are decided on and clarified up top. Meanwhile, decisions on actions (and to an extent, forecasts) are pushed as deep into the system (for their case, up to individual level). outcome targets decided at the top, while solutions decided by people who do the work. Awesome.

Let's translate that to the example above. Imagine, Portfolio-level Management defining Strategic Outcomes (analogue to Ambition). Let's say using OKRs as a tool. Gets their various programs, development teams, release trains, tribes, (or however which way teams are grouped) to look into them. Optionally, formulate their supporting OKRs. Then propose their solutions to work towards them (equivalent to Action). Whether they propose in terms of epics, initiatives, roadmap, etc. it's what will work for their situation.

Here, the solutions were proposed by the teams (and hence, are accountable). High-level management gets to approve what they see connects well with the overall set objectives and the feasibility of the solution presented.

Leaders and Self-Managed Teams

Being accountable to the solution, the teams then have all the reason to be responsible for keeping track of their work and coordinate with the appropriate people. I don’t see the need to push or prescribe any specific way of doing this. Rather, with some coaching, we can trust the teams will use what will work. A chance to self-manage. It’s their solution!

Leadership responsibility then changes from keeping tabs and monitoring progress to making sure the people doing the work have everything they need to be successful. While at the same time, holding them accountable.

The Portfolio can have inspect-and-adapt cadences on how they are tracking with their OKRs. Then make adjustments should they see appropriate (e.g., modify approaches and solutions, change the OKR, expand the work, etc.). Ultimately, success is measured by how close the group came to their ambition.

Agile Leadership

Ultimately, this approach relies heavily on trust and servant-leadership. Circling back to our definition of Agile Leadership.

About Mike Mallete

Mike has been an Agile practitioner since joining his first XP team in the early 2000s.

He eventually went full-time Agile Coaching by 2010 and became a Scrum Alliance Certified Enterprise Coach a few years after - among the first in APAC. Since then, he has helped multiple organisations from start-ups to large multinationals improve and transform their way of working.

Before all these, he has worn multiple hats in the software development industry, from software development, architecture, sales, people management, and training.

He is based in Sydney, Australia by way of Manila, Philippines.

The Daily Scrum — Safe place for Teams

By Rohit Ratan Mani

Do you get these responses when you ask team about why do they attend daily scrum?

“I am here to find out what my tech lead wants me to work on.”
“I am here as I can’t say NO to this invite.”
“I am here to just let everyone know my progress.”
“I am here to update the scrum master.”

Not a nice place to be in, if you get such responses. What do you tell these team members, when you thought everyone knows why they do daily scrum. As a coach, when helping teams achieve their goals, does it become your goal or theirs to make a daily scrum effective? Do you think running a training with the team will motivate them to change?

Well, what I say next might not be the approach many take but might be helpful for some. Over years, my approach how I deal with these situations has changed. I will list down some ways which I have used to change this behavior:

1) Observe, observe and observe:

Before even trying any technique, observe closely on what is going on in daily scrum. What are you really observing? Stand out of the team and check team dynamics,

a) How active is everyone in listening
b) Check how people react when other team members are speaking
c) What is the team talking about, i.e, is it going in too much technical details
d) Who is being addressed when team members speak
e) And the key one these days, is anyone busy on a smartphone

How does that help? Before even taking a step to find a solution, understanding the root cause helps. Formulate your coaching stance based on your observation. Also, gather information from your team. At the end of daily scrum, ask the team to tell one thing that would have made the daily scrum more effective that day?

2) Take the facilitator out of the equation and make everyone accountable:

A team starting their journey may need a facilitator but try to get them out of the system as soon as possible.

But isn't someone responsible to run daily scrum? I would say No, daily scrum is an event for the development team, by the team and for the team. By taking the facilitator out of the equation, team becomes accountable to each other and not to the facilitator.

You will need to coach the teams in the beginning. Stand in a circle, use a whiteboard or an electronic board, ask the person to nominate the next person, get together in a room etc. These are some of the things that can be tried. Introduce these rules by doing it, rather than asking the team to do it, and the team will start mimicking your way. Move away from the team’s circle with each passing day and let them run the show 😊

3)Try pair updates:

To ease out the dynamics and to get everyone engaged, try bringing in pair updates. Get team to update in in pairs, where one updates what the other did yesterday and where are they with it. This requires everyone to be actively listening on what others are planning to do that day. Keep it open for people to fill in the missing details.

For first few days there can be awkward silence but don’t intervene, let the team fill in the silence. You should see, team members speaking up, focused on what others are saying, engaging with each other, reducing fear of conflict and not just co-operating but collaborating.

4) Bring in a buzzer and hand it over to the team:

To have everyone respect the timebox, get the team build in accountability. Give the buzzer to the team and let them buzz it if anyone is going too much in details. Some might think it will create conflicts but coach the teams to stay back and discuss it further if people are interested. Keep a check that the buzzer is not used to to buzz out someone creating conflicts.

You can also use a speaking object, where only the person holding it is allowed to talk. The object will need to be passed on the next team member to speak.

You should see team members keeping each other accountable reducing conflict within the team.

5) Try emotional check-ins:

How are you feeling today? I am not sure if we discuss or tell this everyday to our teams. Emotions drive our behavior and talking about it really gets team to open with each other. I used pictures of cats and dogs with different expressions (funny ones) and asked the team to pick one before they start the daily scrum. And as they start, they tell which emotion does that picture represent. Brings in honest conversation within the team and you should observe more unity and feeling of oneness.

6) Have fun:

Last but not the least, have fun. Don't try to make your daily scrum a routine activity. Use different ways to start: alphabetically by your surname, number of years with the organisation, last one goes first, etc. This one activity at the start of the day can motivate the team for the rest of the day.

These techniques can be used to resolve the dysfunctions of the team (as described by Patrick Lencioni) and in turn resolves the dysfunctions of the daily scrum.

Hope this is helpful and feel free to add on to this if you have more ideas around this. Please let me know your tips and tricks to make the Daily Scrum more awesome.

Read this article online: https://baa.tco.ac/1SyW

About Rohit Ratan Mani

Rohit works as an Agile and Professional coach with leaders, teams, and individuals, helping them achieve their goals and build a growth mindset.

Based in Leeds, UK, he has experience working with co-located, distributed, and fully remote teams as a Scrum Master and Agile Coach.

He is a Certified Team Coach (CTC) and Certified Agile Leader (CAL) from Scrum Alliance and has worked on Agile Transformations across different industries (BFSI, Telecom, and Manufacturing).

LinkedIn: www.linkedin.com/in/rohitratanmani

Blog: www.medium.com/@rohit.ratanmani

Adopting Agility for DataOps

By Mark Marinelli

Once you assemble a high-functioning DataOps team, it's time to establish agile mindsets, skill sets, and toolsets. Second part of a two-part series.

To create a data-driven organization, you first need the right DataOps team, a topic we covered in part 1 of this series. The next step is equally important. True transformation takes agile mindsets, skill sets, and toolsets.

The Agile Mindset

In the past 10 years, we've seen the emergence of "DevOps." It's an approach to agile software development that accelerates the build life cycle (formerly known as release engineering) using automation. DevOps focuses on continuous integration and continuous delivery of software by leveraging on-demand IT resources and by automating integration, testing, and deployment of code. This merging of software development and IT operations ("DEVelopment" and "OPerationS") reduces time to deployment, decreases time to market, minimizes defects, and shortens the time required to resolve issues.

A key aspect of embracing an agile mindset is for data engineering teams to start thinking of themselves not as technicians who move data from source A to report B but rather as software developers who employ agile development practices to rapidly build data applications. As such, teams of data engineers and data scientists must create a sister discipline -- data operations (DataOps) -- to address the needs of data professionals and transform organizations through data. DataOps is the extension of DevOps values and practices into the data analytics world.

The *DevOps* philosophy emphasizes seamless collaboration between developers, quality assurance teams, and IT ops admins. DataOps does the same for the admins and engineers who store, analyze, archive, and deliver data.

Like software engineers using DevOps methods, encourage your DataOps team to build something that works and get feedback before moving forward. This way, the DataOps team learns from its mistakes early and often and pressure-tests its assumptions about the data with real-world usage. Be sure the team embraces agile practices, getting quick wins and iterating instead of engaging in longer-term projects that don't deliver value until they're "done."

The Agile Skill Set

Any DataOps team must include data scientists because today's analytics demands are far more sophisticated than what's possible with traditional reports and dashboards. It also must include data engineers, who know more about the business than the data source owners and more about the sources than the business consumers. Think of them as the bridge between business and functional teams.

The good news: filling these roles is getting easier. The skill set of the average data professional is increasing. More college students learn data analytics and data science, and more traditional information workers are increasing their skills -- adding data analytics and data science to their repertoires.

The organization and management of the work needs to change along with the new approach. Teams acting on individual projects need ways to share tools and best practices as they engage in each new initiative. Projects four and five should benefit from what was learned in projects one, two, and three, and that won't happen without solid coordination built into the project management process.

Agile Toolset

Aligning the right mindset and skills won't get you very far unless you've selected technologies that fuel a new way of working involving agility, scalability, automation, and collaboration.

People have been managing data for a long time, but we're at a point where the quantity, velocity, and variety of data available to a modern enterprise can no longer be managed without a significant change in the fundamental infrastructure. Organizations must manage thousands of sources that are not controlled centrally and frequently change their schema without notice. It's therefore crucial to choose best-in-class tools for each part of the data supply chain versus traditional stack-vendor solutions that deliver pre-integrated, but subpar technology.

Everyone on the DataOps team needs at least one of the following tools to acquire, organize, prepare, and analyze/visualize data:

- Data access/ETL (e.g., Talend, StreamSets)
- Data catalog (e.g., Alation, Waterline)
- Enterprise data unification (e.g., Tamr)
- Data preparation (e.g., Alteryx, Trifacta)
- Data analysis and visualization (e.g., Tableau, Qlik)

All of these tools need to be interoperable by design. Seek out tools with strong API support and open data exchange because you'll need to combine them to build efficient data workflows. This comes at an initial cost, but it brings huge benefits in your ability to mix and match a data pipeline to meet your organization's unique needs. For example, you can replace individual components of the solution as better ones become available.

A Final Word

Today, it's not about why or when to start turning data into a strategic asset, but how to do it efficiently, effectively, and really, really quickly. It's time to get started by building your team and establishing agile practices, skills, and tools to guide them. Then find some challenging problems and start building the modern, DataOps-driven generation of compelling data applications to solve them. Move fast and *fix* things!

Read this article online: https://baa.tco.ac/1SyF

About Mark Marinelli

Mark is Head of Product with Tamr, which builds innovative solutions to help Tamr's customers unify and leverage their key data.

A 20-year veteran of enterprise data management and analytics software, Mark has held engineering, product management, and technology strategy roles at Lucent Technologies, Macrovision, and most recently at Lavastorm, where he was Chief Technology Officer.

The Power of Experimentation

By Blake McMillan

In the corporate world making wrong decisions can lead to negative consequences:

- Loss of customers
- Damage to relationships
- Loss of credibility
- Team member attrition
- Demotion or job loss

In fact, Jeff Bezos the CEO of Amazon is reported as saying that the trait he looks for most when promoting someone into leadership is “I want people who are right most of the time”. This type of environment can cause leaders to spend too much time analyzing and theorizing the effect a change *might* have. It can also prevent them from pivoting away from changes that are not going well and persevering too long supporting a change that doesn’t accomplish the desired results. In my experience a remedy to this is creating an environment where experimentation is understood and is accepted.

With Scrum we do this by limiting the duration of the experiment to one month or less in a Sprint. We define the result we are looking for as a valuable, useable, potentially releasable Product Increment that meets the definition of “Done”. It is only when we release the Product Increment our hypothesis of value is validated. We inspect the results of our increment in the Sprint Review which promotes a dialogue between the customers of the experiment with the possibility of influencing future experiments selected from the Product Backlog. The team also inspects itself in the Sprint Retrospective and determines how it can get better at running experiments and includes a new experiment that applies specifically to themselves in the next Sprint.

A similar model can be applied for making difficult decisions in highly ambiguous situations where best practices are not intuitively known and able to be easily applied and accepted.

- Understand the problem you are trying to solve (easier said than done many times…you can use "5 Whys to help you gain clarity).
- Create an experiment to address the problem…limiting the scope appropriately to contain the risk but also significant enough where you can inspect a result.
- Determine the benefits you are hoping to achieve.
- Predict potential drawbacks of the change.
- Determine the duration of the experiment with a preference to shorter time frames to get feedback sooner.
- Evaluate the results of the experiment and make an adaptation decision:

1. Grow the experiment (high benefits, low drawbacks) – Enjoy these when they happen.
2. Change the experiment (high benefits, high drawbacks or low benefits, low drawbacks) – Continue inspecting and adapting to see how you can improve your results.
3. Question the experiment (low benefits, high drawbacks) – Did you understand the problem? Don't give up…but challenge your assumptions.

If we can make the above as transparent as possible in the organization with the understanding that our desire is always to learn, improve, and provide more value to our customers we can be "right" even when our experiment didn't go as we had hoped.

While this article was working its way through my brain to the keyboard I found an incredibly useful and insightful article by Esther Derby called Change Artist Super Powers: Experimentation. In the article she has ten powerful questions and some great advice on this subject. I would recommend you check out her article and then get to work performing some experiments!

Read this article online: https://baa.tco.ac/1Sxd

About Blake McMillan

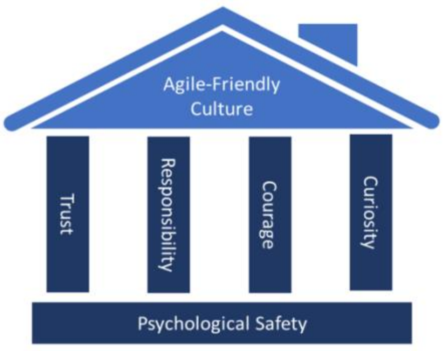
Blake is a Professional Scrum Trainer through Scrum.org and a Principal Consultant at Improving with a focus on Agile/Scrum Coaching and Training.

Areas of focus include: Professional Scrum Master, Professional Scrum Product Owner, Scaled Professional Scrum, and Agile Leadership.

He is the creator and blogger for SoulofScrum.com and is an experienced facilitator, public speaker, and creator of Agile games.

Is your culture agile-friendly?

By Zak Meziane



The majority of agile adoption programmes fail because the **organisational culture is at odds with agile values** (Annual State of Agile Survey). In other words, the adoption and scaling of agile practices are only possible in an ‘agile-friendly’ culture.
If agile values go against your organisational culture or DNA, then agile practices will be rejected in the same way a body rejects a foreign object. It is doomed to failure. It is therefore critical to understand the cultural shift required for any organisation prior to embarking on an agile adoption programme.

What is organisational culture?

Far too often organisational culture is dismissed as an abstract concept that is hard to grasp or change. It couldn’t be further from the truth. **Organisational culture** is a system of shared values, beliefs and assumptions about how people should ***behave*** and ***interact***. According to Daniel Coyle (The Culture Code), “It’s not something you are. It’s something you do”. In other words, it’s a set of behavioural patterns or habits that can be detected and changed to support the transformational objectives, agile or otherwise.

Why is culture important to agile adoption?

Before we delve into what makes an ‘agile-friendly’ culture, it is important to consider the ramifications of agile adoption for the people involved. When we set up an agile team and ask them to adopt scrum, Kanban or any other agile framework, what are we implicitly asking them to do? What does it require of them?

In most cases, especially in mature businesses, we are asking them to take a leap of faith and adopt behaviours that could potentially harm them. For instance, we are asking them to ‘fail fast and learn fast’, to challenge the status quo and go against collective wisdom, and to work collaboratively despite potential tensions between departments and functions.

In reality, this can only happen when we work in an environment that provides people with psychological safety – the belief that they will not be punished when they take a calculated risk, make a mistake or share a conflicting opinion.

Psychological safety

Psychological safety is not only the cornerstone of an ‘agile-friendly’ culture but also a core component of a healthy organisational culture, as professed by the likes of Daniel Coyle (The Culture Code) and Simon Sinek (Leaders Eat Last).

Psychological safety frees our mind from worrying about the potential harm that can come to us from inside the organisation and allows it to focus on external challenges, such as competitive forces. When we feel safe, “we become more open-minded, resilient, motivated and persistent” (Professor Laura Delizonna, Stanford University). These are qualities closely associated with agile values and principles.

Without psychological safety, people will not take calculated risks or try something new. They will not speak up when they have concerns or questions that might benefit others. In other words, they will be unlikely to exhibit the mindsets and behaviours that are essential for agile practices to work.

Finally, psychological safety works like a fertiliser, allowing other core components of an agile-friendly culture to develop and grow. These are: trust, responsibility, courage and curiosity.

Agile at scale requires trust at scale

Trust is one of the 12 principles behind the Agile Manifesto and can be thought of as "the firm belief in someone's ability and reliability to get something done". Every time we micro-manage a team or an individual or 'step in' to control a task or a process, we do the opposite of that.

To appreciate the importance of trust for agile adoption, we need to go back to the essence of agility, including improving the pace at which organisations can react to opportunities and/or threats. This relies on the decentralisation of decision-making and greater empowerment of agile teams to use their knowledge and the feedback their receive from customers to continuously adapt and refine product and service propositions. This can only be achieved by removing the layers of unnecessary bureaucracy and trusting that the people that are closest to the customer can be relied on to do the right thing and do it well.

A few years ago, I had the privilege of leading an agile transformation programme for a large organisation in the UK financial services industry. Senior leaders were quick to recognise that in order to reduce time-to-market for new services and drive the organisational growth agenda, a step change in governance was needed. The result was a fit-for-purpose framework for experimentation that reduced the instances of unnecessary escalation by more than half, allowing agile teams to design and deliver fully compliant services in a highly regulated environment inside a 2-week sprint.
However, trust and empowerment are only one side of the equation. In order to achieve a healthy balance in organisational culture, we need to strive for equally high levels of responsibility and ownership.

With great power comes great responsibility!

The definition for responsibility from dictionary.com highlights the balance between power and responsibility – it defines it as “the state or fact of being responsible, answerable, or accountable for something within one's power, control, or management.”

When we coach organisations and senior leaders about the importance of trust and empowerment, we also need to help those being trusted and empowered to define what being responsible means from their perspectives and clearly articulate the behaviours that will underpin that.

For instance, one of the teams I coached recently identified three key actions that embodied their responsibility. These were:

1. A commitment to identifying potential delivery risks on day one of each sprint and taking appropriate actions to mitigate them;
2. Informing the product owner of any potential impediments as soon as they encounter them or have any inkling they might materialise; and
3. Actively challenging each other at the daily stand up to ensure all perspectives are heard and considered.

Once these actions were articulated, the team updated their working agreement and stuck it on the wall for all to see, including the leaders who empowered them.

Everyday courage

The fourth trait of an agile-friendly culture is courage – the ability to do something that frightens us. Courage is about stepping out of our comfort zone, mastering our fears and acting when we believe it is the right thing to do. It’s not about being fearless or heroic.

In his article about ‘Cultivating Everyday Courage’, James R. Detert highlights that most acts of courage in the workplace don’t come from whistle-blowers and organisational martyrs as some might assume. In fact, his research shows that everyday courage comes from employees at all levels who master their fear and take worthy actions despite potential risks, such as getting fired, demoted, rejected or even embarrassed.

When we talk about the importance of trust in an agile context and the need to decentralise decision-making, we must acknowledge that we are asking those in positions of power to show a great deal of courage to step out of their comfort zone and try a new approach. Equally, when we expect those empowered to reciprocate with equally high levels of responsibility, we are also asking them to overcome a whole set of fears. This is why courage is key to any transformation, agile or otherwise.

Curiosity as an antidote to blame

The final piece of the jigsaw for an agile-friendly culture is curiosity, defined as “the strong desire to know or learn something”. It is key for two reasons. First, it is the fuel that drives some of the agile principles, such as customer centricity, collaboration, quality and reflection. For instance, it enables people and organisations to stay close to their customers’ needs and to continuously adapt products and services to reflect changes in their requirements.

Secondly, cultivating curiosity is a great antidote for blame and a very powerful way for creating an environment of psychological safety. In organisations with prevailing blame cultures, where taking risks is often seen as career suicide, coaching for curiosity can be very effective in supporting a shift in mindset from toxic finger pointing to more healthy behaviours.

Take the example of a senior leader discussing the findings of an experiment that failed to deliver expected benefits and the impact a change in language can have on creating an environment that fosters experimentation and agility. Instead of asking “who is responsible for this experiment?”, which will undoubtedly make people feel threatened, he or she can ask “what have we learned that can help us be more successful next time?”. Not only will it reinforce the message that it is safe to fail but also provide the right leadership in terms of continuously searching for ways to learn and improve.

In addition to these five key traits, there are other closely associated facets that make up an ‘agile-friendly’ culture, such as respect, openness and focus.

Creating an agile-friendly culture

Being aware of what makes an agile-friendly culture is only half the battle, with the real challenge being in effecting the required shift to achieve it. In my experience, any organisational change is hard, with culture change being the hardest.

We can argue that getting people to use a new tool or process is relatively simple, but how do you build trust at scale? How do you create psychological safety?

The answer is both straightforward and hard. The best way to effect culture change is to replace bad habits and behaviours with new healthier ones. And this can only be achieved with role modelling from those in charge, be it at team, departmental or organisational level.

Leadership is everything.

Read this article online: https://baa.tco.ac/1Syg

About Zak Meziane

Zak is an Agile Transformation & Leadership Specialist, with certifications including Certified Enterprise Coach (CEC) and Certified Team Coach (CTC).

Based in London, United Kingdom, he has 15 years of experience in helping organizations and their leaders thrive in a complex and changing economy by transforming the way they think and work.

Zak has supported major agile transformation programs for blue-chip organizations such as Bank of Ireland, Hargreaves Lansdown and Direct Line Group, coaching at team and leadership level.

LinkedIn: www.linkedin.com/in/zakmeziane

Blog: www.zakmeziane.com

Email: zak.meziane@katalusis.com

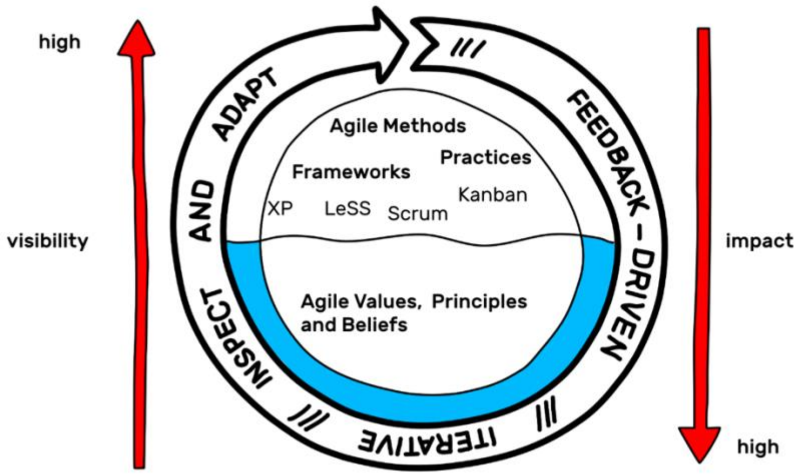
Tools for Agile Transformations: Agile Iterative Iceberg

By Roman Müller

How to explain Agile in a nutshell?

I collected vast experience in articulating, showing, explaining and simulating effects what agile means, why this topic is on nearly every strategic agenda and why organizations these days struggle to be agile.

In cooperation with my beloved colleagues from HR Pioneers and based on the various visualizations from other great colleagues (just search for“agile iceberg”), we developed this simple model which helps me a lot to have an anchored visualization.



The true deepness of the model can only be found beneath the surface of what is visible for the observer above the surface.

Above the water there's every aspect that is visible for the observer:

Methods, tools, roles, frameworks, practices,

Below the surface:

Values & Principles.
More impactful the deeper the stuff below the surface is. The more stable it the iceberg on the top.

What's new?

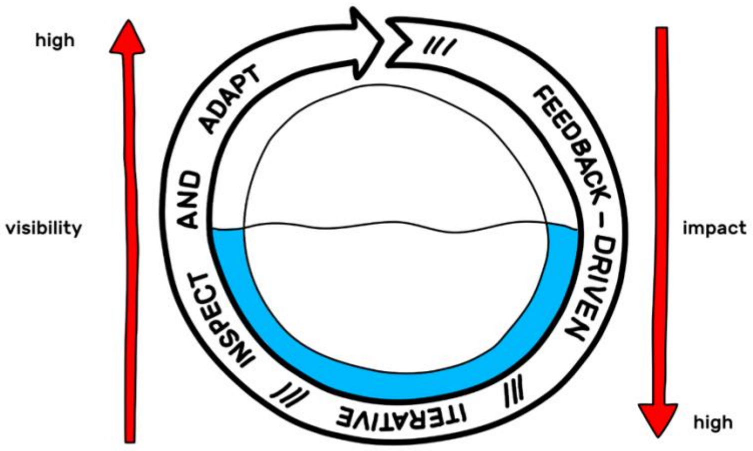
The feedback loop surrounding the iceberg: We can only **be** agile when we take both parts of the iceberg into consideration and learn via feedback and iterations. Only with that approach we find out where to improve and to change in order to get a more fulfilling, fun and purpose driven, learning environment aka **being** agile :)

How can the model be useful?

Since we have our own iceberg visualization we used it in several workshops. With our participants we do simulations like the "Ball Point Game" by Boris Gloger and use the model to debrief the exercise. By questions like "What responsible for your productivity increase?" and then classify the answers within our iceberg model. Like this, we can differentiate their experiences between beneath and above the surface and explain Agile in a nutshell. Whether you use it to debrief a simulation or you just explain it or find a completely different way to integrate it in your work, just do it!

Use it — Check out the template

Find the template here!
You can exchange the terms and text elements and individualize for your context!



Read this article online: https://baa.tco.ac/1SyX

About Roman Müller

Agile Coach

Trainer for Agile Methodologies

Co-Active Coach (CPCC)

Team Coach (CTC Scrum Alliance)

Mentor and Trainer for "coaches in training"

Chief Transformation Officer / Managing Director

As certified Team and Agile Coach, Roman is striving towards fulfillment by transforming organisations, teams and systems into purpose driven environments.

To learn more visit

www.roman-mueller.com

www.linkedin.com/in/roman-müller-49b177122

Here's How I Changed The Way I Fight

By Chris Murman

Like many of you, I got into this business by accident.

I started in IT around the time of the iPhone release.

I became obsessed with mobile technology.

This site began to write about the growing industry. Even co-wrote a now out-of-date book on building apps.

I spent years in the startup world. Small teams building big things. I thought I knew mobile when I started work at one of the largest independent app development companies. The problem is everyone else there did too. So when they asked about other specialties, I mentioned my knowledge of Scrum.

Having earned my CSM and CSPO, I came in with a lot of knowledge of the framework. I even fancied myself as someone who could coach the framework. So they gave me the chance to do so with all our teams. Which taught me the first lesson in coaching.

Book knowledge doesn’t always equate to coaching skills.

Of course, this knowledge I possessed armed me with enough to be dangerous. I was often asked to settle process disputes on teams. I would get texts with a conference room name and the words “scrum battle”.

Hell, even I started them. When I was at local meetups, in the break room, and even with company execs. I would stride in with my knowledge and wage war because I had the right answers.

Took me a while to realize I was cocky and clueless.

There were some new team standards my boss had asked me to roll out, which I thought was easy. Little did I realize getting people to do something was going to be a huge fight. At one point, a project manager pulled me aside. “Chris, nobody here is doubting your ideas are valid. But to ask me to think your ideas are the only valid ones is foolish. Can’t you give me a bunch of ideas and I’ll pick the ones that work for my team?”

I still thank Austin to this day for saying that.

Fighting battles are still something I do in my coaching work today. There are some things that deserve advocacy. I will not quit fighting for diversity on teams. Will always make room for others to speak up. And thanks to Austin, I will always keep learning new ways to do the same things.

What about the other things? The daily grind of helping teams view their work in a new way. Stepping through each day of an iteration. Attempting to inspect and adapt, always finding that one thing they can improve on. Those fights look much different to me today.

How do you fight battles with more than book knowledge?

REMEMBER WHAT FIRST BROUGHT YOU HERE.

Do you remember when you first started working with teams? My entry was through QA. Then I worked on the product side, writing requirements for others. Other times I was a scrum master if I had someone who wanted to wear the PO hat. Point is, I’ve worn many hats on teams, which we all have before getting into our coaching stances.

When you show up fresh off a training, conference, or book you’ve read, there’s so much you desire to bring to the group. You have all the context from the person imparting knowledge and can see more of the big picture.

They just want to get their work done most of the time.

People you are coaching don’t always understand the problem you are trying to solve. Often they don’t even want to know why it’s a problem. They want to work towards the deadline handed from leadership. You were there once too.

This is the worst position to be in to start a fight, and yet often we choose to. Through email, social media, or in-person methods, we want to make them see there is a problem and you’re the person to solve it. Until they are ready to see the problem, you can’t help them by creating one.

The hardest thing I’ve ever done is to let a group keep doing something that is unwise. Which leads me to my next point.

BEING RIGHT DOESN’T ALWAYS MAKE YOU RIGHT.

According to the manual, you aren’t supposed to run a team that way. You may have never said those exact words before, but I feel confident in saying you’ve said something similar. There’s no shame in it, for I know I’ve said it soooooo many times.

One of the reasons the 12 agile principles still apply is that they are broad and open to interpretation. There’s so much to uncover that’s relative to where you are in your journey. A different one will stand out to you than when you first read them.

There are many things “right” about the principles. That version looks different today, though. That means your “right” is different than your previous “right”.

“Right” looks different today to us all.

We learn, evolve, and grow in our journey of working better ways. It’s also relative to the experiences we’ve had in this industry. If you worked for different companies or made different solutions, you would look at work differently.

I had a colleague recently argue with me over UAT being a part of the definition of done. He’s never been on a team that’s done it, so he didn’t think any other team needed it. If he had different experiences, then it wouldn’t have been an argument at all.

So, what’s your “UAT is not part of DoD”?

What have you always thought needed to do that might not apply always? If there’s one thing that might not be the way you’ve always known it, are there others?

WHEN YOU FEEL SURROUNDED, REMEMBER YOU’RE NOT ALONE.

Let’s say you’ve done all that. You’ve empathized with their perspective. They see the issue and have considered there are many paths up the mountain. You’ve given up fighting with people over how they want to work. Awesome!

Fights are still coming your way, and often you will be on an island trying to defend this new way of working.

They tried your approach and it didn’t work. Or their boss told them to go back to doing it the old way. Perhaps they only half did it the way you taught them. When these things happen, the pitchforks come out. A fight is coming your way whether you like it or not.

When I’ve been in these situations, I feel surrounded. Immediately, I’m on my heels in a defensive position. So it’s reasonable for me to react in a defensive manner.

It’s important to remember that even when you feel surrounded, you’re never alone in this. There are people reading this like you and me who’ve struggled to respond in those instances. If I can diffuse the situation, understand with them what went wrong, you can take a break to find a better solution.

Rely on your community. Engage other coaches in the office, send a text to a trusted advisor, or send up a signal flare online. You’re never alone in this.

We have to change how we engage others. In and outside of our little enclave of #agile. I’m never all right or all wrong. I’m always somewhere in between. My hope is to engage with others in a manner that respects where they are and care for them to the best of my ability.

Read this article online: https://baa.tco.ac/1Sxf

About Chris Murman

Chris has spent his IT career focused on how the newest technologies are made and iterate in a better way.

That comes from learning to build smarter, harder, faster, more accurately, and with greater transparency.

Those are the tenets of how he works.

Chris has worked on teams in art/UX direction, quality assurance, product management, project management, and coaching.

Along the way he has accumulated three certifications from the Scrum Alliance, spoken at over a dozen conferences, co-authored a book on building mobile applications, and regularly blogs on his work at www.chrismurman.com

He also serves on the board for the Agile Uprising Coalition.

Is Your Agile Transformation Failing, Too?

A dive into behaviour change to understand why

By Anthony Murphy

Photo by burak kostak from Pexels

"Change is hard because people overestimate the value of what they have and underestimate the value of what they may gain by giving that up."

James Belasco & Ralph Stayer

Change is hard. We all know this. Organisational change starts with personal change. Again we all know this. But if we know these things why do a staggering amount of transformations still fail?

IBM a few years back stated that 84% of Digital Transformations fail — although there is a lot of "grey" area around what constitutes as "failure" still 84% is a staggering number! What interests me the most is why this is the case? As a fellow coach Tanner Wortham brilliantly provoked *"If this agile thing is so great, then why doesn't it always stick?"*

Great question! After all, we all know we need to be more agile, more adaptable and responsive to survive — no one is arguing about the validity of agility anymore — however there is still a gap between what we know and how we behave.

Unfortunately this is largely due to the fact that behavioural change is not so straight forward. Much like how we know we should eat healthy, exercise, spend more time with friends and family, not looking at our phones 80 times a day, but simply knowing these things doesn't mean we do them — and agile is no different.

Knowing ≠ Doing — a very real theory

So, why is it so hard to turn what we know into what we do?

Dysfunction is the gap between what you know, and what you apply.
Random theory

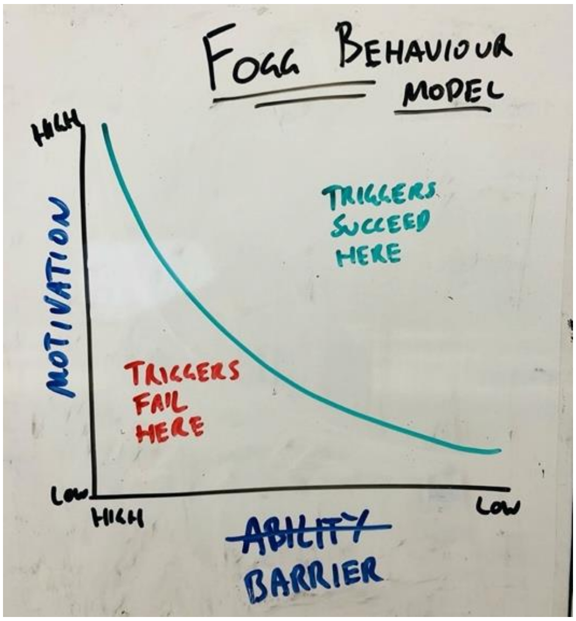
Photo of Dom Price's keynote at X4 Sydney 2019

Behavioural Change and The Fogg Behavior Model

Let's consider a company and those within it have been performing a particular habit for many years, say the habit was Taylorism, or using deadlines and the stick method to get things done. These habits are then often reaffirmed by either a) a promotion, or b) a fat bonus at the end of the year, or even c) simply keeping your job. Over the years the people within this company continue to strengthen these habits and climb the corporate ladder (a perception of success) by doing so. These habits are reinforced further over many years and have now not only become hard to modify but also believed to be impossible to completely get rid off — and therein lies the problem.

"This habit was never really forgotten, […] It's lurking there somewhere, and we've unmasked it by turning off the new one that had been overwritten." — Kyle Smith (McGovern Institute research scientist)

So how do you change a habit or behaviour?

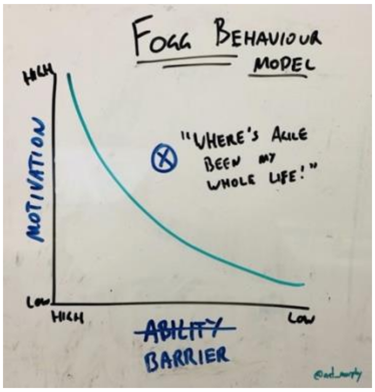


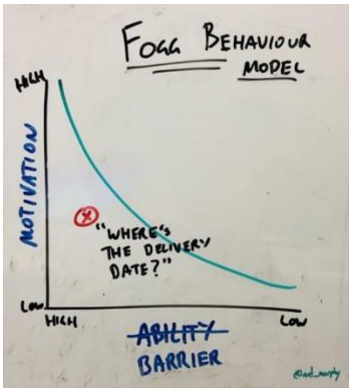
Fogg Behavior model with my adaptation of reframing “Ability” to “Barrier”

The Fogg Behavior model details that in order for a new behaviour (and ultimately a habit) to be taken up there needs to be enough perceived utility (motivation) in the new behaviour and the barrier (the person’s ability) to perform the new behaviour needs to be low enough relative to their motivation — in other words if something is very difficult to do, then one needs to have a higher level of motivation to do it, as opposed to something which is easy would require a lower level of motivation.

For example, think about getting up at 4 am to go to the gym, the barrier is quite high so either you’re motivated enough to do so or you’ll hit that snooze button and tell yourself that tomorrow will be different. Now, compare that to giving up caffeine — you’re at work and 10 am rolls around, your morning coffee habit kicks into gear but deciding to order a decaf instead or perhaps abstaining altogether is a much lower barrier than that 4 am wake up call.

So where does agile sit on the Fogg Behavior Model?





agile will sit on different points for different people for numerous reasons

In my experience agile will sit at different spots based on an individual's perceived value and motivation, as well as how hard it is for that particular person to adopt this new way of thinking.

Take the earlier example, those who have spent years, perhaps decades, building habits of working in a “non-agile” way — for them the barrier to change would likely be quite high which dictates that only an extremely high level of motivation would result in a successful change in behaviour.

“It is difficult to get a man to understand something, when his salary depends on his not understanding it.”
— Upton Sinclair

Conversely, if we took a recent university graduate — besides being fortunate to have a higher level of neuroplasticity — they are likely to have had little-to-no experience working in any other way, their barrier would arguably be much lower than those who have been institutionalised by decades of Taylorism.

💡Ask yourself, what’s your reason for undergoing this transformation? Is it high enough to overcome the change barrier?

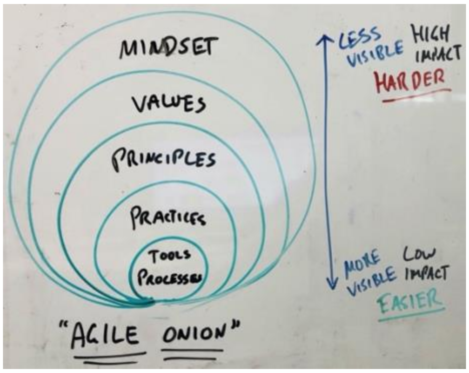
Unfortunately, in my experience the decision to “go agile” and change is often not their own — rather they are part of a big machine, an employee along for the ride as someone towards the top of the ivory tower decided that “going agile” was a good idea.

I had one exchange that will always remind me of this lesson — I was coaching a relatively new-to-agile-team and was helping to facilitate their first retrospective. I cannot remember the specific question which triggered it but I will never forget the response I got! A team member sitting at the back of the room with her arms crossed bluntly said “*Sorry, but right now you are challenging my job satisfaction. I like to write code and develop applications, not sit in meetings!”* — to say I was surprised to hear that is an understatement but I immediately thanked her for the candor and courage to speak her mind. That moment stays with me as a stark reminder that often when places undergo such transformations, it’s less of a “*wooo, can’t wait to go agile!”* and more of a *“wait you want me to do what now!?”.*

Where transformations get stuck — ‘Doing’ vs ‘being’ agile

Dark Scrum, Fake-agile, whatever you call it, we’ve all seen it — organisations adopting agile in name only, reduced to only following a set of processes and tools and failing to transcend and apply the values and mindset — ironically too, as the first value of the agile manifesto states *“individuals and interactions over processes and tools”.*

Many of you will be familiar with the idea of ‘doing’ vs ‘being agile’. Today the number of these companies ‘doing agile’ greatly outnumber those who are living the agile mindset. Many of us change agents are often found in the midst of the ‘doing agile’ tide, furiously paddling against it but with little-to-no avail — *why is that?*



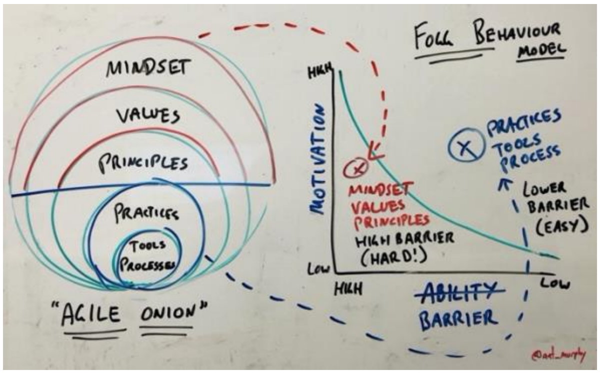
The "agile onion"

This got me thinking about the "agile onion" as it has been commonly dubbed. The "onion" visualises many layers of agile, from tools and processes at the bottom, being things which have high visibility but a low impact. And at the top is values and mindset, which although are less visible have the greatest impact.

If I were to split the onion in two, one could easily delineate between:

- "Doing agile" — i.e. following certain practices, tools and techniques, and;
- "Being agile" — i.e. living the mindset, values and principles.

How would 'doing agile' compare to 'being agile' on the Fogg Behavior Model?



A generalised mapping of the 'agile onion' onto the Fogg Behavior Model

This by no means is how it would look for everyone — I wouldn't be writing this if I hadn't managed to make the transition myself — for me and many others, the intersect of the whole "agile onion" falls within the successful change side of the curve. But for many organisations their agile journey looks much like this, with 'doing agile' on the successful adoption side and the 'being agile' on the non-successful side.

But what about those, like us, who do change?

Yes, statistically there will be some who will make the switch to 'being agile' — either they have a greater personal motivation for change or their barrier to change is lower than the rest. Whatever the reason there will be some who make it — but the burning question is, are the few who change enough to change the rest of the organisation? *Can the few change the many?*

That I'm not sure about, potentially in certain situations yes, but my experience has not shown this to be true. Rather **the law of critical mass** tends to make its mark — *if you are not familiar with the law of critical mass, it is a social dynamic which states that only when a large enough portion of people have adopted the change will it become self-perpetuating and much like a snowball rolling downhill will also attract change in others.*

An interesting concept and one which warrants further exploration but at least at face value, it suggests that organisations who have had years of reinforcing anti-agile behaviours are perhaps doomed to never transcend beyond agile practices — simply because the majority of the organisation have been pushed into the unsuccessful side of the Fogg Behavior Model — which starts to draw the conclusion that perhaps only younger organisations or those whose culture are already agile-friendly will be able to achieve an agile mindset.

"Changing behaviour works only if it can be based on the existing "culture"" — Peter Drucker

If so, what hope does that leave for older organisations where they have been institutionalised by decades of anti-agile patterns? Will they ever be able to realise the agile mindset or forever reduced to only ‘doing agile’?

Surely, there’s something we can do about it?

Michael Sahota is often known to suggest working the culture angle — as he states in his book — “An Agile Adoption and Transformation Survival Guide: Working with Organizational Culture” — we should be starting with culture and work with it rather than oppose it, which as he states is often what agile does — create friction against the organisations current culture — Sahota takes no prisoners in his suggestion to stop doing this, to stop what he calls “accidental transformations” — forcing agile into a company whose culture is simply not ready for it.

