

Agile at 20: The Failed Rebellion

[Originalartikel](#)

[Backup](#)

<html> <div class=„default st-post-thumbnail -st-not-illo st-box-shadow st-anim-trigger“ data-anim=„post-image“><noscript><img width=„1084“ height=„594“ src=„<https://www.simplethread.com/wp-content/uploads/2021/07/cropped.jpg>“ class=„attachment-medium-large size-medium-large wp-post-image“ alt=„Agile at 20: The Failed Rebellion“ srcset=„<https://www.simplethread.com/wp-content/uploads/2021/07/cropped.jpg> 1084w, <https://www.simplethread.com/wp-content/uploads/2021/07/cropped-300x164.jpg> 300w, <https://www.simplethread.com/wp-content/uploads/2021/07/cropped-1024x561.jpg> 1024w, <https://www.simplethread.com/wp-content/uploads/2021/07/cropped-768x421.jpg> 768w“ referrerpolicy=„no-referrer“ /></noscript><img width=„1084“ height=„594“ src=„<https://www.simplethread.com/wp-content/uploads/2021/07/cropped.jpg>“ class=„lazyload attachment-medium-large size-medium-large wp-post-image“ alt=„Agile at 20: The Failed Rebellion“ data-sizes=„(max-width: 1084px) 100vw, 1084px“ srcset=„<https://www.simplethread.com/wp-content/uploads/2021/07/cropped.jpg> 1084w, <https://www.simplethread.com/wp-content/uploads/2021/07/cropped-300x164.jpg> 300w, <https://www.simplethread.com/wp-content/uploads/2021/07/cropped-1024x561.jpg> 1024w, <https://www.simplethread.com/wp-content/uploads/2021/07/cropped-768x421.jpg> 768w“ referrerpolicy=„no-referrer“ /></div><div class=„st-blogpost-subscribe“><p class=„st-emphasized mb-0“>Worried you’ll miss us?</p><p class=„mb-2“>Subscribe to get our articles and updates in your inbox.</p></div><p>The <a href=„<https://agilemanifesto.org/>“>Agile Manifesto turned 20 this year, and there are two facts that seem self-evident:</p>Agile, as a label, won; nobody wants to be called non-Agile.Agile, as it is practiced, falls woefully short of the revolutionary ideas of its founders.<p>How did we get to this point? Everybody says they do Agile and yet almost nobody is Agile.</p><h2>Whence The Manifesto</h2><p>In February, 2001, a group of seventeen expert software practitioners met at The Lodge at Snowbird ski resort in the Wasatch mountains of Utah. Over the course of a few days of discussion and debate, they collaboratively wrote the “Agile Software Development Manifesto”.</p><aside class=„st-aside“><p>* I don’t know the personal history of every signatory, but of those I know, they were all either still actively writing code or had a long and recent history of it.</p></aside><p>The first point to highlight is that these were practitioners. They weren’t project managers or CTOs or VP Engs. They were developers, programmers, scientists, and engineers. They were still slingin’ code* and working with their stakeholders to solve problems. This is important.</p><p>The other point is that the Agile Manifesto wasn’t created in a vacuum. Many of these people already had a methodology they had created and/or were proselytizing. I might have my timing slightly off, but I think all of these methodologies pre-existed “capital A Agile”: Extreme Programming, Scrum, DSDM, Adaptive Software Development, Crystal, Feature-Driven Development, Pragmatic Programming. I know Schwaber and Sutherland spoke publicly about Scrum in 1995, and Beck and Jeffries started talking about Extreme Programming (XP) in 1996, I think.</p><p>Everyone in this group had deep experience writing software, and they were all looking for an alternative to documentation-driven, heavyweight software development processes that ruled the day. The heart of the manifesto was four statements of value:</p><hr /><div class=„c2“><p>We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:</p><p>Individuals and interactions over processes and tools<br