> “In my experience, many Agile change agents have done our industry a disservice by unwittingly undertaking a transformation without full buy-in or understanding of the organizational consequences.”
> — Michael Sahota

Perhaps he’s right, I’ve often felt like I was trying to jam a square peg into a round hole — rather we should look for that round peg? but what is it? Because it’s not agile.

Sahota suggests adopting only tools and practices (even if they aren’t from the agile family) which are compatible with their existing culture and only if/when their culture is ready should you start to bring the agile values and mindset into the picture. This feels a lot like working with where the org sits on the Fogg Behavior Model, no point in pushing something like ‘being agile’ if it’s on the wrong side of the change curve because it’s just going to fail.

Sahota’s focus on culture and ‘working with it’ tends to agree with the idea that only companies which are “agile friendly” will be able to successfully adopt an agile mindset.

But what if the culture is not ready, then what?

In his book, Sahota cites a number of case-studies suggesting that yes organisational change can still be successful albeit extremely hard.

One source which Sahota references is the book Leading Change by John Kotter. Kotter outlines an 8 step process to organisational change with step one being ‘Creating a Sense of Urgency’. A sense of urgency has interesting parallels to the motivation axis on the Fogg Behavior Model — *is your motivation high enough? Is your sense of urgency for the change giving you a high enough level of motivation to keep you on the success-side of the curve?*

“Ken Schwaber spoke of companies that were in really desperate situations (e.g. company survival) as good candidates for Scrum adoption since they had nothing to lose. It is clear in such circumstances that the first step — a sense of urgency — would be fully satisfied.” — Michael Sahota

Even more thought-provoking is that step four of the Kotter change process is ‘Enlisting a Volunteer Army’ — whereas Kotter puts *it “Large-scale change can only occur when massive numbers of people rally around a common opportunity.”* — this again has compelling parallels to the importance of critical mass reinforcing the idea that change alone is not enough, rather getting enough people to change is.

So perhaps change is only successful when both motivation and critical mass are satisfied.

If so, it would perhaps suggest that the only hope for large-traditional organisations who have had decades of building anti-agile habits would be to fire and rehire the majority of the organisation — an extreme thought — and one which doesn't sit well with me, as firing people in the name of change usually does nothing but build a negative connotation and further resistance — but it could very well perhaps be the terrible truth for successfully changing these traditional organisations.

On that notion, Sahota actually shares a case study in his book where GM did exactly that — they fired the majority of their middle management team and replaced them. As a result, they were apparently successful in changing their culture. Intriguing, but perhaps the exception and not the rule.

Check out the Corporate Rebels if you haven't heard of them already! (Photo by Robert Anasch on Unsplash)

However to rebuttal the notion of mass-scale firing, I recently read a case study from the Corporate Rebels blog — if you don't know about them and haven't read their stuff I suggest you do! — the case study was around the consultancy K2K Emocionando and their radical approach to organisational transformation. An approach which hasn't changed for over 20 years, it was inspiring to see that they worked both the motivation and critical mass angles all without firing a single person!

As the Corporate Rebels case study explores, the consultancy K2K starts any transformation much like Sahota suggests, with 100% buy-in from the CEO — no complete buy-in, no start, it's as simple as that for them. This buy-in is not just a verbal one, it is something that is backed with action. K2K require agreement from the CEO that the change comes first, this means that they require the CEO to agree that they can be replaced at any time by either K2K or another person in the organisation if K2K deem it necessary — now that's radical! — and if you don't believe that they'll follow through on that promise, they apparently have on more than one occasions!

They require a 100% buy-in from the owners of the organization and require them to commit to their change process. If not, they don't even bother to start. — Corporate Rebels NER case study

Second, they make the CEO agree that no one gets fired — this is core to their values and the way they approach transformation — *but what about all that stuff on critical mass if they are not going to fire anyone? Is it not still important?* K2K's radical approach to transformations is not absent of critical mass, rather on the contrary their approach hinders on obtaining critical mass, which is step three.

K2K next step in the process is to get the company to close-up-shop for 2 full days where they get everyone together and explain the change in detail. At the end of the two days they conduct an anonymous vote — K2K require over 80% of the employees to vote yes to the change otherwise it again is a no go for the change — *80% sounds like critical mass to me.*

Interestingly Kotter put's a similar benchmark against his "sense of urgency" step, stating that when over "75% of a company's management is honestly convinced that business-as-usual is totally unacceptable" is when a sense of urgency is high enough for successful change.

Complete buy-in (motivation) and now critical mass, again reinforcing the idea that the two aren't mutually exclusive when it comes to large-scale change. Empirically, K2K's numbers speak for themselves *"In total, K2K has intervened in 51 organizations, finalizing the entire transformation process in 31 of them."*

All this starts to make me draw the hypothesis that perhaps transformations should be similar to K2K's approach — only when critical mass and a high enough level of motivation is achieved should an organisation embark on a transformation, otherwise they are likely setting themselves up for failure.

"No motivation + no critical mass = no transformation" — a developing theory

K2K are not alone either.

Jared Spool's company UIE share a similar approach, although for UX and not as extreme, their approach is similar in regards to only taking on work where they have satisfied a similar criteria. As Jared shares on UIE's blog, their company policy is to only take on work when there is complete buy-in — no buy-in, no go, just like K2K.

"As a policy here at UIE, we only take on work we can guarantee results from. I know from experience that I have no chance in hell to convince any executives of anything, so I politely decline the gig. […] Our success has always come from projects where the client team, including the senior management, already understood the value of great user experiences. I haven't convinced them because they didn't need convincing."

So perhaps there is hope after all, but only for those who want it bad enough.

Photo by Javier García on Unsplash

But what about the x-axis of the Fogg Behavior Model? What about influencing the barrier to change?

So far it would suggest that everyone has looked to the y-axis, the motivation side of the equation. Perhaps this is because they subconsciously believe that the barrier is fixed — change is hard and it's not going to suddenly become any easier…I'm not sure but I think this deserves further exploration.

I often find as coaches we try to influence both angles — we try to increase motivation by helping to realise the value, whilst also focusing on breaking things down into digestible chunks not to overwhelm them and keep the barrier as low as possible — baby steps right? But even after doing so, for many, we still seem to inevitably hit a ceiling — perhaps we can influence it, but only so much.

“You can take a horse to water, but you can’t make it drink” — proverb

So where does that leave us?

Not sure about you, but I’m becoming less and less convinced that our current approach to transformation is working. Einstein said that the definition of insanity was doing the same thing over again and expecting different results — and I’m fast becoming convinced that agile transformations are becoming the definition of insanity!

Motivation is a crucial component to personal change, especially when the barrier to change is quite high. But when applying this to the context of organisational change, you are but changing one individual in a very large pool of people. This is where critical mass becomes just as pertinent, as there needs to be enough people to adopt the change to perpetuate the rest of the organisation too — without it you are left to be like those 300 brave Spartans against a hundred thousand Persians — you may put up a good fight, but defeat is inevitable.

So, perhaps Michael Sahota, UIE and K2K (among others) have it right, don’t take on any transformation where their level of motivation or critical mass are not sufficient enough. Perhaps as an agile community we need to get better at saying no to these kinds of doomed-transformations as it’s only doing the community harm.

I would love to hear your thoughts on this topic!

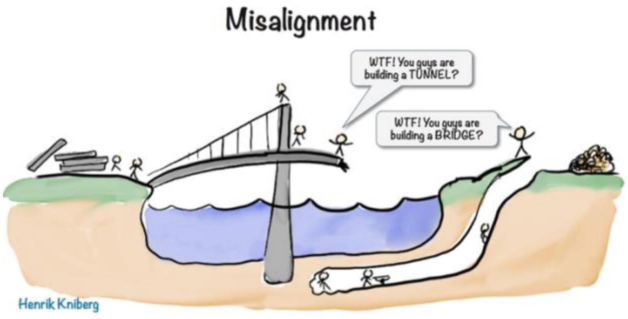
“Consider how hard it is to change yourself and you’ll understand what little chance you have in trying to change others. ” — Benjamin Franklin

References

1. IBM Blog — 84% of Companies Fail at Digital Transformation by Judy List (February 23, 2017)
2. Spikes and Stories — But Why Is It So (Fr)Agile? by Tanner (July 3, 2019)
3. Business Insider — The average iPhone is unlocked 80 times per day by Kif Lewwing (April 19, 2016)
4. Wikipedia — Scientific Management
5. MIT News — How the brain controls our habits by Anne Trafton (October 29, 2012)
6. Fogg Behavior Model by Dr BJ Fogg
7. Ron Jeffries — Dark Scrum (September 8, 2016)
8. Wikipedia — Critical Mass (sociodynamics)
9. Michael Sahota — An Agile Adoption and Transformation Survival Guide: Working with Organizational Culture (2012)
10. John Kotter — Leading Change: An Action Plan from the World's Foremost Expert on Business Leadership (January 1, 1988)
11. Corporate Rebels — A RADICAL AND PROVEN APPROACH TO SELF-MANAGEMENT (July 26, 2017)
12. HBR — Leading Change: Why Transformation Efforts Fail by John Kotter (May, 1995)
13. UIE Blog — Why I can't convince executives to invest in UX (and neither can you) by Jared Spool (June 8, 2011)

Alignment through OKR's and Hypotheses

By Anthony Murphy


Misalignment
WTF! You guys are building a TUNNEL?
WTF! You guys are building a BRIDGE?
Henrik Kniberg

When you start to scale and have multiple products and/or teams, alignment becomes paramount. The general trap is to try and control things to stop misalignment form every happening by adding many layers of bureaucracy — or as I like to call "forced-alignment". This stifles creativity, does very little to keep your people motivated and usually degrades team velocity.
But there is another way, a way to keep teams and product aligned whilst still allowing for the speed of autonomy.

Two tools that are often my go-to for helping Product Leaders and organisations I work with achieve this, are OKRs and Hypothesis Driven Development — both are relatively simple and complement each other, but most importantly they scale easily!

I'll walk through how to get this going in your organisation but first, a quick 101 on OKRs and Hypothesis Driven Development for those who aren't familiar.

"OKR" 101

OKR is an acronym for Objectives, Key Results. Its origins can be traced back to Peter Drucker in 1954 when he invented MBO (Management by Objectives). MBO was then largely used by Intel and later became widespread and now commonly known as OKRs through Google by John Doerr — an ex-Intel employee and early advisor to the Google founders, Larry Page and Sergey Brin.

As the name suggests an OKR is broken down into:
Objective — the goal we wish to achieve, and;

The **key results** we expect the objective to give us. General rule of thumb is to have no more than five key results per objective.

OKRs are designed to describe an outcome — the impact you wish to achieve but not the how. A common mistake is describing an output rather than an outcome as your OKR — something like "*Launch app x*" is **not** an OKR, it's not the objective — it's an output, rather launching app x to achieve *blah*, that's your outcome.

🔥**Pro tip #1:** If you ever find yourself in this trap, simply ask *"why?" — why are we launching app x? What benefit are we expecting to see?* The answers to these questions are likely be the outcome you're trying to achieve.

The second crucial part of an OKR is the **key result** you expect to see — in other words, how are you going to measure the outcome? Fluffy statements like "*drive customer engagement*" are neither useful nor measurable. Key results need to be measurable like "*increase retention by 5%*" — think SMART goals.

OKRs can be used at all levels — they can be high level larger long-running, lagging indicators or low level tactical, leading indicators. Often in Product-led organisations, OKRs are leverage across all levels, from high level company OKRs to low level Product development OKRs.

Hypothesis Driven Development 101

In response to the increasingly turbulent environment that product development has become today saw the rise of approaches such as Lean Startup and rapid experimentation. A natural extension of this is the idea of replacing the items in your product backlog with experiments rather than user stories — this has become known as Hypothesis Driven Development.

Unlike OKRs, I don't know it's origins (*although I would love to find out so please share if you do know!*), Hypothesis Driven Development has become a popular practice in product development today.

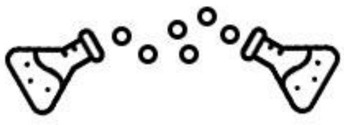
The Hypothesis framework which is commonly used is relatively simple. At a core level it consists of two statements, one which is the experiment that you wish to try (or "*bet*" as I like to call it) and second is the outcome you expect it to make and more importantly how you are going to measure its impact.

We believe that ____________________
We'll know we have succeeded when we see ____________________
A more detailed version commonly used separates the outcome from the measurement and includes a targeted customer (this is my preferred version).

We believe that ____________________
For ____________________
Will result in ____________________
We'll know we have succeeded when we see ____________________

I'm personally a big fan of Hypothesis Driven Development — the idea that everything is an experiment, diving into the unknown, trying something and hoping it will achieve an intended and measurable outcome, resonates well with me. Another reason for my love of Hypotheses is the way they help combat the classic feature factory mindset — I'm actually a big fan of ditching user stories altogether and to have the majority of my backlog written as Hypotheses — again this forces the experimentation and outcome mindset rather than just blindly building stuff.

Just like OKRs, the Hypotheses framework is agnostic to the size of work and for that reason, it too scales very well — you can have large experiments or extremely small ones, I may even have a larger higher-level Hypothesis which gets broken down into smaller experiments (hypotheses).



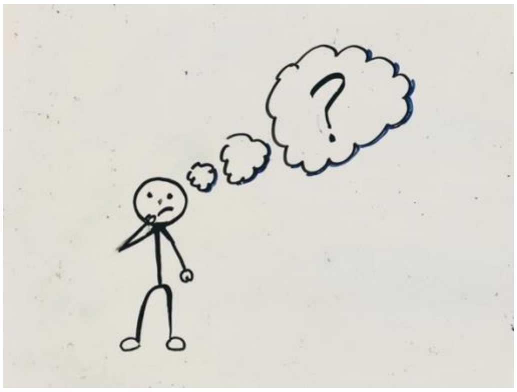
Bringing them together

As mentioned before, OKRs and Hypotheses work quite nicely together this is because they are both focused on outcomes and cold hard metrics around said outcomes. The only key differentiator between them really is that OKR’s are deliberately solution agnostic whereas Hypotheses aren’t.

OKR’s are deliberately solution agnostic to allow the freedom for the “how” part — OKR’s don’t care how you achieve the objective, they just want the objective met. This is where Hypotheses come in.

Similarly to OKRs, Hypotheses also illustrate the intended outcome and how we are going to measure it but it also has a leading statement — *the experiment* — or in layman’s terms, the “what” we are going to do in hope that it gives us a particular outcome. This bridges the gap nicely between the outcome (we want to achieve) and what we are going to try in order to achieve it (the “how”).

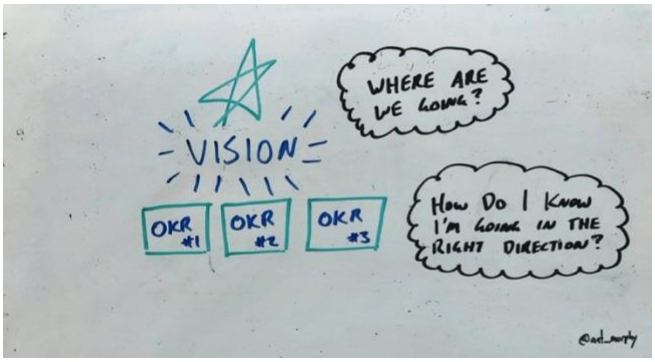
So where do you start?



A Product point of view

Like with anything I often start with the vision — the end goal, where are we heading and why.

The next question to ask yourself as a Product Manager is "*how am I going to measure progress towards the vision? how will I know I am swimming in the right direction or not?"* — these are your top level OKRs for your product. They will change but for now, they are your product-compass.

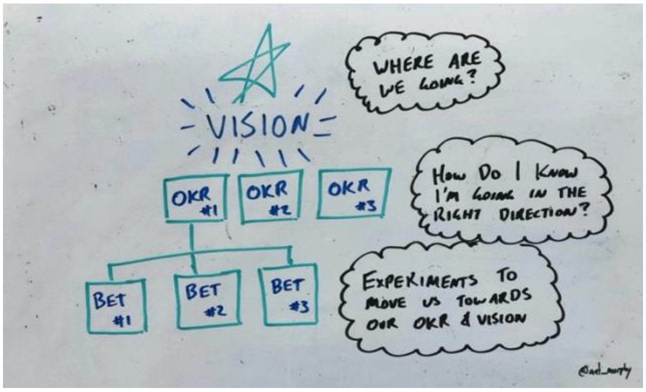


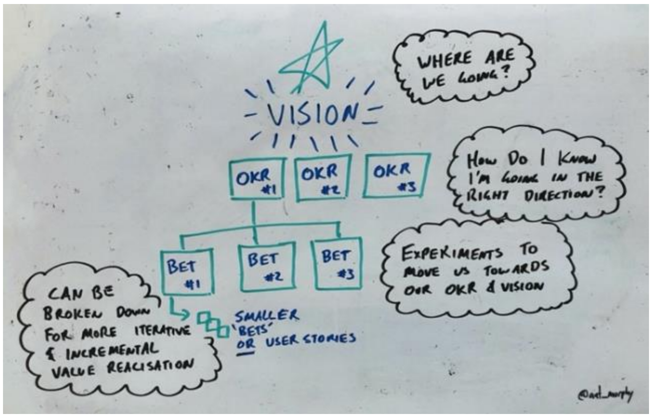
Vision > OKRs

🔥**Pro tip #2:** The fewer OKRs the better (I like no more than 3 but am still flexible on that, 4 sure, that's ok, 5 starting to push it but if it makes sense ok). Key here is to provide focus, more than 5 and you start to lose focus.

🔥**Pro tip #3:** Consider diversity in your metrics on your OKRs — i.e. you don't want to increase sign-up conversion at the detriment of retention — *beware of tunnel vision!*

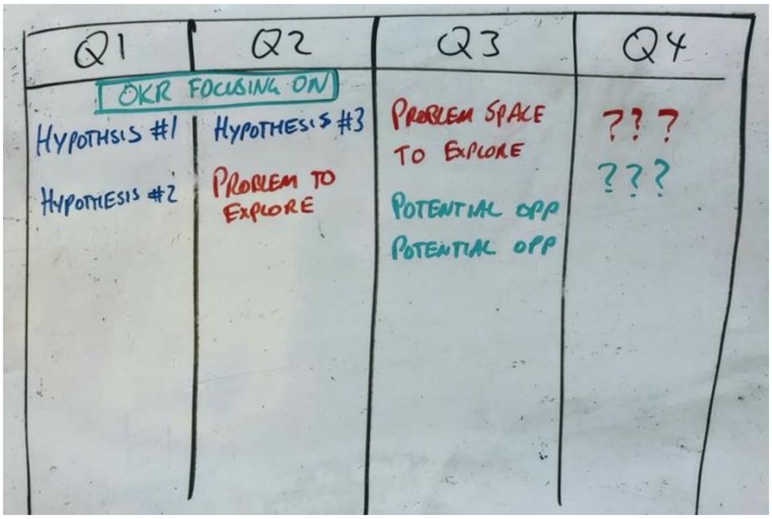
From there I will start to do experiments, or "bets" as I like to call them. This is usually informed by product discovery but are essentially the first lot of experiments you wish to perform in order to help bring the product closer to your vision.


WHERE ARE WE GOING?
VISION
OKR #1
OKR #2
OKR #3
HOW DO I KNOW I'M GOING IN THE RIGHT DIRECTION?
BET #1
BET #2
BET #3
EXPERIMENTS TO MOVE US TOWARDS OUR OKR & VISION
Vision > OKRs > Hypothesis


WHERE ARE WE GOING?
VISION
OKR #1
OKR #2
OKR #3
HOW DO I KNOW I'M GOING IN THE RIGHT DIRECTION?
BET #1
BET #2
BET #3
EXPERIMENTS TO MOVE US TOWARDS OUR OKR & VISION
CAN BE BROKEN DOWN FOR MORE ITERATIVE & INCREMENTAL VALUE REALISATION
SMALLER 'BETS' OR USER STORIES
Vision > OKRs > Hypothesis > User Stories/Smaller Hypotheses

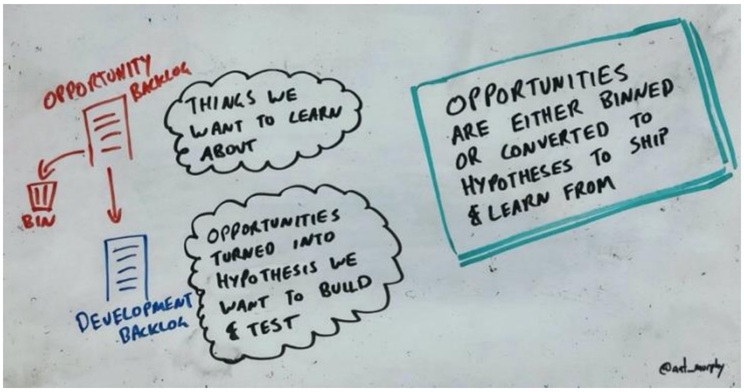
An important thing to note is that this doesn't supersede discovery, rather the contrary, discovery should be informing your bets.

Typically I will still have an opportunity backlog for discovery which will still contain potentially vague opportunities or things that are further down the roadmap. This means I may not yet have the information needed to turn them into a hypothesis just yet. This means your roadmap will likely have planned bets in the near-term and "fuzzy" opportunities still in the far-term — all still aligned to an OKR.



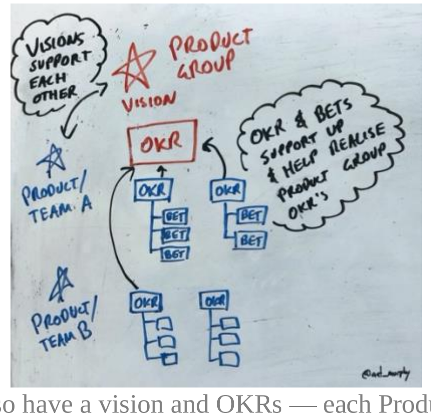
Roadmap, hypotheses in the near term but further out are opportunities/problem spaces to explore

Over time through discovery, you will gain more information and likely turn those opportunities into hypotheses but be cautious of doing this too preemptively — don't forget to validate and understand what problems you are solving before trying to solve them!


OPPORTUNITY BACKLOG
BIN
DEVELOPMENT BACKLOG
THINGS WE WANT TO LEARN ABOUT
OPPORTUNITIES TURNED INTO HYPOTHESIS WE WANT TO BUILD & TEST
OPPORTUNITIES ARE EITHER BINNED OR CONVERTED TO HYPOTHESES TO SHIP & LEARN FROM
@ant_murphy

Often we can’t jump straight to a Hypothesis (nor should we) we need to explore opportunities, validate them before we can truely understand what kind of experiment we want to do.

Scale to Product Group point of view


VISIONS SUPPORT EACH OTHER
PRODUCT GROUP
VISION
OKR
OKR & BETS SUPPORT UP & HELP REALISE PRODUCT GROUP OKR'S
PRODUCT/ TEAM A
PRODUCT/ TEAM B
OKR
BET
@ant_murphy

Product Group will also have a vision and OKRs — each Product will align and have autonomy in how they decide to contribute and help realise the groups OKR/Visio

As a Group Product Manager, you will usually have a vision of your own for the Product Group. Typically the products within a group are somewhat aligned hence why it makes sense to group them and have someone leading that group. Often when I work with Group Product Managers I like to remind them that their product craft doesn't stop just because they no longer have a specific product to manage, rather they now have a portfolio of products which is more complex but the same skills still apply —*sometimes you've just got to get creative with how.*

Having a shared vision and measurable outcomes (whether they are expressed in OKRs or not) is a powerful way for creating alignment across the multiple teams and products — and I would argue is fundamental at this level. Without it and there is the potential for the multiple products and teams to diverge.

The devil is often in the details with how you maintain alignment. The key is to provide direction and boundaries, not to cascade solutions or micro-manage (I often say to leaders I coach, *"facilitate don't dictate")*.

This means focusing on outcomes and providing alignment rather than giving orders to the product teams. It's pertinent that each Product Manager and product team still have the autonomy to decide how they are going to contribute towards their group's OKR and vision.

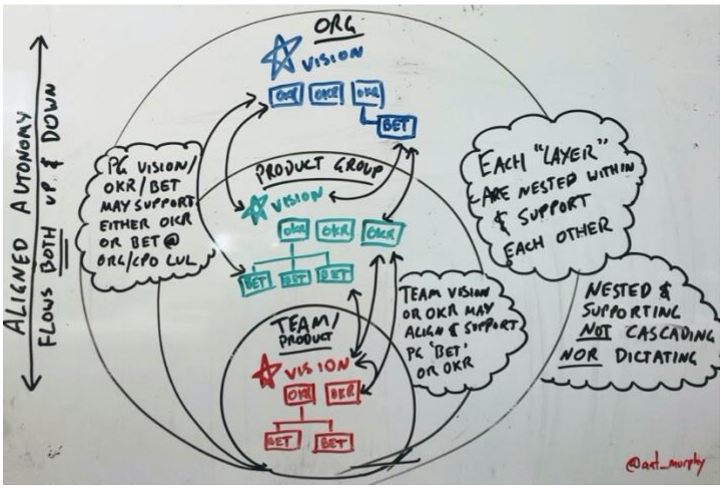
🔥**Pro tip #4:** You can still have "bets" at this level (and higher). Typically at this level you are thinking slightly bigger picture, across the product group this means your "bets" are usually linked to investment mix and things like exploring new opportunities, new markets, new product ideas, etc — often such bets result in changes in team shape — "*do I spin up a new team to explore this opportunity? or refocus an existing team? Do I sunset a product or build a new one?"* — again these are not orders cascading down and should ideally be created collaboratively with your team.

Scale again, Chief Product Officer point of view

If you're big enough it may warrant having multiple product groups. My general advice around scaling is always, "*less is more" —* in other words, only scale when absolutely necessary.

A general rule of thumb for me is Dunbar's number — Dunbar's number is a cognitive limitation of where an individual can maintain relationships and continue to feel a sense of purpose and belonging. Although the exact number varies depending on the context, a widely used number is ~150 people.

Applying this to scaling when the total number of people across your product teams approaches or exceeded ~150 people in size it may be better for numerous reasons to split them into two or more Product Groups — as a general rule of thumb, this is the limit I work within and by no means a hard and fast rule. Depending on your context, your products, and company size it may make sense to have multiple Product Groups of ~30 people, again nothing wrong with this but its often important, especially as you scale the be mindful of Dunbar's number.


ORG
VISION
OKR OKR OKR
BET
PRODUCT GROUP
VISION
OKR OKR OKR
BET BET BET
TEAM/PRODUCT
VISION
OKR OKR
BET BET
ALIGNED AUTONOMY
FLOWS BOTH UP & DOWN
PG VISION/OKR/BET MAY SUPPORT EITHER OKR OR BET @ ORG/CPO LVL
EACH "LAYER" ARE NESTED WITHIN & SUPPORT EACH OTHER
TEAM VISION OR OKR MAY ALIGN & SUPPORT PG 'BET' OR OKR
NESTED & SUPPORTING NOT CASCADING NOR DICTATING
@ant_murphy

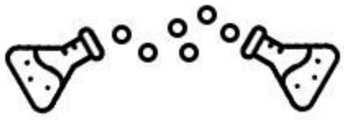
Company vision and direction <> Product Group <> Individual Products — t*hink in terms of supporting networks and things being nested. Each "layer" supports both upwards and down.*

Again no different than at the Product Group level — OKRs are outcomes and “the why” and similarly, each Product Group has the autonomy to decide how they are going to contribute to the organisations visions and objectives and so forth.

🔥**Pro tip #5:** OKRs and Hypotheses should correlate, not cascade down. They are common goal posts to keep teams loosely aligned — not for telling the teams what to do. Rather each team, group, department, tribe (whatever you have) are free to decide how they are going to help attribute towards the higher goal, or not if they have a great reason why.

🔥**Pro tip #6:** Equally things don’t cascading up either — a common anti-pattern I see is the need for things to go up the chain for someone’s “blessing” — like the Head of Product needs to “prioritise it” or give the thumbs up. There should be no need to getting “someone’s blessing” this only creates a dependency and bottleneck, rather teams should be empowered and trusted to make their own decisions.

🔥**Pro tip #7:** Split your work into past, present and future buckets.



✨Bonus Tips for using OKRs and Hypotheses for the first time✨

The number one area I see people struggle with is the measurement part. Either a) they haven’t been measuring impact or outcomes before so they have no idea of current value, where to start or whether how to even obtain such metric, or b) they get way too caught up with coming up with the worlds perfect metric.

So here’s some quick advice to help avoid the common traps for those OKR and Hypothesis virgins out there:

1. **Just start somewhere** — if you don't know where to start just pick a metric and start figuring out a way to measure it, one by one you'll start to build out your metric base. And the best part is if that metric "wasn't the right metric", that's ok, you've now learned something and you can try something else — chances are that metric, which by the way you are now measuring, will be useful in the future.

2. **Don't have too many OKRs** — less is more — ideally, I look to have no more than 3 OKRs at the vision/top level. If I was to use them in a roadmap, I would then look to have only one per month/quarter or whatever cycle I am working too.

3. **Don't pick too many metrics either** — again, less is more! I generally work off the same rule-of-thumb as for OKRs and go for no more than 3 metrics/key results for both OKRs and Hypotheses — ideally I would want to have only one but I know that one doesn't always make sense.

4. **Focus on making your metrics SMART goals** — which stand for specific, measurable (obviously), attainable, realistic and time-bound.

5. **Don't get too caught up on getting it perfect** or right to begin with — in a similar vein to the first tip, once you've done your experiment if your measurement part is off you will know really quickly when you start to get feedback. No need to worry, simply pivot your metric and measure something else. The more practice and experience you get, the better your metrics will get over time.

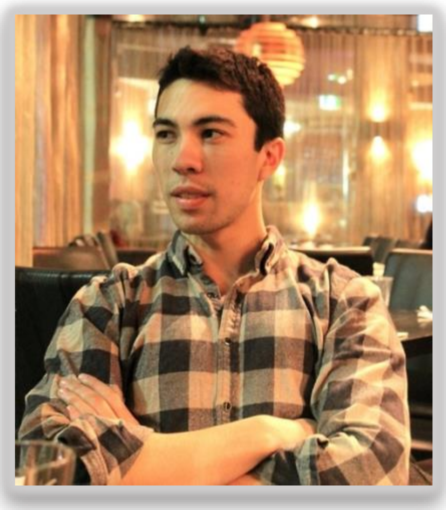
6. **Think about leading vs lagging indicators** — it can be useful to think about OKRs as more lagging indicators or ones which will see the needle move over a much longer time period

whereas, since your Hypothesis are your experiments to meet the OKR, their metrics are often leading indicators, as they require a much shorter feedback loop.

7. **OKRs and Hypotheses should flow both top-down and bottom-up**. It is important that neither of these cascades down — they're not there to tell teams what to do! Rather they're a tool for creating the conditions for aligned autonomy. What do I mean by that? Teams should have the autonomy to decide *how* they are going to contribute to a higher OKR/Hypothesis. Think of them as a guiding light in the distance, designed to keep everyone rowing in the same direction whilst allowing the autonomy and freedom to forge their own paths there.

8. **Be collaborative and create both together** — don't be a dictator and prescribe what OKRs or Hypothesis people should do rather involved them in the journey and co-create them together — many minds are better than one.

About Anthony Murphy



Anthony is a Product Management and Leadership coach who has trained and mentored Product Managers and leaders all over the world.

He currently leads a team of 20+ Product and Agile consultants.

Anthony's experience spans across several different industries, from start-ups to large corporates.

He has done everything from early-stage product discovery to pivoting products and even sunsetting them. Often called in at the CPO level he helps organisations become more product-led.

Anthony loves a good coffee, having a laugh and making products that have an impact on the world.

LinkedIn: www.linkedin.com/in/ant-murphy

Blog: www.medium.com/@antmurphy

Twitter: @ant_murphy

4 Paradoxes in Agile (and Life)

By Stephanie Ockerman

A paradox is something that is seemingly absurd but really true. When I experience complexity and uncertainty, I find comfort and power in paradox. It opens up creativity, possibility, and collaboration. Let's take a look at 4 paradoxes we need to navigate in the agile world and beyond.

Agile Paradox #1: You have no control.   And you always have control.

Agility is about accepting that the work is complex and unpredictable and committing to working in a way that honors the truth of this. In Scrum, we take an empirical approach to minimize risk and enable informed decisions along the way.

You cannot control everything that happens, but you can choose how you respond. You likely made choices that led you to the situation to begin with. And you will continue to make choices that lead you to new situations. Leadership means taking responsibility for your world.

Create transparency. Inspect and adapt.

This is how you always have some control even when you cannot control the variables that impact your work - *by frequently making choices with new information.*

Agile Paradox #2: It's not about you. And everything is about you.

Servant leadership means your growth and success is measured by the growth and success of others, the people who are choosing to follow you.

And this will be 100% impacted by how you are showing up.

I recently came across this statement in the book Contagious Culture by Anese Cavanaugh:

> "Waiting for "they" to show up differently, to act differently, to do something differently, to be different is a waste of time and energy at best, and completely uninspiring. You have no control there."

If you want to create impact and help others grow and succeed, **you need to show up in service of others**. And to do that, you have to take care of your own needs to show up with intention and be fully present. You need to hold a clear vision aligned with your values. Believe the best about people, and include them in solving problems and tackling new challenges.

Are you bringing your full authentic self to your team and other relationships?

Are you modeling integrity and responsibility?

Are you leading by example with openness, curiosity, and compassion?

Agile Paradox #3: Show people compassion and understanding. And hold people accountable.

I can care deeply about you and believe that you are doing the best you can with what you have at the time. Yet that does not mean I should give you a pass when you have done something hurtful or damaging.

Perhaps the compassionate thing to do is to provide honest feedback and support you in closing the gap between where you are today and where you need to be. It is compassionate to believe someone can learn and grow, discovering new perspectives. And perhaps the compassionate thing to do is help you find a better fit for your skills, purpose, and passion.

The key is to set boundaries and hold people accountable. Boundaries can be as simple as saying what is okay and what is not okay, and they are essential for growing trust.

When someone does something that is not okay, we hold them accountable AND show them compassion and understanding.

This also applies to yourself. Hold yourself accountable and have compassion for yourself when you make a mistake. Own it, apologize, and do the work to make it right. Do the work on yourself to learn and grow from the experience.

Agile Paradox #4: Value the individual. And protect the team.

This often goes along with #3. Sometimes an individual needs to hear a difficult message, and sometimes they need to hear it in a more direct and challenging way if the "gentle and nice" way isn't working. Do this from a place of integrity.

Don't sacrifice the health and safety of the team because you're worried about upsetting someone (this is often about protecting their ego and/or your own discomfort). Sometimes you have to take decisive action in a timely manner to show that you value the other individuals in the team.

And you can do this while valuing the individual - assuming the most generous thing, giving them a chance to learn and redeem themselves by earning back the team's trust.

Conclusion

Carl Jung described the paradox as one of our most valued spiritual possessions.

> "Only the paradox comes anywhere near to comprehending the fullness of life.

The ability to hold two contradictory ideas in our minds leads to possibilities we may not have seen before. It is what enables us to productively and creatively collaborate with others when there are few "right" answers. We can use our experience and wisdom, yet stay open to trying something new. This is what makes life interesting.

What other paradoxes have you noticed in agile and other areas of life? How do you hold them?

Read this article online: https://baa.tco.ac/1Syc

The #1 Thing That Kills Your Influence

By Stephanie Ockerman

When I teach Professional Scrum courses and leadership workshops for women, the topic of how to influence people always comes up. Through experiential learning, people begin to see that the old way of doing things doesn't work and that we need a new approach based on embracing change and incremental learning.

But how do you encourage and support others in taking a risk, to try something new?

How do you inspire trust and a desire to work together?

Well, let me tell you the one thing I see most frequently that does the exact opposite.

Being competitive and aggressive is killing your influence.

This shows up in non-profits, start-ups, large multinational companies, politics, community groups, families… everywhere.

If you're thinking, *that's not me* or *that's not my co-worker/ team/ family*, I encourage you to take a closer look. It is not always intentional, but consider how your approach may come across to others.

Here are some examples:

- Continuing to push a perspective to get others to agree (rather than being okay with hearing other perspectives that are different).

- Repeating the same thing, trying to wear people down so that they will "give up" and agree with you (rather than creating space for new ideas and more perspectives).
- Challenging other people's perspectives with assumptions, extreme hypotheticals, or actually putting words in someone's mouth (rather than keeping it factual and neutral).
- Feeling a need to share your opinion (even when that's not what others need right now).

These behaviors lead people to believe that you are out to win or be right.

They may not feel you are seeking the best outcomes for everyone. They may feel like it's not worth the risk of being attacked or criticized to enter into the conversation, so they simply don't. Even if personal risk isn't of concern, they may not feel it is worth the effort to spend their energy engaging in competition.
Ultimately, the downside is:

- People don't trust your motivations.
- People don't feel safe.
- You lose diversity and creativity because people don't engage.

In some cultures or environments, people are socialized to be competitive and aggressive. Beyond the messages I got in school, I have always worked in a male-dominated industry. I had to speak a little louder, work a little harder, be more prepared, and be more persistent in order to be heard, to be appreciated, and to get the opportunities I desired.

While I was successful in navigating this world, I always balanced that with a facilitative approach. I brought people together and created a space to focus on solving problems.

Enabling openness and collaboration grows your influence.

I eventually learned that my true nature is to be open and collaborative, to understand different perspectives, to co-create the best possible solutions. These are my superpowers. And most importantly, I learned that this is servant leadership, and it is the more effective leadership style for the complex and challenging problems we face in our world.

Now that I know this about myself, I can show up as the leader I want to be.

Yes, I still can easily get sucked into other people's competitive and aggressive energy. And it's a constant battle to stay self-aware enough to check myself when I do get drawn in. Passionate debate can be a good thing, but you have to consider the environment and your goals. Here are the questions I use to check in with myself when I am feeling frustrated or hurt:

- Do people feel like I am trying to convince them to change, or do they feel like I am inviting them into a conversation?
- Am I okay with my idea or perspective not being used as the basis for moving forward as long as it was heard?
- Am I trying to understand the other perspectives? Can I truly appreciate them even if I don't agree?
- Am I okay with that person having their own perspective even if it is different from my own? (Note this means I am not judging them for it.)
- Do I feel people are trying to understand my perspective? (Usually, if the answer is no, this is what triggers my own aggressiveness.)

Now that you are aware, what do you do with that?

Neutralize it with patience, appreciation, openness, and curiosity.

I don't judge the person - I meet them with empathy. I also don't sit back and let it happen. I kindly and directly bring transparency to the situation. And there are times when I need to walk away for a bit to protect my own energy and stay in alignment - so that I can keep showing up as the leader I want to be.

Consider areas of your life where you might be engaging in a competitive way (even if unintentional) or where you are being drawn into competition.

- *What approach will create the outcomes you truly desire?*
- *How do you want people to see you as a leader?*
- *Do you need to listen more or share more? What conditions will help you do that?*
- *How can you change your language to create an open and collaborative space?*
- *Is it worth it? Is it best to walk away, at least for now?*

P.S. If you would like some support in this process, check out the Scrum Master: Grow Program.

Read this article online: https://baa.tco.ac/1Syd

About Stephanie Ockerman

Stephanie is a Scrum.org certified Professional Scrum Trainer (PST), Scrum Master, and Coach.

She has been a Steward of the Professional Scrum Master (PSM) Curriculum since 2016, helping to improve and evolve the course while maintaining the integrity of Ken Schwaber's vision.

She started her agile training and coaching business Agile Socks LLC to empower and enable teams and organizations to confidently navigate complexity and uncertainty in order to deliver greater value and thrive in a constantly changing world.

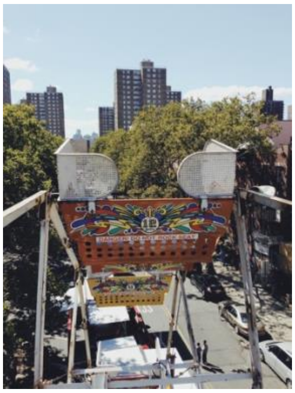
Stephanie believes that with alignment, focus, and smart execution, we can do amazing things together. She combines training and coaching to help people level-up their skills and amplify their impact. You can read Stephanie’s musings on Scrum and agility, as well as enroll in her online courses and training classes at AgileSocks.com.

To learn more visit

www.linkedin.com/in/stephanieockerman

Making meaning of cultural values and preferences through retrospectives

By Brandi Olson



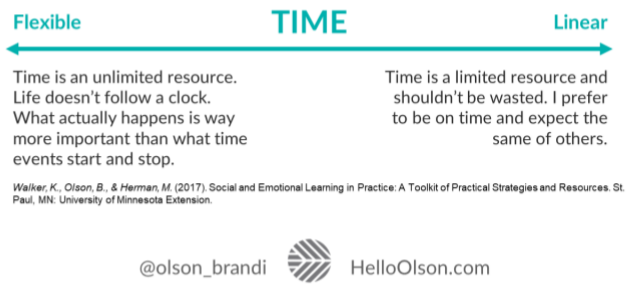
Failing to recognize the diverse values and cultural preferences of colleagues usually lead to harmful assumptions that a difference in behavior is the result of a personal flaw, instead of recognizing it as a difference in social-emotional skills or cultural values.

We each bring social and emotional skills to the ways we show up and interact with others. Our ways of feeling, ways of relating, ways of getting work done are deeply influenced by our cultural values and preferences. And yet, our personal cultural values are often unexamined, leading us to make value judgments about the "right" way or the "wrong" way to communicate, resolve conflict, make decisions, etc.

Our ways of feeling, ways of relating, ways of getting work done are deeply influenced by our cultural values and preferences.

@olson_brandi HelloOlson.com

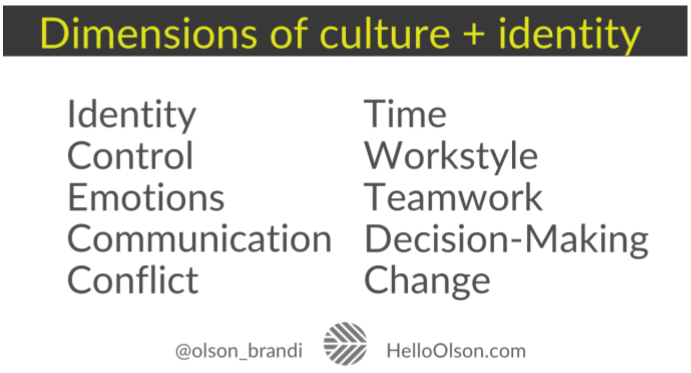
For example, what's your value around time? Is time a scarce resource? If so, starting a meeting when the clock says it's time is the "right" way to behave. What if you see time as an unlimited resource? This value might lead you to believe that the "right" time to start a meeting is when everyone who needs to be there has arrived. Even the Agile Manifesto and Principles are steeped in a set of cultural values + preferences that privilege some behaviors over others.


Flexible
TIME
Linear
Time is an unlimited resource. Life doesn't follow a clock. What actually happens is way more important than what time events start and stop.
Time is a limited resource and shouldn't be wasted. I prefer to be on time and expect the same of others.
Walker, K., Olson, B., & Herman, M. (2017). Social and Emotional Learning in Practice: A Toolkit of Practical Strategies and Resources. St. Paul, MN: University of Minnesota Extension.
@olson_brandi
HelloOlson.com

Failing to recognize the diverse values and cultural preferences of colleagues usually lead to harmful assumptions that a difference in behavior is the result of a personal flaw, instead of recognizing it as a difference in social-emotional skills or cultural values. This reality plays out frequently as we try to make sense of team behavior.

Understanding the cultural values and preferences represented on a team or in an organization can bring powerful insights into ways to work better together.

In collaboration with the University of Minnesota, I co-created the Ways of Being Model for teaching the connection between cultural values and social-emotional skills. One of the most powerful tools we developed is the Cultural Values Map It's a simple way to help a group of people explore how their own values and preferences shape the culture of their teams and organizations.



Dimensions of culture + identity

Identity	Time
Control	Workstyle
Emotions	Teamwork
Communication	Decision-Making
Conflict	Change

@olson_brandi HelloOlson.com

Cultural values mapping in a team

At a recent scrum master workshop, I introduced Cultural Values Mapping. One of the participants sent me a note a few weeks later. She had been infuriated with her scrum team-- they were so undisciplined!

Standups were a mess, she said. They never started on time and she felt like she wasn't getting through to the team on the importance of standups.

She invited the team to create a cultural values map at their retrospective. They learned that the entire delivery team held a deeply shared value around time--it's a flexible, abundant resource.

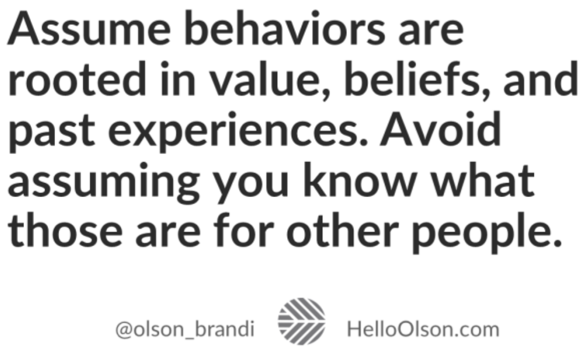
Can you guess where the scrum master landed? She found herself on the opposite end of the spectrum!

They spent the rest of the retrospective talking about their mutual frustration around standups. The team was frustrated when the scrum master tried to get them to start the standup before everyone arrived.

They collectively believed it was more important to have everyone in the standup conversation than to start on time.

The team didn't lack discipline. They were prioritizing a different value.

The ways we behave are rooted in our values. My values around time or communication or celebration are not more right or more wrong than your values. But if I don't know what your value is, it's all too easy for me to assume that you just don't know the "right" way to act.



Assume behaviors are rooted in value, beliefs, and past experiences. Avoid assuming you know what those are for other people.

@olson_brandi HelloOlson.com

In the case of our scrum master, she assumed that her belief around when to start standups was the **RIGHT** way to do it. She judged her team to be generally disorganized and undisciplined because standups frequently got a late start. While she stated often and loudly her expectation that standups should start "on-time", she never explored why they were starting late and if it was a problem for anyone else on the team.

In any workplace, there are behaviors and norms that are privileged because they align with the implicit cultural values of the organization. Shared expectations around behavior aren't inherently bad--a team has to work together towards shared goals. However, if we want to work better, together, then we need to make these expectations -- and the values behind them, explicit.

Two ways to use cultural values mapping in your next retrospective:

1. Team Retrospective: Map the group
Invite team members to complete the cultural values mapping exercise on their own and bring their results to the retrospective (get the cultural values map and template right here). Map each person's responses into a single group view.

- What trend does the team notice?
- What dimensions have strong shared values?
- What dimensions have value diversity?
- How do those differences or similarities show up in your team interactions?

2. Team of Teams Retrospective: Human Graphing
At your next PI planning event or product design session, use one cultural dimension from the Cultural Values Map as a conversation starter for a group retrospective. (Here is a simple deck of all of the cultural value dimensions and descriptions, ready for you to use with your group!)
Share the dimension so everyone can see it. Ask people to move to specific areas of the room to represent their preferences on the spectrum.

- What are our group norms around this value?
- How might someone who has a different value be viewed?
- When do our different values around this dimension help us as a group?
- What values (and resulting behaviors) are privileged and rewarded in our organization?

Turn it into action

The Scrum Master from the workshop reached out to ask, "What now?" What do we do with our understanding of each other's cultural values? Here's what I told her…

Once you've mapped individual values and preferences, map out the values underlying your team working agreements.

> What are the implicit values represented in your working agreements?
> How do your team's values compare?
> Are there opportunities to make the values behind the working agreements more explicit?
> Does the team want to change any working agreements?

As a result of their retrospective on cultural values mapping, the Scrum Master's team made their expectations and values around standup explicit--the daily standup would start when everyone was ready--even if that meant it started a few minutes past the scheduled time. The Scrum Master agreed to prioritize the team's value around time, and they all agreed to check-in on this expectation in a few sprints to see if they needed to adjust.

If you'd like to get additional resources to use Cultural Values Mapping in your retrospective, head over to Brandi's Cultural Values Mapping Resource Page - created just for Scatterspoke readers. There, you'll find individual templates and a deck you can use to introduce the concepts to your teams.

Read this article online: https://baa.tco.ac/1Sxe

About Brandi Olson

Brandi believes that you shouldn't have to choose between doing good, important work, and your own humanity.

She works at the intersection between learning, organizational agility, and human-centered design. With over 15 years of experience consulting with leaders across diverse sectors who share a commitment to people, learning fast, and doing good,

Brandi knows that the best way to solve big problems is through happy, high-performing teams.

To learn more visit

www.HelloOlson.com

www.linkedin.com/in/brandiolson

Liberating Structures - an Antidote to Zombie Scrum

By Barry Overeem, Christiaan Verwijs and Johannes Schartau

- Scrum is huge! It’s been massively adopted by organizations and has become the software development framework of choice.
- Many organizations and teams think they’re doing Scrum, when in reality they’re not even close. They’ve got all the Scrum roles, events and artifacts in place, but fail to meet their potential.
- The number one cause of Zombie Scrum is that most organizations and teams don’t have a clue as to why they are doing Scrum.
- There are multiple antidotes available to prevent & fix Zombie Scrum. Liberating Structures is an example of such an antidote to Zombie Scrum.
- Stop Zombie Scrum in your organization by making the current situation painfully transparent. Join the Zombie Scrum resistance, invent new and use existing antidotes to Zombie Scrum, and discover the true potential of the Scrum framework!

Scrum is a simple, yet sufficient framework for complex product delivery. It helps organizations thrive on complexity. Scrum provides the minimal boundaries for teams to self-organize and solve complex problems with an empirical approach.

However, we’ve noticed that although many organizations use Scrum, the majority struggle to grasp both the purpose of Scrum as well as its benefits. Instead of increasing their organizational agility and delivering value to customers sooner, they achieve the opposite. We’ve come to call this Zombie Scrum; something that looks like Scrum from a distance, but you quickly notice that things are amiss when you move closer. There is no beating heart of valuable and working software, customers are not involved, and there is no drive to improve nor room for self-organization.

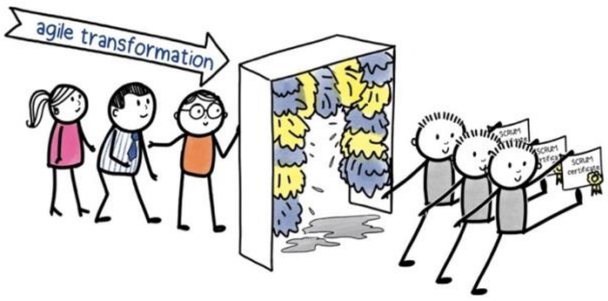
One antidote we've found helpful is to rethink how teams interact, both within the team as well as with stakeholders and the broader organization. For this, we found help in Liberating Structures. Rooted in complexity science and based on simple recipes that allow everyone to be involved and unleashed, we've found them helpful to promote a fast-paced interaction, self-organization, high involvement, radical transparency and creative destruction that we feel are vital to Scrum.

This article describes what Zombie Scrum is about and gives you tangible examples of how to recognize, treat and prevent Zombie Scrum by using Liberating Structures.

The State of Scrum

Scrum is huge! It's widely adopted by organizations. Worldwide, there are hundreds of professional Scrum trainers. Probably more than a million people are certified through one of the official Scrum institutes. Countless books and articles on Scrum have been written and every country has its own user group. Scrum has become *the* framework of choice for many organizations, each following their own promise of increased agility and faster delivery.

This is inspiring and is a cause for celebration. However, we've noticed that while many organizations and teams think they're doing Scrum, they're only touching the surface of what is possible. Most are stuck in painful mediocrity and struggle to find their way out, and we feel that our community is not taking enough responsibility in helping them get out.



(image by Thea Schukken - Beeld in Werking)

For one, many organizations and teams think they’re doing Scrum by having all the roles, events and artifacts in place, certified and all. And there’s a slew of coaches around to help them. But all these changes are superficial if there’s no working and valuable software at the end of *each* Sprint, ready to be delivered to stakeholders. All these roles are useless when there’s no drive to improve anything and no involvement from customers and users during development.

Here is one example: a couple of years ago we worked for a large financial institute. They had the seemingly perfect transformation plan to rollout 50+ Scrum teams in one year. Every week, a couple of new Scrum teams were launched. The organizations buzzed with excitement. "Scrum of Scrums" started. Big room planning sessions were organized. Release Trains were planned. At the end of the year, the transformation plan was completed and it was time for a big party. The Agile transformation was a success!

However, until that moment, they only used metrics focused on utilization and efficiency. Local optimization was encouraged without taking the bigger picture into account. People were being manipulated, and everyone felt as though they were being monitored and controlled. Although the utilization & efficiency metrics showed good results, everyone felt something was fishy ...

Two years after the Agile transformation had kicked-off, they started exploring different kinds of metrics. Instead of focusing on efficiency and utilization, they selected metrics that were useful and relevant for measuring the success of the empirical process implemented through the Scrum Framework. They started using metrics focused on Agility and on value delivered, such as cycle time, customer satisfaction, team happiness, innovation rate, return on investment, and total defects.

When the first results of these metrics became visible, the whole organization was in a state of shock. Their cycle time had increased, customer satisfaction had worsened, teams were unhappy, the return of investment was very low, the amount of defects seemed to go through the roof, and as a result, there wasn't any time for innovation anymore.

What was going on? They had implemented all parts of the Scrum framework. All the artifacts, roles and events were in place. They even added some extra practices like Scrum of Scrum and a big room planning… Why wasn't Scrum delivering on its promise?

The answer is that although what they did certainly looked like Scrum from a distance, they actually missed the essence. This is a prime example of Zombie Scrum, and we need to work together as a community to help prevent it. Let's stop going mindlessly through the motions of Scrum, and let's start getting real value, while having serious fun!

(image by Thea Schukken - Beeld in Werking)

Zombie Scrum

For years, we have done extensive research and studied hundreds of teams and organizations. We’ve noticed the following patterns:

- Scrum cherry picking: not all the roles, artifacts, and events of the Scrum framework are being used, or teams don’t use Sprint Goals because it’s too hard.
- Although Scrum is used, contracts remain defined by fixed scope, budget and planning.
- Scrum teams are not allowed to use any physical boards to create transparency. Everyone must use the same digital tool with tons of mandatory fields.
- Scrum has become a heavy-weight framework full of "complementary practices" like story points, velocity and mandatory use of user stories.
- Scrum is scaled throughout the entire organization without even one team applying it successfully. Or, scaling is done without understanding its purpose, for example by mindlessly copying the "Spotify Model".

The number one cause of Zombie Scrum, however, is that most organizations and teams don’t have a clue as to *why* they are doing Scrum. We often hear reasons like ....

- "We are doing Scrum because our Agile Coach said this was a great idea."
- "The CEO was promised an increase in productivity of 400%."
- "All our competitors are using it as well."
- "If we don’t increase our agility, we'll become the next Blockbuster or Kodak."

(image by Thea Schukken - Beeld in Werking)

Over the years working with different Scrum Teams and organizations, we've seen that reasons like these don't demonstrate a real understanding of why Scrum and empiricism matter in the first place. Without such understanding, any implementation will remain superficial and flat, as people don't have a good reason to persist in the face of the really hard challenges they are bound to face when trying to work empirically. It will just look like Scrum from a distance, but it will not actually help to learn faster about what works and what doesn't. Nor will it help deliver value to customers more quickly.

Luckily, there's at least one antidote we've found helpful: Liberating Structures.

What are Liberating Structures?

Liberating Structures are a collection of interaction patterns that participants learn to unflatten, enrich and deepen their interactions in groups through. With roots in complexity science, and collected and curated by Keith McCandless, Henri Lipmanowicz and a thriving community, they were initially developed twenty years ago in response to the stale and non-dynamic nature of interactions in organizations. The founders noticed how purposeless meetings, unengaging presentations and chaotic brainstorms were often dominated by a few voices. With this, the creative ideas and different perspectives from a silent majority were lost, as was the potential for novel solutions to persistent challenges.

Each Liberating Structure is specified on five design elements and embodies ten principles. Examples of design elements are the way space is arranged, how participation is distributed, and what invitation is used to start the interaction. Examples of principles are "include and unleash everyone", "build trust as you go", and "learn by failing forward". New structures are being developed all the time and existing structures are refined or creatively destroyed.

Liberating Structures are simple enough for anyone to learn, and able to "go viral" within organizations as people discover how well they work. By reshaping how people interact, the potential for self-organization and finding break-through solutions is unleashed.

Using Liberating Structures as an Antidote

Liberating Structures fit very naturally within Scrum, as many Scrum teams have discovered since we started spreading them within our community. They open a wide variety of applications, like problem solving, determining strategies, improving collaboration, and creating shared understanding. When working with Scrum, all of this is relevant. To make this tangible, I’ll give three examples of how we’ve used Liberating Structures as an antidote to Zombie Scrum.

1. Clarifying the purpose of doing Scrum

As mentioned earlier, the number one cause of Zombie Scrum is that many organizations and teams don’t know *why* they are using the Scrum framework. This lack of shared understanding can quickly turn into Zombie Scrum, as people mindlessly go through the process they think is Scrum. Teams only used the mechanics of Scrum because they were told to do so.

Liberating Structures can help organizations with defining a common purpose and strategy, and make working with Scrum a joint effort; not as a one-time trick, but as a continuously recurring activity focused on using Scrum in such a way that it's tailor-made for your organization.

Liberating Structures that help clarify the purpose of doing Scrum are Nine Whys, What, So What, Now What, and Critical Uncertainties.

Nine Whys

Discover the purpose of your work together as a group

The Liberating Structure Nine Whys helps groups to identify their purpose. It starts by inviting participants to make a list of the activities they are engaging in regarding the task or challenge at hand. It invites them to ask each other why these activities are important. What is it that you're trying to stop or start? Why is this necessary? What would be missing if you simply did something else? After interviewing each other, encourage everyone to write a short description of their purpose: "My/our work exists in order to…".

Examples and ideas of how to use it include:

- For Scrum teams as part of a team liftoff to identify the deepest reason for working together.
- As part of strategy and roadmapping initiatives to first clarify purpose and then actual goals.
- For Agile transition teams to shape the direction and intention of the change.

What, So What, Now What

Make sense of progress to date and decide on next steps together

What, So What, Now What is based on Chris Argyris' Ladder of Inference. Ask the participants to first list all the facts they have, observations they made and things they noticed (the What). After they have shared these items and created a common understanding, ask them to interpret what this means and what conclusions they draw from these facts (the So What). Finally, they are invited to discuss what next steps make sense based on their conclusions (the Now What).

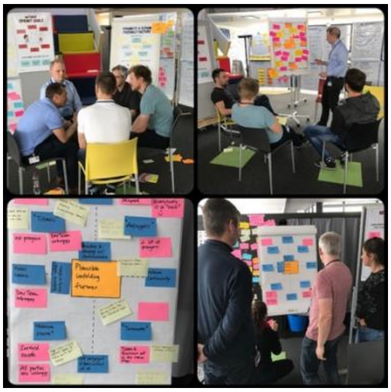
Examples and ideas of how to use it include:

- As part of a Sprint Review to digest and make sense of what was delivered during the past Sprint, any feedback that was received, and to decide on next steps together.
- Together with developers, to determine what strategies to use for optimizing codebases or reduce technical debt.
- To periodically reflect on how the organization is doing in their Agile journey. Invite teams and stakeholders to reflect on the process so far.

Critical Uncertainties

Develop strategies for operating in a range of plausible-yet-unpredictable futures

Critical Uncertainties is a concise, collaborative version of scenario planning. Instead of placing all your bets on one (often very optimistic) version of the future and assuming predictability, Critical Uncertainties works with multiple scenarios. This technique helps generate much more robust strategies while providing an excellent tool for collaborative sensemaking.



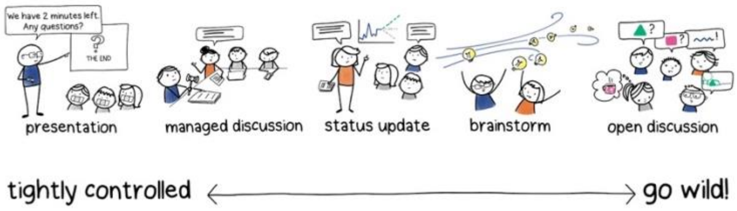
Pictures of strategy-making with Critical Uncertainties

Examples and ideas of how to use it include:

- Supporting the product owner in testing the viability of its product strategy. This helps expose assumptions and uncertainties and define next steps. Based on the different scenarios, the product owner could set Sprint Goals, update the product roadmap, engage with stakeholders, and offer clarity to the development team.
- Showing management how to involve everyone in shaping the future of the organization. As a joint effort, the possible scenarios and strategies for how to be successful were determined. This helped develop more organization-wide confidence in managing the unknowable future.
- Growing the self-organizing capabilities of the Scrum Team by having them think of the most critical and uncertain realities they might face. Doing this together not only bonded them as a team, but also allowed them to adapt to change more quickly.

2. Improving the Scrum events

When focusing on the Scrum events, what happens frequently is that they become too structured and inhibiting, with one person talking while the rest "listen" — like during *presentations, status meetings,* and *managed discussions*. Or the Scrum events are too unstructured and loose, with only a handful of people talking while the rest struggle to keep up — like during *brainstorms and open discussions*.



(image by Thea Schukken - Beeld in Werking)

Examples of how these conventional ways of interaction often manifest in the Scrum events include:

- The **Sprint Review** becomes a boring presentation (demo) given by the product owner with stakeholders (if present) as static listeners.
- The **Sprint Planning** results in a managed discussion with the Scrum master as the chairman discussing the Sprint Backlog in excruciating detail.
- The **Daily Scrum** becomes a dreadful meetup in which the development team mechanically answers all the work they've done and are planning to do, without any connection to the Sprint Goal.
- The **Sprint Retrospective** results in a brainstorming and open discussion, continuously identifying the same improvements; let's improve communication and collaboration.

By using Liberating Structures, the full wisdom, experience, and perspectives of the Scrum team or organization can be used to resolve impediments, make decisions, share knowledge and innovate. By doing so, it prevents the Scrum events from having these unproductive presentations, status meetings, etc.

Examples of Liberating Structures that will help improve the Scrum events are Impromptu Networking, Conversation Cafe, and TRIZ.

Impromptu Networking

Rapidly share novel ideas and make personal connections

Impromptu Networking allows a group of any size to form personal connections and share ideas in less than 20 minutes. It invites everyone to participate from the very start and share stories, challenges or experiences with each other. Not only is it a good way to "break the ice", but it also doubles as a clever way to use the collective brainpower of the group to rapidly identify patterns.

Examples and ideas of how to use it include:

- A couple of months ago we facilitated a Sprint Review with a large group of participants. It consisted of several Scrum teams and about 25 stakeholders. We used Impromptu Networking as an icebreaker exercise and offered them the questions: "If the product was a living, talking entity, what would it say right now? What secrets would you be worried that it might tell? What would you want it to say?" The result was a very fun start to the Sprint Review, which offered some valuable insights as well.
- We recently used it during the Sprint Retrospective by offering the questions: "If you could invite a special guest to the Retrospective, who would you invite?" and "How could you make the next Sprint dramatically worse than the one you have just had?" Of course, these questions are simply examples, yet the exercise will definitely kickstart your retrospective!
- Try Impromptu Networking during the Daily Scrum. In three rounds, each time in different pairs, have a conversation about the progress towards achieving the Sprint Goal. After the three rounds, identify patterns and determine a plan for the upcoming 24 hours.

Conversation Cafe

Engage everyone in making sense of profound challenges
Conversation Cafe uses two quick highly-structured rounds, one after another, talking very briefly and with a talking object to then lead into a longer, less-structured round in which an open conversation can take place. This leads to a totally different kind of engagement and spread of ideas. Conversation Cafe concludes with a final round of a structured exchange. Conversation Cafe helps people have calm and profound conversations in which there is less debating and arguing, and more listening.

Impressions of a group participating in Conversation Cafe

Examples and ideas of how to use it include:

- In the Sprint Review, use Conversation Cafe to make sense of the challenges you're facing with the development of a product. Give everyone the opportunity to share their perspectives and jointly gain insights on how to move forward.
- During the Sprint Retrospective, use Conversation Cafe to create a safe environment in which every member of the Scrum Team has the opportunity to speak their mind and share worries, anxieties or positive experiences.
- During the Sprint Planning, use Conversation Cafe to discuss the objectives the product owner has in mind and the work the development team needs to do in order to create a Done increment. It helps team members make sense of complexities, difficulties, or unclarities and lay the ground for being able to move on.

TRIZ

Make space for innovation by stopping counterproductive activities and behavior

The Liberating Structure 'TRIZ' invites the creative destruction of activities that limit innovation and productivity. It does so in a cathartic way that is fun, engages and involves everyone, and is bound to create some laughs along the way. This structure was inspired by an element in a problem solving approach created by Genrich Altshuller.

Examples and ideas of how to use it include:

- Use TRIZ during one of the first Sprint Reviews with stakeholders by asking, "What should we do to make this *the* nightmare project in the history of our company?" The result will be a list of activities that often went wrong in previous projects. This triggers a conversation on how to prevent this from happening again.

- During a Sprint Retrospectives in Scrum to stop counterproductive activities and behavior. Invite the team members to make a list of all they could do to guarantee that the upcoming Sprint will be the worst possible Sprint ever! After the Scrum team has created the list, ask them to be brutally honest and circle the items they already recognize from the previous Sprints. As a final step, help them define improvements to stop these activities.

- Use TRIZ during the Sprint Planning with a focus on delivering a "Done" increment. For the first round, ask, "What can we do to deliver the most un-done Increment imaginable?" Eventually, it will help the team create a more realistic plan for the upcoming Sprint in order to build a "Done" increment.

3. Encouraging Scrum teams' self-organizational capabilities

In a complex domain, a context's influence is so important that there are no "best practices". Therefore, we prefer the term "complementary practices". Examples of complementary practices are Story Points, User Stories, or doing the Daily Scrum standing. For some teams they're useful, for others they're not. Therefore, it's also up to the development team to determine the best way to accomplish its work and build "Done" increments. Figuring this out requires self-organization. Scrum supports self-organization by offering a lightweight framework that contains only three roles, five events, and three artifacts.

Liberating Structures support self-organization by offering 33 interaction patterns that are specified on five micro-organizing elements:

1. The invitation, the question or topic you want people to explore
2. How space is arranged and what materials are used
3. How participation is distributed among participants
4. How groups are configured
5. The sequence of steps and the time allocated to each step

As with the Scrum framework, Liberating Structures offer clear boundaries and constraints. It's up to the participants to self-organize within these constraints. When using Liberating Structures, the Scrum team is encouraged to explore local solutions that fit their context.

Although we can't mention all of them, some Liberating Structures that strongly encourage self-organization include Troika Consulting, Ecocycle Planning and Improv Prototyping.

Troika Consulting

Give and get practical help from peers

Troika Consulting helps people to gain insights on issues they face and unleash local wisdom for addressing them. In rapid rounds of "consultations", individuals ask for help and get advice immediately from two others. Peer-to-peer coaching helps the "client" in refining their skills in asking for help. They will learn to formulate problems and challenges clearly. It enables the "consultants" to improve their listening and consulting skills. Overall, this structure builds trust within a group through mutual support, builds the capacity to self-organize, and creates conditions for unimagined solutions to emerge.

Giving and getting help with Troika Consulting

Examples and ideas of how to use it include:

- During a Sprint Retrospective to facilitate the problem-solving process of the Scrum team. Gather a list of the problems the team is facing. Divide the Scrum team into groups of three. One person is the "client", the other two are "consultants". The client explains the problem, the consultants listen and offer advice. It encourages the Scrum team's self-organizing capabilities by having them think of solutions. As a Scrum Master, you facilitate the entire process.

- As part of backlog refinement to explore upcoming features, risks or technical challenges with each other and define possible next steps.
- During a Sprint Review to have stakeholders and the Scrum team gain insights on the issues they face. The goal of a Sprint Review is to collect feedback and to determine the best way to move forward. By using Troika Consulting, you can remove the possible barrier between stakeholders and the Scrum team and have them jointly explore problems and define solutions.

Ecocycle Planning

Bring clarity and focus in the activities you're doing individually or as a group

The Ecocycle is a powerful concept that helps Scrum teams engage in meaningful conversations. Any system in nature goes through the phases of Gestation (something is starting to take shape but might not be visible yet), Birth (something has taken form but needs nourishment to provide benefits), Maturity (something is well-established and you profit from it) and Creative Destruction (something has outlived its purpose and needs to either be decomposed or reinvented). By using this metaphor, teams can inspect a variety of topics, look at the progression of individual items and the distribution of items across the whole Ecocycle, and discuss next steps to advance items to the next phase.

Examples and ideas of how to use it include:

- Cleaning up a product backlog or an entire portfolio of products together with a representation of (or everyone) involved in the product(s).

- Continuously inspect how strategic ambitions match with the reality of product portfolio distribution on the Ecocycle.
- Plotting the activities that Scrum teams do and to discover what they should spend more time on, and most importantly, what to let go of.
- Helping Scrum masters and product owners develop themselves. What are they spending a lot of time on that isn't delivering value?

Improv Prototyping

Develop effective solutions to chronic challenges while having serious fun

Improv Prototyping uses the power of bodily re-enactment in an improv theater style. Instead of talking about ideas, the participants get to feel what it's like to actually act them out. First, a scene that needs to be improved is performed in front of everyone. The participants are then invited to develop solutions in small groups. In fast-paced interactions, improvements are made until a satisfactory solution is found.

Examples and ideas of how to use it include:

- We've used this approach to let Scrum masters act out significant challenges they face when working with development teams, and explore different strategies and behaviors.
- In our Professional Scrum Master-classes, we've used it to let participants enact Scrum events "gone wrong" — like a Daily Scrum that turns into a status meeting — and explore different behaviors and interventions.
- You can use Improv Prototyping during Sprint Retrospectives to help teams find better ways to interact during, and outside of, the Scrum events. You could also use it to explore pair/mob programming and swarming.

Closing Thoughts

The amount of organizations getting affected by Zombie Scrum is increasing rapidly. It's a serious problem that needs to be solved. Luckily there are multiple antidotes available, Liberating Structures being one of them. Also, there's a strong Scrum community focused on learning & sharing experiences and willing to explore novel solutions to persistent challenges; a community that offers opportunities for deepening skills, developing strategies and learning new techniques. The strength and influence of a community is determined by its members. Therefore, our hope is that you will join the community, with The Liberators Network as an example, and help us fight Zombie Scrum!

Want to Learn More?

If you want to learn more about Liberating Structures, the liberatingstructures.com website offers a vast amount of information. If you're interested in how to combine Liberating Structures with Scrum, check out this series of articles on liberating structures we wrote. However, Liberating Structures is something you should experience. Therefore, attending a local user group or an Immersion Workshop is strongly recommended.

About the Co-Authors

Barry Overeem - Co-founder of The Liberators. Barry liberates organizations from outdated modes of working and learning. Bringing in fresh energy and creativity, he creates space for everyone to be involved in shaping the future and making a positive impact. As a Professional Scrum Trainer and course steward, Barry is connected to Scrum.org. By providing training, facilitating workshops, speaking at conferences and writing blog posts, he shares insights, ideas, and lessons learned. Barry speaks Dutch & English fluently. However, when working with groups, he enjoys using the language of Liberating Structures the most.

Christiaan Verwijs - Co-founder of The Liberators. As a Scrum Master, Professional Scrum Trainer and Course Steward for Scrum.org, Christiaan helps teams become awesome. He is a passionate developer, statistics geek and full-time Nerd.

Johannes Schartau is a consultant, trainer and coach for Agile product development and organizational improvement. His interests in ethnology (with a focus on Amazonian shamanism), psychology, technology, integral thinking, complexity science and stand up comedy finally coalesced when he was introduced to Scrum in 2010. Since then he has dedicated himself to exploring organizations from all possible angles together with their members. His mission is to bring life and meaning back to the workplace by spreading Healthy Agile and Liberating Structures around the world. LinkedIn: www.linkedin.com/in/johannes-schartau

Blog: www.zombiescrum.org

Twitter: @IntegralAgile

Decide how you are going to decide

By Dhaval Panchal

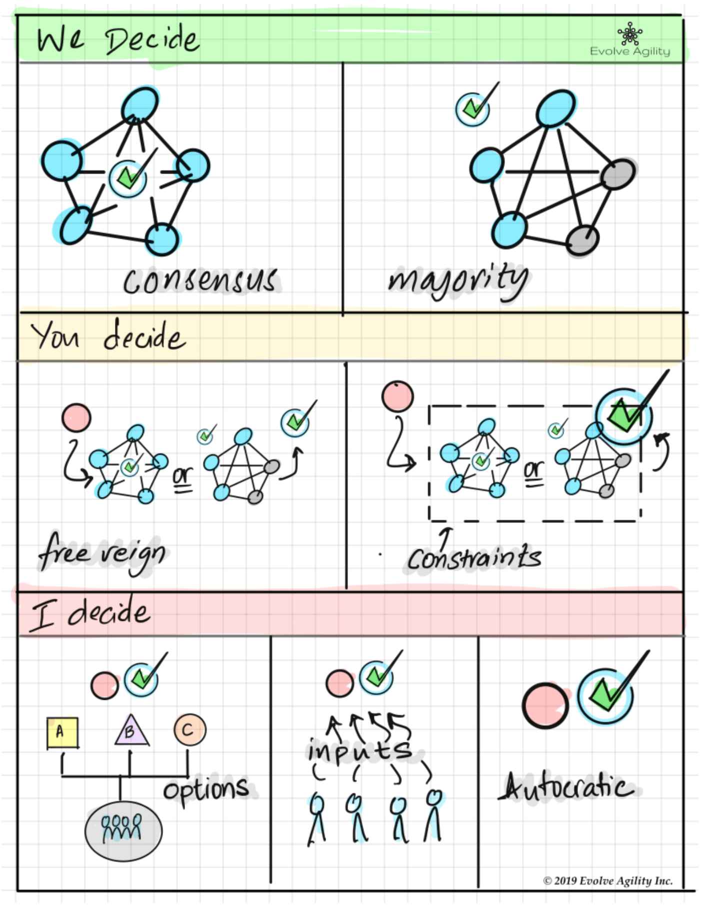
Meetings serve purpose of sharing information and/or enabling groups to make decisions. Team effectiveness can be judged based on quality of their decisions, and more importantly on cohesiveness of team membership after these decisions were made. Because in teams, decisions are not made on singular basis of expert judgement. Often a a sub-optimal decision that all team members support is more valuable than an optimal expert decision that divides a team into camps. For long term team health, it is important to decide how you are going to decide.

The consequences of not deciding how you are going to decide, can lead to misunderstandings. In dysfunctional team meetings most common kind of confusion is from simplest of misunderstandings – “Did we make a decision?”

Without a clear and explicit indicator that a decision was in fact made, people can remain under impression that further discussion remains to be had. Therefore people acting towards implementation can be perceived by others to be “impulsive”, “having their own agenda”. While on the other hand people who think more discussions are needed can be perceived to be “insubordinate”, “passive-aggressive”.

For teams that want to avoid these misunderstandings, it helps to setup decision making norms. Agreeing on these norms prior to a decision making moment is important so participants know beforehand, that when a decision needs to be made – HOW a decision will be made. It can be quiet frustrating to contribute but have no influence over final decision making. While people do not expect to have a say in every single decision, not knowing how their input will get used is common root cause for frustrations.

Here are three main modes of decision making:



We decide

For high stakes decisions that affect everybody, it is preferable that everybody is involved in making and agreeing on the final decision.

• **Consensus:** Unanimous agreements ensure everybody's buy-in and have long term 'stickiness'. Building consensus takes longer and cannot be rushed. Useful in high-stakes situations when impact of decision will impact everyone on the team.

For Example – Fist of five to check for team agreement on Sprint commitment at end of sprint planning.

• **Majority vote:** Creates win/lose situations. It could be simple majority or 2/3rd majority. It is important to remember that when majority wins, the minority looses. Popularity based voting can be effective in filtering from many ideas ideas *(example: dot voting*) or generating debate on competing options. Useful in high-stakes situations, when time is not on your side.

You decide

The person-in-charge delegates decision making to the group, and agrees to live with and support the decision made by the group. Delegating authority to make decision at team level where expertise and information resides is better than making decision by fiat.

• **No reservations delegation:** The person-in-charge gives complete unfiltered autonomy to the group to make a decision. The group can use consensus or majority vote techniques to arrive at their final decision.

For example: While deciding on place for lunch, a team member may express that they are "fine with anything" to express their willingness to go along with decision of rest of group.

Caution!: Carte blanche over decisions can be empowering to teams and yield unexpected results. So when managers have reservations for decision outcomes, they must express their reservations explicitly as constraints, or guiding principles before hand. Otherwise they risk being perceived as "unsupportive" or "insincere".

• **Delegation with constrains:** The person-in-charge provides simple set of rules, or constraints, or guiding principles to the group so the group can make informed decisions. These boundaries can be helpful in clarifying operating environment. Careful though, over prescription can feel farcical.

For example: In organization that promote self selecting teams, organization Sr. Management sets constraints that each team must be cross-functional, can meet product definition of done, and has 4-9 people each – and leaves specific team membership up to individual preferences.

I decide

When the person-in-charge makes unanimous decisions they assume full responsibility for the decision and its consequences. For personal matters, this is appropriate level of decision making. But when decisions impact other members of team, and for high-stakes decisions, group decision making is preferable. However in rare moments of true crisis there are advantages to making a decision in order to stabilize the situation first.

• **You give me options**, I decide: The group provides clear decision options with trade-offs for consideration. The person-in-charge retains final decision making authority.

For example: Team members provide multiple options to implement a feature and Product Owners makes final decision.

• **Consult, then I decide:** The group provides input to person-in-charge in order for them to make a decision. This approach is not recommended for group decisions, as it is difficult for the group to know what inputs were relevant in final decision made by person-in-charge.

For example: Manager conducts one-on-one meetings with subordinates on important organizational decision and finally makes a decision on their own.

• **Autocratic**, I decide: The manager makes a decision for the group. In rare moments of true crisis this can be useful.

For example: In case of fire in the building, resident fire warden direction is not questioned or debated to generally better outcomes. Unfortunately many organizations are stuck in some versions of “I decide” decision making mode. These are least effective and unnecessary most of the times. Taking time to decide how your team is going to decide can avoid many misunderstandings and lead to better long term outcomes.

About Dhaval Panchal

Experienced Executive and organizational agility Coach. Dhaval brings results oriented, people-centric perspective.

Dhaval helps his clients to focus on signals over noise. Exposes trade-offs for executives to make effective decisions.

He is one of a handful of Agile practitioners to reach CEC, CST, and CAL-E accreditation.

Dhaval is at the cutting-edge of implementing Agile in real-world situations. He is founder of Evolve Agility® - www.evolveagility.com - a Texas based Agile coaching, training, and transformation agency.

We have led large scale Agile transformations for companies in Oil & Gas, Banking, Insurance, Gaming, and Medical industries.

Big Room Planning and Looking at Dependencies Differently

By Allison Pollard

PHOTO BY TALLIS PHOTOGRAPHY

"We're going to invite people from 40 product teams to meet for 2 days. Our goal is to map dependencies between the teams for our enterprise initiatives and determine how we might need to shift priorities for feature delivery across teams.

It was a big ask. Roughly 200 people in a large room. Mapping out future work and dependencies on the walls, tables, and out into the hallway. It can be overwhelming to have so many people in a working session together. The goal was to create smoother delivery plans across so many teams. In my experience, big collaborative work sessions benefit from a very small amount of pre-work by attendees. The session is for work to happen. When people come in and have done a lot of pre-planning, we lose some benefits of the event:

The session becomes more of a read-out than collaborative event.
Engagement decreases, and potential discoveries go overlooked
The fidelity of information changes, and people are more averse to modifying plans in the room because of the effort already invested in it
There's potential frustration if the pre-work ended up being a wasteful activity

Interesting discoveries often happen in events like Big Room Planning. Folks recognize which products/teams are potential bottlenecks. A few teams realize they are expected to deliver something they had not planned for. Connections of names and faces are made between teams.

At least one person is surprised to see just how many dependencies exist in the organization.

My colleagues and I knew the challenges teams would continue to face after this event. They could be struggling to deliver overly complex solutions, losing sight of the intended business results, and waiting on other teams for days or even weeks at a time. Dependencies run the world.

What if we could disrupt the current thinking around dependencies?

Dependencies indicate that multiple products and teams are needed to deliver a solution that will (hopefully) create business value. The more teams involved in delivery, the harder it can be to coordinate efforts and integrate. That typically elongates schedules and delays deployment to production where we learn how customers respond to the new functionality. It’s easy to get caught up in delivering a particular solution rather than trying to move the needle for business outcomes. In any case, we need to define how business results will be measured and then ensure the proper metrics are captured. How did the thing we delivered help or not?

Going back to our Big Room Planning event, how could we use the time with 200 people differently when it came to dependencies? Instead of coming up with a plan for managing them, let’s reveal the dependencies and see how we can break them. Say 8 products show dependencies between them for a given effort--what’s the business result being pursued? Could those 8 product teams imagine a smaller experiment that would allow them to remove or avoid some of the dependencies? Maybe not all 8 teams need to be involved in the first deliverable that we can learn from. Create something simpler that can be deployed more quickly and measure customer behaviors. Validate if we’re on the right track to solving customers’ problems or not and enable the product to evolve from there based on customer usage. We allotted time for group breakouts to discuss the objective and outcomes.

People were excited to attend Big Room Planning. I’m still surprised at the buzz that was in the room those 2 days. Discoveries and connections were made. One team abandoned the plan they’d had and pivoted direction to focus on something more valuable. Conversations are continuing about how to “start smaller” so every product team can get measurable customer feedback from production every 6 weeks or less. Two days of Big Room Planning, and the group did not formulate a plan. It jumpstarted more planning, which is what is needed to ruthlessly break down dependencies and focus on delivering business outcomes.

Read this article online: https://baa.tco.ac/1SxH

Beginnings of an Agile Coaching Team

By Allison Pollard

There's been enough success with agile to justify forming a coaching team—congrats! Now what?

PHOTO BY TOM WOODWARD

A few big questions come up:

- How will we know if agile coaching is successful or not?
- How will coaching be structured in the organization?
- Which teams or groups will agile coaches work with?

That first question is a doozy! Agile coaches can be squeamish about metrics to evaluate how effective their efforts are because ultimately results are outside of their control. Yet we all like to know that our work adds value and makes a difference; when we feel we are not seeing positive results despite our best efforts, we will look to make a change in that coaching relationship. Metrics could support conversations with the people we coach about how things are going and if coaching should continue with them or not.

Coaching could be structured around working with groups for a particular timeframe or until a team reaches a particular set of capabilities. There's often an underlying assumption that every team will need agile coaching, and that leads to agile coaches having more teams to coach than they can handle at once. Organizations typically need a diverse coaching group that includes technical skills, product/business skills, and organizational change/team dynamics skills—this is important in enabling longer-term benefits of agile. Hopefully these coaches are working together (rather than in silos by specialty or disparate areas of the organization) and are aligned with a shared goal.

Which brings us to the final big question from above. There are many options to consider in determine which groups or teams to start coaching:

- Management/leadership so they “go first”
- Teams that ask for coaching because they are open and motivated
- Teams whose managers request coaching for them because they must be important and have management backing
- Teams working on the highest priority products/work for the organization so that they are more likely to be successful and create visible wins
- Teams who will be working on new products so they get started on a good path and we can make use of the fresh start from a timing perspective
- Teams that are bottlenecks for programs/other teams because these will have a multiplier effect for the organization

Other options probably exist, and there’s no clear “use this approach in all cases” answer. In fact, some agile coaching teams offer different products or services based on those “customer” personas or needs.

Newly formed agile coaching teams need to take some time to think through these questions and create their own charter. It would be easy to just start coaching and become so busy that we forget to reflect on our efforts. Let’s get clarity about our plan to help the organization because doing so will enable us to replan later as needed. An agile coaching team that can pivot based on organizational needs is quite amazing.

Read this article online: https://baa.tco.ac/1SxF

About Allison Pollard

Allison helps people discover their agile instincts and develop their coaching abilities.

As an agile coach with Improving in Dallas, Allison enjoys collaborating with coaches and leaders to unlock high performance and become trusted change agents in their organization and the community. In her experience, applying agile methods improves delivery, strengthens relationships, and builds trust between business and IT.

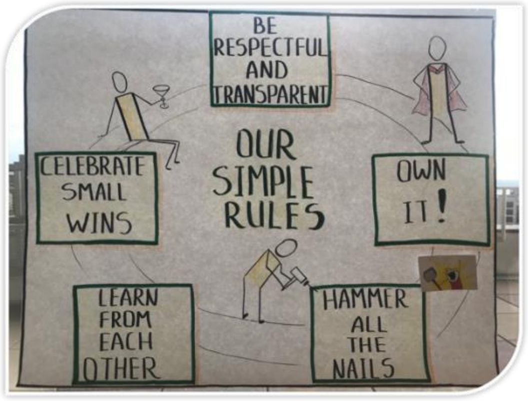
Allison is also a Certified Professional Co-Active Coach, a foodie, and proud glasses wearer.

To learn more visit

www.allisonpollard.com

Every team needs a Working Agreement

By Dana Pylayeva



Do you have a working agreement in your team

Unless your team is highly skilled in mindreading and doesn’t ever make any assumptions, you may want to consider creating one!

Working agreements (aka Team Norms, Simple Rules) – is a short list of behaviours or actions that a team agrees to engage in (and hold themselves accountable to) on a regular basis. One of the really cool things about Working Agreements – they are created by the team! And they get created in the language and the choice of word of the team. One of my favourite working agreements story is from a team of interns I worked with over the summer. Their Team Norms included a statement like

“One diva, one mic”.