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan</p><p>That is, while there is value in the items on the right, we value the items on the left more.</p></div><hr><h2>A New Hope</h2><p>From our vantage point in 2021, it's easy to take so much of modern software development practice for granted, but in 2001, these ideas were wildly radical.</p><p>What do you mean you're going to start building the software before you've gathered all of the requirements and estimated every piece of functionality? That's insane!</p><p>The important piece that gets forgotten is that Agile was openly, militantly anti-management in the beginning. For example, Ken Schwaber was vocal and explicit about his goal to get rid of all project managers; not just get the people off his projects, eradicate the profession from our industry.</p><h4>Agility & PMI</h4><blockquote class=„st-quotation c4“><p class=„c3“>We have found that the role of the project manager is counterproductive in complex, creative work. The project manager's thinking, as represented by the project plan, constrains the creativity and intelligence of everyone else on the project to that of the plan, rather than engaging everyone's intelligence to best solve the problems.</p><h4><cite>Ken Schwaber, manifesto signatory and Scrum co-creator</cite></h4></blockquote><p>Scrum Masters had almost no authority, no votes on issues. They were servant leaders, helping shield and unblock the team but not managing the team. XP was similar. If I recall correctly, XP originally had trackers and coaches, which had a similar facilitating, supportive vibe.</p><p>Alistair Cockburn, manifesto signatory and creator of the Crystal methodology and hexagonal architecture, had a marvelous, insightful thread on this recently, including this idea (paraphrased):</p><blockquote class=„st-quotation c4“><p class=„c3“>Scrum struck a magnificent bargain in hostile territory:</p>Management got 12 times per year to change direction in any way they wanted, after each sprint.Teams got an entire month of total quiet time with no interruptions or changes of direction to do heavy thinking and working.Teams got to announce what they could and couldn't do in the month, with no management interference in their bid.<p class=„c3“>No executives ever got a better deal.
No development team ever got a better deal.</p></blockquote><p>I'm a certified Scrum Master, working on Agile teams for 15+ years, and I've read many of the popular books in the space. I've never seen anyone frame the idea so explicitly and succinctly (again paraphrasing Cockburn):</p><blockquote class=„st-quotation c4“><p class=„c3“>Scrum was invented to function in hostile environments. It's a contract between hard-pushing managers and developers needing time to think and explore.</p></blockquote><h2>The Empire Strikes Back</h2><p>In some ways, Agile was a grassroots labor movement. It certainly started with the practitioners on the ground and got pushed upwards into management. How did this ever succeed?</p><p>It's partially due to developers growing in number and value to their businesses, gaining clout. But the biggest factor, in my view, is that the traditional waterfall approach simply wasn't working. As software got more complicated and the pace of business accelerated and the sophistication of users rose, trying to plan everything up front became impossible. Embracing iterative development was logical, if a bit scary for managers used to planning everything.</p><p>I remember meetings in the mid-2000s where you could tell management wasn't really buying it, but they were out of ideas.</p><p>What the hell, let's try this crazy idea the engineers keep talking about.</p>

We're not hitting deadlines now. How much worse can it get?

Then to their surprise, it started working, kind of, in fits and starts. Teams would thrash for a while and then slowly gain their legs, discovering what patterns worked for that individual team, picking up momentum. After a few sprints, you'd start to see the real power of prioritizing working software, collaboration, taking time to inspect and adapt, and all the rest.

In the course of about 5 years, Agile went from a methodology that you'd heard of but probably not used to something everybody did. In 2005, I was switching jobs, and I remember the fact that I knew a bit about Agile and TDD was a real differentiator. By 2010, it was assumed modern software teams were doing some flavor of Agile. At least, this was true for my bubble in the consulting world; large enterprises always move slower.

🎉🎉🎉 We did it! We won! Congratulations everybody! 🎉🎉🎉

And that's the end of the story. You can go ahead and close the browser tab.