This was their unique and beautiful way to express the need to take turns, when speaking as well as expect each one on the team to be respectful, listen without interruption.

As a Scrum Master you can certainly help to facilitate the process, offer techniques to engage everyone’s voice, but you can’t create their

Working Agreements on behalf of your team! It is for them to bring out what's the most important.

Why does a team need working agreements?

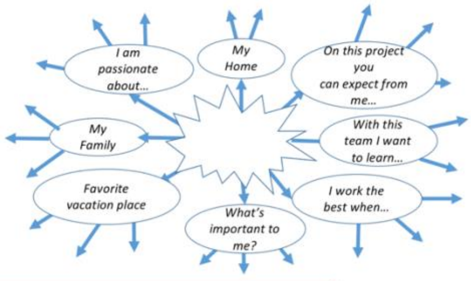
1. To clarify assumptions about what's acceptable in this team.
2. To acknowledge and embrace the differences. (Hey, not everyone is a morning person!)
3. To help them stay as a team when things "go south".

How do you create working agreements?

There are many ways to do it.
It could be as simple as starting with a silent brainstorming, followed by affinity mapping and dot-voting to help select the top 5 – 7 ideas.

I like to add a bit more fun and a personal touch to this by asking team members to pair up and interview each other, using modified personal maps.

For the personal map questions, I mix in a few about family, hobbies, favourite vacation spots with the ones that lead to the working agreements conversation:



Modified Personal Map with Working Agreement with Leading Questions

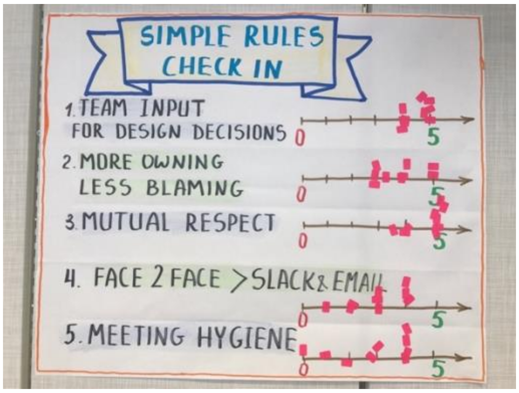
- I work the best when
- With this team I want to learn…
- Something that can really trigger me is…
- Top 3 things you can expect from me are…

I also ask each pair to come up with one answer that is an “exaggerated truth”. After about 6 minutes, team members take turns and introduce their pair to the rest of the team (using the personal maps they created in their interviews).
This is when the team gets to learn about everyone, laugh, while trying to guess the exaggerated answer…
And this is when a facilitator gets to capture highlights of these introductions in a “Same/Different” poster. These highlights become the basis for narrowing down on the Working Agreements choices for this team. (and some team members get to find their new skiing buddies too!)

Are Working Agreements carved in stone?

Of course not!
You can help the team and keep them alive by reviewing them in your retrospectives.

Why not facilitate a self-assessment on Working agreements? Ask your team next time:
“How are we holding ourselves to our team norms? On a scale from 0 to 5, how would we rate ourselves? Is everything still relevant?”


SIMPLE RULES CHECK IN
1. TEAM INPUT FOR DESIGN DECISIONS
2. MORE OWNING LESS BLAMING
3. MUTUAL RESPECT
4. FACE 2 FACE > SLACK & EMAIL
5. MEETING HYGIENE
0
5

Working Agreements check-in at a retrospective

Another opportunity to bring them up, when you have a team composition change: a new team member joining or someone leaving the team.
Last but definitely not least, team working agreements will help your team to stay civil during a conflict. Refer to them when things get

heated. Then discuss with the team what needs to be added to the Working Agreements to navigate conflict more effectively next time.

What are you waiting for? Go ahead, give it a try. Experiment with Working Agreements in your team and have fun doing it!

About Dana Pylayeva

Dana is an international speaker (50 conferences in 15 countries), trainer, experienced remote facilitator, author of several publications and agile games, including "DevOps with Lego and Chocolate", "Fear in the Workplace" and "Safety in the Workplace".

She draws her inspiration from the 20+ years of hands-on experience in IT, Business Agility/ DevOps culture coaching, working with international clients and facilitating change in Fortune 500 companies.

Dana is a certified Training from the Back of the Room trainer, Open Space facilitator, CSM, CSP, CSPO, and Certified LeSS Practitioner.

Dana is the founder of the Big Apple Scrum Day conference in NYC, a leader of NYC Liberating Structures community, and a program chair of Agile2020 conference with Agile Alliance.

Team Coaching and the Scrum Master Role

By Tom Reynolds

What is the relationship between Team Coaching and the Scrum Master role?

Fundamentally, a Scrum Master is a team coach and servant leader and as such team coaching and the Scrum Master role are synonymous with one another. The exact definition of the role in the Scrum Guide, vis-à-vis how the Scrum Master works and serves the development team, is as follows:

How does the Scrum Master serve the Development Team?

The Scrum Master serves the Development Team in several ways, including:

- Coaching the Development Team in self-organization and cross-functionality
- Helping the Development Team to create high-value products
- Removing impediments to the Development Team’s progress
- Facilitating Scrum events as requested or needed, and
- Coaching the Development Team in organizational environments in which Scrum is not yet fully adopted nor understood.

There is a large emphasis on coaching within the role of the Scrum Master. So, we must ask, what in fact is team coaching? How is it different to 1-to-1 coaching? And how does this definition relate to the role of the Scrum Master and indeed the Scrum Framework?

A new study in 2019 examined team coaching and how it differs to 1-to-1 coaching and found that team coaching is focused on four primary issues:

1. Developing and achieving a common goal.
2. Helping the team develop a higher level of performance.
3. Developing a culture of learning and reflection.
4. Developing inter-team trust, communications and awareness.

The study revealed that, to achieve the above, team coaches need to:

- See the team as a system.
- Have advanced coaching and group facilitation skills.
- Have a higher level of awareness of a range of issues not required in one-to-one coaching.
- Be a long-term fixture with the team.

This study was conducted by researchers from Henley Business School, University of Reading, Aston Business School and the Management School and the University of Liverpool. Collectively, they have looked at how team coaching is developing as an idea and as a concept. Through this research they found that team development broadly sat within four types of activity and that the following definitions were the most common:

What constitutes a team development process?

- Team building
  - Some form of team engagement in an activity, followed by reflection on the behaviour and relationships of individual members and how they contribute to the team's overall performance.
- Team training
  - Usually considered to be a systematic and staged strategy to improve performance on a task or a series of tasks, or to improve team cohesion and performance.
- Team development
  - An overarching term for the process of using any method to develop a team, or sense of team, and a more specific set of activities to improve team performance.
- Team coaching
  - A continual developmental process whereby there is a coach whose aim is to provide feedback and encourage reflection about how the team is performing. It's considered to be a series of ongoing direct interventions to increase coordination, task performance and making good use of their collective resources.

They further found that a coach is separate to the team in respect that they are not supervisors or team leaders. A 2016 study demonstrated that this independent team coach has a very different form of authority and power dynamic than a traditional manager or leader.

Where is the synergy between team coaching and the Scrum Master role?

The 2019 research outlined four primary themes, which I have listed below. I have also, alongside their findings, given examples of how these findings map to the ScrumMaster role and the Scrum Framework. You will see how the role is inherently a coaching role and is supported naturally by the Scrum framework to enable coaching to take place.

Findings	The Scrum Master role and Scrum Framework overlap
The team needs a common goal and the coach is there to help the team define the common goal. This ensures every member of the team has a similar understanding of the purpose of the team at that moment in time.	As part of the Scrum Master's service to the Product Owner, the Scrum Guide says "Ensuring that goals, scope, and product domain are understood by everyone on the Scrum Team as well as possible" The Scrum Master during Sprint Planning will facilitate the formulation of a Sprint Goal and ensure that it is understood by the whole Scrum Team, more so each "increment is a step toward a vision or goal". So, not only does the Scrum Master facilitate the production of the Sprint Goal, they commonly facilitate or, at the very least, ensure that there is a clear over arching Vision for the Product or Project. Moreover, they ensure that everyone is clear about this vision so that the whole team has a common goal and a shared understanding of the goal.this vision so that the whole team has a common goal and a shared understanding of the goal.

Team performance is found to be the next concern of team coaching and this branches on two levels: • How the team performs as a team (how they interact, make decisions, deal with conflict etc.) • How the team completes tasks (skills, knowledge, application etc.)	With reference to the Scrum Guide, the Scrum Master serves the Development Team in several ways including: • Coaching the Development Team in self-organization and cross-functionality. • Helping the Development Team to create high-value products. • Removing impediments to the Development Team's progress. The Scrum Master also serves the organisation by • Causing change that increases the productivity of the Scrum Team. And serves the Product Owner by • Finding techniques for effective Product Backlog management • Helping the Scrum Team understand the need for clear and concise Product Backlog items • Understanding product planning in an empirical environment • Ensuring the Product Owner knows how to arrange the Product Backlog to maximize value • Understanding and practicing agility

	The Scrum Master achieves this through various coaching interventions; for example, during Sprint Planning, the Daily Scrum, Product Backlog Refinement, Sprint Planning and, most importantly, the Sprint Retrospective.
Team learning and reflection. The study found that a primary aim of team coaching is to ensure that the team constructs knowledge and develops the skills and attitudes needed, for example, to help the team to learn and think about itself and how it operates. This means creating a safe reflective space.	The Scrum Master would most generally achieve this through their skilled facilitation of the Sprint Retrospective. The Scrum Master creates a safe place for team members to say what needs to be said and a safe space for the team to explore, reflect and learn so that they can take positive action going forward to improve. The purpose of the Sprint Retrospective is to: • Inspect how the last Sprint went with regards to people, relationships, process, and tools; • Identify and order the major items that went well and potential improvements; and, • Create a plan for implementing improvements to the way the Scrum Team does its work. The Scrum Master encourages the Scrum Team to improve, within the Scrum process framework, its

	development process and practices to make it more effective and enjoyable for the next Sprint. During each Sprint Retrospective, the Scrum Team plans ways to increase product quality by improving work processes.
Capability development. This involves helping to develop trust, improve communications and increase interpersonal, personal, organisational and situational awareness	Ultimately, the Scrum Master achieves this through the coaching actions they take against the three previous findings and undertaking their responsibilities as defined within their role in the Scrum Guide, namely serving the Product Owner, The Development Team and The Organisation. The Scrum Master achieves this via the Scrum Events, through coaching interventions, and by creating a safe place for the team to explore, reflect and learn. There are many techniques that a Scrum Master may use to help the team achieve this outcome and ultimately through their skilled coaching and facilitation they are enablers for team growth and development.

Coach the team as a system

The findings also revealed that the team coach views the team as a system and focuses on that system as an entity, whilst still recognising the varying skills, knowledge, needs and perspectives of the individuals.

What is interesting about the above finding is that this view is perfectly reflected in Organisation and Relationship Systems Coaching (ORSC). This is an International Coaching Federation (ICF) certified and approved coaching framework. The four cornerstones of the Relationship Systems Model in the Workplace are:

1. The Relationship System is the Focus
2. The Relationship System is Naturally Intelligent and Creative
3. Work With the Whole Relationship System
4. Reveal the System to Itself

The findings of the study and the ORSC model fit in perfectly with the mindset of a great Scrum Master and it is interesting to read Eben Halfords blog "You are probably not part of a real team" which also talks about teams as systems.

Moreover, the study found that coaching a group of people together requires advanced coaching and facilitation skills which entails:

- The ability to deal with the risks involved of taking on a group.
- Being able to navigate a complex and continually changing, developing and revealing series of relationships.
- The ability to listen to and work with multiple and often conflicting or competing perspectives, ways of thinking and beliefs/value systems at the same time.
- The ability to observe and interpret interactions and spot emerging properties and to 'read' relationships.
- Understanding and being able to work with and use group dynamics, processes and facilitate a group 'live'.
- Developing a collective conversation around issues of importance for the team.

Lastly, the researchers identified that team coaches are usually invested in the long-term performance of the team and are usually not engaged on short-term discrete outcomes. As such, they need to build a trusting and powerful relationship with the entire team.

In conclusion, the 2019 study found that organisational team coaching is ‘live’, with real-life organisational issues being dealt with and changing as the coaching is ongoing. Organisational team coaching is in a live context with ‘in-the-moment’ consequences and outcomes. This requires an advanced level of awareness and skills on the coach’s behalf.

Conclusion – The Scrum Master is a Team Coach

This study for me is totally compatible with the role of the Scrum Master, after all this role is a coaching role, team coaching and the Scrum Master role are totally aligned. To be a great Scrum Master is however an on-going journey of discovery and learning. Some ideas of how to develop as a Scrum Master and agile coach can be found as part of our Agile Coaching Growth Wheel. There are many areas to explore and many areas of self-learning. There are also elements that you can develop on a more formal basis through direct training and coaching. However you choose to proceed in the journey, it will be very fulfilling, but can often be difficult and challenging. The most important takeaway from this: the ScrumMaster is a team coach.

References

Jones, R. J., Napiersky, U., & Lyubovnikova, J. (2019). Conceptualizing the distinctiveness of team coaching. Journal of Managerial Psychology, 34(2), 62-78. In Team coaching and how it is different to 1-to-1 coaching. Research Briefing. The Oxford Review. www.oxford-review.com

Schwaber, K & Sutherland, J. (2017). The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game.

About Tom Reynolds

Tom is a highly experienced coach whose insights into the Agile mindset make his training and coaching both effective and genuinely transformative.

He has spent more than 30 years working in IT, focusing on Agile since 2007. Today he is a certified Scrum trainer, Kanban trainer, Agile coach, Relationship Systems Coach and Personal/Life Coach who believes passionately that Agile is not just a new set of processes – it's a new way of thinking.

Tom's training and coaching are delivered with a freshness and energy that make the process of embracing change an enjoyable as well as challenging experience.

To learn more visit

www.beliminal.com

www.linkedin.com/in/tomjreynolds

Agile Catalyst Co-Creation Advanced Learning Experience Cube Coaching Block Toddler Interaction Session

By Andrew Rose

The block pulley contraption (Photo: Andrew Rose)

Unpacking a Buzzword Compliant Article Heading

My wife and 2-year-old son were playing with building blocks.

A single adjustment to his construct was the genesis of this post.

I made one adjustment to a toy and delighted myself on how the successive improvements flowed from one to another.

My son was delighted that he had a new toy.

If this post brings me some mirth when I reflect on it a few years from now, then I'll consider it a success.

For the reader, this post intends to showcase some concepts using observations of my family playing with a construction set. There are a couple of tools introduced here and my hope is to put some tools out there for others to learn.

I would also like to show how play, and an iterative approach to problem solving can form an iterative, experimental opportunity generating mindset.

Agility is not just "a thing" that technology people do. It should be "a thing" that all business people do, unless such business owners want to stay romantic with how they make money. The speed at which markets are changing means that businesses cannot become too comfortable with their business model.

Here are the unpacks for this article title:

Agile - Learning through doing, getting feedback, incrementing on the product design with the new feedback.

Catalyst Co-Creation - There are no expert opinions being forced. We build on each other's ideas. The resulting product is far better than what any one of us could produce as it was all of our design.

Advanced Learning - Any form of advanced learning is indistinguishable from Play. How often would you and your friends pick up something new and... just play with it to determine what it does?

Experience Cube - A method of structuring our thoughts, emotions and wants around an observation. Everyone makes the same observation but has different experiences about it.

Toddler Interaction Session - An increment of playing with my kid!
The title sounds somewhat important. Should I create an alt-doctorate program?

What do you want?

I returned home with my son after some time at a playground. We were giving mom a break so she could plant her balcony garden. Upon return, my son wanted to watch train videos. We are ensuring he has limits on screen time, so my wife started to play blocks with him while I worked on some educational materials.

The contraption they evolved with the blocks was a rolling tower similar to a medieval siege tower. He's more of a tinker toddler rather than an imperium toddler so we will call it a ball drop tower. Knights and castles may come at a later time in life!

My son likes to play with wooden balls that came from a Connect-4 inspired game. He finds it fun to have the balls drop through the tower and be ejected at the bottom.

What does my son enjoy:

- Construction toys
- Cause and effect when experimenting with gravity
- The randomness caused by watching things drop under the power of gravity

How to measure success?

My son is not quite 2 and a half years old at the time of this writing. His needs are not very complex so having fun is at the top of his list.

How do we measure "fun" for toddlers? I found that my little guy just wants to have fun and receive appreciation from mom and dad. He also runs his own Boredom Minimization Framework, of which the rules are still not fully known by mom and dad.

We will measure success using the following:

- The loudness of his laughs
- Number of spins in his victory dance
- Length of focused, quiet engagement with the toy after the initial enthusiasm.

Experience Cube

You need to be aware of your experience before you can describe it, and you need to know what experience is in order to use your curiosity well. (Busche, G., 2009)

My paraphrasing of Busche's work is: An experience is our interpretation of an observation we make. We have an awareness of an event that creates an emotional response within ourselves. There is then a voice in our heads that forms a narrative. Then what follows is a want or moment of curiosity.

An example of using the experience cube to step through an event may look like this: It is a rainy day, then the sun breaks out and starts shining, it warms up outside. When looking at this there is a feeling of happiness. The thought running through my head is the sun that comes out after a rain is very refreshing. A moment of need arises, I want to go outside to be in the sun as I am curious if I will feel that same refreshing feeling as I have in the past.

Another can observe the sun shining and feel a sense of despair. Their thought: I now need to honour my agreement to go out and finish the yard work, I wanted to stay inside and binge watch a Netflix series while it was raining!"

One observable truth leads to two completely different experiences. The Experience Cube helps us to "create data" by expressing our feelings, thoughts, and wants. Teams that exercise this tool create greater understanding between their members.

Leadership Stances

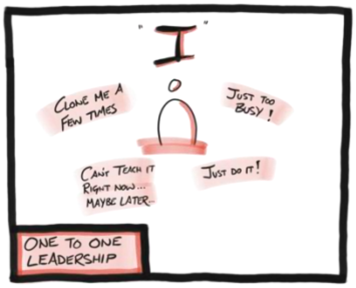
Leaders in knowledge based industries need to adopt new ways of adapting to rapidly changing market conditions. (Joiner, B., 2007). Industrial command and control style may have limited success with knowledge based product development. Effective executives work to grow themselves, their superiors, their reports and the peers that work around them. (Drucker, P. F., 2002)

I will provide a short summary of the leadership stances. My hope here is it sparks a reader to investigate these further. I believe we would be better served to have more people with catalyst based leadership toolkits in the world.

Here is my summary of Joiner's Leadership stances. No leadership stance is truly better than the other. They each have their representative toolkits. However, there is a learned progression of the stances. A Catalyst stance is an evolution of the Achiever stance, which is an evolution of the Expert stance. A Catalyst leader has access to all the tools of the three stances. An Expert leader only has the tools of the Expert stance. The Catalyst leader stands the greatest chance to create more options for their teams, increasing their chance to discover breakthrough solutions.

Expert Stance

Expert Leaders focus on solving the problem as fast as possible. They are very skilled and tend to default to creating groups rather than building high performing teams. Providing opportunities for learning, growth, and opening paths for serendipity may be viewed as "inefficient". Often the Expert Leader is an individual contributor who is elevated to a management or supervisory role because of their performance while on the line.



In my toy observations later in in this article, a pulley system that we implemented on the toy, broke. From an Expert Leadership stance, I go over and fix it before returning to my writing.

Experience Cube - I saw that the pulley system broke. I feeling apprehensive because I thought I would get aggravated if my son disengaged with the toy (he would come to the laptop and start pressing keys at random). I wanted him to continue playing with the toy so I could write this article!

Achiever Stance

Achiever Leaders focus on achieving goals as fast as possible and landing the quick results. Training and education is looked upon as a means to an end. Achiever leaders set in their ways tend to approach people as commodities which packaged skills. Those that produce are rewarded, those that flounder are discarded.

A softer approach to Achiever leadership is to set the measures for success and have the team provide the "how". The intention here is not to colour this leadership style in a negative light. There is a time and place for each style.

"MY WAY"
PEOPLE AS "RESOURCES"
STRETCH GOALS
ON THE SAME PAGE OR GET OFF THE BUS!
ONE TO MANY LEADERSHIP

I ask my wife to cut a piece of string so I can create the pulley system. If I was in an expert leader stance, I could have found the scissors myself and cut the string. However, I did not know where the scissors were at the time and I wanted to implement "my" idea.

Experience Cube - I saw that he was manually lifting the balls to the top of the tower. I felt a bit of a fun feeling coming on with the thought that I could create a magnet lifting system (he likes magnets) as I wanted him to engage with the toy rather than asking for videos.

Catalyst Stance

Catalyst leaders show a passion for a vision and grow their people in order to achieve it. There are many approaches to a problem and they will engage with the team to create many diverging options before having the team converge on a solution. This leadership style engages all members of the team to be active participants in generating options. This can lead to solutions that are significant breakthroughs that no single member of the team could have created individually.

EXPERIMENT TO LEARN
ALL BRAINS IN THE GAME!
"WE"
"WITH"
CONSIDER ALL IDEAS!
MANY TO MANY LEADERSHIP

Iterative Crane Development

Iteration 1

While building the toy contraption, my wife added this mechanism to the top of the tower. She then operated it in a pincer-like fashion. I looked up from my work and thought (Achiever Stance), "Neat idea, but there is an opportunity for more fun".

Iteration 2

Selecting a single blue unit block, I add it to the pincer contraption so that there is a larger range of motion. The yellow boom can now swivel 360 degrees. My thoughts were to add a slight improvement on the design (Catalyst Stance) to open up more options for play. My wife started to swivel the unit while making a mechanical sound.

Laughter from my son.

Feedback noted.

Iteration 3

Looking up from my work again, I see that they are manually lifting small wooden balls to the top of the tower then dropping them down the centre. This is a tower ball drop design that my son enjoys.

Iteration 4

There is a piece of string tied to a toy car which is used to pull it around the room. The Expert leader in me would like to take that string and improve the contraption, so I start to tinker with it.

I'm not getting anywhere. My craft skills are rusty.

The Achiever leader in me wants to ask my wife to get something to cut the string. The Catalyst leader in me has me explain the vision of what I want to do to my wife. So she has some ideas, gets the scissors, cuts the string, ties it off on a block, slings it over the yellow boom.

Iteration 5

The contraption needs to elevate stuff up to the top of the tower. So my intention is to connect blocks to the block that mimics a crane hook, then elevate them up to the top.

Iteration 6

My son finds that the rope slips too much and the payload drops back down.

He installs a flag on top of the tower. I think this was just to add some fun to the tower. He has seen flags on top of buildings so may have been mimicking that design.

However, that little bit of genius trapped the string between the flagpole block and another block on the top of the tower. This provided friction on the string and the payload would no longer fall.

Experience Cube - I was seeing the design evolve. I noticed I had some urges to jump in and "fix it and make it better". I caught this urge then thought I would just stop and watch, wondering what would happen next.

There was a lot of quiet building time during the last few iterations. A key observation of high performing teams is they get very quiet when they are focused and in flow. As an important "Public Service Announcement" - Do not go into a team room when the members are heads down focusing, shout out how quiet it is, then attempt to bring "life to the party".

Iteration 7

We discovered that the balls fit perfectly into the elevation block when we flipped it upside down. We tinkered with it a bit and my son helped QA the change before we put the friction block back on. A couple of slight modifications including moving the elevating block to the other side of the boom so that the balls can be dropped into the tower with one move rather than pulling the device over the boom. Too much kludge.

Lots of laughter and "Wows" from my son.

Success!

And everyone is happy - we all played our parts!

My son now knows the basics of a pulley system. Coming back to the toy after a couple of hours, he re-installed the elevator device and started to raise and lower the payloads, air dropping the balls, reloading them, overloading the container, moving the entire contraption back and forth while raising and lowering the payloads, fast lowering the payloads so they spill and see how the balls move, creating an unbalanced load, integrating the contraption with a Brio train toy.

My tired son eventually deconstructed the crane platform. Later in the evening, he rebuilt it with the help of my wife and I showed them how the flag piece provides friction on the pull string in order to suspend the load. Looking at it now, there might be an opportunity to evolve it into a sort of gantry crane...

A few days later while editing this article my son walks up to me, with the string and attached elevator brick in hand.

"Error, Error", he exclaims.

The pulley system was disconnected from the tower and he wanted to report the incident...

References

- Joiner, B., & Josephs, S. (2007). *Leadership agility: Five levels of mastery for anticipating and initiating change.* San Francisco: Jossey-Bass.

- Drucker, P. F. (2002). *The effective executive*. New York: HarperBusiness Essentials.

- Bushe, Gervase, R. (2009). Ch. 4. The Four Elements of Experience: The Experience Cube. In Clear Leadership: Sustaining Real Collaboration and Partnership at Work (pp. 91-107). Mountain View, CA: Davies-Black Pub.

Read this article online: https://baa.tco.ac/1SxL

About Andrew Rose

Andrew has over 25 years experience working with product delivery organizations, including startups, small-medium business and local government.

Joining amazon.com back in 2000 set a foundation of agility which he carries with him today. He is passionate about transforming the world of work by exploring whole human practices, through a blend of team coaching and agile methods.

He holds a Certified Team Coach credential with the Scrum Alliance, Associate Certified Coach with the International Coaching Federation, and is a Certified Organizational Coach through the University of British Columbia.

His servant leadership style has sparked the creation of mentoring and coaching programs at various organizations he has been a part of.

He moved to Vancouver in 2011 to contribute to the local professional community through speaking, workshops and master classes.

Andrew can be found in his local communities of practice, including Agile Vancouver and the International Coaching Federation Vancouver Chapter. He has presented workshops at global conferences including Global Scrum Gathering and Building Business Capability.

Email Andrew at

andrew@forxia.com

Large-Scale Agile: Where Do You Want Your Complexity?

By James Shore

One of the pernicious problems in large-scale software development is cross-team coordination. Most large-scale Agile methods focus on product and portfolio coordination, but there's a harder problem out there: coordinating the *engineering* work.

Poor engineering coordination leads to major problems: bugs, delays, production outages. Cross-team inefficiencies creep in, leading to Ron Jeffries' description: "a hundred-person project is a ten-person project, with overhead." One of my large-scale clients had teams that were taking three months to deliver five days of work—all due to cross-team coordination challenges.

How do you prevent these problems? One of the key ideas is to isolate your teams: to carefully arrange responsibilities so they *don't* need to coordinate so much. But even then, as Michael Feathers says, there's a law of conservation of complexity in software. We can move the complexity around, but we can't eliminate it entirely.

So where should your complexity live?

Monolith: Design Complexity

In the beginning, applications were monoliths. A monolith is a single program that runs in a single process, perhaps replicated across multiple computers, maybe talking to a back-end database or two. Different teams can be assigned to work on different parts of the program. Nice and simple.

Too simple. A monolith encourages bad habits. If two parts of the program need to communicate, sometimes the easiest way is to create a global variable or a singleton. If you need a piece of data, sometimes the easiest way is to duplicate the SQL query. These shortcuts introduce coupling that make the code harder to understand and change.

To be successful with a monolith, you need careful design to make sure different parts of the program are isolated. The more teams you have, the more difficult this discipline is to maintain. Monoliths don't provide any guardrails to prevent well-meaning programmers from crossing the line.

Microservices: Ops Complexity

At first glance, microservices seem like the perfect solution to the promiscuous chaos of the monolith. Each microservice is a small, self-contained program. Each is developed completely independently, with its own repository, database, and tooling. Services are deployed and run separately, so it's literally impossible for one team to inappropriately touch the implementation details of another.

Unfortunately, microservices move the coordination burden from development to operations. Now, instead of deploying and monitoring a single application, ops has to deploy and monitor dozens or even hundreds of services. Applications often can't be tested on developers' machines, requiring additional ops effort to create test environments. And when a service's behavior changes, other services that are affected by those changes have to be carefully orchestrated to ensure that they're deployed in the right order.

I see it so often, I've given it a name: "Angry Ops Syndrome."

The microservice ops burden is often underestimated. I see it so often, I've given it a name: "Angry Ops Syndrome." As dev grows, the ops burden multiplies, but ops hiring doesn't keep pace. Problems pile up and a firefighting mentality takes over, leaving no time for systemic improvements. Sleepless nights, bitterness, and burnout result, leading to infighting between ops and dev.

Microservices also introduce complexity for programmers. Because of the stringent isolation, it's difficult to refactor across service boundaries. This can result in janky cross-service division of responsibilities. Programmers also need to be careful about dealing with network latency and errors. It's easy to forget, leading to production failures and more work for ops.

To be successful with microservices, you need well-staffed ops, a culture of dev-ops collaboration, and tooling that allows you to coordinate all your disparate services. Because refactoring across service boundaries is so difficult, Martin Fowler also suggests you start with a monolith so you can get your division of responsibilities correct.

Nanoservices: Versioning Complexity

What's a nanoservice? It's just like a microservice, but without all the networking complexity and overhead. In other words, it's a library.

Joking aside, the key innovation in microservices isn't the network layer, which introduces all that undesired complexity, but the idea of small, purpose-specific databases[1]. Instead of microservices, you can create libraries that each connect to their own database, just like a microservice. (Or even a separate set of tables in a single master database. The key here is *separate*, though.) And each library can have its own repository and tooling.

*[1]Thanks to **Matteo Vaccari** for this insight.*

Using a library is just a matter of installing it and making a function call. No network or deployment issues to worry about.

But now you have versioning hell. When you update your library, how do make sure everybody installs the new version? It's particularly bad when you want to update your database schema. You either have to maintain multiple versions of your database or wait for 100% adoption before deploying the updates.

To be successful with libraries, you need a way to ensure that new versions are consumed quickly and the software that uses those versions is deployed promptly. Without it, your ability to make changes will stall.

Monorepo: Tooling Complexity

Maybe total isolation isn't the answer. Some large companies, including Google and Facebook, take a hybrid approach. They keep all their code in a single repository, like a monolith, but they divide the repository into multiple projects that can be built and deployed independently. When you update your project, you can directly make changes to its consumers, but the people who are affected get to sign off on your changes.

The problem with this approach is that it requires custom tooling. Google built their own version control system. Facebook patched Mercurial. Microsoft built a virtual filesystem for git.

You also need a way of building software and resolving cross-team dependencies. Google's solution is called Blaze, and they've open-sourced a version of it called Bazel. Additional tooling is needed for cross-team ownership and sign-offs.

These tools are starting to enter the mainstream, but they aren't there yet. Until then, to be successful with a monorepo, you need to devote extra time to tooling.

Which Way is Best?

I'm partial to the monorepo approach. Of all the options, it seems to have the best ability to actually reduce coordination costs (via tooling) rather than just sliding the costs around to different parts of the organization. If I were starting from scratch, I would start with a monorepo and scale my tooling support for it along with the rest of the organization. In a large organization, a percentage of development teams should be devoted to enabling other developers, and one of them can be responsible for monorepo tooling.

But you probably don't have the luxury of starting from scratch. In that case, it's a matter of choosing your poison. What is your organization best at? If you have a great ops team and embedded devops, microservices could work very well for you. On the other hand, if you aren't great at ops, but your programmers are great at keeping their designs clean, a monolith could work well.

As always, engineering is a matter of trade-offs. Which one is best? Wherever you want your complexity to live.

(Thanks to Michael Feathers for reviewing an early draft of this essay.)

About James Shore

James teaches, writes, and consults on Agile development processes.

He is a recipient of the Agile Alliance's Gordon Pask Award for Contributions to Agile Practice, co-author of The Art of Agile Development, and co-creator of the Agile Fluency™ Model.

You can find more of his writing on his

Art of Agile blog at

www.jamesshore.com

Gearing the Leadership Runway

By Sourav Singla and Lalita Chandel

Credit: https://www.regalunlimited.com/tag/executive-coaching/

Fill in the Blanks.

_________ plays a key role in the Agile Transformation for Organizations. True agility comes from a shift in the _________ mindset.

What words are coming to your mind?

Growth? Leadership? Organizational Learning? Goals? Mindset? If yes, we are aligned, Bingo!

Leading and sustaining any change hits the biggest blockers without **Leadership Support**. As per our experience so far, it is one of the biggest setback agile coaches face during their respective assignments. Here are some Questions for those who are in "**coaching**" roles in some capacity?

“Do you experience any of the below traits in the leadership where you worked?”

- *Missing alignment, aspiration and value for an agile transformation*
- *Ignorant of the cultural and change-management implications of agile*
- *Lack of scaling strategy and roadmap beyond pilots*
- *Highly tuned towards planning and controlling instead of testing and learning, experimentation and failures considered as a threat to the organization*

In **90%** of the transformation assignments, we have worked, we have seen most of these **de-railers**. Now here are some questions for our "**managers, executives and leaders**".

- *Do you believe that you generally know what is best for your team?*
- *Do you frequently and willingly offer solutions/advise to your team?*
- *Do you actively involve yourself into problem solving? Do you wait to receive invitations from your team for problem solving?*
- *When interacting with teams, how do you put forward questions? Are they close ended questions? Example: Who is doing this work? When will this work be completed?*

If your answer is majorly "Yes", then this blog is surely for you. Bingo!

Indeed, our brain is hard wired to answer questions, solve problems and fix issues. When rubber hits the road, unknowingly, you may be the one, who is slowing down the whole transformation journey. Interestingly, this first needs a conscious observation, unlearning and the subtle art of letting go of the control. To change the strong "anchored" patterns and to learn how to pull insights out of your team members, we need to adopt fresh beliefs that can enable others to explore innovative ideas.

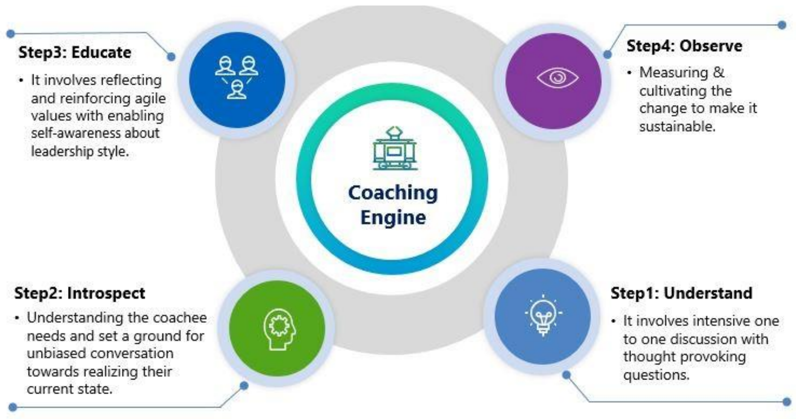
This blog's objective is to take a laser focus and explore some tools and techniques, that can be used by Executives, Leaders and Coaches, while dealing with above mentioned situations. Agile Coaches who are working with "hard-wired" leaders, this blog can enable you to navigate through these challenges from leadership buy-in to their mindset transformation.

The Fundamental Question: As a leader how to impact your people mindset?

The core area of any Agile transformation is to assist in changing the mental models and short-cuts to enable people unlearn many of these models and re-learn based on agile values and principles. For leaders of those organizations, it's about how to lead in a new way. It triggers with a 'wake-up' call, while also recognizing that we have saturated our limits when we employ the same old strategies to solve new and more complex business problems of today's world.

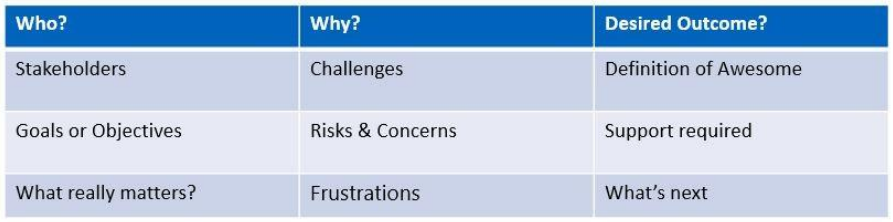
While approaching executives and leadership, we need to understand *"Where they're coming from", "What they're motivated by", "What's important to them, "How do they see the problems in the Organization"*. Direct selling a solution may be counterproductive as it may lack agreement and consensus. Hence, we don't recommend directive coaching.

Here, we are presenting "Coaching Engine" reference model that helped us in attaining buy-in from leadership for their mindset shift.


Coaching Engine
Step3: Educate
• It involves reflecting and reinforcing agile values with enabling self-awareness about leadership style.
Step4: Observe
• Measuring & cultivating the change to make it sustainable.
Step2: Introspect
• Understanding the coachee needs and set a ground for unbiased conversation towards realizing their current state.
Step1: Understand
• It involves intensive one to one discussion with thought provoking questions.

Step 1: Understand

Intensive One to One Discussion with thought provoking questions to get actual insights of the real issue. As said by Albert Einstein *"I would spend 55 minutes defining the problem and then 5 minutes solving it"*? So before working on any problem statement we must make sure we are working on actual root and not the symptom of the problem. Use **Interview Canvas** (Credit: Peter Stevens) where below questions can help in understanding the other person perspective.



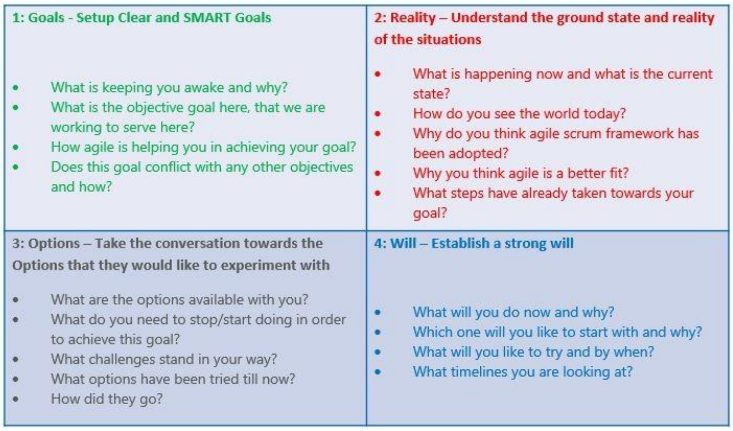
Who?	Why?	Desired Outcome?
Stakeholders	Challenges	Definition of Awesome
Goals or Objectives	Risks & Concerns	Support required
What really matters?	Frustrations	What's next

Main idea here is to make leadership visualize how the new world could look like, generating excitement.

Step 2: Introspect

Understand their needs and set a ground for unbiased conversation towards realizing their current state with open ended questions

GROW Model can enable to assess the current goals, challenges, understanding the ground reality and discovering what options can be taken to achieve the stated goal with the necessary will and motivation needed to achieve it. It works on the foundation of powerful questions. Here is an illustration.



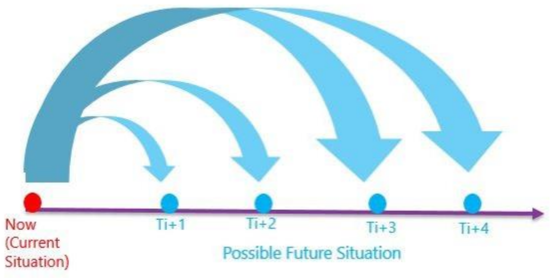
1: Goals - Setup Clear and SMART Goals	2: Reality – Understand the ground state and reality of the situations
• What is keeping you awake and why? • What is the objective goal here, that we are working to serve here? • How agile is helping you in achieving your goal? • Does this goal conflict with any other objectives and how?	• What is happening now and what is the current state? • How do you see the world today? • Why do you think agile scrum framework has been adopted? • Why you think agile is a better fit? • What steps have already taken towards your goal?
3: Options – Take the conversation towards the Options that they would like to experiment with	**4: Will – Establish a strong will**
• What are the options available with you? • What do you need to stop/start doing in order to achieve this goal? • What challenges stand in your way? • What options have been tried till now? • How did they go?	• What will you do now and why? • Which one will you like to start with and why? • What will you like to try and by when? • What timelines you are looking at?

Based on the responses, further enable leaders to prioritize their actions items and **commit to actions** proposed by them.

Did Grow model always work?

There can be scenarios where people are not responding to your powerful questions, not opening enough, there grow model may not provide the intended results. An alternative deep explorer, Future Pacing Method was quite useful in those cases.

Future Pacing enables leadership to build the visual steps required to achieve their goal I.e. visualizing a new future. This exercise will also give an opportunity to fit the different outcomes together and create a single view of what needs to be done to achieve the goal.



It is a three step process:

1) Acknowledge where they are at right now 2) Taking them to future
3) Committing to the future

By creating the visual representation of the future pacing, leaders can relate with their mental map, of what needs to be done, by when and what emotions will really define the success which can enable them in understanding the gaps that they need to work on. What resources that are available to them. Based on all these factors what steps can be taken to achieve the goal and committing to that actions.

Step 3: Educate

Make them realize "Leadership is about building Trust and Climate enroll, creating an eco-system of possibility and not command and control". It's like holding up a mirror up to a leader so that they see both a) What they are doing and b) The impact on others.

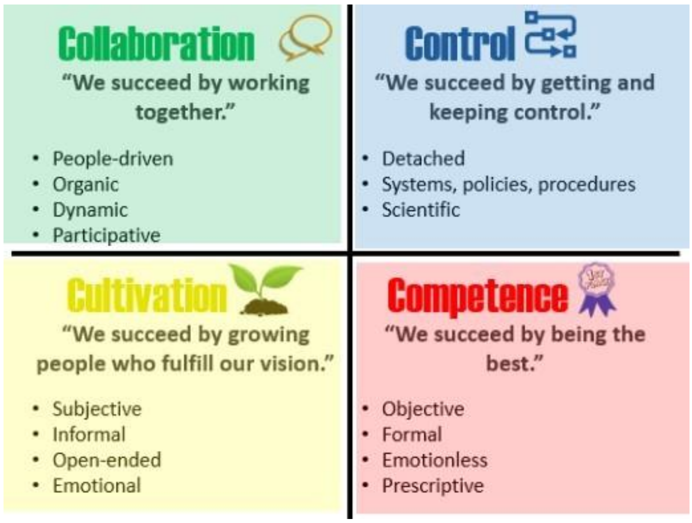
A) Reflect and Reinforce Agile Values

Pygmalion effect, which states that higher expectations lead to an increased level in performance and educate (help of videos and exercises) about co-active coaching principle that *"people are naturally creative, resourceful, we need to create an environment for them"*. Increase awareness about NLP and its models like "Dr. David model of engagement" which states how autonomy plays a critical role in team performance, impacting quality of delivery. Also teams cannot experiment (inspect and adapt) if failures are always being discouraged.

Create awareness about neuroscience of engagement which states teams who have goals clarity, what is expected from them, feels more motivated. Guide them about psychological safety (PS) quoting some of the world class organizations like google and other case studies of PS. According to Google Aristotle project, to maximize collaboration, team members must feel safe in sharing ideas and concerns. Without safety, team members cannot grow or improve. When it is absent from the workplace, teams lose the individual knowledge and expertise each member brings to the table, what is known as the **"Common Knowledge Effect"**. Reinforce Agile Principles like - *"Build projects around motivated individuals, give them the environment and support they need, and trust them to get the job done"* highlight the importance of leadership behavior in creating right environment.

B) Self-awareness about leadership style

Through surveys, enable leadership to identify their leadership style (Democratic, Autocratic, Transformational, Bureaucratic (Command and Control), Transactional). Here we tried "Schneider Culture Model" successfully to the leadership, through active observation and organization level surveys which depict their style of working in the four quadrants. Here questions are asked like *"What counts most in the organization", the primary way decisions are made in the organization, "attitude toward mistakes", "role of the individual employee", definition of success, etc.* Then sharing the results in form of four quadrants as shown below:



Credit: https://www.pinterest.com/pin/810507264171167448/

Based on organization principles, values and goals they want to achieve, enable leadership to spark new ideas and decide what shift they want through questions like *"What problems are there due to misalignment?"* Here existing culture is questioned followed by a period of reformulation where leadership will consider what new beliefs they need to develop to move out of control quadrant.

Finally, as part of brainstorming session with leadership, we generally present this problem statement *"How we can make a shift from command and control to collaboration and cultivate culture"* and let leadership come up with innovative ways to change habits, style of working/leadership and culture. After education and mentoring, challenge leadership towards action, we use 25/10 crowd-sourcing liberating structure to enable leadership to come up with creative ideas which can influence their leadership styles.

Step 4: Observe

Measure and Cultivate the change After giving your best, now it's time to observe, "*Is what has been committed is coming to life?* Is the new behavior becoming a reality?

Let's understand this by considering an example of new year resolutions where we create a plan for reinventing our self like going to gym, spending more time on your long-term goals, etc. Fast-forward a month and mostly we are back to our old habits. Right?

Reason being, sustaining change is extremely hard.

First part for sustenance is **validation/measurement** since it's hard to progress unless you establish a feedback loop and milestone checkpoint about the change process. Measurement is crucial for sustaining the change. KPI's of any form be it team happiness index, productivity, it needs to be defined and measured.

Second part to it is **continuous support and enablement** to make that change their part of life. As coaches we believe in being an active sounding boat and mentoring through real-time feedback. **Hence taking pause, reflecting and stepping back are the critical elements of sustenance.**

About the Co-Authors

Sourav Singla works with TCS Enterprise Agility practice, helping organizations to deliver better value sooner through the application of agile and DevOps principles and practices. He believes the path to successful agile adoption is through inculcating an environment which fosters the continuous improvement of the processes and practices. He collaborates with leaders and teams to create more responsive, effective and resilient teams.
As Peter Drucker famously said "Culture eats Strategy for Breakfast", it is a cultural transformation that he focuses on through training, coaching and consulting assignments. He shares his agile & devops tools experience regularly on TrustRadius. As a coach he partner with people to help them achieve their intrinsic business goals. A continuous learner at heart who loves to reinvent himself, currently living in Mumbai, India

www.linkedin.com/in/sourav-singla-certified-team-coach-ctc-35b65924

Lalita Chandel is a passionate Agilist and Transformation Leader. She believes in **people centric development and purpose driven change** that sticks to an organization and evolves organically. Lalita has worked on various large scale and global distributed programs with a multicultural employee base. Her specializations are in the area of Organizational Change Management, Coaching, Operating Model and Agile Leadership development. She is a well recognized thought leader in the Agile Community and has presented 25+ talks in various Agile DevOps conferences in South Asia. Lalita has travelled across the globe and works extensively on bridging the cultural gap. Lalita is an active Trainer, Blogger, Coach, Speaker and a lifelong learner. She is based out of Mumbai, India where she lives with her family. She is working with TCS Enterprise Agility group.

www.linkedin.com/in/lalita-chandel-27184825
Twitter: @LalitaChnd

Agility: Build the right thing

From discrete Output to continuous Outcomes

By Jonathan Smart

A milestone

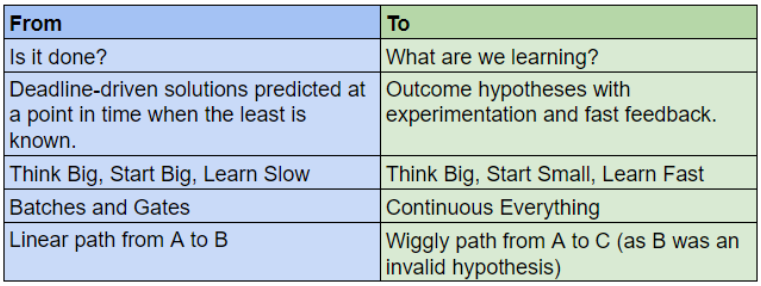
Milestones mark miles. They are made of stone. They are hard to move and are no longer valid once moved. They don’t exhibit agility, nor do they need to. They resemble tombstones and are often referred to as ‘dead-lines’, or ‘drop dead’ dates.

They don’t convey any expectations on **why** you want to get to where you’re going, how you might get there, **how else** you might get there, **if** you’ll make it there at all (the level of **safety** en-route), the quality and **experience** of your journey nor **if** it’s even the right destination, given **your intent.**

They are one dimensional. The distance between two fixed points along one fixed route. Originating 2,300 years ago in the Roman Empire, milestones were set more than 2 feet into the ground, stood 5 feet tall and weighed more than 2 tons. They measured 1,000 paces. They were immovable.

You don’t know **when** you’re going to reach a milestone. The last 20% of the journey might take 80% of the time. There might be roadworks, an accident or a flooded river crossing. You only know you’ve reached a milestone when you reach a milestone. And even then, given your **intent**, you might find that it’s not the optimal place to be.

Are milestones a good mental construct to use for unique change with a rapidly changing terrain? I think not.



From	To
Is it done?	What are we learning?
Deadline-driven solutions predicted at a point in time when the least is known.	Outcome hypotheses with experimentation and fast feedback.
Think Big, Start Big, Learn Slow	Think Big, Start Small, Learn Fast
Batches and Gates	Continuous Everything
Linear path from A to B	Wiggly path from A to C (as B was an invalid hypothesis)

This is the fourth post in a series sharing a number of observed **antipatterns** and corresponding **patterns** on the topic of organisational agility, for organisations who want to deliver Better Value Sooner Safer & Happier.

We are in a new Deployment Period in a 50 year cycle, in the Age of Digital, as well articulated by Carlota Perez in 'Technological Revolutions and Financial Capital'.

Today, for complex knowledge work, many large organisations are still applying a management model from the early 1900s optimised for manual labour in factories.

With new means of production and consumption, with on-demand compute, information and communication power which is in the cloud and in our pockets, if traditional organisations want to survive and thrive, want to keep up with competitors and challengers, there is a need to adopt new ways of working to maximise the outcomes from the new means of production.

Survival is not mandatory

This is of course optional, as survival is not mandatory, a path that UK department store Debenhams has taken. Debenhams, founded in 1778, went into administration 241 years later, on 9th April 2019 with 165 stores and 25,000 employees. To quote Richard Lim, chief executive of industry analysts Retail Economics, *"Put simply, the business has been outmanoeuvred by more nimble competitors, failed to embrace change and was left with a tiring proposition. The industry is evolving fast and it paid the ultimate price." (source).*

Time will tell if Debenhams is following in the footsteps of Toys R Us, Maplins, BHS, House of Fraser, HMV, Kodak, GM, Blockbuster, Nokia and Sears as firms who have chosen, implicitly, to not be a going concern either at all or in what was their core market.

Takeaways

This post focuses specifically on how organisations choose (implicitly or explicitly) what work to do and how it is approached, in the context of product development, as organisational agility increases or needs to increase.

A scenario often observed is that processes *upstream* of product development teams become the primary constraint to valuable flow, especially as product development teams increase their agility.

How do large, traditional, organisations make sure that teams are working on the best possible things as agility increases? Done badly, and anything can be done badly, it's fragile rather than *agile*. As empowerment and autonomy are increased, the system of work could become chaotic and disconnected. Or, capital 'A' Agile is imposed with a culture of fear, with all change still having drop-dead predictive output milestones, waterfall dressed up as agile, which doesn't maximise the benefits of agility.

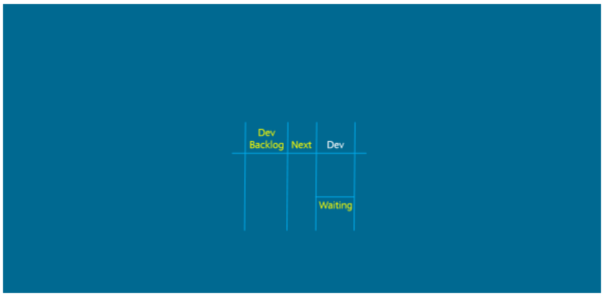
First we take a look at four antipatterns: *(1) Local Optimisation & the Urgency Paradox (2) Milestone Driven Predicted Solutions (3) Headless Chickens and (4) Start Starting.* Then we look at the corresponding pattern per antipattern, namely: *(1) End to End Flow (2) Outcome Hypotheses (3) Strategy Alignment and (4) Stop Starting, Start Finishing.*

These antipatterns and patterns are based on lessons learnt through doing, through learning, through running experiments (the only failed experiment is one which generates no learning. Even then that itself is a learning). This includes being a servant leader on better ways of working, applying agile, lean, DevOps, systems thinking and so on, across a large (80,000 people), old (300+ years old), global, not-born-agile, highly regulated enterprise. I have been delivering change with an agile mindset, principles and practices since the early 1990s, about a decade prior to the Agile Manifesto, when the term 'lightweight processes' was used. The antipatterns and patterns are also based on learnings from many horses (rather than unicorns) on similar journeys.

Antipattern 1: Local Optimisation & the Urgency Paradox

The following scenario is a common one I've experienced, especially with business areas early in the journey to delivering Better Value Sooner Safer and Happier.

I had met with a software development team who were rightly proud of their progress with increasing their agility. The **Team Outcome Lead** (aka Scrum Master) showed me their improvement of Cycle Time, from the time work is pulled off the backlog to development done. It was an impressive improvement with the 85th percentile Cycle Time approximately halved from what it used to be 12 months previously. During this time the team had become multidisciplinary, limiting Work In Progress with more smaller stories, which had reduced complexity per change, de-risked delivery and led to faster feedback and learning.



However, something wasn't quite right. The recipients of the change were complaining that they weren't seeing the claimed benefits of agile. Things were taking just as long as before.

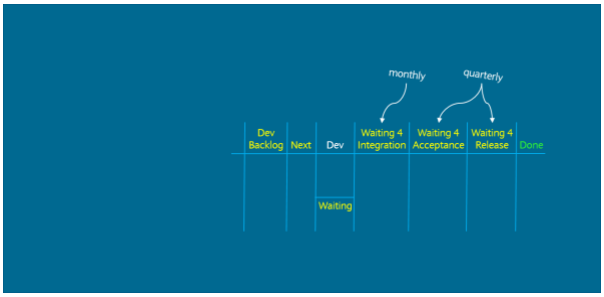
I asked what happened after development and testing was done to get the product into the hands of users. *"Well, we need to wait for integration testing because we've got lots of dependencies, we haven't decoupled the architecture, and we've got three or four or five systems that all need to be deployed together. So we have to wait for integration".*

Okay, so when you've done integration testing, are you then good? Are you then ready for production? *"Well, no, because we then have to wait for user acceptance testing".*

Okay, fine, so you do your acceptance testing, and then anything else before you can then put that into production? *"Actually yes, because we batch up releases. So we have to wait for the release cadence, we have to wait for that to happen and for IT Ops to be happy to do the release."*

Okay, is there anything else after that in order to reduce the time to market? *"No, after that, then we're done".*

And so then I'll ask, okay, so what's the cadence on this, what's the frequency?. *"Well, integration testing is monthly, acceptance testing is quarterly, and releasing is quarterly".*



I ask, to the left, prior to the work getting to the dev team what happens?

"Well, prior to the work getting to the dev team there is a Product Backlog with epics which need to be refined, analysed and broken down into stories. In some other teams that work is predictively planned into multiple iterations over a quarter, with a focus on commitment of output, rather than experimenting to best achieve outcomes, which feels kind of traditional".

"Prior to that, a detailed design needs to be reviewed by the organisation-wide Technical Design Authority and before that there is a Detailed Business Case which needs to be completed for each programme or project, funding agreed and approved by a business line steering committee".

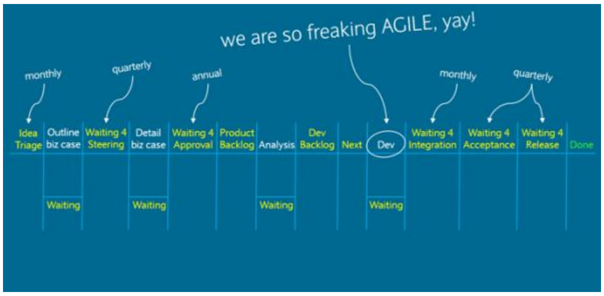
"Before all of **that** *there is an Outline Business Case approval step to filter the business cases going into Detailed Business Case review and ahead of that there is Idea Triage".*

How often do these happen, I ask.

"That's a good question. It's monthly for the Idea Triage, quarterly for the Outline Business Case and an annual planning process which typically takes 6 months to complete, using the Detailed Business Cases, which determines what programmes and projects we work on for the following calendar year. It's always been this way, we start planning in September for the following year, with endless cycles of big up front planning and around March we know our level of funding, on the premise that it delivers the big up front plans, which by the end of the year are 18 months old. Usually there's gaming of the system, going in high, knowing that it will be knocked back, usually ending up the same or just lower than last year".

However, when you speak to the dev teams they say:

‘WE’RE SO FREAKING AGILE, YAY!”



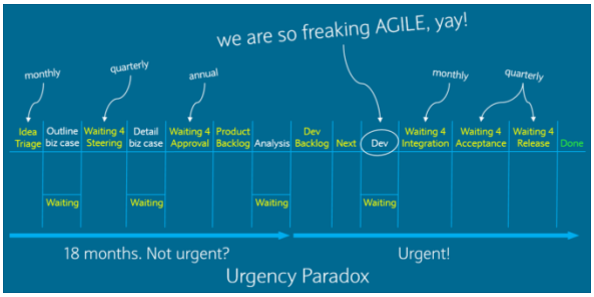
“We’re so freaking Agile, yay!” (credit: Klaus Leopold)

Software development teams can improve all they like, however the time from customer need identified to need met, the time to pivot based on a market event or learning, hasn’t materially reduced. Often I’ve heard recipients of change comment, quite rightly, that they have not seen an improvement in time to market.

“A system of local optimums is not an optimum system”, Eli Goldratt, The Goal

Delivery risk is lower with faster learning within development, however it might be the wrong thing being built. Time to learning, time to value, time to delighting customers, flow, has not materially changed end to end. This was a lesson learnt in manufacturing a long time ago. Don’t improve the performance of a non-bottleneck as it will stack up more inventory at the bottleneck and not improve end to end flow of value. There is, however, a human benefit in the local optimisation, where it is likely making working lives more humane and rewarding within the local optimisation, even if the overall human endeavour is not generating broader benefits.

What was also witnessed was an **Urgency Paradox** (Don Reinertsen & Preston Smith, “Developing Products in Half the Time”). Valuable ideas sit in 12m to 18m of big up front planning with no sense of urgency, no questioning if they might add more value than what has already been locked in the plans for the year. As soon as they reach the product development team they are urgent. However, often by then the Cost of Delay profile has plateaued.



The Urgency Paradox

As per Douglas Hubbard, author of "How to Measure Anything", the most valuable data for investment decisions is (1) whether the product will be used at all and (2) how quickly it will be used. These two factors trump all other data, including expected cost (source).

This adds further weight to the Urgency Paradox of not waiting one rotation of Earth around the Sun before deciding what to invest in for the next orbit. A lack of flow end to end and an inability to pivot reduces the probability that teams are working on the most valuable thing, both due to a lack of feedback and an inability to respond to a changing environment.

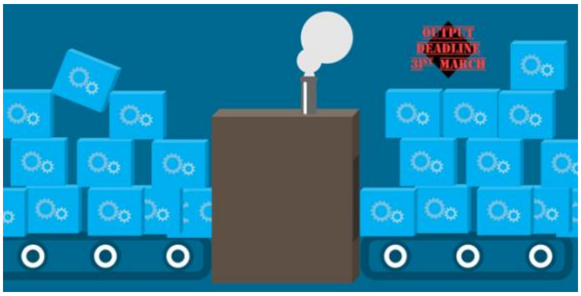
Why wait one rotation of the Earth around the Sun before deciding what to invest in next?

Antipattern 2: Milestone Driven Predicted Solutions

Traditionally, milestones in a Gantt chart focus human endeavour on activity completion, on output, predicted at the time of knowing the least, instead of continually thinking about how to maximise outcomes based on strategic intent and ever increasing knowledge of the landscape, in the context of unique product development.

"Have we achieved the activity", rather than "What shall we do next, given what we've learnt so far, in order to try to best achieve the desired outcome". The former is order-giver and order-taker with thinking put on hold. The latter requires everyone to use their brains.

"Don't mistake activity for achievement", John Wooden



Milestones are an unsuitable metaphor to use for unique product development (which is emergent), within organisations (which are emergent), to continually delight customers (who are emergent), in a landscape (you guessed it, which is emergent), where they are all changing faster than ever. The metaphor, in this context, traditionally has been overloaded to predict distance (work done), the quality of the destination and time.

Add in managers vs. workers, business vs. IT, low psychological safety with fear, blame and reprisals, individual incentivisation over team incentivisation, tribal identity by task based job role and not surprisingly, success is low at 29% (Standish Group, 2015, source) and employee engagement, while rising, is low with 34% of people actively engaged, 53% not engaged and 13% are actively disengaged (Gallup, 2018, source). The Standish Group survey also reports that an agile approach is four times more successful than a waterfall approach.

This is not to say that fixed dates don't exist, they do. More on this later. Specifically, the word and metaphor of "milestone" is sub-optimal, perpetuating old behaviours and outdated ways of thinking for unique change.

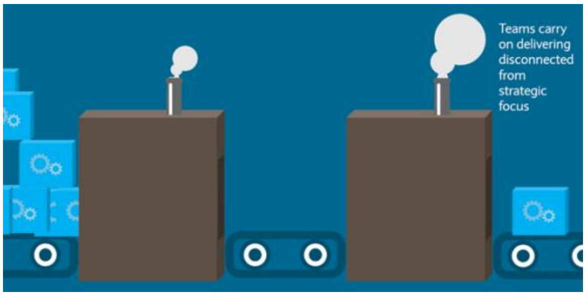
I've often heard milestones being referred to as 'drop dead dates'. With a long lead time, big bang, traditional approach, the activities to reach the drop dead date have at times been called a 'death march'. Indeed, the term **deadline** has its origins from prisoner of war camps in the American Civil War in 1864, where the 'dead line' was literally a railing or a ditch which if crossed would result in death.

In one organisation I worked at, project managers set *deadlines*, fixed output milestones, in advance, without consulting those doing the work, at the point of knowing the least. There was big batch, feast to famine work passing by role, with finger pointing. The *deadlines* never moved. During the 'feast' phase of work per role, people were working unsustainably. I asked some of the people why they were still working there. The answer I got, said seriously: *"We are united through a common suffering"*. A form of organisational Stockholm Syndrome. An undercurrent of do this thing by this date or die. How's that for colleague engagement and satisfaction at work?

In the Age of Digital, with a new means of production, with a rapidly changing terrain, there are better ways of working, if an organisation wants better outcomes. This is of course optional. Surviving and thriving is not mandatory.

Antipattern 3: Headless Chickens

In this scenario, product development teams have adopted agile ways of working and have become a **Feature Factory** churning out stuff, disconnected to company strategy and typically incentivised by activity or output (e.g. velocity, Say Do ratio, commitment), rather than by Outcomes. Teams become a self-fulfilling prophecy of backlog replenishment.



Teams are disconnected from strategic intent, yet continue producing stuff

This could be because there is a lack of flow coming from upstream, as the PMO, portfolio management, investment case approval and technical design authority processes are big batch and infrequent.

It could be due to a lack of clear strategic alignment, a lack of clear North Star, no link between the work being done and strategic goals. There is low alignment and high autonomy, resulting in brownian motion.

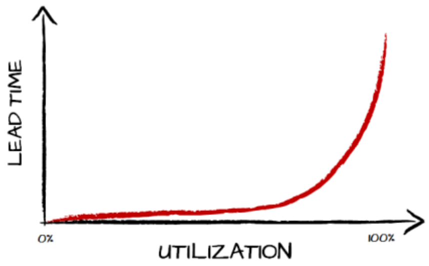
It could also be due to disconnect from the customer, a lack of seeking or responding to feedback. A behaviour I've observed is Product Owners (especially new POs who used to be project managers) being so focussed on 'I **OWN** this Product' that they forget to speak to the customer and understand what might be most valuable for them.

As an aside, to help address this point, we've been experimenting with renaming the Product Owner role to **Stakeholder Outcome Lead**, to make the self identity more clear. The Stakeholder Outcome Lead is a servant leader focussed on the **What**, an outward focus on the customer, economic, societal and environmental outcomes (**Value**. See Triple Bottom Line and this significant statement from the Business Roundtable signed August 2019 by 181 US CEOs, moving away from short-term shareholder primacy to all stakeholders, including society and the environment). This complements the **Team Outcome Lead** (aka Scrum Master in Scrum) who is a servant leader with an inward focus on the **How** and building the muscle of continuous improvement as a daily habit practiced by everyone **(Better Sooner Safer Happier).**

Another possible reason for this anti pattern of teams churning out stuff agnostic of strategy, feedback or value, is a traditional mindset around resource utilisation with a view that people need to always be busy, leading to overproduction in siloes, rather than coming together, swarming and helping to alleviate the constraints of a lack of flow and strategy alignment, coming from upstream.

"An [operation] in which everyone is working all the time is very inefficient", Eli Goldratt

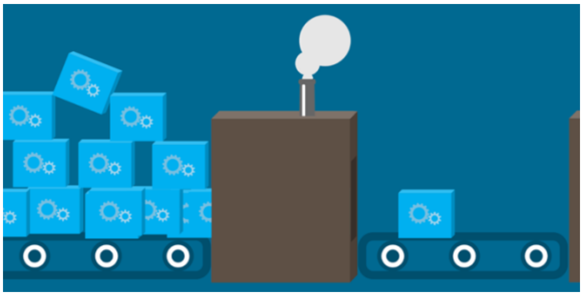
If there is a traditional mindset around resource utilisation, with a desire for people to always be busy, this will have the opposite effect to that desired, in that Lead Time rises *exponentially* as utilisation increases in a system of work with variability, as is the case with product development. There is no time for continuous improvement, for resolving impediments, for reflecting, for pivoting, for swarming and the system of work goes slower. People are fully occupied pushing the square wheels. For a great online simulation of the effect of utilisation on lead time, see Troy Magennis's interactive article here.



Antipattern 4: Start starting

In this antipattern, organisations keep on *'start starting'* instead of *'stop starting, start finishing'*. This is like adding more cars to an already busy motorway, the cars go slower.

Blissfully unaware of the capacity of the system of work and treating it as a push based rather than pull based system, organisations keep on starting more initiatives, more projects, saying yes to customers or stakeholders demands.


Start starting

In one specific case, an area was saying yes to twice as many requests from an internal customer, than the natural capacity of the system of work, such that each month the backlog of committed work grew by another month. Applying traditional thinking, the focus was on SLAs at each sequential stage and people incentivised to work harder to meet the SLA, a series of local optimisations, pushing work and actually making the end to end lead time longer by having unlimited work in progress, building up queues at each stage.

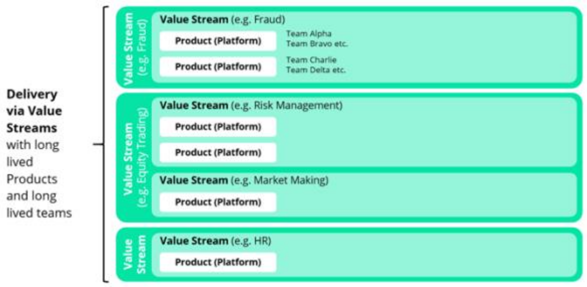
The result in this context was an acceptance by the recipients of change that things take a long time to get done, driving a desire to start more things otherwise they'll never get done ('the best time to plant a tree is 20 years ago, the second best time is now'), an us and them behavioural norm and growing piles of invisible inventory. This results in lower quality as people try to work harder rather than smarter to meet local optimisation SLAs, and lower engagement. It's a no win situation with a lack of awareness of the system of work.

The most important measures for teams, and teams of teams, to know are their Flow measures: Lead Time, Work In Progress, Throughput and Flow Efficiency. This determines the health of the system of work. For more on this see Know Your Flow.

Pattern 1: End to end Flow

Where the antipattern is local optimisation, the pattern is to focus on end to end flow, looking at the whole system of work.

First, identify long lived value streams. A value stream is producing something of value, with one or more internal and or external customers, going from needs identified to needs met.


Delivery via Value Streams with long lived Products and long lived teams
Value Stream (e.g. Fraud)
Value Stream (e.g. Fraud)
Product (Platform)
Team Alpha
Team Bravo etc.
Product (Platform)
Team Charlie
Team Delta etc.
Value Stream (e.g. Equity Trading)
Value Stream (e.g. Risk Management)
Product (Platform)
Product (Platform)
Value Stream (e.g. Market Making)
Product (Platform)
Value Stream
Value Stream (e.g. HR)
Product (Platform)

It is likely that the organisational structure of the business is already set up as value streams, for example, mortgages, savings, investment advice, luxury bags, customer onboarding, helicopter engines, recruitment, finance, internal audit and so on. Key is that a value stream should be end to end and ideally with as few dependencies as possible. That is, the value stream should have high cohesion and low coupling. Importantly, value streams are ***nested***. For example, Bank ➜ Investment Bank ➜ Capital Markets ➜ Trading ➜ Equities ➜ Market Making.

Value Streams are represented horizontally, with a logical flow of value from left to right, from need identified to need met, from hypothesis to validated learning. Personas (e.g. young person, family, retired couple) are not Value Streams. They consume the Value Streams. Rebadging existing role based teams, fiefdoms and calling them Value Streams, are likely also not Value Streams.

Value Stream Mapping is a technique which can be used to shine a light on the steps involved in the end to end value creation flow, including wait time vs. value adding time. This provides a Flow Efficiency measure. For most large organisations, Flow Efficiency for knowledge work is very low at around 10%, which means that work is waiting 90% of the time.

The value streams should be as self contained as possible, with dependencies eliminated or mitigated over time, part of a network of interdependent services. This won't be the case at the beginning, there will likely be many dependencies. Spend time breaking dependencies rather than just managing them. Ways to do this for software products include shared code ownership, internal open source, service virtualisation and breaking monolithic balls of mud into microservices.

In my experience typically 80% of value streams are customer facing and 20% are internal shared services, for example HR and Finance.

Long Lived Products

Having identified the value streams that enable the organisation to economically, socially and environmentally serve its purpose, next, map long lived products to the long lived value streams.

These products are often IT systems. For example an Equities Trading System which enables the Equities Market Making value stream, or an HR system which supports the onboarding of new employees, or a smartphone app which enables management of personal finances or to shop online or to publish a photo of your lunch. The mapping of IT products to value stream shines a light on duplication of systems and eases simplification and decommissioning efforts.

It also shines a light on big monolithic IT systems which straddle multiple value streams and inhibit agility due to in-built hard dependencies. In some cases, the mapping of products to value streams leads to an organisation actually having a product inventory for the first time.

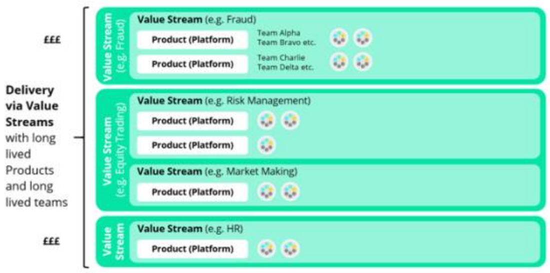
Long Lived Teams

The long lived value streams should have long lived small (<10 people) multi-disciplinary teams on them. This results in teams going through forming, storming, norming, performing (Tuckman) and staying together, unlike temporal project teams. The teams get to understand the unarticulated needs of the customers. The multidisciplinary nature of the teams means that most of the time they have all the skills necessary to deliver end to end value. Team members are 'T-shaped" generalising specialists. This minimises dependencies on individuals and other teams and hence enables flow, fast feedback and learning. It should not require multiple teams to get value to production, early and often.

The alignment of long lived teams to long lived value streams, creates a tribal identity around the value stream, away from a tribal identity of 'I'm business' or 'I'm IT' or by job role. Instead 'we are mortgages' or 'we are luxury bags' or 'we are helicopter engines'. The nature of the organisation of human endeavour has people working *together* daily to iterate towards common business outcome hypotheses.

Funding

Funding is aligned to the value streams. Each value stream, with long lived teams, is capacity funded. *Funding is a constraint on which to maximise the value curve*. For knowledge work, the main cost for value streams are people, who are aligned to value streams. Funding is value stream aligned, with a rolling roadmap of value outcome hypotheses.


£££
Delivery via Value Streams with long lived Products and long lived teams
£££
Value Stream (e.g. Fraud)
Value Stream (e.g. Fraud)
Product (Platform)
Team Alpha
Team Bravo etc.
Product (Platform)
Team Charlie
Team Delta etc.
Value Stream (e.g. Equity Trading)
Value Stream (e.g. Risk Management)
Product (Platform)
Product (Platform)
Value Stream (e.g. Market Making)
Product (Platform)
Value Stream
Value Stream (e.g. HR)
Product (Platform)

Long lived Value Streams with long lived multidisciplinary teams . Value Streams are funded.

Pattern 2: Outcomes Hypotheses

"It's better to do the right thing wrong than the wrong thing right. The righter you do the wrong thing the wronger you become", Russell Ackoff (source)

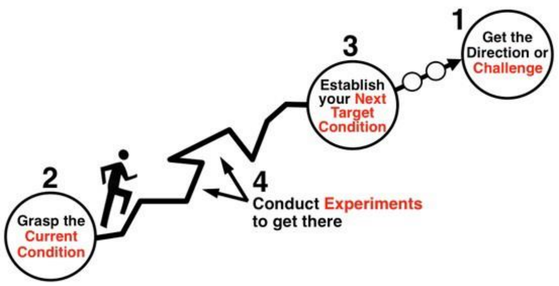
Where the antipattern is big up front milestone driven predicted solutions, the pattern is to focus on outcome hypotheses with experimentation. From deterministic output to outcome hypotheses.

Emergence

We can't travel in time, product development is emergent, it is unknowable, what people want, how it's going to be built, what obstacles we'll come across on the way and how external market forces will change over time. Acting in the space, changes the space. The North Star should not be a fixed *solution* or *output* by a date. It is not a knowable straight path from A (current condition) to B (desired outcome). The bigger the distance from A to B is, the less knowable the journey is.

It is beyond our knowledge threshold. A traditional approach is placing a fixed bet at the beginning of the race when the least is known.
Instead it is a wiggly journey from A to B, optimised with fast, real, feedback and learning, with safe to fail experiments, in order to test the Outcome Hypothesis. For unique work, we want to maximise variability and run multiple experiments, maximising learning, early and often.

This approach enables changing bets on the race while the race is running.

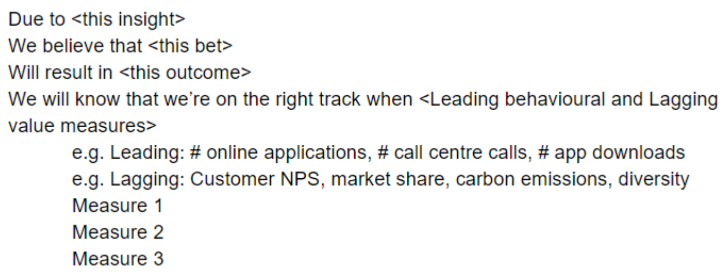

1
Get the Direction or Challenge
3
Establish your Next Target Condition
2
Grasp the Current Condition
4
Conduct Experiments to get there

Toyota Improvement Kata, Mike Rother

Outcomes

The outcomes are ***nested***, at multiple cadences. Daily, weekly, monthly, quarterly, yearly, multi-year. The key focus is the quarterly Business Outcome (also known as an OKR). A three month time period is not too close and not too far away. This is broken down into monthly Features, which are experiments to test the hypothesis. These are in turn broken down into Stories which are daily and optionally into hourly Tasks. Therefore there is learning and innovating many times a day, on the desired outcome hypothesis.

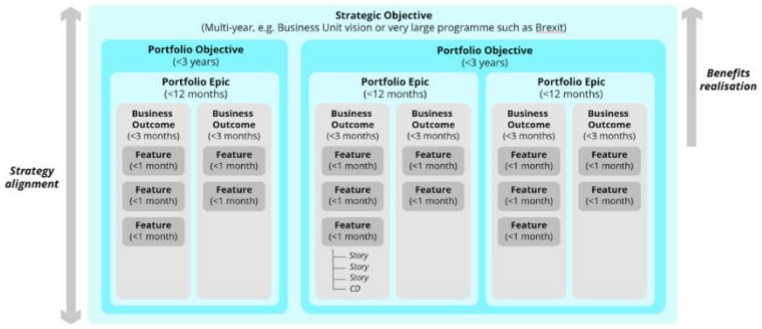
The Business Outcome hypothesis is written along the lines of the following:



Due to <this insight>
We believe that <this bet>
Will result in <this outcome>
We will know that we're on the right track when <Leading behavioural and Lagging value measures>

- e.g. Leading: # online applications, # call centre calls, # app downloads
- e.g. Lagging: Customer NPS, market share, carbon emissions, diversity
- Measure 1
- Measure 2
- Measure 3

Each quarterly Business Outcomes has a Portfolio Epic parent, a higher level North Star outcome hypothesis. Each business area has a handful of Portfolio Epics, limiting work in progress. This is strategy alignment. The <12 month Portfolio Epics are nested within multi-year Portfolio Objectives, which are business unit level strategic items. And for larger firms, they are contained within a handful of Strategic Objectives which is the top, group-level, strategic intent, spanning all business units.



The nested Outcomes are represented vertically, from multi-year strategic intent down to daily stories. Typically each level has its own Kanban board, with WIP being limited at all levels.

Value, which is unique and not aggregatable, is expressed in the Lagging Value Measures. It is important that these are measurable, that they are quantitative.

Example
A multi-year Portfolio Objective might be a hypothesis to be in the top 3 of market share for credit card providers, in order to reverse a decline in market share, to increase profitability, to keep people in gainful employment and to help more people with responsible financial liquidity.

An in-year Portfolio Epic might be a hypothesis to launch a partner card with an airline, in order to increase market share. A quarterly Business Outcome might be a hypothesis that issuing one partner credit card, to at least one customer and using it for at least one transaction end-to-end, in a live environment, will de-risk delivery, enable fast learning in a safe to fail environment, and will result in delighted customers, hence helping to increase market share. The approach is outcome-oriented, in business language (not IT activities), with real feedback and learning and the ability to pivot.

Rather than eat the elephant in one go, aim for *elephant carpaccio*, try to get the thinnest vertical slice of real value, to maximise learning and making it safe to fail, which is necessary in order to learn.

Elephant carpaccio, over eating the elephant whole

The key shift in paradigm here, is moving to being able to pivot to maximise the desired outcome bet, changing the bet mid race, either because the hypothesis is not valid or to be able to *'cut the tail'* when it appears that the high value, low effort features have been implemented.

Rolling roadmap of outcome hypotheses

Also, importantly, it is a rolling roadmap of outcome hypotheses. More fine-grained up close and more coarse-grained further out, with a logarithmic scale. For example, the current Quarter, Quarter+1, 6 months. A great example of this is from the GOV.UK experiments in roadmapping here, where they intentionally picked a curved glass partition, so that the future is out of view, they literally can't see what's around the corner.

With multiple nested cadences, daily, weekly, monthly, quarterly there are actionable learning loops. This way, multi year strategic intent is getting daily feedback. And, those who are closest to the customer, to the detail, closest to understanding unarticulated needs, are able to see clearly the strategic intent and are best placed to innovate.

Dealing with fixed scope and fixed dates

Don't artificially lock in a fixed date, if it doesn't need to be fixed. Don't back yourself into a corner unnecessarily. I've seen many cases of leaders insisting on fixed output with fixed dates unnecessarily, which limits the ability to respond to learning, in order to best achieve the desired outcomes. Outcomes, which are quarterly, defacto have a date. The ninja move here is to define the *Outcome* not the *Output*. This is synonymous with the move from detailed command to mission command in the military.

There are cases where there are fixed dates with fixed scope, such as mandatory regulatory change. Examples of this include GDPR and the Dodd-Frank legislation after the 2008 credit crisis. Having implemented many regulatory initiatives pre and post 2008, all fixed date and fixed scope, all of them agile, all completed early, even if you think that the scope is fixed, in reality legislation is usually written in a way which enables near infinite ways to implement it.

Also, even if you think the rule writing is fixed, it may not be. For example, around 2012, we implemented the most risky, least understood bit of the UK version of the Dodd-Frank legislation within the first month, tested it with production data, got insights back, took those insights to the Bank of England, who updated the legislation as it would have put the UK at a competitive disadvantage to other financial centres globally.

A waterfall approach with value and learning delayed to the end, is far too risky an approach for fixed date, fixed scope work. Everything should be done to remove ignorance, to maximise early and often learning, with the most time to act on it.

From PMO to Value Enablement Team

The role of the traditional Project Management Office (PMO) changes in the pivot from temporal projects to long lived products with long lived teams and a rolling cadence of outcome hypotheses. The pivot is to become more of a servant leader team with a focus on the Value part of *Better* **Value** *Sooner Safer Happier*, in particular the Lagging Value Measures which are on each quarterly Business Outcome. To make the change in identity clear, it can help to rename the team. For example *Value Enablement Team (VET) or Value Realisation Office*. The VET coaches teams in articulating and measuring Business Outcomes, gathers the Leading and Lagging Measure data together and provides consolidated data for a monthly cadence with senior leaders to inspect & adapt on the quarterly and annual outcome hypotheses. More on this in this talk here.

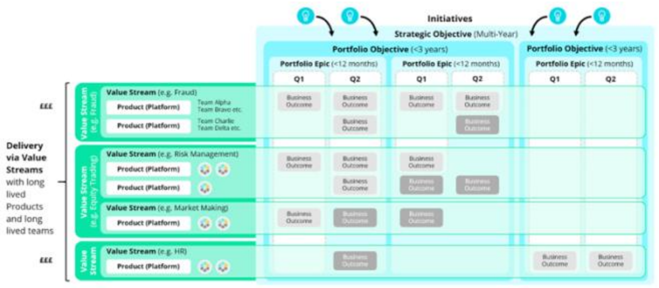
Pattern 3: Strategy Alignment

Where the antipattern is headless chickens, teams producing output disconnected from the strategy, the pattern is strategy alignment, by bringing together the horizontal nested value streams with the vertical nested outcome hierarchy.

Each long lived team, on a long lived value stream, with long lived products have clear Business Outcomes hypotheses to work on, with clear lineage up to the top handful of Strategic Objectives.

This enables High Alignment (clear nested North Star outcomes) with High Autonomy (the multi-disciplinary teams empowered to use their own brains and mastery to achieve the nested North Stars).

There is strategy alignment and the Business Outcome value metrics articulate value realisation.


Initiatives
Strategic Objective (Multi-Year)
Portfolio Objective (<3 years)
Portfolio Epic (<12 months)
Q1
Q2
Delivery via Value Streams with long lived Products and long lived teams
£££
Value Stream (e.g. Fraud)
Product (Platform)
Team Alpha
Team Bravo etc.
Team Charlie
Team Delta etc.
Value Stream (e.g. Equity Trading)
Value Stream (e.g. Risk Management)
Value Stream (e.g. Market Making)
Value Stream (e.g. HR)
Business Outcome

With a tribal identity of the value that is being produced, with multi-disciplinary teams (business, IT, finance, compliance, etc.), there is a shared purpose, clear outcome bets, written in business language, to experiment on. In my experience, this really starts to remove the behavioural boundary that traditionally-organised companies have between ‘the business’ and ‘IT’ due to the previous order-giver, order-taker ways of working. Instead people work together, as a team, to achieve and test desired outcomes. The ‘tribe’ becomes the value stream rather than by job role.

No longer are there headless chickens, where agile teams are disconnected from strategic intent, being a self-fulfilling feature factories. Instead, there is clear strategic alignment, with two-way strategy, down and back up. Strategy with rapid testing and learning. This two-way strategy in Japan is known as ‘Hoshin Kanri’. Traditionally Western firms have practiced one way strategy, top down.

Pattern 4: Stop Starting, Start Finishing

Having organised the horizontal long lived nested value streams with long lived teams and the vertical outcomes, running at multiple nested cadences, the next thing to do, in order to reduce Lead Time, the time to learning, the time to market, and to increase flow efficiency is to **Stop Starting and Start Finishing**, instead of Start Starting.

Doing less work concurrently and ensuring that people are less than 80% allocated (see antipattern 2), significantly reduces the lead time, the time to value, learning, de-risking, pivoting, avoiding sunk cost and allows time to improve the system of work.

Go slower to go faster. Or you will end up going slower.

This is analogous to cars on a motorway. The more cars, the slower they go.

Reducing Work In Progress (WIP) is a forcing function, an enabling constraint. As less work is done in parallel with shorter lead times, it shines a light on the impediments to flow. If an item of value is blocked and cannot progress, the team do not park it, instead they swarm on it, to clear the blocker. This means that the system of work is permanently improved. Like the tide going down, you see the rocks and shopping trolleys, the impediments to flow efficiency, that were always there, just not visible previously.

Impediments are not in the path. Impediments ARE the path.

Work in Progress should be limited at all levels in the Outcome Hierarchy.

Visualise

Information should be radiated. Ideally, there should be one or more Enterprise Visibility Rooms (EVR). Here the system of work is made visible. Very quickly, too much WIP and blocked items become visible. The previously invisible system of work for knowledge work becomes visible. Often, it’s the first time that people have actually seen the full end to end value stream. There may be multiple kanban boards, representing the multiple cadences (multi year, year, quarter, month, daily) along with Cumulative Flow Diagrams which is an incredibly insightful visualisation.

Pull based over Push based

In line with ‘Stop Starting, Start Finishing’, an Outcome, Feature or Story is not pulled until another one has been done. This results in the system of work becoming a pull based system, rather than a push based system. Every system of work has a natural capacity. By being a pull based system of work, the work flows at the capacity of the system, the natural capacity of the system of work can be identified and then improved.

Focus on the work not the workers

Pattern Summary: Nested Value Streams with Nested Outcomes

From by Role to by Value

From temporal Projects to rolling Outcome Hypotheses

Focus on the whole system of work, the whole value stream network.

The whole organisation is in scope, starting small. There are small, multi-disciplinary, long lived teams, aligned to long lived, nested, value streams, with autonomy, psychological safety and support in experimenting to achieve the nested outcome hypotheses. The tribal identity is to the value stream (e.g. Mortgages, equity trading, cloud provisioning, payments, luxury bags, taxi dispatch, customer sign up, etc.) bringing together many roles with a common purpose and working as a team, not an alignment by job role with work passing.

Outcome oriented teams focus on maximising outcomes

Project oriented teams focus on predetermined output

Focus on Outcome hypotheses, nested North Stars.
Multi-year, yearly and quarterly outcome hypotheses. There is a clear strategic intent with autonomy in how to best achieve it. The focus of human endeavour becomes experimenting and learning to achieve the desired outcomes, working as multidisciplinary teams, with a capability to pivot based on fast feedback.

Outcome hypotheses over predictive solutions

Summary

1. End to end Flow over Local Optimisation : nested horizontal Value Streams

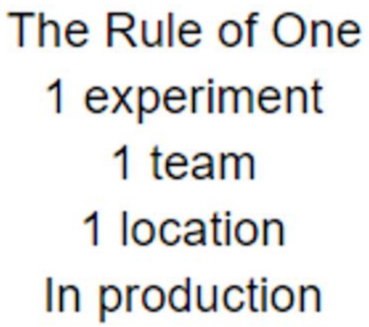
2. Outcome Hypotheses over Predictive Milestones : nested vertical Outcomes

3. Strategy Alignment over Headless Chickens : the intersection of both

4. Stop Starting, Start Finishing over Start Starting : limit WIP at all levels

This requires experimentation. Experimentation requires psychological safety.

How to start?



The Rule of One

1 experiment

1 team

1 location

In production

Start small. Learn Fast. ‘S’ curve adoption, by invitation.

About Jonathan Smart

Jon leads Deloitte's UK Business Agility practice, helping organisations deliver **Better Value Sooner Safer Happier**.

Previously, Jon led Ways of Working globally across Barclays Bank, which is 328 years old, with 80,000 colleagues in 40 countries. After three years, teams were on average delivering three times as much in a third of the time with 23x fewer production incidents and the highest ever employee engagement scores.

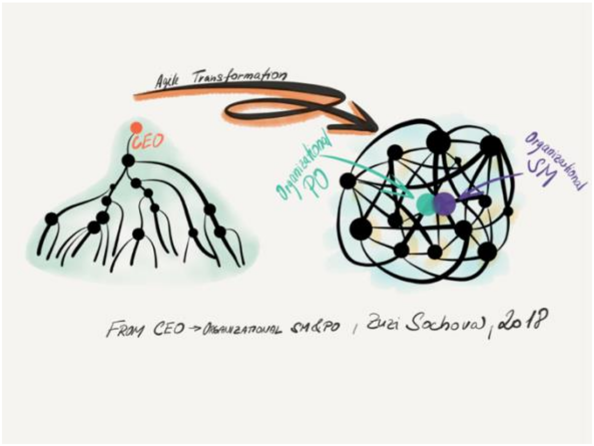
Jon has been an agile practitioner since the early 1990s, is the Enterprise Agility Leaders Network founder, is a member of the Business Agility Institute advisory council and is a guest speaker at London Business School.

Do we need CEO in Agile Organization?

By Zuzana Zuzi Sochova

Agile at the organizational level changes everything. I wrote here already about Agile HR, Agile Board of Directors, and Agile at the Executive Team Level. Let's have a look at the top of the organization. Should it remain the same? Or shall we go one step further and change it as well?

Searching for an "agile mindset CEO" is frankly a nightmare. Everyone who tried it can confirm that. There are not many CEOs with enough Agile experience yet in the world and those few who has it are not likely looking for a job as they are usually quite happy in their current organization. So, no matter which executive search firm you choose, or what you write to the position description, the search firms are no real help here. Unless you already spent effort up front on growing the talent internally, you need huge luck to get someone who understands agile more than a basic theory. No matter how desperate you are searching for a CEO, this is still just a small obstacle, not the real reason for a structural change.



The real need for top-level change starts when most of the organization is having an agile mindset. The Agile transformation is about being agile. How many times you've heard this request: *"How about if our management give us more support on your Agile journey?", "How about if our executive team is Agile and don't just push it to us?", "How about if they practice the same way we work as well?*"Would that make it a difference? Maybe. So why don't we go one more step ahead and change from the top and become a role model for the entire organization? As the organization has changed and Agile is not solely the domain of the IT anymore, but the business agility got into every department, the need for a change at the top level is inevitable. Why do we need a CEO in the first place? Just because that's how organizations always been? Shouldn't we 'eat our own lunch' and change the way we work entirely? Shouldn't we apply the same principles the team do? Sounds simple, right.

And as usually it is simple to understand, hard to do as it needs courage. Courage to say "We are going to be different." We will have Organizational ScrumMaster and Organizational Product Owner instead of a CEO because it will be closer to the way we work at this organization, fits our values, and last but not least we believe it will help us to be more flexible and adaptive at the organizational level. And that's worth of trying.

The Organizational ScrumMaster would be focusing on the right culture, mindset and structure, so we become high-performing innovative organizations which embraced agility, and Organizational Product Owner would be focusing externally to the business, vision a purpose to we know where are we heading and why and are value driven. Both roles need to respect each other and be open with each other, the same way as it is in a single Scrum team, as together they will be part of the Organizational Leadership Team, and the network structure of self-organizational teams. There are two roles in Scrum by a reason instead of one. When you ask people if they would suggest to combine them, no one feels it is a good idea. And at the top organizational level, we would still do it? The similar reasons are valid at this level as well. When you think about it, it fits the way we work much better, then a single CEO – supports the right organizational mindset, transparency, and collaboration, it is consistent with who we are.

From a legal perspective, it is perfectly possible, and it's not that much work either. You might need to change the bylaws a little, but there is no reason why you can't do it. From a hiring perspective, it's much simpler, as you are not looking for that 'superhero' personality who would be great with both internal and external sides. Try it. As I said already, all you need is courage. And that's one of the Scrum values anyway. Experiment, and from that stage inspect and adapt. Now, do I believe that this SM-CEO or PO-CEO will eventually make themselves out of job? No, I don't. It's the same as the team level. Even if the team is self-organized and knows the business well, there is still work for ScrumMaster and Product Owner. Similarly, at the organizational level, there is still a need for Organizational ScrumMaster and Organizational Product Owner even when the network of collaborative teams got self-organized, business value-driven, and customer-centric. The Organizational ScrumMaster and Organizational Product Owner would use Leader – Leader style to build other leaders around them, and if they are successful, the organization will become purpose driven where leadership will be emergent and structure liquid. The same way as it is in the Scrum team, the Organizational ScrumMaster and Organizational Product Owner will move from those who explain, tell and share, to those who coach, facilitate, and keep the system spinning. And that's what is the Agile Leadership about.

About Zuzana Zuzi Sochova

Zuzana **"Zuzi"** Šochová is an independent Agile coach and trainer and a Certified Scrum Trainer with the Scrum Alliance with more than 15 years' experience.

She has implemented Agile transformation and implementation in many companies and teams around the world. By creating and sustaining Agile leadership, she believes the worlds of work and life can be happier and more successful.

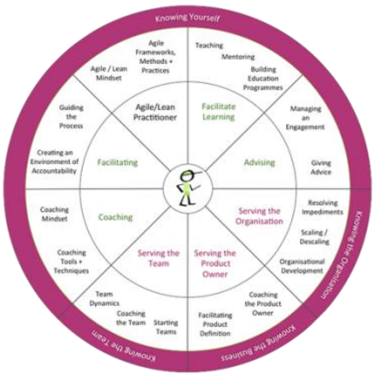
She is the founder of the Czech Agile Association and the AgilePrague conference, a member of the Scrum Alliance board of directors, and an author of The Great Scrum Master: #ScrumMasterWay.

Email Zuzi at
zuzi@sochova.com

Follow Zuzi on Twitter
@zuzuzka

The Agile Coaching Growth Wheel

By Mark Summers



Authors and Acknowledgments
The Agile Coaching Growth Wheel is a collaboration between a number of contributors including:

John Barratt, Shannon Carter, Rickard Jones, Martin Lambert, Stacey Louie, Zia Malik, Helen Meek, Rohit Ratan, Tom Reynolds, Andre Rubin, Kubair Shirazee and Mark Summers.

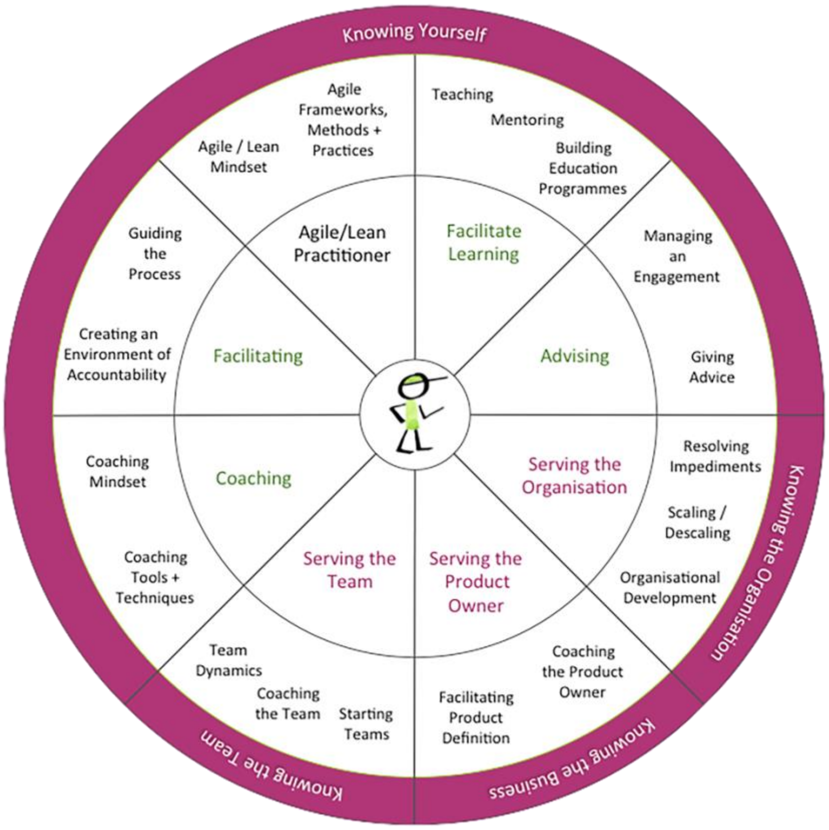
We would also like to thank all the other Agile Coaches who have provided feedback.

What is Agile Coaching?

Agile Coaching is a collaboration with people in a thought provoking and creative journey using coaching approaches with an agile mindset and principles to help individuals, teams and organisations be the best they can be.

What is the "Agile Coaching Growth Wheel"?

The "Agile Coaching Growth Wheel" is a tool for Agile Coaches and ScrumMasters to help them reflect and grow themselves on their Agile journey. This tool is also best used with another coach to help support them.



Agile Coaching Growth Wheel by John Barratt, Shannon Carter, Rickard Jones, Martin Lambert, Stacey Louie, Zia Malik, Helen Meek, Rohit Ratan, Tom Reynolds, Andre Rubin, Kubair Shirazee, Mark Summers is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License

The wheel has 8 segments or spokes which represent main competency areas. Within each competency area, there are one or more competencies that an individual can reflect on. This guidance identifies 5 levels for each of those competencies.

5 Levels of assessment

1. Beginner
    - Knows the theory but has no real practical experience of the application
2. Practitioner
    - Has applied in at least one situation and may still require support in the application
3. Journeyperson
    - Can apply in most situations independently
4. Craftsperson
    - Unconscious competence has mastered the application and knows when to bend and when to break the rules
5. Guide/Innovator
    - Capability to change to meet the current situation and innovate to create new techniques

The tread around the outside represents the supporting competencies, these are knowledge areas that in-turn support the skills of the other 8 competency areas.

Why create this wheel?

Misconceptions exist with clients and Agile Coaches with regards to what Agile Coaching is. This confusion has resulted in unqualified people presenting themselves as Agile Coaches with little experience and low competence. This creates something of a lottery for clients choosing the right Agile Coach for them.

How does one become a great Agile Coach? There is no clear pathway, Agile Coaching is not yet a fully-fledged profession. This Agile Coaching growth wheel lays down some core competencies, that allows an Agile Coach through a reflective process to go from good to great.

In 2011 Lyssa Atkins and Michael Spayed created a competency framework for Agile Coaches. Intentionally this was not a competency model, as it did not define specific behaviours, skills, knowledge or levels of proficiency. However, the creators of WhatIsAgileCoaching.org and the creators of this Agile Coaching Growth Wheel believe that more definition is required in order to professionalise the world of Agile Coaching.

We believe that defining the Agile Coaching journey will allow educators and other coaches to better support the growth of Agile Coaches by developing learning and development programmes. It will also build confidence in the industry around the future profession of Agile Coaching. Making it easier for an organisation to select the right coach for them with confidence.

How to use the wheel and guidance

This part of the guidance is written from the perspective of a coach helping an Agile Coach to reflect. There are many different ways that the wheel could be used in a coaching conversation, but it could go something like this ….

Step 1: Identify an area of improvement

Talk through each of the competency areas (the 8 spokes and 4 tread areas), use the guidance below to make sure the coachee has a high-level understanding of each area. You can't improve everything at once, so get the coachee to select an initial area of focus to work on.

Step 2: Reflect on a competency area

For each competency within the competency area, go through the guidance and get the coachee to assess their own competence against the 5 levels of assessment.

Some people will sell themselves short, others will overestimate their competence, your job as a coach is to try and hold them accountable to a true representation of themselves, ask for examples and be curious.

Step 3: Brainstorm options and generate actions

Use the insight generated in the reflection to brainstorm options for growth and then formulate a plan of action.

The rest of the guidance is just that: guidance. The detail against each level for a specific competence is just meant as reflection, not as a checklist. There may be guidance at the practitioner level that you cannot fulfil 100%, perhaps they are not important to you or your context, but as you explore the journeyperson guidance you might find a better fit for where your coachee is at. Ultimately the coachee (Agile Coach) decides.

Agile and Lean practitioner

Agile Coaching is coaching in an Agile context. To work as an Agile Coach most clients would expect knowledge and experience here. Most Agile Coaches come from Agile or Lean backgrounds, but reflecting here helps us stay rooted. If you are coming to Agile Coaching from a non-Agile background, then investment in personal growth is likely to start here. There is also a lot of synergy between an Agile/Lean Mindset and a Coaching Mindset, an underlying belief in people, the idea that change is possible and people can be the best that they can be.

Agile/Lean Mindset

This includes the Agile values and principles, which guide our thinking and actions when approaching new situations. A deep understanding of Agile allows an Agile Coach to apply frameworks and practices in the way they were intended. An Agile Mindset requires belief in yourself and in others, people are the foundation of Agile working. We trust, support and nurture people to unleash their full human potential. Being Agile over doing Agile.

Lean Manufacturing and Lean Product Development provide us with some foundational concepts that underpin the Agile frameworks and methods.

- Focusing on the value that gives the most delight to our customers.
- Optimising our organisations for Flow with small batch sizes with the shortest possible lead time.
- Maximizing quality and minimizing waste.

At its heart Lean is about total respect for the people involved and a continuous improvement mindset

Level	Reflection
Beginner	Able to summarize the Agile values. Able to describe the Agile Manifesto and principles.
Practitioner	Can contrast an Agile mindset with a non-Agile mindset. Demonstrate how the values and principles of the Agile Manifesto are present in how their team works. Able to demonstrate an Agile mindset. Able to explain the core concepts of Lean Thinking. Recognizes when decisions help or hinder the adoption of agile principles. Can help teams apply existing practices in a more Agile way, i.e. collaborative design over design upfront, testing right from the start.
Journeyperson	Models the values and principles. Able to analyse their personal fulfilment of Agile values and identify how they could improve. Able to help those outside of their immediate area adopt Agile principles. Can associate Lean principles and Agile approaches. Can illustrate at least two concrete examples of how they actively applied Agile value(s) in their work.

Craftsperson	Describe an experience in which there is no obvious resolution to an impediment, requiring them to leverage Agile values or principles to help their teams or organisation select possible solutions. Can judge Agile practices adopted at a team and organisation level that are disconnected from the underlying Agile principles.
Guide / Innovator	Thought leadership through creating their own new values and principles that help people achieve greater levels of agility.

Agile approaches - frameworks, methods and practices

There are many flavours of Agile, an Agile Coach understands that there is no one correct way, and therefore has experience with many Agile approaches.

Level	Reflection
Beginner	Can describe how at least one Agile approach and how it relates to the Agile Manifesto. Can explain at least 3 Agile practices commonly used by Agile teams

Practitioner	Able to use a prescribed framework or method, applying all of its elements in one situation. Can describe at least three Lean/Agile development frameworks/methods. Is aware of changing Agile trends and newer methods in the industry. Can compare and contrast different Agile approaches and apply where needed.
Journeyperson	Can associate at least three Agile engineering practices to Lean practices. Analyse the benefits of a wide range of Agile practices and can help the team adopt them as appropriate. Can apply Agile practices beyond the team. Respected outside of the immediate work environment as somebody who knows about Agile practices. Applied at least one framework or method in multiple situations.
Craftsperson	Able to evaluate different practices and evolve them to meet the organisational need. Helps the team evaluate the process that is most suitable for them. Can describe a situation in which they might advise a client to apply XP, Lean, or a non•-Agile approach to a workflow instead of Scrum. Can describe the reasoning behind their advice.

	Applies many frameworks and adapts to different situations.
Guide / Innovator	Invents and modifies practices to match the context. Active contributor to the Agile community, consistently identifying and developing, sharing existing and emerging Agile approaches. Helps the organisation evolve the process that is of most value for them.

Core Skill Competencies

The core skill areas of Facilitating, Coaching, Advising and Facilitate Learning draw on other existing professions. These are the key skills that a growing Agile Coach will learn.

Facilitating

Facilitation is the practical neutral craft (an informed blend of techniques and insights) of creating environments of openness, safety and innovation[1].

Facilitation increases the effectiveness of helping everyone align in a collaborative way, to interpret their context and mutually identify the most valuable outcomes desired so that they can be the best they can be.

Guiding the process

Help individuals and teams set goals and manage their coaching interactions to support the journey in pursuit of their goals.

Level	Reflection
Beginner	Understands the role that listening plays in facilitation. Can list at least three ways they may facilitate the process.
Practitioner	Plans the content and agenda for a collaborative meeting and can facilitate the meeting. Able to facilitate a small group towards a goal. Creates an environment where the whole group are involved. Prepares well for the meeting. Facilitates the process over participating in the discussion. Identifies at least three indicators when a group is engaged in divergent thinking and at least three indicators when a group is engaged in convergent thinking. Identifies at least three challenges of integrating multiple frames of reference (i.e. the “Groan Zone”). Describes at least three ways a group could reach their final decision. Describes at least five facilitative listening techniques (e.g. paraphrasing, mirroring, making space, stacking, etc.) for effective meetings/events and can apply at least two of them. Understands when conflict is arising and

	has at least one strategy for dealing with it.
Journeyperson	Has practised at least two alternatives to open discussion, in multiple contexts (e.g. structured go-arounds, individual writing, listing ideas, dialogue in pairs or small groups, etc.) and can explain when they may be effective. Identifies at least one action the facilitator can perform to support meeting participants during divergent thinking, integration, convergent thinking, and closure that will support the development of an inclusive solution (e.g. powerful questions). Can apply five visual facilitation techniques for a collaborative session (e.g. card question, clustering, dot voting, visual note-taking). Analyses situations where conflict arises and selects an appropriate strategy to deal with the situation. Can apply gamification techniques. Able to apply multiple strategies for dealing with conflict, depending on the context.
Craftsperson	Able to facilitate in any context. Can facilitate large events, such as Big Room sessions, Bazaars, organisational change events, Conferences, Fests, Gatherings, Retreats. Works with other facilitators as a mentor to help them develop.

Guide / Innovator	Invents and modifies practices to match the context. Active contributor to the community consistently identifying, developing, sharing existing and emerging facilitation practices.

Creating an environment of accountability

Hold attention on what is important for the individual or team, and leave responsibility with them for action. Hold the team accountable to what they say they will do and their plan.

Level	Reflection
Beginner	Describes three obstacles to clear communication and describes their impacts on both the sender and receiver. Describes at least four ground rules to foster clear communication in a collaborative meeting and describes how the introduction of the ground rules impacts the interaction. Understands the importance of the team following up on their actions.

Practitioner	Describes, using two concrete examples, when the Coach should not act as the facilitator for the group. Able to hold the team accountable to the actions that have been agreed. Has created at least three working agreements to foster clear communication in a collaborative meeting and describes how the working agreements impacted the interaction.
Journeyperson	Demonstrates at least two techniques for raising team accountability. Demonstrates the ability to maintain unbiased views and leverage collaboration and consensus strategies to identify creative opportunities.
Craftsperson	Able to create an environment of trust and respect in any situation. Works to build accountability within the team to reduce dependence on the coach. Helps teams create the necessary mechanisms for the team to reach for high-performance. Holds the team accountable for building and sticking to these behaviours.
Guide / Innovator	Creates new innovative techniques that enhance facilitated events, growing team ownership mind-set.

Coaching

The International Coaching Federation (2013) defines coaching as: “Partnering with clients in a thought-provoking and creative process that inspires them to maximize their personal and professional potential... Coaches honour the client as the expert in his or her life and work and believe every client is creative, resourceful and whole. Standing on this foundation, the coach's responsibility is to:

- Discover, clarify, and align with what the client wants to achieve
- Encourage client self-discovery
- Elicit client-generated solutions and strategies
- Hold the client responsible and accountable”

Coaching Mindset

Coaching is not about fixing people problems; it is about believing in people and helping them grow to be the best that they want to be.

Level	Reflection
Beginner	Aware of coaching principles and ethics.
Practitioner	Demonstrates a coaching stance in an interaction with one or more people (i.e. neutrality, emotional intelligence, client agenda, etc.) and describes how that coaching stance impacted the interaction. Able to focus on the coachees' agenda and believes the coachee has the answer.

Journeyperson	Actively living the coaching principles. Ability to demonstrate coaching stance in multiple situations. Able to create a safe, supportive environment that produces ongoing mutual respect, trust, creative self-expression and opportunities for new learning. Awareness that coachees are the highest priority. Ability to create a spontaneous relationship with an open, flexible, confident style e.g. dancing in the moment, "goes with the gut", open to not knowing, willing to take some risks, lightness and energy, is confident with strong emotions. Recognises when the coachee requires specialist support (e.g. counselling). Has a coach themselves.
Craftsperson	Can act as a personal coach outside of context. Skilled in treating people according to their emotional reactions. Able to be optimistic even in the face of failure. Works with other coaches to help them develop as coaches. Able to analyse coaching approach and make adaption. Participates in the coaching community.

Guide / Innovator	Discovering new coaching principles/concepts and sharing them with the coaching community.

Coaching Tools and Techniques

There are a number of different approaches to coaching, each of which may contain different models, practices, and tools that can help a coach given different contexts.

Level	Reflection
Beginner	Aware of one coaching tool/technique. Able to describe the difference between facilitating, teaching, mentoring, consulting, and coaching.
Practitioner	Can apply at least three coaching techniques (e.g. active listening, powerful questions, reflection, feedback, GROW model, etc.) and describe how the coaching technique impacted each interaction. Able to formulate a basic coaching agreement and contract. Able to actively listen, without trying to solve the coachees' problem some of the time. Able to help the coachee create opportunities for learning and for taking new actions. Helps them explore alternatives, promotes experimentation

	and self-discovery, celebrates successes and capabilities, helps “do it now”.
Journeyperson	Received coaching and coaching supervision. Received formal training or mentoring on coaching skills. Describes at least five elements of a fundamental coaching agreement (e.g. role of the coach, duration, expectations, feedback, responsibilities). Able to actively listen, without trying to solve the coachees' problem most of the time. Asks questions for maximum benefit, they evoke discovery and insight, challenge assumption, open-ended, forward-looking and pre-supposing success. Understands when to use a coaching approach.
Craftsperson	Undertaken a coaching education, accredited by the International Coaching Federation (ICF) or equivalent. Has multiple coaching approaches to bring to bare at any time. Complete focus on what coachee is/is not saying to understand the meaning of what is said e.g. client’s agenda, hear concerns, values, beliefs, summarises and mirrors back without judgement.

Guide / Innovator	Invents and modifies practices to match the context and published the result. Active contributor to the coaching community consistently identifying and sharing existing and emerging coaching practices.

Facilitate Learning

Agile is all about learning, as an Agile Coach, you will need to facilitate the learning of other people around you, helping them learn new skills and knowledge.

Mentoring

As a mentor you are able to use your expertise to show others new skills and/or to develop existing ones, working alongside the mentee as they do their job. As a mentor, you are more experienced than your mentee.

Level	Reflection
Beginner	Able to describe mentoring.
Practitioner	Able to give feedback in a way the encourages growth. Has mentored on at least one Agile practice. Has been mentored.

Journeyperson	Gives feedback without interpretation or judgement. Able to mentor in a number of areas. Called on to act as a mentor outside of current area of work. Able to use storytelling and metaphors to convey ideas. Able to identify mentees needs and adopt the approach appropriately. Fosters an ongoing mentoring relationship with the mentee. Knows when to stop mentoring. Learns from mentoring relationship. Understands the impact they are having on the mentee.
Craftsperson	Regarded as a mentor and leader in developing understanding and awareness of agility, within the organisation. Able to challenge individuals or teams limiting beliefs and assumptions. Able to guide/mentor other mentors.
Guide / Innovator	Invents and modifies practices to match the context.

Teaching

The ability to convey information in a way that is understood and useful to the recipient. You will have to be adept at integrating information to help people gain awareness.

Level	Reflection
Beginner	Able to convey knowledge to a group of people. Able to describe teaching.
Practitioner	Has facilitated at least one training workshop (e.g. half a days introduction to Agile). Able to present concepts in depth to a group of people. Applies at least one appropriate teaching style.
Journeyperson	Has co-trained with a more experienced trainer. Demonstrates the ability to create a suitable learning environment making use of the physical space. Regularly gathers feedback and uses this to adapt their approach to teaching. Practises with cultural sensitivity and adapts accordingly. Creates an environment of emotional safety, so that learners feel safe to engage. Able to create a learning environment where students can learn from each other. Has applied multiple teaching styles in multiple training workshops. Able to create an opportunity for different

	learning styles. Uses effective storytelling to convey key concepts. Able to deal with disruptive situations in the training room.
Craftsperson	Focuses on stabilizing principles and varying practices to situationally align the client’s maturity with effective application of agility. Ability to maintain the required energy level. Practises effective classroom management. Identifies and utilises effective instructional techniques (gamification, visual aids, etc.) to impart key concepts. Has co-trained other aspiring teachers and can give them constructive feedback. Able to assess audience response and adjust accordingly, to maximise the learning experience. Understands and can apply the principles of adult learning theory. Creates own games and activities for their training.
Guide / Innovator	Able to develop new instructional techniques/styles. Invents and modifies practices to match the context. Active contributor to the Teaching community consistently identifying and

	sharing existing and emerging training practices. Shares new games/activities with the wider community.

Building education programmes

In order to facilitate growth, especially deeper learning there is often a need to support individuals and teams on longer development programmes. This is likely to include working with different parts of an organisation, such as HR and learning & development, to design and deliver suitable programmes.

Level	Reflection
Beginner	Able to describe the learning needs of an individual or team.
Practitioner	Defines clear learning objectives, which are used to create and execute training for a team and/or stakeholders. Has demonstrated the ability to integrate learning materials, to meet the need and objectives of at least one training event.

Journeyperson	Able to design learning and capability goals for one individual (or team) and analyse the execution of these goals. Has demonstrated the ability to integrate learning materials, to meet the need and objectives for multiple training events. Able to build new learning materials, in a variety of forms to cater to different learning styles. Able to check for understanding of learning. Can apply at least 1 model or strategy that promotes lesson stickiness (e.g. 10, 24, 7).
Craftsperson	Able to design and build a bespoke training and development programme, with multiple learning interventions and can evaluate the success of the programme.
Guide / Innovator	Invents and modifies practices to match the context.

Advising

As an advisor or consultant with an individual, team or within the wider organisation you are responsible for setting the workup for success and may also be called on to play the role of a trusted advisor.

Managing an engagement

A common area of frustration and failure is because Agile Coaches have not agreed to clear goals for their work with an individual, team or organisation. Also as organisations are complex systems any work should be carried out with appropriate inspect points, that allow the client and the coach to adapt the work or even the goals.

Level	Reflection
Beginner	Understands the importance of setting goals, boundaries and rules. Understands that organisations are complex, therefore engagements should be empirical in nature, with effective feedback loops.
Practitioner	Able to create an agreement with individuals and/or with a team, that defines how they will work together. Has identified clear goals within at least one engagement and identifies how those goals will be measured. Can describe an occasion with at least one team/individual that allowed the coach and coachee to inspect and adapt towards the defined goals. One or more occasion the individual/group is satisfied that the goals have been met or alternate goals have been identified. Either the engagement is at an end, or new goals have been identified. Is clear and transparent with the engagement stakeholder and has appropriate feedback loops in place.

Journeyperson	Creates a coordinated agreement with a team of teams (multiple teams, e.g. through a coaching plan). Identifies clear goals and measures that are aligned to organisational outcomes. Can describe multiple instances that allowed the coach and coachee(s) to inspect and adapt towards the defined goals. Able to describe the effective closing of multiple coaching relationships. Able to grow relationships across an organisation. Champions transparency within the engagement. Practices that change is not done to people but facilitated, encouraging full participation. Able to manage engagement.
Craftsperson	Creates agreements with leadership for engagements across an organisation. Able to manage engagement that involves a number of Agile Coaches. Able to manage budgets for organisational change. Able to look after the wellbeing of others involved in change.
Guide / Innovator	Thought leadership on such things as organisational change. Creates and publishes tools and

	techniques for others to use.

Giving advice

As an Agile Coach you may take on the role of a trusted advisor, you have experience which will be invaluable. This can be especially important when the client has low confidence and low competence in something.

Level	Reflection
Beginner	Understands that sometimes advice needs to be given.
Practitioner	Able to use direct communication with a team, language is clear with feedback, reframes, clearly explains techniques or exercises, and does so in a respectful way. Creates awareness by presenting hard facts to the team even if difficult – Be the mirror to the team. Knows the right time to give advice and when not too.

Journeyperson	Has examples of using direct communication with multiple teams and can describe examples where this has been used effectively with management. Creates awareness by presenting hard facts to the organisation. Had practised multiple techniques to provide feedback. Aware of the limitations of knowledge and has the courage to say I don't know.
Craftsperson	Able to use multiple styles and techniques to raise awareness within an organisation.
Guide / Innovator	A thought leader who is sought out by organisations for their advice.

Servant Leadership Competencies

While servant leadership is a timeless concept, the phrase "servant leadership" was coined by Robert K. Greenleaf in The Servant as Leader, an essay that he first published in 1970. In that essay, Greenleaf said:

"The servant-leader is a servant first… It begins with the natural feeling that one wants to serve, to serve first. Then conscious choice brings one to aspire to lead. That person is sharply different from one who is a leader first, perhaps because of the need to assuage an unusual power drive or to acquire material possessions…The leader-first and the servant-first are two extreme types. Between them, there are shadings and blends that are part of the infinite variety of human nature.

“The difference manifests itself in the care taken by the servant first to make sure that other people’s highest priority needs are being served. The best test, and difficult to administer, is: Do those served grow as persons? Do they, while being
served, become healthier, wiser, freer, more autonomous, more likely themselves to become servants? And, what is the effect on the least privileged in society? Will,
they benefit or at least not be further deprived?“.

A servant-leader focuses primarily on the growth and well-being of people and the communities to which they belong. While traditional leadership generally involves the accumulation and exercise of power by one at the “top of the pyramid,” servant leadership is different. The servant-leader shares power puts the needs of others first and helps people develop and perform as highly as possible.

Being a Servant Leader is the very being of an Agile Coach.

Serving the team

Agile Coaching is focused on helping teams become the best they can be using the core competencies you have already covered. This section covers a more focused view of specifically serving the team through a journey to high performance.

Team dynamics

A team is more than a collection of individuals, it is a human system with its own characteristics, needs and growth potential. Moments of conflict or collaboration difficulty should be seen as human systems dynamics, rather than solely personal to the individuals involved.

Level	Reflection
Beginner	Can list at least three different challenges facing a self-organising team. Can list at least three impediments that can impact a team. Understand the importance of conflict in a team.
Practitioner	Explains the difference between a working group and a team. Identifies at least three key attributes of effective Agile Teams (e.g. ground rules in place, awareness of capabilities and capacities, effective and efficient collaboration). Applies at least two methods for improving team performance (e.g. common goals/purpose, shared accountability, working agreement, psychological safety, etc.). Identifies at least two pitfalls of a homogenous team (i.e. lack of different perspectives, experiences, and viewpoints). Describes at least one multi-staged model for team formation and development (e.g. the Tuckman model). Knows how to surface conflict in a positive manner to improve the team’s conflict competency.

Journeyperson	Able to apply at least two different models for team development (e.g., Tuckman model, team performance curve, etc.) and can adapt the coaching approach based on the outcome. Appraise the effectiveness of at least two different development frameworks for supporting an Agile team's growth. Knows when and how to engage leadership in solving organizational impediments that are impacting the team.
Craftsperson	Contrast the different team dynamics across multiple teams with whom you have worked, and evaluate the effects on team results. Can demonstrate how to increase the team's capacity for both self- awareness and self- management.
Guide / Innovator	Has created/adapted a new model to support team development and published it to the wider community.

Coaching the team

Assess the health of an Agile team and adjust one's individual coaching style to facilitate the team's journey to high-performance.

Level	Reflection
Beginner	Able to list three ways in which you serve the team.
Practitioner	Applies at least two techniques to foster greater self-organisation within teams (e.g., powerful questions, autonomy/mastery/purpose, active listening, etc.). Applied a countermeasure to reduce the impact of a challenge facing a self-organising team. Able to describe how a self-organising team approaches at least three challenges.
Journeyperson	Demonstrates at least two tangible examples of how they developed and changed the culture of at least one team. Identify two team formation and development challenges commonly encountered while introducing Agile. For each, describe a coaching approach to address the challenge. Applies at least three techniques for addressing team dysfunctions and has used these techniques multiple times.

	Applies at least three techniques or activities for building trust in a team, in multiple contexts. Able to create a coaching agreement with the development team. Can explain how they have helped a team increase the quality of delivery through technical excellence practices. (e.g Pair Working). Introduced one or more models for assessing healthy team functioning, especially the ability to identify clearly dysfunctional aspects.
Craftsperson	Able to evaluate different techniques used to increase team effectiveness across multiple teams with whom they have worked, and evaluate the effects on team results. Ability to adapt leadership approach based on team maturity, and has an awareness to decrease the level of direct involvement as they mature.
Guide / Innovator	Has created new coaching tools to support coaching teams, and shared with the wider community.

Starting teams

Studies show a team launch/relaunch can represent up to 30% of team performance this should include elements such as helping the team get to know one another, create a culture, align on a vision, setting up their work environment and establishing team agreements and/or ground rules.

Level	Reflection
Beginner	Understands the importance of a positive team launch.
Practitioner	Illustrates what is important for a new team. Explains how purpose, alignment, and context are set and used during the start-up of a team to accelerate teamwork. Able to organise and facilitate a change of context for an existing team that defines purpose, alignment, and context. Demonstrate ways to help people get to know each other (e.g. constellations, tribes). Has experienced a positive team launch.
Journeyperson	Explains at least three reasons why the start of a new team should be handled differently from a traditional project kick-off/charter (e.g. level of collaboration, lack of experience in Agile environments, (the importance of

	shared understanding). Explains how seeing the whole system, emphasizing collaborative work, focusing on a good start, continuous learning, and “good enough for now” support the launch of a new Agile team. Describes at least three responsibilities each for the sponsor (e.g. clarify constraints, context, and stakeholder expectations), Has organised and facilitated the launch of at least one new Agile team. Has helped a team create a team vision aligned with the organisation's vision.
Craftsperson	Has successfully organised and facilitated the launch of a number of new diverse teams. Has supported an organisation in changing the environment in order to provide the best possible start to a team. Has supported an organisation in articulating a clear vision and goals (e.g. OKRs) that can support teams in aligning to the vision and goals.
Guide/ Innovator	Has published a tool or guide to launching teams.

Serving the Product Owner

An Agile Coach serves the Product Owner in several ways, including [2]:

- Ensuring those goals, scope, and product domain are understood by everyone on the Team as well as possible;
- Finding techniques for effective product backlog management, that maximises value;
- Helping others to understand product planning in an empirical environment;
- Supports the Product Owner and business stakeholders understanding and practice of agility;

Facilitating product definition

Customers do not know the exact product they want even if sometimes they think they do. They only know the problem they are trying to solve or the business results they are looking for. It is our job to support the organisation in building the product to provide the customer with the outcome they desire.

Level	Reflection
Beginner	Has experienced a Product vision being formed

Practitioner	Has facilitated the creation (or refinement) of the product vision between the Product Owner and the development team. Can explain at least two techniques for moving from product vision to product backlog (e.g., product vision board, business model or Lean canvas, customer journey, impact mapping, user story mapping). Organises and facilitates a product backlog refinement session with one team and stakeholders. Can explain two techniques that could be used to create product backlog items that are ready to be taken into the next sprints. Has an understanding of what a Minimal Viable Product is (the Smallest amount of effort to test a hypothesis).
Journeyperson	Organises and facilitates the creation (or refinement) of the product vision between the Product Owner and stakeholders, with multiple teams. Can apply at least two techniques for moving from product vision to product backlog (e.g. innovation games, user story mapping, user story workshops, brainstorming, etc.). Can appraise at least three criteria that can be used for structuring a complex or multi-team product backlog (e.g., feature area, team). Has a customer focus and able to support activities such as user research & testing.

	Able to facilitate Lean experiments. Can express commercial awareness (e.g. funnels, understanding OPEX & CAPEX, action and vanity metrics, types of value) Can facilitate the agreement of a Minimal Viable Product and support its release and measured outcome.
Craftsperson	Able to support product kick-offs in almost any situation engaging multiple customers, stakeholders, leadership and team members. Has successfully defined and launched a product they had responsibility for.
Guide / Innovator	Has created tools and techniques others can use to define a product. Speaks at conferences and other community events on Product topics.

Coaching the Product Owner

Initially, an Agile Coach may help educate a Product Owner, but in the long term they look to support the Product Owner's learning and growth through Coaching.

Level	Reflection
Beginner	Can list at least three ways in which you could serve the Product Owner. Can identify at least three effective collaboration techniques that a Product Owner can use to work with the team. Able to discuss at least three negative impacts that arise when the Product Owner applies excessive time pressure to the development team.
Practitioner	Lists three benefits that arise if a Product Owner participates in the retrospective. Can explain agile to business stakeholders. Has built a coaching relationship with at least one Product Owner and helped them become more effective.

Journeyperson	Has built a coaching relationship with multiple Product Owners and helped them become more effective. Has worked with the organisation to align around Products using tools such as value stream mapping. Is able to support the Product Owner in using product data to make an informed decision on what to build next. Supported the Product owner in sharing ROI information to the team.
Craftsperson	Able to evaluate the effectiveness of previous coaching done with Product Owners and uses this to continually improve how to better serve others. Able to support a Product Owner community in their growth. Has changed the focus on Product success to be outcome-focused.
Guide / Innovator	Shares experiences and tools with the community on coaching the Product Owner. Creates new tools to help Product Owners grow and/or do their job.

Serving the organisation

An Agile Coach serves the organisation in several ways, including [2]:

- Leading and coaching the organisation in its Agile adoption;
- Planning Agile adoption within the organisation;
- Helping employees and stakeholders understand Agile delivery;
- Nurturing change that increases the effectiveness of the teams; and,
- Working with other Agile Coaches to increase the effectiveness of the application of Agile in the organisation.

Organisational Development

Organisations are complex, and changing them is an even more complex proposition. An empirical and informed approach to the change process improves the chances of success of an Agile transition.

Level	Reflection
Beginner	Lists three ways in which you serve the organisation. Can describe one example of a major organisational design change implied by adopting Agile (e.g. elimination of single-function groups, traditional career paths, or annual appraisals). Able to list at least three ways that traditional management changes in the Agile workplace. Can describes at least two stakeholder behaviours that support the team's success and at least two behaviours that do not support the team's success.

Practitioner	Has applied at least two techniques to effect change in an organisation in order to help teams be more productive.
Journeyperson	Can describe the nature of complex systems (eg. cause-and-effect only visible after the event, high level of uncertainty and disagreement, emerging systems, products and practices). Able to explain the importance of taking a systemic view (i.e. help a stakeholder understand that the system as a whole needs to be optimized, understand causal loops and delayed effects). Can describe at least two systematic methods for helping organisations improve their Agile adoption (e.g. causal loop analysis, value stream mapping). Has applied at least one systematic development approach (e.g. systems thinking). Can describe at least two frameworks for catalyzing organisational change (e.g., Kotter's 8-Step model, the Grief Cycle, 4D Model/Appreciative Inquiry). Can demonstrate at least two tangible examples of how they developed and changed the culture of his/her team (or organisation) from a command-and-control to an Agile mindset. Able to describe an approach to a complex intervention that addresses the root cause(s) of an organisational dysfunction and analyse the long-term impact.

	Outside of their team, they are seen as someone who develops the organisation
Craftsperson	Identifies three factors to introduce and cultivate in an organisation (business unit, department, programme) that can promote improvement in agility and value delivery. Some examples are collaboration tools, technical practices, and structural changes. For each, can describe how it enables and enhances agility and success. Able to design and facilitate a retrospective with senior leaders and executives to foster improvement at the organisational level. Able to explain a variety of approaches for creating an organizational change strategy. An effective approach should leverage Agile principles such as co-creation through collaboration, incremental change, transparency, fast feedback, and frequent inspect and adapt cycles. In addition, it should take into consideration that psychological transition happens at different paces for different people and groups, and an overall change process in an organization may take many years. Can Identify and explain not only the limits of their current skills and leadership maturity but also the boundaries of what they will and will not tolerate with regards to Agile and other personal values. Has identified their own gaps and can collaborate with other coaches, in order to serve the client organisation.

Guide / Innovator	Has created a new model or approach to support organisations through change. Has worked with several large organisations and supported them through large changes, furthermore has published the case studies and talks about them regularly.

Scaling / Descaling

Level	Reflection
Beginner	Aware of at least two approaches for scaling.
Practitioner	Illustrates, with at least two reasons why scaling might not be such a great idea (e.g. products created by small teams, communication overhead, TCO). Can identify at least three techniques for visualizing, managing, or reducing dependencies between teams. Recognize at least three different scaling frameworks or approaches. Experiment with at least one large-scale, participatory meeting format (Open Space, World Cafe, etc.) to scale meetings/workshops.

Journeyperson	Able to differentiate the impact of feature teams versus component teams on the delivery of value. Describes an organisational design that enables multiple teams to work on the same Product. Explains the pitfalls of too much or too little prescription (e.g. lack of autonomy, lack of alignment, no slack, integration mess, overly detailed planning, not meeting the Definition of Done, overly slow pace, death by meetings, etc.). Contrast two patterns for scaling the Product Owner role (e.g., shifting clarification responsibility to the development team, defining feature areas or different sub-products, PO team, Chief Product Owner). Can describe at least five techniques to improve inter-team collaboration and has experimented with at least two of them. Able to explain at least three benefits of supporting strong technical practices when working with multiple teams. Has organised and facilitated multiple large-scale, participatory meeting format (Open Space, World Cafe, etc.) to scale meetings/workshops.
Craftsperson	Evaluate an experience with supporting the work of multiple teams in an organisation; identify how you would do things differently. Able to connect interdependencies and impact teams’ reflection, learning, and

	growth. Knowledge and application of multiple change management frameworks. Demonstrates competency in successfully applying the frameworks.
Guide / Innovator	Has written case studies on multiple large organisations including what did and did not work in their continuous pursuit to agility Has invented a Scaling / Descaling framework or principles that have been used within multiple organisations

Resolving Impediments

Level	Reflection
Beginner	Can discuss at least two ways to help the team with responding to impediments (e.g. makes impediments visible, works with the team to resolve impediments). Can identify and explain at least three common organisational impediments outside the scope of a team that can affect the effectiveness of teams (e.g. geographical distribution, people in multiple project teams, incentives and HR policies, no constructive safe-to-fail culture).

Practitioner	Able to identify at least three typical impediments for a team and describe at least one way to address them (e.g. late attendance in meetings, blocked work, supplier issues). List at least three techniques to evaluate impediments in-depth (e.g. root- cause analysis, fishbone, 5 whys) and describe when they might not be working. Able to analyse an impediment and identify a root cause(s) and/or underlying issue(s). Is able to highlight when an impediment is within their or a team's sphere of control
Journeyperson	Has demonstrated the ability to remove impediments from multiple teams in multiple contexts. Has the ability to see impediments that span multiple teams at the same time and facilitate the removal of the impediments including working with leadership Facilitated large scale retrospectives in order to uncover systemic organisational impediments,
Craftsperson	Has demonstrated the ability to remove organisational impediments e.g. (Changing the environment, Organisational Structure) Is Continually looking to remove bureaucracy, waste and reduce top- down hierarchies in order that teams can operate as an interacting network, all focused on working together to deliver

	relentless value to customers. Coached senior leaders to support teams in the removal of systemic impediments
Guide / Innovator	Is specifically brought into an organisation to resolve specific systemic problems

Knowledge Areas

The knowledge areas represent your domain expertise. This contextual expertise may help you build trust with the team or organisation with whom you are working. However, the risk is too much domain expertise may make it more difficult to be objective in your coaching. Therefore, it may be valid for an Agile Coach to allow a reduction of expertise in some areas (i.e. choosing not to stay up to date with the latest changes in technology), but still to seek to increase knowledge in other areas.

Knowing the team

Expertise in the technical work of the team, including:

- Understanding of current technical practices, and practices that could be improved or adopted to increase agility; and
- Technical understanding of the product a team is using or products across the organisation.

Knowing the business

Expertise in the business domain of the team or organisation, including:

- Understanding of current market place in which business is being conducted; and
- Understanding the needs and concerns of users, customers and other business stakeholders;

Knowing the organisation
Expertise in how the organisation works, including:

- Knowledge of structures, policies, operating models
- Understanding of relationships between people, teams, departments
- An understanding of the organisational culture

Knowing Yourself
If you are going to be helping the growth of others then this needs to start with you, knowing and growing yourself. A deep understanding of your drives, beliefs, values and strengths can be valuable to manage your emotions when interacting with others

Acknowledgements
The original concept was conceived at the London Scrum Coaching Retreat in 2018, with thanks to the team:

- Shannon Carter
- Rickard Jones
- Martin Lambert
- Rohit Ratan
- Stacey Louie
- Tom Reynolds
- Andre Rubin
- Kubair Shirazee
- Mark Summers

Thanks to other reviewers who are numerous, some anonymous. Contributors who we know and can thank explicitly include:

- John Barratt
- Dean Bryan
- Matt Hoskins
- Helen Meek

- Tim Robinson
- Andy Spence
- Christian Zander
- Zia Malik
- Simon Lawrence

If you are not listed here but have given feedback, a big thanks, let us know and we will add you to the acknowledgements.

References

1. "What is Agile Facilitation?" by Cara Turner - source: https://facilitatingagility.com/2012/03/05/what-is-agile-facilitation/
2. Scrum Alliance Learning Objectives for Path to Certified Scrum Professional®.
3. ICAgile Learning objectives for Agile Coaching
4. Greenleaf, R. (2002). Servant Leadership 25th Anniversary Addition: A journey into the Nature of Legitimate Power & Greatness

About Mark Summers

Mark is a Certified Scrum Trainer and Certified Enterprise Coach, helping individuals, teams and organisations on their journey since 2006.

Mark is committed to bringing more joy to work and strives to professionalise the world of Agile Coaching.   Within the Agile community

Mark has been spotted facilitating coaching clinics, running workshops, acting as ScrumMaster for a gathering organising team, organising coaching retreats across Europe, and mentoring other coaches.

Mark is a professionally trained coach in business and personal coaching, accredited to Professional Certified Coach (PCC) by the International Coach Federation.

11 Tips for Creating Innovation in Business Business Outcomes Part III

By Jessica Thomason

We live in a rapidly changing world–one in which customers change their minds quickly and every business has to fight to stay relevant in their market. If someone else has a great idea before you–or worse, the ability to implement their great idea before you–it could mean the end. This is why organizations across the globe have put more emphasis on their need for innovation in business.

What is Innovation in Business?

Business innovation is defined in the Path to Agility® transformation model as the "new ideas, creative thoughts, or novel imaginations that provide better solutions to meet new requirements, unarticulated needs, or known market needs".

In other words, an innovative organization has:

1. A deep understanding of its market's strengths, challenges, and needs
2. The ability to ideate and provide new solutions for the problems and needs of that market

To encourage and create innovation in the workplace, there are 11 core functions organizations should aim to achieve.

11 Tips for Creating Innovation in Business

1. Building Experiment-Driven Development
Level: Team

Experiment-Driven Development is the ability to recognize market assumptions and to create hypotheses/execute experiments to validate or invalidate assumptions quickly. This ability starts with empowered teams, who typically work closest to problems and can perform a number of small-scale tests over time.

By building up this ability across many teams, the organization as a whole can respond quickly to new challenges by efficiently testing out a range of potential solutions and implementing the best of them. As a result, the organization has the space to innovate and customers reap the benefits of those tests sooner.

2. Aligning Teams to Value
Level: System

It's not enough for teams to experiment with new ideas if they don't know why. Without a "why", practices that should result in innovation might instead create chaos within the organization.

It's important that all teams first understand their role in the greater system. How does work and value flow through the organization? Where can teams see their direct impact?

Further, teams should be cross-functional, meaning they're set up to reduce dependencies and optimize the flow of value. This will ensure experiments can be conducted quickly and without jumping through bureaucratic hoops, allowing teams to pivot as soon as the results are in.

3. Making Objective Data-Driven Decisions

Level: System

Objective data is our greatest source of truth when it comes to understanding market needs. When teams and leaders have the ability to leverage truly objective data in their experimentation and decision making processes, it drives impactful innovation for the business.

This data will be particularly helpful when creating and tracking progress towards your business goals. You can learn more about creating solid business goals in our article on Objectives and Key Results (OKRs).

4. Building Market Responsiveness

Level: System

The ability to respond quickly to market changes is one of the greatest hallmarks of innovation in business. In order to achieve this, testing and validation of experiments must be shorter than the rate of change in the market.

Often easier said than done in today's volatility and uncertainty. However, organizations that create systems for objective data-gathering, experimentation, and creativity can combat this.

5. Creating Faster Time to Value

Level: System

As we discussed in our last Business Outcomes article on Speed, one team's speed might not be indicative of the entire organization.

Examine your organization's system as a whole. Are there bottlenecks outside of teams that compromise their speed? Are there dependencies on other areas of the organization that could be slowing or preventing their progress?

Leaders across all areas of the organization should work together to ensure that optimizations are being made to improve the overall system, not just one team or even one department. As a result, teams will move more quickly in their experiments, staying ahead of market volatility and driving innovation.

6. Communicating A Compelling Purpose
Level: Leadership

Creating more innovation in the workplace is likely going to require some level of change in your organization. When undergoing any kind of change, we recommend leaders take some time to clearly identify and communicate their compelling purpose.

We define a compelling purpose as the meaningful reason behind why the organization should change. Once identified, this purpose should be communicated across the organization so that alignment is achieved. For example, you don’t want to simply demand more innovation, but communicate why innovation is valuable. This could be the need for better serving customers, finding new market solutions, etc.

7. Empowering Teams
Level: Leadership

As we talk about the importance of experimentation and test validation at the team level, it’s worth mentioning teams need to feel as though they have power to make certain decisions for and based on these experiments. Without this, they will fall back to their leaders and leave decisions to them.

Not only does this limit the number of brains working on a problem, but it also creates more complexity in the system. Precious time is wasted as problems and solutions travel up and down the chain of command. Ultimately, this will slow progress, market responsiveness, and rate of innovation for the business.

8. Implementing Agile Leadership Practices
Level: Leadership

When teams are empowered and we remove the traditional Command and Control processes, there is often a shift in the role of leadership. This may cause strain among employees, particularly middle management, as they find their new place in the system.

However, leaders still play an important part in successfully supporting teams in their more innovative style of work. When leaders implement Agile leadership practices, they recognize and embrace the shift in their roles from directing the work to supporting their people. They enable high-performing teams. This means a wide range of responsibilities for the leader, including:

- Continuously working to remove bottlenecks
- Providing clarity on how individual team's work impacts the organization's goals
- Optimizing the flow of value for increased speed and quality.

9. Creating Decision Agility
Level: Leadership

As we've mentioned, in traditional hierarchical companies, it takes a long time for decisions to go up the chain and come back down. And the decisions are often being made by people too far removed to fully understand the problem.

You can build decision agility through empowered teams, Agile leadership practices, etc.

Increased agility in decision-making means that empowered teams can make quick decisions (based on clear business objectives from leaders and meaningful data from validated tests). As a result, new ideas take less time to come to fruition, meaning the market sees the results of your organization's innovation.

10. Fostering Whole Organization Agility
Level: Leadership

It takes more than the development teams. It takes more than one department. In order to really gain all the benefits of innovation, all groups in the organization must embrace an agile mindset and innovative culture.

This often comes back to leadership and their dedication to communicating the compelling purpose for change. If the entire organization is included and they understand why innovation is important to the business, they are more likely to actively participate in a new way of working.

11. Creating Financial Agility
Level: Leadership

It's important for all systems to support your new culture of innovation–that includes finance.

A final tip for creating innovation in business is to shift from rigid budget processes to fluid funding tied to goals and objectives. While the traditional budgeting process feels safe, it can limit your ability to innovate. When we plan upfront, we tend to spend our time managing to that plan over maximizing value. As a result, we forget to account for learning and adjustments we could (and should) make along the way to better provide value. Instead, try treating each project as an investment. Run experiments. If they prove valuable, continue investing time and money into fully fleshing out the idea.

I hope this provided a little insight into ways you can increase innovation in your organization. If you have more suggestions, we'd love to hear about them in the comments below.

Don't forget to explore the other articles in our Business Outcomes Series:

- Part I: Speed
- Part II: Predictability
- Part IV: Coming soon!

If you'd like to learn more about how you can achieve business outcomes like innovation, explore our Path to Agility®, a transformation framework that allows organizations to break down large business goals into day-to-day capabilities.

Read this article online: https://baa.tco.ac/1Sx_

About Jessica Thomason

Agile Velocity's Content Coordinator, Jessica Thomason, joined the team in 2017 after graduating from Baylor University.

Her creative background lends a hand to content ideation and creation.

In her free time, Jessica likes to paint, read, and take her Australian Shepard on adventures.

Becoming Agile: Transitioning to Transformation

By Christine Thompson

I recently joined a new organisation to support them in what they had termed an ‘Agile Transformation’. As I transitioned from my previous company to the new one, I stopped to reflect on what an Agile Transformation really means and what was likely to affect the success of this. I very quickly came to one conclusion, based on my observations from all the companies in which I have worked as an Agile Coach. Transitioning teams is easy; transitioning organisations is hard. Why would this be so?

My observations were this. Often, we see management directing teams towards an Agile approach. Or, in other cases, supporting teams in their request to make such a change. But what we seem to miss is the understanding from the management in our organisations that Agility depends on them too. Their mind-set; their behaviours; the culture that they instill in the organisation. They issue the directive and stand-back to await the results, seemingly unaware that success depends on them too.

As I began my new role, with this understanding in mind, I turned to Michael Sahota’s book, An Agile Adoption and Survival Guide: Working with organizational culture. (Michael has also published a set of short articles on Agile Transformation Traps which nicely complement his book.) In his book, Michael clearly defines the difference between an Agile Adoption (“doing” Agile) and an Agile Transformation (“being” Agile) and he notes that Agile processes alone will not transform the culture of an organisation. They may be a catalyst for change but will not by themselves induce a change. Michael also notes that most organisations do not actually want what transformation means. Managers want problems solved with as little effort and risk as possible. However, Transformation requires great effort, significant risk and sustained energy over a long period of time. This requires a specific and well supported change effort as well as a sense of urgency from management that the status quo is unacceptable. Transformation is likely to fail without this.

This guidance is clearly reflected in my experience of working in organisations with a desire to “be” Agile. I have found that the adoption of Agile frameworks into the team is relatively easy, assuming the team desire this change. With the right framework and the right engagement, the effectiveness of the framework is visible within only a matter of a few weeks. With the right support and coaching, teams quickly see the benefits of the framework and find their ways of working more effective. Several years ago, as a less experienced Coach, this would have delighted me. Now I see this as the easy part! A team with an effective Agile framework can adopt great Agile behaviours and a great Agile mind-set but none of the full benefits of Agility will be seen whilst this team remains in an isolated ‘box’ inside a non-Agile organisation.

It’s clear that the role of management is essential. Can management adapt? Can they do their part? Organizational culture reflects the management’s culture and therefore management must model Agile behaviours. This also relates to the Management X versus Management Y theory, postulated by Douglas McGregor, which notes that Management X style uses carrot and stick to motivate people who are assumed to be inherently disinclined to work. Management Y theory, however, says that people are inherently challenged by, and engaged in, their job and can be trusted to work effectively by themselves. Only one of these styles, Management Y, is compatible with Agile.

For management to effectively support an Agile organisation they need to inspire, rather than direct and focus on removing obstacles to achievement. If managers can't adopt an Agile mindset and manage in a theory Y style, we can't achieve Agile transformation. Only adoption is possible. Therefore, we must be realistic about what is possible in the organisation when we embark on an Agile Transformation.

Michael Sahota points out that an Agile Transformation will fail if people don't understand what it is. He also references Martin’s Fowler’s observations that the definition of Agile has been weakened by wide usage and has lost its meaning. People understand Agile practices but not Agile principles, values and behaviours. The hype and inflated expectations from Agile has led to disillusionment as people fail to reach their expectations.

Having understood the difference between an Agile Adoption (process) and an Agile Transformation (culture) this led me to consider how we can better understand the steps that are required to achieve the latter. I can see 3 stages or aspects to this:

- Basic adoption
- Team agility
- Organizational agility

Basic Adoption

This relates to the movement to an Agile framework or process which allows a team to achieve some basic Agile approaches, including delivering effectively whilst responding to change. Depending on the framework chosen, this is fairly easy to define and understand. For example, the Scrum Guide gives a clear outline of how the Scrum framework can be adopted.

Achieving this basic adoption is likely to give teams a better way of working with some small, measurable benefits. It can be achieved quickly and easily, with the right support. The scope of this change is limited to the team level.

Team Agility

This is around the team adopting Agile principles and behaviours and is still only limited to the team level. It's the step beyond following the framework, where the team begins to inherently follow the Agile Principles, as defined alongside the Agile Manifesto.

As I began thinking about this stage in the journey towards Agility, it occurred to me that almost all of the 12 Agile Principles relate to the team level only. And it is these 12 things that really begin to make a team more Agile, because it defines behaviours and outcomes rather than just steps in a process.

Organisational Agility

This brings us to our third and final level, which we understand as Agile Transformation, related to the culture of the organisation and which is dependent upon the actions and behaviours of management. But how do we define what this is? What is it that we need in order to achieve this level of Organisational Agility?

Edgar Schein, in his book Organisational Culture and Leadership, defines 6 things which affect organizational culture:

- what leaders pay attention to, measure and control
- how leaders react to critical incidents
- how leaders allocate resources
- leadership role modelling, teaching and coaching
- allocation of rewards and status
- recruitment, selection and promotion criteria

So what, specifically, are the behaviours that we need from managers to achieve Organisational Agility? From my own experience, I began to compile a list:

- decisions are passed down to those with best information
- teams are given problems to solve, not solutions to implement
- management ask about impediments, not metrics

- teams are not asked for anything that does not help them to deliver more effectively
- recognition is given for teamwork not individual heroism
- investment is made in skills and learning
- innovation is considered part of the job
- managers focus on improving the working conditions in the organisation
- trust is actively demonstrated and is commonplace
- ‘failure’ is routinely perceived as learning

These are just a few of the examples of the kinds of behaviours that we must see from management in an Agile organisation. Only when our managers live by the Agile values, lead by example, are transparent and support an appropriate culture, can we then achieve Organisational Agility, finally leading to Agile Transformation. Unlike Agile Adoption, which can be achieved in only a matter of weeks, Agile Transformation is likely to take months or even years to achieve and, even then, is never “done” as we are constantly uncovering better ways of working in an Agile environment.

In summary, Agile adoption is likely to bring benefits to the team from working with frameworks that help them to deliver and behaviours that support them in doing this better. If we desire Agile transformation, our managers must also engage and commit to their role in creating the organisational behaviours and culture necessary for this to be successful. Without this, Agility will always be confined to the team-level but with it, we can achieve the organisational Agility that we so often pursue.

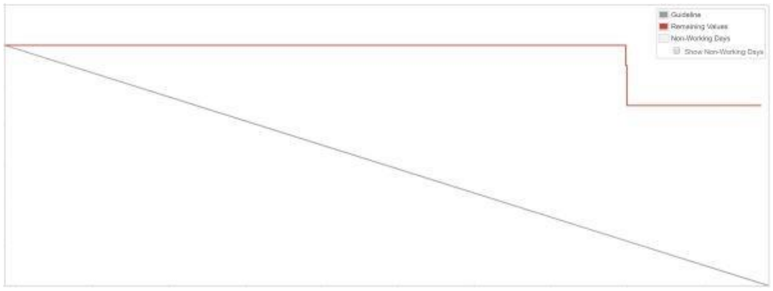
Read this article online: https://baa.tco.ac/1Sxh

Burn-downs: Insights into the Sprint

By Christine Thompson

As a Scrum Master and Agile Coach over the last 8 years, I have seen many stories that the sprint burn-down tells about our team's progress. I have even named them according to their shape, so that the teams I am working with can quickly spot which pattern they are in and why. Let me share these with you. All of them are real examples from real teams.

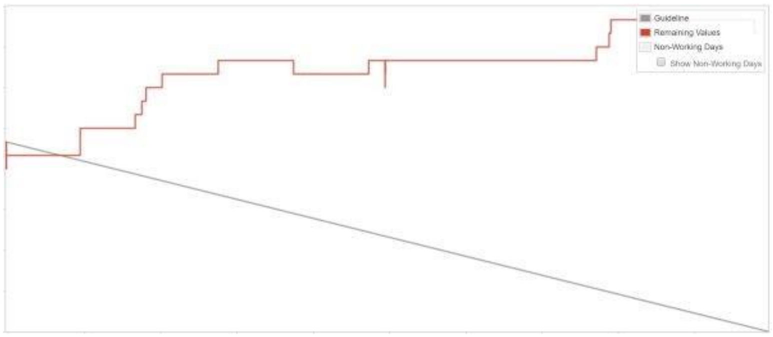
The Flat-line:



This pattern suggests that the team have done almost nothing for the whole sprint. Something has stopped the team from being able to progress their work. Perhaps they were distracted from their sprint and spent their time responding to other demands. Perhaps they brought in a small number of very large stories that they hadn't a hope of completing. Perhaps no reviews or QA was able to be done and so the stories stalled at that point.

So what are the team doing to address this? If the stories were too large, how can they adjust this next time? If they were being distracted, what by, were the distractions warranted and how can they manage this better if it occurs again? Questions for the team to ask here are going to be around either what stopped them *doing* what they planned or what stopped them *completing* what they planned for the sprint.

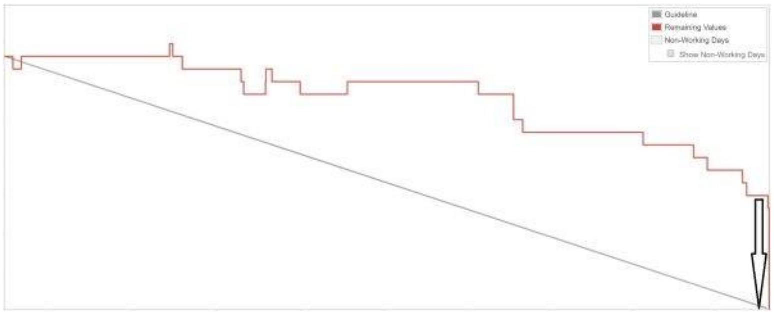
The Burn-up:



The team are adding more and more into the sprint without completing anything. Their workload goes up and up, far exceeding the capacity they committed to at the start of the sprint and creating themselves a mountain to climb.

What caused this increase in workload? Perhaps they committed to some feature work which got overtaken by urgent support tasks that took priority. Perhaps they are bringing tasks in without taking other work out. Perhaps they simply planned the wrong work. The questions for the team to ask here are around how they respond better to changing priorities in future sprints and how these should be handled in their case.

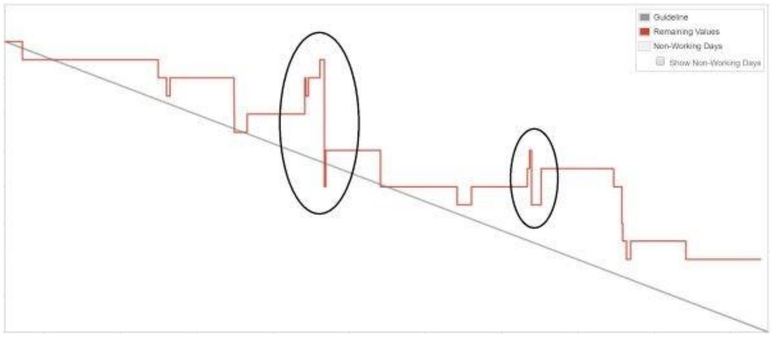
The Cliff Edge:



In this pattern, we see a significant amount of work being completed right at the end of the sprint. Rather than a gradual progression during the sprint, everything gets done at the last minute. Little value is delivered until the final moment.

There could be many reasons for this occurring and the team are going to want to explore what happened in their case. Perhaps items were blocked during the sprint, in review or QA. Perhaps the tasks in the sprint were so large that every one of them took the entire sprint to complete. I have even seen this happen when the team had ‘forgotten’ to update the board until they closed their sprint at the end. In which case, they may be asking themselves how the board serves them, if they are not using it on a regular basis. Whatever the reason, the team need to ask themselves how they can deliver value, more regularly, in smaller increments, during the course of the next sprint.

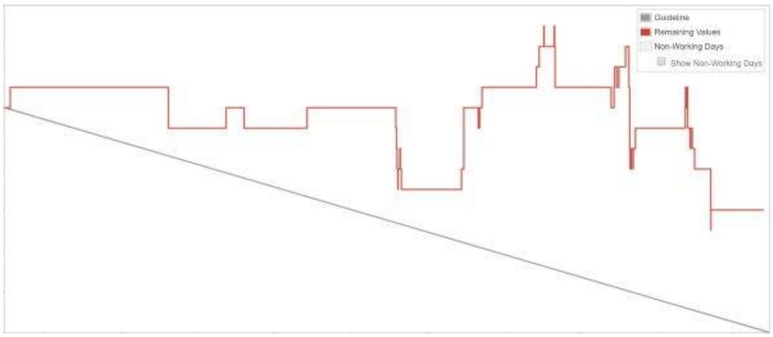
The ECG:



This pattern looks like the beats in an Electrocardiogram. Every time the team does something to burn down, they burn up too! They are making progress but what’s happening each time we get this blip? It’s like the goal post is moving and it takes a couple of attempts to get things done.

Sometimes this can be as simple as an error on the team board. Something got moved to done too soon and needed putting back before it could be properly closed off again. In which case, perhaps the team needs clarity around their Definition of Done. Alternatively, it may be an instance of the following pattern, the New York Skyline. The particular pattern in this case can be slightly easier to solve because there are fewer, specific instances for the team to investigate and understand what caused these individual blips. In each case, they can be asking themselves what happened, why, and how they can best correct for this kind of thing in future.

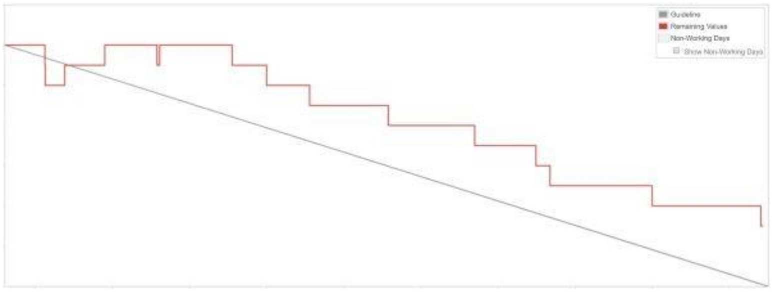
The New York Sky Line:



This is an erratic sprint that is constantly changing. The team are chasing their tails as their world shifts around them. Despite their ability to complete things, the team and their work are in constant flux. After a sprint like this, the team are probably exhausted from bouncing around all over the place.

There may be a number of factors at play that they want to look into and discuss further and they may be well aware that their efforts to commit to a sprint and maintain a steady pace were thwarted. The team will probably be asking some questions in this case about what interrupted them, what priorities changed and how they responded to this. What is paramount is that they start to identify these factors, one by one, in order to address and eliminate them. If this pattern persists, then they may need to ask themselves some higher-level questions about what framework would work best for them to manage their flow of work in future.

The Burn-down:



Finally, here we have the nirvana which is the burn-down. A steady progress by the team, throughout the sprint, as value was added, piece by piece. So there are no questions to ask here, right? Well, I can think of two things to ask the team in this instance.

Firstly, does the burn-down accurately represent their experience of the sprint. Was the sprint successful, in their opinion? Or is the burn-down masking anything that we need to consider? Just seeing a nice burn-down isn't quite enough to assume that all is well. Let's ask that question to be sure.

If the team agrees that, yes, the burn-down accurately reflects a sprint where they delivered value frequently, whilst maintaining a steady pace, and all is well in their world then we have a very important question to ask. What is it that's working well? We ask this question so that we know what it is that we want to repeat. If we can identify all those things that are helping the team to work effectively, then we can focus on doing more of these things again.

Summary

These charts are the starting point for a team to have a conversation. The six patterns give us a window on our world that helps us to ask the right questions at the right time in order to identify what's working and what's not. This can help the team to take corrective action on the problems they face, as well as repeating the positive patterns that are helping them to deliver value most effectively.

About Christine Thompson

Christine is an Agile Coach, a Team Coach and an Individual Coach.

She is committed to helping you to become the best version of yourself. She works with teams to promote an Agile mindset, and a set of behaviours resulting from that mindset, which allows them to focus on delivering value whilst being responsive to change, in an environment that is inclusive, safe and challenging to all its members.

She is an ICF Associate Certified Coach, a Scrum Alliance Certified Team Coach, an NLP Master Coach and is ORSC Trained.

Christine has a clear focus on people, coaching them to be the best they can and forming teams which are both effective and fun.

Key challenges for Agile: Confronting the bad and the ugly

By Michiel van Gerven

Agile is helping to change the future of work. Over the past few years Agile ways of working have spread over a wide range of disciplines and organizations. This happy trend brings with it some key challenges that we as an Agile community need to confront. Some of these challenges are trivial, while others may be a threat to Agile itself. This article seeks to identify these key trends and offer a glimpse of a hopeful future, but also issue the warning that Agile is in need of careful stewardship.

This article came about by talking to various Agile people; Scrum Masters, Product Owners and Agile Coaches in a number of organizations, both public and private and of course my own colleagues. What you will find below are the challenges they mentioned most, coupled with some of my own thoughts on the subject.

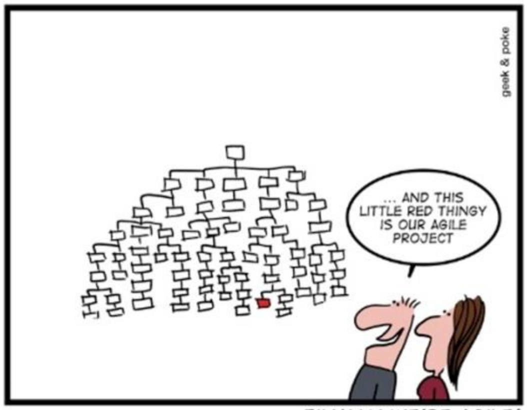
Agile In Name Only

The primary challenge is Agile In Name Only. It can manifest itself in a number of ways. The first common way is for a situation to carry the name Agile, and maybe even look Agile, when in reality it is anything but. A situation a senior Product Owner at a financial institution described to me as: '*Agile 3.0, now with extra top down.*' He was referring to a major project with multiple Agile teams that was using Agile on a day-to-day basis but was also expected to deliver the project fixed budget, fixed scope, and fixed deadline. This was in an otherwise mature Agile organization.

The second way is on the personal level. Like Prince2 before it, the certification happy nature of the Agile crowd has created an abundance of people that have received training but can't do what it says on the box. When I met Tanner Wortham in Silicon Valley in October 2017 he aptly described this as '*certificated rather than certified'*. In other words, someone who talks the talk, but can't walk the walk. Fortunately, help is available in identifying and avoiding such people.

This trend is mirrored in the fact that many organizations pretend to be more Agile than they actually are. This may be partially driven by an interesting megatrend affecting Agile today: the war on talent. Qualified people are getting scarcer and scarcer. This week for instance, I spoke to a leading Agile Coach at a well know brand that was struggling to find good scrum masters and Agile Coaches. Presenting yourself as an Agile organization may help in recruiting young people especially (not just scrum masters, coaches etc.). Therefore expect this trend to continue in 2019 and the foreseeable future.

A rather special kind of Agile In Name Only is *machine room agilism*. Many coaches report seeing this more and more as Agile starts to be adopted by follower, rather than early adopter organizations. It is personified in managers that seem to champion Agile, but only do so to serve their personal interest. While they can lever the power of Agile teams to increase innovation and development speed in their organization or department they will support the Agile movement in an organization. But as soon as they will have to change their own behavior they will drop their support. To their mind Agile is only for the production floor. This is a dangerous trend, because it weakens an Agile movement at a key turning point in a transformation.



FINALLY WE'RE AGILE!

Everybody's doing it

As it turns out, there is such a thing as a conservative Agile transformation. A type of transformation that is not driven by a true desire to 'go Agile', but one that seems to start out of a fear of lagging behind and the feeling that 'everyone is doing it'. The trouble with these is that what has worked somewhere else, will not necessarily work for you.

The banking sector for instance seem to suffer from Spotify-syndrome. Across the world we see virtual carbon copy implementations of the same model (hold your comments, I am perfectly aware the spotify model is not a model). It comes in many guises; bricks, squads and cells are all names for the same thing, an end-to-end Agile team.

Most coaches and consultants I spoke to are expecting to see more and more of these types of transformations in 2019 as Agile starts to become more and more mainstream. Such cookie-cutter transformations carry with them an inherent risk, which brings us to my next point.

Incomplete transformations and commodification

As Agile starts to become more widespread the danger of a backlash increases exponentially. Typically, an incomplete or conservative implementation (yes in this case it is usually an implementation rather than careful adoption) involves a large scale training effort, a phased roll-out of the selected Agile framework across the organization, and a steep consulting fee. Clearly there is more to transformations than rolling out a framework. Without serious consideration for what goes unspoken any transformation is doomed to fail.

Agile is fast becoming a commodity of sorts. This is clearly exemplified by the entry of large consultancy firms into the market. On the one hand they are very welcome, they have the clout to convince the upper echelons of management and their entry is a clear signal that Agile is starting to become the new normal. On the other hand they are a clear reflection of what is sometimes referred to as the Agile Industrial Complex. There is money to be made in selling Agile. Which is fine, if what is being sold is true Agile. Too often, however, selling Agile involves selling Agile frameworks or methodologies without consideration for the things that truly matter, the mindset and ideas behind the frameworks.

Inevitably, we will (start to) see failed or incomplete transformations. This will cause many to become even more skeptical towards Agile. In the past Lean has suffered from a similar backlash. In many organizations it is not uncommon to hear *'we survived Lean, we will survive this'*. The value of Lean principles has not gone away, yet the inherent inertia and resistance to change in many organizations has resulted in an unsuccessful adoption. As more organizations will try to adopt Agile and fail because they underestimate the challenges involved or simply because they want the benefits without putting in the work and making the hard choices.

Leaders are afraid to admit they don't know what Agile is

One of the most curious challenges pertains to leadership in organizations that first start out on a path towards transformation. Being a senior leader in an organization is not an easy job at the best of times. Now imagine coming face-to-face with something that goes completely against everything you have always known, something like Agile. Agile likes to turn things completely on its head, in fact that is one of my favorite things about it. However this, coupled with the fact that Agile is now trendy, makes it a difficult beast to handle for many managers. For them, it has until now been a thing that '*some of those IT teams do*'.

Many of the Agile coaches and scrum masters I spoke to signaled that senior managers are often lacking in knowledge of what Agile truly is. Moreover, they are often in the unenviable position that admitting to this doesn't feel like an option to them. Educating leadership on Agile should therefore be a key issue in 2019, especially since without them, wholesale adoption of Agile in an organization is downright impossible

Conclusion

Agile In Name Only and the commodification of Agile may be the two most dangerous trends facing Agile today as they bring more and more people into contact with a faux form of Agile. Not only does faux Agile not ask the real, hard questions, it might also put many people off Agile as it fails to deliver on its promises.

But, let's not forget there is also plenty of cause for celebration. Agile is clearly starting to move into mature territory. Adoption is more widespread than ever and Agile ways of working are steadily becoming the new standard in an ever in widening range of disciplines. In Finance, and HR especially Agile is really taking off.

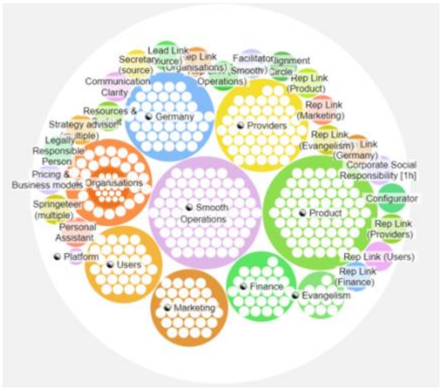
As an Agile community we should be ready to confront the challenges in this article and make sure we can continue to change the future of work for the better.

What do you see as the key challenges to Agile in 2019? Personally, I am hoping to be wrong about a few of the more depressing ones. Please feel free to share your own thoughts below so we can revisit them later during the year.

Read this article online: https://baa.tco.ac/1Syp

Experiments with Holacracy: Why we stopped doing it, and what we learned along the way

By Michiel Van Gerven



My colleague Minke's recent article 'Getting started as an employee experience chief' triggered some questions about why we at Organize Agile abandoned Holacracy. It's not an easy question to answer and while I have my own thoughts on the subject I also asked my coworkers to offer their opinions. This article will offer some insights from our journey, what Holacracy did (and didn't do) for us, and some recommendations should you want to start your own Holacracy experiment.

What is Holacracy anyways?

If you're unfamiliar with Holacracy, I like to describe it as a way to constantly reorganize and refactor your organization. While it might seem terrible to work in an organization that is constantly reorganizing itself, the point of course is that by constantly changing, change becomes an incremental process rather than occasional and infrequent shock therapy. Ideally, Holacracy makes organizational change easy and helps to ensure your organization is always in tune with what outside forces and the people in the organization demand of it.

Key to this is the concept of a 'tension'; something in the organization that isn't quite right, could be better, or is simply missing altogether or superfluous. Anyone can signal a tension and can offer a proposal on how to address the issue. Such a solution might be to assign a new responsibility to a pre-existing role, create a new role, or change an organization policy. To discuss and decide on tensions and solutions Holacracy uses clearly structured governance- and tactical meetings.

Roles are not the same thing as job descriptions, rather they are flexible things that can be created, adapted or disbanded according to the organizations needs. Similar roles or roles that contribute to a shared purpose are grouped together in a 'circle'. An organization is therefore likely to consist of one main circle with several subcircles. People can have multiple roles at any given time, meaning they should be able to use all their talents and carry all manner of responsibilities, rather than just do what is in their job description. We tried to keep things pretty light-hearted. Minke's quite seriously named Employee Experience Chief role was contrasted by an IT Troll (one of my roles, basically responsible for making sure we had the right hardware such as phones and laptops), and a Metrics Monster (One of Mikel's roles responsible for helping us work in a Data informed way).

Crucially, Holacracy is also about distribution of power. To do 'pure Holacracy' is to have the owners of a company relinquish their power over it by signing a 'constitution'. It therefore goes beyond self organization into the realm of self-management.

Should you want to read more about Holacracy, ’Holacracy’ by Brian Robertson and 'Getting teams done' by Diederick Janse and Marco Bogers (In Dutch) are good places to start, although I found 'Holacracy' a pretty tough read. Alternatively, there is the Holacracy blog.

What did Holacracy ever do for us?

I'm not out to bash Holacracy. We stopped doing it, but before we did, it helped us do many things. So to paraphrase Monty Python: What did Holacracy ever do for us?

Monty Python discussing the merits of Holacracy circa 30 AD.

Like Agile, Holacracy doesn’t solve all your problems, but it is pretty good at making them visible. Holacracy’s meeting structure and the concept of tensions helped us to **bring issues to the surface** and forced us to have a conversation about them, even when sometimes maybe we didn’t really want to.

Some of the things it forced us to discuss where our company values, strategies and policies. Any mismatch between them would often result in a **‘tension**’ which we would then have to resolve. That wasn’t always easy but at least we were actively discussing them. We also found many of our company policies were implicit. ‘Everybody knows this, right?’ Making our policies explicit definitely raised the quality of our work and helped us to build a culture by doing certain things the same way.

Explicit policies also proved valuable when on-boarding new people, it gave them (including me) something to fall back on. Please note, our entire policy book consists of a thirty row excel sheet, so it’s pretty lightweight.

The Holacracy meeting structure also allowed us to **involve everyone** in our governance process. We’re not a large organization but it is still nice to be able to use everyone’s knowledge and experience when dealing with something. Holaractic meetings can give everyone a voice.

When our **strategy was in flux** we were constantly able to adapt the organization to this. Don’t get me wrong, this still hurt, but it gave us a mechanism for clearing up organizational debt.

Finally, we never became **stagnant.** My favorite thing about Holacracy is that it’s an anti-consensus model. Rather than not deciding anything Holacracy helped us make a decision and experiment with a solution to a tension. If a solution proved ineffective we could always bring up a new tension next week (and we often did).

So in fairness, Holacracy gave us lots. So why did we abandon it?

Why did we stop?

There is not a single reason why we stopped doing Holacracy, rather there are many reasons both large and small which added up to the disadvantages outweighing the benefits. These disadvantages were not necessarily caused by Holacracy, rather they are what we experienced when we used it.

By using Holacracy we sometimes ended up trying to address an issue by assigning a new responsibility to a role or changing a company policy. Sometimes there is another way; simply **solve the problem**. Holacracy does allow ‘one-off actions’ and you can ask a role to pick up an issue but having a framework for discussing tensions in place sometimes resulted in waiting for the next governance meeting to address something when it could have been solved well before then. In other words in some cases we allowed Holacracy’s framework to limit our thinking in coming up with the right solution.

One such solution we tended to forget about was **leveraged assets**, meaning simply hire someone or some organization to do the job for us. We moved certain responsibilities between roles (even several times), or sometimes created new ones to solve problems that would have been easier to solve by outsourcing rather than by trying to finetune our internal division of responsibilities.

In Holacracy tensions are dealt with by proposals that can create or adapt roles or policies. If there are objections to such a proposal these need to be validated and if proven valid they need to be taken into account. In our organization **the validation process created its own tensions**. A colleague would have an objection to a proposal but would know from experience this would not be a valid objection. This resulted in people not even bringing up these objections, or already mentioning that they would not be valid. It didn't make for a very constructive debates and was a cause for some rumbling in the undercurrent of our organization.

Holacracy is also about how power is divided in an organization. It can energize an organization by giving individuals the power and mandate to act. However, we sometimes ran into friction when a role would clash with **the *implicit* hierarchy** in our company. It is all very well to put to paper what role is responsible for what but at some point that can clash with what founders (who still own the company) feel the organization should do. This can be partially tackled by giving them the Lead link role, but it doesn't solve everything. Distribution of power is not just about (explicit) power structures, but also what's going on beneath the surface.

A key thing my colleague Jens mentioned is the nature of our company. We are a consultancy company, and spend most of our time with our clients. Friday is the only day we all always spend at our own office, in order to work on our shared backlog, share knowledge and improve as an organization. As such governance and tactical meetings were quite a drain on our time. Our time together is very precious to us and we do not take kindly to unnecessary overhead.

Last but not least. We are a company of Agile Coaches, and it shows. Meaning, we tend to have an opinion on pretty much anything, especially on anything in the realms of process, governance and decision making. Our **meetings sometimes devolved into soul-sucking discussions**. Especially when people felt strongly about issues, and personal tensions might get mixed up with professional ones or elevated to the level of team tensions. I have facilitated my fair share of our tactical and governance meetings and some of those rank among the most difficult sessions I have ever facilitated. I tried following both the process to the letter and being more lenient with the rules, both created issues. It was very hard to get right.

I am fully aware that most of the things above were at least partially our own mistakes, but here's the thing: Holacracy as a framework should have helped us make our lives, and organizational governance, easier. It did not. Holacracy is anything but simple, and the holacratic road is riddled with pitfalls. Maybe we simply were not good enough, but I believe there are simpler ways to achieve what we set out to do with Holacracy, one of them is organizing ourselves into Agile teams and working from a shared backlog.

How did we stop?

We didn't just stop doing Holacracy overnight. In fact, for a long time we weren't even sure whether we had actually stopped doing it. The roles were still there, but we stopped having regular tactical and governance meetings. Basically Holacracy was slowly bleeding out.

We found for instance that administrative roles, or components thereof had all ended up with our Office Manager. Theoretically we could still move these roles to another person, but we never did. In other words, these roles had become waste because in reality they corresponded to a job description.

It took us a long time to make the decision explicit but eventually we decided to formalize what in reality we stopped doing for a while.

What are we doing now?

We still use roles in our company, but there are far fewer than there used to be. We love the flexibility roles offer, but not everything should be a role. For instance, Minke, one of our Agile coaches, has taken on the role of Employee Experience Chief, but most of our colleagues no longer have 'extra' roles besides their regular job of Agile Coach.

We have learned from our mistakes and have explicitly given Minke time to fulfill her role. Also we will evaluate her performance in this role on a set date. Not to judge her, but to help her and our organization improve.

In doing so we have also shifted towards Sociocracy 3.0. Where Holacracy felt very formalistic this feels more human to us. It allows more space for discussion but isn't soft.

One of the key concepts that allows us to keep our momentum is 'good enough for now, safe enough to try'. Instead of wanting to get everything right, this helps us to take a more experimental approach. Something that is close to our hearts, but was sometimes lost along the way.

Putting responsibilities in roles meant work was often done by individuals. As mentioned elsewhere we now use our own stable agile teams to do our development and maintenance work in a structure that could be described as a very lightweight version of LeSS.

Combining elements of Sociocracy 3.0 and stable agile teams allows us to have the benefits of flexible roles, but also simply get stuff done. We still have all sorts of discussions, but the focus has returned to delivering value.

Conclusions, and is Holacracy right for your Organization?

I still like Holacracy, my favorite bit about it is that it is basically an anti-consensus model. It allowed us to be a different organization pretty much every week and experiment rather than get stuck merely talking. 'Doing' holacracy is definitely not easy though and bits of it feel overly restrictive to me. Conceptually it is very strong, but execution is hard and to my mind many of its goals can be achieved in simpler ways.

In some ways Holacracy reminds me of yoga. Can yoga help you become more limber? Maybe. Probably. But quite possibly yoga is suited most to people who are already very limber. In the same vein Holacracy can help you become more agile and responsive to change, however it is probably best implemented by organizations that are already very agile. Starting to do Holacracy in an organization that is not ready for it is likely to result in accidents.

Read this article online: https://baa.tco.ac/1SyN

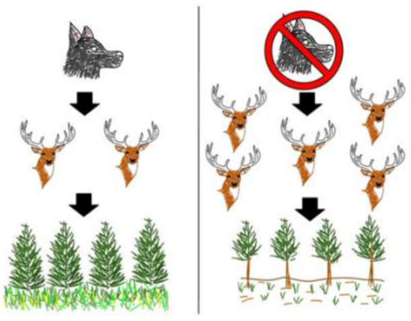
Why true agile transformation requires apex predators

By Michiel Van Gerven

Most of you have probably seen the video 'How wolves change rivers' (it's embedded below). It famously describes how the reintroduction of wolves into Yellowstone national park changes the entire ecosystem. It triggers changes in deer behavior, which in turn allows vegetation to regrow, which eventually even changes the course of the rivers in the park. It is a powerful story of how you can make a small change and have a big impact. Or is it? It has certainly been used that way. Simply start working in an agile way in a few teams and watch the organization change around you as a result. If that were the truth, then why is true agile transformation so hard?

Let’s watch that video again. Crucially, it describes the reintroduction of *wolves*. It is quite unlikely that the introduction of squirrels, even in quite large numbers, would have had the same effect. **Not all small changes are created equally.**

In other words, making a small but significant change **at the top of the food chain** can change the entire ecosystem. This is what biologists call *trophic cascades*. So how does this translate to the work environment? Making a small but significant change *at the right level* can change the entire organizational system and culture. It can change the behavior of local deer, diversify the flora and fauna, and reduce erosion. The right behavior at the top – setting the right example – will change the behavior of the entire organization, will make your organization more attractive to a more diverse workforce by being a great place to work, and thus improve employee retention.


Source: The City University of New York

Fundamentally, what *ultimately* changes the ecosystem is different behavior by the deer, who start to graze in different areas. But what *triggers* the change in this behavior is the wolves. You simply can't expect all the deer to change their behavior without a catalyst. Organizational agile transformation is the same.

What does this mean for agile transformations?

Many agile transformations start out as grassroots movements that gradually attempt to scale the organizational ladder. The movement is admirable, but it lacks what is ultimately required to make wholesale significant change – the deer can't change the ecosystem by themselves. The simple truth is that the grassroots movement rarely includes the most significant apex predators in the organization. Agile transformation cannot succeed without leadership support, but more importantly *changes in behavior* from those in the position to truly influence the behavior of mammals down the food chain.

One of the ways the wolves in Yellowstone National Park have helped to change their environment is by rewarding and discouraging certain behavior. Driving deer from valleys where they are vulnerable is analogous to setting clear boundaries, changing decision making processes and reward structures. For the deer to change their behavior they need to see the value in doing so. In effect the deer in Yellowstone have self-organized towards the areas that deliver the most value. Leaders in organizations can do the same by setting clear boundaries, funding and supporting the right initiatives but at the same time affording teams the freedom to self-organize around the right areas.

Crucially wolves, like leaders, give new life to an ecosystem exactly because they kill coyotes that prey on the organization. In that sense, they clear unsuitable projects and initiatives that take up considerable resources but have no place in the value/food chain.

Wolves not dinosaurs

So what does the organizational apex predator look like? It is not necessarily the loudest, the largest or the most dangerous individual. It is not your typical organizational carnivorous dinosaur. Large, powerful, extremely dangerous, but slow to change its mind. Let’s consider the wolf more closely. It is a *social*, *engaged* and *intelligent team player*. It is the individuals, or rather close knit packs of such individuals that have the power to change the ecosystem in the organization.

So who might this include? Certainly the top management in an organization comes to mind. But how do they operate? Is it a true leadership *team*? Do they serve to optimize *the whole* or do they seek to maximize their own interests? Do they engage in constant infighting and turf wars? Is their goal to strengthen their own position or do they put the needs of the pack above their own? Like the wolf pack a leadership team cannot function well – and ultimately will not survive – if its members go it alone. Like the wolves in Yellowstone National Park, top organizational leadership and its behavior have a massive impact on organizations. Conflict between directors of different departments is reflected in proxy wars between those departments themselves. Ultimately, it slows down the whole organization. Conversely, setting the right example can energize and streamline an entire organization.

Apex predators can also be found outside the ranks of top leadership. Crucially these informal leaders will still need to have the clout to truly impact the organization. Someone leading a highly visible strategic initiative can still have a strong impact, but it is hard to truly influence the center from the periphery.

Conclusion

You cannot change an ecosystem by only working with squirrels and deer. Small changes will not result in large scale transformation unless they are made in the right place, by the right people. True lasting change requires introducing or changing apex predators. Organizational transformation requires changes at the top of the food chain, be it from formal or informal (but influential) leaders. The good news is changes at the top can be introduced to the top of the organizational food chain. A new CEO for instance, who ‘get’s it’.

Another way to reorganize the food chain is to change the existing top dinosaurs into a cooperative, social, intelligent wolfpack. That starts with awareness amongst leaders that they are in fact influencing an ecosystem instead of a machine. Which is why ecosystem thinking is a central theme in our Agile Leadership training.

Like organizational change, forging a top wolf pack can be a hard road, but if you want to change the course of the river, that’s where you need to start.

Finally, please remember ecosystems don't change overnight. The reintroduction of wolves into Yellowstone National Park eventually resulted in a quintupled average tree height, but it still took time for the trees to grow.

About Michiel van Gerven

Michiel is an Agile Coach, Consultant and Trainer and international speaker at Organize Agile and Scrum Company, based in the Netherlands.

He has a penchant for Agile in unconventional environments and likes a challenging organizational puzzle.

Currently he is involved as the Lead Transformation Coach at an international insurance agency and coaches, trains and advises the Dutch National Police in adopting agile ways of working and thinking.

Michiel's personal motto is 'This shit could be so much better!'

These are the 5 benefits of Scrum

By Saskia Vermeer-Ooms

Lately, I have been giving several Scrum training sessions, mostly in-house to non-IT folks. When I start explaining the basics of the Scrum framework to them, they eagerly want to hear examples and benefits of using Scrum. It always helps to have a good story ready, this blog contains five examples of the benefits.

1. Create Focus during the Sprint Planning

Usually, when I start working with a new team I ask them to put all the activities they are currently doing onto post-its. As soon as all post-its are on the wall, the conclusion is that they are working on a lot of things. People tend to promise to accomplish multiple things at the same time and then start working on five things simultaneously. This results in getting these five things half way done instead of finishing at least one of them. In other words, there is no focus.

Scrum uses a time-boxed period, called a Sprint, for example, a period of 2 weeks. At the start of the Sprint, during the Sprint planning, the team needs to decide what their Sprint goal is and how they are going to accomplish it within the timeframe. Yes, all things on the Product backlog are important, but as a Scrum team we must choose the items which will satisfy our Sprint goal at the end of the sprint. The Sprint backlog will make it transparent what we are working on and show us that we are focusing on our current Sprint goal. At the end of the Sprint we have a “Done” increment, which meets the Scrum team’s Definition of Done. This means there are no more loose ends to tie up and we can move our focus to a new goal to work on in the next Sprint.

2. Courage to work on tough problems during the Sprint

During the Sprint we work on the items from the Sprint backlog. This is usually where the hard part starts. We don’t have all the answers to how to tackle unforeseen problems and there are no written out plans to follow, so we need to figure it out as we go along. A big dose of courage is expected from all team members. The courage to take risks needed to solve tough problems, but also courage to ask questions or to admit you are in need of help from others. A Scrum team is self organizing, which means there is no manager from the outside telling them what and how to do it. The team itself needs to figure this out and organize as they see best. Yes, this may result in making decisions that end up wrong in the end. But this process helps them learn to do it better.

3. Make Commitment transparent during the Daily Scrum

Are you in or are you out? When working with other team members it is always good to know if they are equally dedicated to the task at hand. Ideally a Scrum team consists of team members who are all fully available to accomplish the work together. The key thing is that you have volunteered your time and attention to this task. You need to have a sense of ‘we are all in this together’. If you are only there to give comments and suggestions, but won’t be there to do the work, you will not be perceived as a committed team member. This usually manifests itself during the Daily Scrum, where we discuss with each other if we are still on track to achieve the Sprint Goal. Commitment becomes transparent by personally asking each other what their contribution to the Sprint Goal will be today.

4. Openness about challenges and uncertainties at the Sprint Review

At the end of the Sprint there is the Sprint Review. This is where team members share the increments which have been completed during the Sprint and stakeholders and/or end users are able to give their feedback.

The visibility of the product and progress of the work being done ensures that risks can be tackled early. Openness about challenges and uncertainties that team members encountered during the Sprint increases trust amongst everyone involved. What you see is what you get, there are no hidden agendas and no need to hide things. It’s not always possible, but I encourage teams to ask for feedback from stakeholders as soon as the work is done. The feedback loop happens throughout the Sprint and at the Sprint Review there are no surprises.

5. Effectiveness of the Sprint Retrospective is based on Respect

Finally, at the end of the Sprint we have the Sprint Retrospective. This is the event where the team inspects itself and finds solutions to improve the way of working. At the start of the event, I like starting out by stating the Retrospective Prime Directive by Norm Kerth:

> *Regardless of what we discover, we understand and truly believe that everyone did the best job they could, given what they knew at the time, their skills and abilities, the resources available, and the situation at hand.*

Basically, what it says is that as team members, we respect each other. There is no one to blame and everyone deserves to be heard, regardless of background or expertise.¶A team can only flourish when individual members flourish. Individuals need to feel respected for who they are, so they are able to increase their effectiveness within the team.

The Scrum values

Have you noticed the 5 benefits as described above not only fit nicely in the five Scrum events, but also in the five Scrum values? In an attempt to make these Scrum values more tangible within the Scrum events, I’ve attached each Scrum value to an event. Here is how:

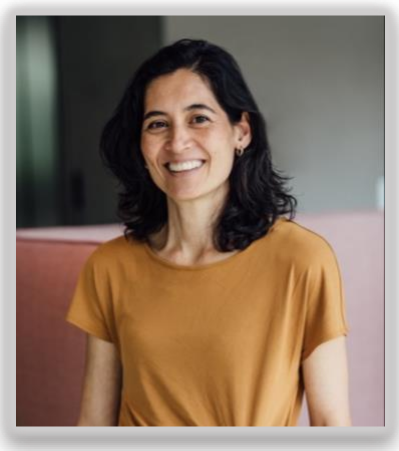
- Sprint Planning → Focus: Everyone focuses on the work of the Sprint and the goals of the Scrum Team
- Sprint → Courage: Scrum Team members have courage to do the right thing and work on tough problems
- Daily Scrum → Commitment: People personally commit to achieving the goals of the Scrum Team
- Sprint Review → Openness: The Scrum Team and its stakeholders agree to be open about all the work and the challenges with performing the work
- Sprint Retrospective → Respect: Scrum Team members respect each other to be capable, independent people

SCRUM VALUES
RESPECT
COMMITMENT
FOCUS
COURAGE
OPENNESS

Although they have been added to the Scrum guide in 2016, people still tend to skip over these Scrum values. That's a shame, because understanding what these values mean in practice makes it easier to embed the rest of the Scrum framework. Making things work within your team is not only about understanding the definition of Scrum. In fact, it should start by understanding what these Scrum values mean to you and how you can apply them to becoming an effective team member and team.

Read this article online: https://baa.tco.ac/1SyY

About Saskia Vermeer-Ooms



Saskia started her career as a developer and has worked in many different fields.

In 2012 she became a Scrum Master and has never let go of the role since. She has also ventured out to non-IT environments to help improve teams in their way of working.

With a smile and her enthusiasm she creates the right environment for teams to excel in what they do best.

On Product Manager in the Team

By Bas Vodde and Craig Larman

On Product Manager in the Team

This post is the second in a series related to product management and product teams. Like the previous post, this one is partly influenced by Marty Cagan's article on "Product Teams" vs "Feature Teams". Marty seems to see the role of Product Manager different than we do, but the difference is probably more about the team than about the Product Manager. That said, this post is especially influenced by the frequent recurring question, "Should a Product Manager be on the Team?"

Being on the Team in LeSS

First, let us clarify that a little bit by exploring what "part of the Team" means in LeSS.

The Team has a *shared responsibility* for the result of a Sprint. A shared responsibility means that, even though people on the team probably have a primary specialization and a preferred focus area (which may not be the same, because of learning), *all* of the members of the team are responsible for *all* the work that the team does. In practice, this results in vague boundaries between the individual team members primary specialization, and in some teams not even a meaningful distinction.

A Team in LeSS (and Scrum) is also *self-managing*. What's the implication of that on a team taking a shared responsibility? Key point: the team will need to have (1) **a clear shared goal**, (2) **at the same time**. We want to stress the *temporal* aspect here as it is rarely discussed but really important! When a new self-managing team is formed in which previously the team members were used to traditional siloed single-function roles, then if they continue that pattern, the team **will** struggle with finding ways to *parallelize* their work. The first and easiest questions is, how can we test *before* the implementation is done? Then, the same question needs to be answered related to the other roles that used to work in a sequential lifecycle with handoff, "How can we do the

UI design *at the same time* as the implementation?”, “How can we do the analysis *at the same time* as the implementation”.

The fundamental problem that self-managing teams with shared responsibility have to solve is: How can we together in parallel work on activities that were previously considered sequential?

What would it mean if the Product Manager were part of the team?

When a Product Manager is part of the team, it will mean that *the Product Management responsibilities are a shared responsibility within the team*. However, one problem to solve here is that a significant amount of Product Management activities usually do not – or even can’t – happen at the same time as the activities of the rest of the Team for the selected work in the Sprint. This is mostly because of the delayed feedback of some of these activities. For example, if a Product Manager is out observing users or talking with them, then this activity is usually not at the same time and for the same goal as what the team is working on right now.

Just to be clear, we are not saying that it’s good that Product Managers don’t work on the same goals at the same time as the Team, we are simply observing that this is often so, and that it’s hard to change that in the context of the customers they work with and the organization they work in.

Considering this, we have seen two common alternatives of Product Management working with the team:

Product Manager as part of the team

This means the whole team takes on Product Management responsibilities (not just the person with Product Management preference). The team together discusses how to get closer to customers, understand them better, emphasize better, and discover potential new outcomes to achieve or new features to create.

It is likely to happen that the Product Management activities do not directly relate directly to the other work selected in the Sprint. In that case, the team needs to cope with the temporal differences, the work selected in the Sprint and the Product Management responsibilities related to understanding the customer, without creating separate roles within the team. For example, th *entire* team gets together at the beginning of the Sprint (Sprint Planning 2) to understand the Product Management activities and decides how they together are going to work on these.

Note that there is a similarity and overlap between Product Management activities and Product Backlog Refinement. Product Backlog Refinement is similar look-ahead work. A difference is that it is often possible to do Product Backlog Refinement work with the entire team together while it might not be feasible to always send the entire team to the customer (though maybe that's not a bad idea…)

Product Manager who works with a team

Although in LeSS we love the Product Manager – and hence *Product Management responsibility* – to be part of the team, in practice we often see a Product Manager working *with* teams. In this case, the Product Manager focuses on Product Management activities (such as competitor and market analysis, and much more) and helps the teams to understand the market, customer, and problems to solve. Sometimes a Product Manager might work very close with one or several teams for a long time when the team works with customers or features that they know most about. However, in that case the Product Management responsibilities are not (yet) fully shared within the team.

In some cases, this is a first step to eventually moving the Product Management responsibilities into the team. In other cases, that won't happen because (1) the focus of the team changes whereas the focus of the Product Management might not (e.g. in case the Product Manager is focused on a *particular customer or customer segment*), or (2) the Product Manager's responsibility is intrinsically hard to share in the team, which might be the case when there's lots of travel and customer relationships that needs to be maintained.

Conclusion

So, should the Product Manager be a part of the team? Which also means, should Product Management be a shared responsibility in the team? If you can do that, great! Alternatively, Product Managers can work with the team to help them understand the customers better.

On the Product Owner Role

Final note: We are talking about Product Management here and *not* the role of Product Owner. In LeSS, the Product Owner focuses on overall Product vision, prioritization, and investment decisions. In product companies, the Product Owner tends to be one senior person from a Product Management group, but not all Product Managers will be Product Owners.

Read this article online: https://baa.tco.ac/1SxP

About the Co-Authors

Bas Vodde is a coach, programmer, trainer, and author related to modern product development. He is the creator of the LeSS framework for scaling agile development and author of three books. He is an active developer as he loves that and strongly believes that you can't be fast and flexible without a well-factored codebase. He is the maintainer of several open source projects, including CppUTest, a unit test framework for C++.

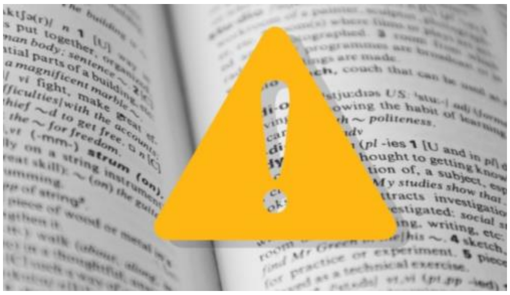
Bas works for Odd-e, a company which supports organizations in improving their product development. He currently lives in the Netherlands and can be reached through www.less.works.

Craig Larman works as both a programmer -- most recently on a self-driving car -- and as a coach in learning to think about and experiment with organizational designs.

With his friend and colleague Bas Vodde they help NOT scaling, but help *de-scaling* organizations for more with LeSS. www.less.works

Name Your Trigger – Team Vulnerability Activity

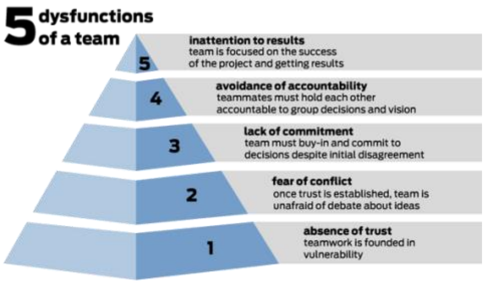
By Natalie Warnert

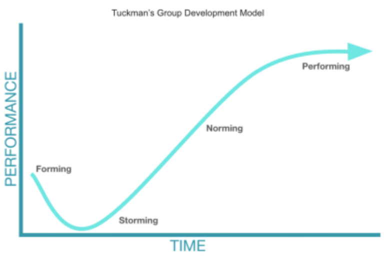


In order to build trust, we need to show vulnerability, we need to have trying experiences together, and maybe even get into trouble together (that is what my dad says about why my brother is still close to his high school friends).

Vulnerability, however is hard to produce in a work environment without a real moment that causes it. We are taught that emotions are not supposed to be in the workplace, but let's face it, we're emotional people. We work for huge parts of our day and sometimes see our co-workers more than our families. So doesn't that mean we should have plenty of opportunities to build trust? Most of the time new teams, or even teams that have been together for a long time, that trust can be really hard to build or regain if lost.

We have all heard of Patrick Lencioni's 5 Dysfunctions of a team and Tuckman's Team and Group Development Model (and if you haven't, here they are). Teams need to have trust, but to get to trust, they need to get through Tuckman's model. Starting out in forming, teams have nothing in common, not much history, and are often on their best behavior. It's often when they move into storming that the issues start (hence the name), but storming is a vital stage in building trust.


5 dysfunctions of a team
5 inattention to results
team is focused on the success of the project and getting results
4 avoidance of accountability
teammates must hold each other accountable to group decisions and vision
3 lack of commitment
team must buy-in and commit to decisions despite initial disagreement
2 fear of conflict
once trust is established, team is unafraid of debate about ideas
1 absence of trust
teamwork is founded in vulnerability



There is an activity that I have tried a few times in order to form a Minimum Viable Relationship (MVR – copyright Natalie Warnert and Jenny Tarwater – Agile2018) and to start building trust. Even if it doesn't work at least we come out with a few ideas of how the others operate that make working together easier to navigate. It's called **Name Your Trigger** and here is how it works. Everyone answers the following three questions to the group:

- **What is one emotional trigger you have at work?**
- **How do you react when you are triggered?**
- **Why you think it's a trigger?**

It seems simple but it isn't. It actually extremely awkward to talk about a trigger and reaction with other people. It's vulnerable. We've all had the interview question about our weaknesses and we've all tried to make them look better. But a trigger? This is about being real, helping others to understand what we're like at our worst, and explaining why (often an emotional issue) it happens from the root or source.

Here's an example of mine. An emotional trigger I have at work is when people misspell things in emails or have poor grammar because they are being sloppy or lazy. I react by being passive aggressive, making faces or comments (almost involuntarily as I do in many other situations) when I read it, and by looking at that person in a worse light. Unprofessional writing is a trigger for me because when I was a child I struggled to keep up in reading and was put in a special class for a few months. It was difficult to be below average in the age of millennial children being told we were all the smartest. Even though I got past that in elementary school, I had a similar experience with a graduate school professor criticizing my writing and passive voice on my thesis. I truly went back to that childlike state of feeling unintelligent and like I could not do anything right. This is why I trigger hard on people not taking the time to re-read their emails and making small errors because I have had to try so hard to overcome it

Telling these stories show others that we are real people and we let them into something that is close to us. I originally got this idea of vulnerability and trigger stories from The Advantage (Patrick Lencioni), but many others have also coined similar ideas. Adam Grant, the host of the *Work Life* podcast says this in one of my favorite episodes, "How to Trust People You Don't Like":

"Most of us assume that you have to build trust in order to be vulnerable. But actually, the opposite is true.

Our intuition has it backwards. You do not build trust in order to be vulnerable. When you're vulnerable, it builds trust. Being vulnerable together builds closeness and creates it."

Part two of the activity follows with a set of more questions and action planning to hold ourselves and others accountable for our triggers. It's another round of vulnerability that can help to bring the group together.

- **Identify the underlying reasons for your trigger**
- **Understand and recognize your right to your trigger**
- **Understand and recognize that people who have different triggers or who trigger you have valid feelings and reasons**
- **Understand how this trigger influences your behaviors and interactions**
- **How will you remain self-aware of this trigger?**

This allows the group to co-create action plans and establish more vulnerability while giving permission and trusting others to help. It also establishes a feeling of ownership which is important. If you own that you have the right to something and have control over it, it feels less shameful. To own something is to take the power away from it and you do that by understanding how it influences you and remaining self-aware (with others helping, too). This begins the process into team norming and even performing.

Please feel free to try this activity as an exercise to bring teams and groups closer together and to start seeing each other as humans. It’s been proven especially helpful when a team is stuck in a storming state and it can help get them over the hump into norming. As Adam Grant says: “Building trust requires risk.   Putting yourself out there isn’t always safe.   But sometimes, the riskiest move is to never take a risk **at all. “**

Read this article online: https://baa.tco.ac/1SyT

About Natalie Warnert

Natalie is founder, president and executive director of Women in Agile Inc., a nonprofit organization which enables diversity and inclusion in the Agile communities worldwide.

She is passionate about guiding companies as they design, execute, and support their approach to cultural change as an independent consultant.

Her vast thought leadership experience is recognized by her numerous keynotes, podcasts, and publications including being a SAFe framework contributor and creator of the UX Runway.

Natalie has earned various accolades including her Master of Arts in Organizational Leadership, SAFe Program Consultant, Certified Scrum Professional, and Six Sigma Yellow Belt.

To learn more visit

www.nataliewarnert.com

www.womeninagile.org

Lessons From An Aircraft Carrier

By Robert Weidner

The aircraft carrier displaced 94,000 tons of water as it separated the Atlantic Ocean in two halves, going far in excess of the 30 knots where the top speed becomes classified information. We went as fast as we could, and then turned as sharp as we could, to see if the ship would break. A torrent of water roared over the forward, top-side of our steel coffin. We all held on to some fixture that was welded or bolted firmly in place. This was a break from the norm... from our day-to-day tasks of swabbing the deck and running drills. This was the adventure we had signed up for.

Massive waves lapped over the bow of the ship, which looked to be sinking into the depths, before suddenly careening back up and out. Later, we would be informed that a whale had crossed our path during these high-speed maneuvers. The ship survived, the whale did not. Imagine a ship the size of the Empire State Building traveling faster than any speedboat you've ever ridden in, thrust forward by two nuclear reactors.

This was builder sea trials. We were fresh off a 10-year build cycle. The United States Navy was now conducting user acceptance testing prior to taking ownership of the vessel from Newport News Shipbuilding, after which it would be commissioned a United States Ship. We spent two weeks doing everything in our power to demonstrate that the ship wouldn't be ready for the perils of war. But it was, and so were we, the first "plankowner" crew of the USS Harry S. Truman.

I remember the proud day when President Bill Clinton gave the keynote address at our commissioning ceremony. I had met him only a few hours earlier, crossing paths with his security detail in the hangar bay. I was in my dress white uniform, dispatched as a member of the working party assigned to carry the camera equipment for the event. It was a hot day – July 25, 1998 – and the anticipation was months in the making. Receiving orders to a pre-commissioned unit meant arduous duty, and arduous it had been. But our moment had finally arrived.

Six months later, we would return to the shipyard, where the Truman would be gutted and the rebuilding process would begin. We had not been bombed, either by enemy or friendly fire. We did not suffer a hull breach, combustion, radiation leak, or some other catastrophic failure. We did not have a hurricane ravage our ship and pin it to side of the pier, which is why naval units will sortie during inclement weather. No, the cause of our demise was none of these. Instead, we fell victim to bad planning.

A Nimitz-class aircraft carrier such as the Truman is built to specification, over the course of a decade. The shipbuilders had been contractually bound to requirements that were defined before the real work ever began. Progress was tracked in accordance with the iron triangle. Risk was mitigated and change requests minimized. The ship was delivered on time, seemingly within budget, and met the original, locked-in scope. The Project Management Institute would have considered this project a success. The real question however, is whether or not the ship delivered the value that was intended to the customer. Did the ship meet the market need it was trying to solve?

Sure, the Truman floated. We could go fast and turn at high speed. But could we fight in a war? Could we lead a battle group? As it turns out, we could barely communicate. Our equipment was outdated. The technology had advanced far beyond our capabilities. We didn’t even have a proper level of encryption.

The United States Navy accepted the ship, while knowing they would soon return to the shipyard for a complete overhaul. Throughout the next six months, every piece of electronic equipment onboard the ship was torn out and replaced. These additional enhancements would come at a significant cost to taxpayers, but they wouldn't be considered a budget overrun. It was just the cost of how we did business. I remember thinking at the time, *there must be a better way.*

In 2001, 17 pioneers met at a three-day retreat in Snowbird, Utah, for some skiing. And to summarize a different way of designing products. Many of them were computer scientists, and had spent considerable time studying game theory. They recognized the current waterfall process for managing projects was based on the *contract game*, which uses predictive planning techniques, punishes change, limits transparency, and results in CYA behaviors. This group of experts wanted to shift from the *contract game* to the *cooperative game*, where the product development team uses adaptive planning techniques, embraces change, operates with radical transparency, and works together with their customers to evolve a product that will truly meet their needs.

These visionaries applied the umbrella term "Agile" to the sub-set of frameworks that used adaptive planning techniques and codified the values and principles in the Agile Manifesto. These frameworks are rooted in empirical process control, which has three pillars: transparency, inspection, and adaptation. If the state of the work is transparent, then the product can be adequately inspected. Based on the insight garnered from inspection, we can then choose to adapt the plan in accordance with what we've discovered. This method of validated learning helps the product emerge in lock-step with the market it's being designed for, even if the market conditions should shift over time.

Agile isn't just a revolution. It's a response to a world full of volatility, uncertainty, complexity, and ambiguity… one that is spinning ever faster and in more directions than ever before. With the accelerating pace of technological innovation, disruption can come from anywhere. Business agility is how we embrace change to sense and respond to these emergent conditions.

The cost to build the USS Harry S. Truman was $4.5 billion dollars. How much could taxpayers have saved if the shipbuilders had used adaptive planning techniques rather than predictive ones? What if, instead of locking scope, they used capped time and materials, but left scope variable and simply focused on high value work throughout the build cycle? If the ship had been developed in collaboration, rather than through contract negotiation, what are the odds it would have better met the needs of the customer? In the end, it was a full year after our commissioning ceremony before we were battle ready.

According to the Standish Group CHAOS report (2015), large projects are six times more successful when they use Agile instead of Waterfall. Small to medium projects, which are easier to predict, are four times more successful with Agile. What's the average cycle time for your products? How long before you get market feedback? The best way to mitigate risk is to increase transparency and reduce the time between inspections. If you want to ensure the product will solve a market problem, even as the market continues to evolve, then embrace adaptive, not predictive, planning techniques. In a VUCA world, it's adapt or die.

Read this article online: https://baa.tco.ac/1SyV

About Robert Weidner

Robert has over 20 years of experience in adaptive product development, leading large-scale organizational redesign to promote business agility. During this time, he's helped numerous companies — from startups to the Fortune 5 — implement customer-centric software development practices. From training to executive coaching, Robert has worked with hundreds of Product Teams to remove waste and optimize flow.

To learn more visit

www.livemindllc.com

BEST AGILE ARTICLES

Companion Site

Visit the Companion Site to

Download Publications for Free

View Publications Online

Nominate Articles for Publication

Volunteer on the BAA Publication Team

Receive updates on Publication Activities

https://baa.tco.ac/2019

Publication Curators

Cherie Silas is an ICF Master Certified Coach (MCC) and a Scrum Alliance Certified Enterprise Coach® (CEC), who has been coaching, leading, training, and mentoring people for most of her career.

She has a strong desire to help people arrive at the place they define as success in both personal and professional life.

Her goal is to invest the experience and talents she has gathered through years of learning, often times the hard way, into people whom she hopes will become greater than she can ever dream to be.

Cherie's life mission that drives every interaction with every individual she encounters is simply this: To leave you better than I found you with each encounter.

Tandem Coaching ACADEMY

To learn more visit
https://tandemcoaching.academy/

Michael de la Maza is a Scrum Alliance Certified Enterprise Coach (CEC).

As an Agile consultant, his major engagements have been with Paypal, State Street, edX, Carbonite, Unum, and Symantec.

He is the co-editor of Agile Coaching: Wisdom from Practitioners and Why Agile Works: The Values Behind The Results.

He holds a PhD in Computer Science from MIT.

HEART HEALTHY **SCRUM**

To learn more visit
www.hearthealthyscrum.com

Agile Community Publication Volunteers

Managing Editor
Christine (Christie) Murray

As an Enterprise Agile Coach, Christie specializes in helping leadership with their most challenging business and organizational needs. She helps them to expose and understand the hidden strengths and weaknesses within their culture and amplify behaviors to achieve greater agility.

Using her US trademarked Agile PS Culture and Communication Framework, she helps build environments of trust to pave the way to accelerate high-performing teams within organizations.

Her work has taken her all over the world assisting leaders and their teams in Fortune 500 companies. For 10+ years, she has enjoyed partnering with organizational leaders at Cisco, American Express, TIAA, Micron, and Cummins. She is proud to be part of their learning and growth journey, which has resulted in them having an organizational environment and culture that is more effective, adaptive, and empowering for those they lead.

She holds multiple Agile and Coaching certifications with Scrum Alliance, WBECS, Kanban University, SAFe and ICAgile.

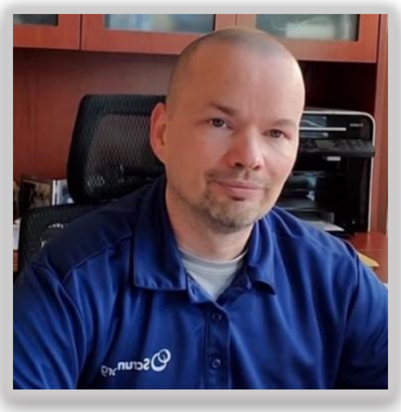
When not coaching, she volunteers her time in the service of others within various Agile communities and Agile organizations.

To learn more visit

www.ChristieMurray.com
www.linkedin.com/in/christinemurrayagilecoach

Online Publication & Technology

Alex Kudinov



Alex is a Technologist, Professional Coach, and Trainer. He delivers business competitive advantage to his customers by growing and nurturing great agile development organizations.

At Tandem Coaching Academy (www.tandemcoaching.academy) where we keep Agile non-denominational, Alex works with Agilists of all walks, bringing agile and professional coaching closer together.

Alex holds the International Coach Federation Professional Certified Coach (PCC) accreditation. He is also a Professional Scrum Trainer (PST) with Scrum.org, Scrum Alliance Certified Enterprise Coach (CEC), and an Accredited Kanban Trainer with Kanban University.

He holds an MBA degree from the University of Texas at Austin.

To learn more visit
https://lnk.tco.ac/ak

Article Committee Members

Michael Anderson

Mike has been practicing, learning and driving the spirit of high quality Agile solutions for over a decade in diverse roles including Scrum Master, Agile Coach, Director of PMO, Transformation Leader, Mentor, teacher and consultant. He understood the potential of agile when he first saw it in action. He has led agile teams, established agile practices in technology and traditionally "non-agile" domains, and helped executives improve their organization's ability to consistently deliver value to customers. In his early career as a Producer in the Live Events field Mike had the insight to instill Agile Methodologies successfully.

James Black

James has spent the past 15 years in software development in various roles from Software Developer, Project Manager, and now serving his organization in the role of an Agile Coach & Scrum Master. His goal is to help organizations understand different frameworks and processes that can lead to successful outcomes. James believes that by encouraging teams to continue learning, challenging our current processes and empowering people to make a difference is how we succeed and deliver true value to our customers. James can be reached at www.linkedin.com/in/james-black-csp-pmp-7853b12 or email Jamesblack190@gmail.com

Raj Chander

Raj is a Certified Scrum Professional (CSP) and is actively pursuing the Certified Team Coach qualification from the Scrum Alliance. For over 15 years, Raj has supported large enterprises in transforming the delivery of complex enterprise solutions from traditional project management to Agile. Recently Raj has been focusing on supporting non-technical teams embrace and adopt Agile values, principles, and practices to enhance value delivery. Raj is passionate about developing high-performing teams based on commitment, accountability, communication, trust, and fun. Raj currently works as an Agile Coach leading the journey of a large enterprise through a Digital (Agile) Transformation. Reach Raj at www.linkedin.com/in/rajchander or email rajeevchander@gmail.com

Christine Thompson

Christine is an Agile Coach, a Team Coach and an Individual Coach. She is committed to helping you to become the best version of yourself. She works with teams to promote an Agile mindset, and a set of behaviours resulting from that mindset, which allows them to focus on delivering value whilst being responsive to change, in an environment that is inclusive, safe and challenging to all its members. She is an ICF Associate Certified Coach, a Scrum Alliance Certified Team Coach, an NLP Master Coach and is ORSC Trained. Christine has a clear focus on people, coaching them to be the best they can and forming teams which are both effective and fun. You can find out more, including how to reach Christine, at www.christinethompson.coach.

Matt Kirilov

Matt has a well-rounded set of skills that deepen in both the coaching, teaming (team development), and group psychodynamics / systems modeling competencies. He is passionate about partnering with organizational systems and coaching them to explore and reach their goals. Matt is the organizer or co-organizer of several Agile and professional coaching meetups in Omaha (U.S.) and in Bulgaria. He is an ICF-ACC accredited coach. Matt currently is an Enterprise Agile Coach and is the co-creator of the course From Scrum Master to Coach Training Series

Paulo Dias, CSP-SM, CSP-PO

Paulo is a passionate agile practitioner on the journey to become a CTC/CEC by the Scrum Alliance and an ACC/PCC by the International Coach Federation. He has 24 years' experience in Software Development having started his career as a software engineer in 1995. In the last 11 years, Paulo has been working on a variety of different roles as Enterprise and Business Agility Coach, Consultant, Trainer and Mentor. As a coach, Paulo has been partnering with clients at all levels of the enterprise from the senior leadership to team level coaching. As a leadership consultant, Paulo has been helping organizations to gain strategic alignment, becoming more adaptive to fast changing market conditions, reducing lead time and increasing flow efficiency. The bulk of his experience has been in large financial institutions, having also worked in the airlines, telecommunications and automobile industries. Paulo can be reached at www.linkedin.com/in/paulo-dias-uk or email paulorcd@gmail.com

Allison Pollard

Allison helps people discover their agile instincts and develop their coaching abilities. As an agile coach with Improving in Dallas, Allison enjoys collaborating with coaches and leaders to unlock high performance and become trusted change agents in their organization and the community. In her experience, applying agile methods improves delivery, strengthens relationships, and builds trust between business and IT. Allison is also a Certified Professional Co-Active Coach, a foodie, and proud glasses wearer. To learn more visit www.allisonpollard.com

Madhavi Ledalla

Madhavi is a transformational enthusiast with technology background and end-to-end development experience with MS technologies. Her expertise lies in coaching, to help people meet their professional and personal aspirations. She works with leadership & teams to guide them through transformation. Her practices include training and coaching in varied agile methods including Scrum and Kanban and Scaling frameworks. She played multiple roles of Scrum Master, Coach, Project Manager and a Technologist. This has given her the ability to understand team dynamics to navigate transformational challenges. She is committed to serving the agile community by getting involved in various activities that include volunteering and organizing community events. Madhavi is author of the book " Retrospectives for Everyone". She shares some powerful metaphors for doing effective retrospectives in this book. She occasionally blogs at www.lmadhavi.wordpress.com

Gary Hansen

Gary is a self-proclaimed "learning instigator." He has worked with agile approaches since 2009 and his major engagements have been with United Health Group, Target, 3M, Best Buy, Thrivent Financial, & TCF Bank. He works with all levels of an organization to ensure that transformation efforts are aligned. Gary is a Scrum Alliance Certified Team Coach (CTC), International Coach Federation Associate Certified Coach (ACC), and has 13 additional certifications. Gary has an M.Ed. in Human Resource Development from the University of Minnesota. Email Gary at hansengarywa@gmail.com

Andrew Lin

Andrew has been practicing and teaching Scrum for about 15 years. He trains and coaches Scrum Masters, Product Owners, Dev. teams, and executives. He helps teams and organizations to deliver twice the value in half the time. He hosts Meetups, volunteers in translation Scrum Guide, S@S Guide, and is involved in many agile/coaching communities. One time he taught Scrum in front of Jeff Sutherland. Andrew is the first LST (Licensed Scrum Trainer) trained and certified by Jeff Sutherland, Scrum Inc. in the Greater China area. He works and lives in Fairfax, VA, and Taiwan. Reach Andrew at www.licensedscrum.com/trainer/andrew-lin or email him at andrew.lin.agile@gmail.com

Joseph Jones

Joseph is a servant leader and agile change agent. He has his CSM, SA, ICP-ATF and ICP- BAF certifications. He's an advocate of the growth mindset, practicing agile approaches and coaching. He's active in the agile/coaching community and volunteers on the Agile Alliance's Diversity and Inclusion committee. He's a member of several meetup groups such as Agile NOVA, Tandem Coaching Academy, Agile Coaching Circles and Tri-State Agile Coaching to name a few. He holds a MFA from The George Washington University in Visual Communication. He's passionate about Creative Thinking. Joseph can be reached at www.linkedin.com/in/josephkylejones or josephkjones@gmail.com

Daren Edmiston

Daren is an Agile Evangelist, a Certified Scrum Professional (CSP) and is pursuing the Certified Team Coach qualification from the Scrum Alliance. Working in Agile environments for over 12 years, Daren has supported Agile transformation in small and large enterprises. Daren is passionate about helping teams be the best they can be. Daren currently works as an Agile Coach/ScrumMaster in a large state government setting. Daren can be reached at www.linkedin.com/in/darenedmiston

Gitta Toussi

Gitta is an Enterprise Lean Agile Coach, using agile practices for Digital Transformation in diverse industries. Her specialty is in practical and lean technology implementation with focus on business agility and market competitiveness. She is equally at home with various delivery methodologies, and her passion as a servant leader is to bring out the best in people and their talent. Gitta has 8 years of hands-on agile experience and earned her agile certifications from Scaled Agile Inc., Scrum Alliance, PMI-DA, and Chicago State University. Learn more at www.linkedin.com/mwlite/in/1worldinfreedom or www.1worldinfreedom.com

Ted Wallace

Ted is currently an Agile Coach at Principal Financial Group. He has completed Master of Science degrees in Computer Science and Physiology and is a certified ScrumMaster Professional (CSM, CSPO, CSP) and a registered corporate coach (RCC) with thousands of hours of coaching sessions completed.

He has co-authored two books on habit change, “Total Brain Coaching”, and “The Coherence Code”. You can learn more about these efforts at www.totalbraincoaching.com. His talented wife, Danielle, is from the Netherlands. They have three amazing children, Jace, Kyran, and Myka.

Special Thanks to Our Volunteers

Mike Anderson
James Black
Raj Chander
Paulo Dias
Daren Edmiston
Gary Hansen
Joseph Jones
Matt Kirilov
Alex Kudinov
Madhavi Ledalla
Andrew Lin
Christine (Christie) Murray
Allison Pollard
Christine Thompson
Gitta Toussi
Ted Wallace

We are grateful to Our 2019 Sponsors

MURAL

Scrum Alliance®