> Winning was easy, young man. Governing's harder.
>
> ### <cite>George Washington, as portrayed in Hamilton</cite>

Unfortunately, like many revolutions, the history of Agile didn't unfold how the founders envisioned.

- It turns out that prioritizing individuals and interactions is a hard concept to sell. It's much easier to sell processes and tools.
- It turns out that working software is harder to produce than unrealistic plans and reams of documentation.
- It turns out that collaborating with customers requires trust and vulnerability, not always present in a business setting.
- It turns out that responding to change often gets outweighed by executives wanting to feel in control and still legitimately needing to make long-term plans for their businesses.

It turns out that Agile done poorly often feels like chaos.

That doesn't mean the four values are wrong. It just means this whole thing takes some effort to get right, some courage to accept that software is inherently messy and chaotic at times. You have to understand and believe that if you keep learning and adapting and improving and shipping, you'll eventually get to a much better place, a much more honest and realistic and productive place than you ever could with waterfall.

> The Agile movement is not anti-methodology, in fact many of us want to restore credibility to the word methodology. We want to restore a balance. We embrace modeling, but not in order to file some diagram in a dusty corporate repository. We embrace documentation, but not hundreds of pages of never-maintained and rarely-used tomes. We plan, but recognize the limits of planning in a turbulent environment. Those who would brand proponents of XP or SCRUM or any of the other Agile Methodologies as “hackers” are ignorant of both the methodologies and the original definition of the term hacker.
>
> ### <cite>Jim Highsmith, History: The Agile Manifesto</cite>

Those are important points. We still need to [plan](https://www.simplethread.com/dear-agile-can-we-please-start-planning-things-again/) and document and have rigor in Agile. **It's about balance.** However if your organization is struggling with an Agile transformation, drowning in chaos, you're going to leap when someone offers you a lifeboat in the form of certifications and processes and tools. Executives understand processes and tools much more than they grok self-organizing teams.

Return of the Rogue One?

This is where my three-act structure breaks down a bit, because unfortunately, I don't see the plucky rebels coming back on this one, at least not under the Agile label.

It's been overrun with tool vendors and process consultants and experts making promises that can never be delivered. This is how we've ended up with SAFe and Scaled Scrum and all of the other enterprise Agile flavors. These frameworks weren't created with malicious intent, and they probably even have some value in the right context. But I wouldn't call them Agile. Trying to scale a methodology that focuses on individuals and interactions will inevitably lead to problems and erode the original value of the methodology.

This is how we've ended up with

this famous 2018 piece by Ron Jeffries, manifesto signatory and XP co-creator.

Developers Should Abandon Agile

When “Agile” ideas are applied poorly, they often lead to more interference with developers, less time to do the work, higher pressure, and demands to “go faster”. This is bad for the developers, and, ultimately, bad for the enterprise as well, because doing “Agile” poorly will result, more often than not, in far more defects and much slower progress than could be attained. Often, good developers leave such organizations, resulting in a less effective enterprise than prior to installing “Agile”.

This is how we’ve ended up with the famous 2014 piece by Dave Thomas, manifesto signatory and Pragmatic Programming co-creator.

Agile is Dead (Long Live Agility)

The word “agile” has been subverted to the point where it is effectively meaningless, and what passes for an agile community seems to be largely an arena for consultants and vendors to hawk services and products… Once the Manifesto became popular, the word agile became a magnet for anyone with points to espouse, hours to bill, or products to sell. It became a marketing term.

So I think it is time to retire the word “Agile.”

Retro

The really sad part of this for me is seeing young developers denigrate Agile and think of it as a means for management to extract unrealistic promises and push dev teams to work crazy hours. I get it. The only Agile they’ve ever known has been a mechanism of control imposed on them, not a tool of self-empowerment they joyously embraced. But I hope kicking off some discussions around the history and original vision can help us remember how things were supposed to go.

The good news in all of this is that the principles of the Agile Manifesto are as wise and relevant today as they were 20 years ago. And even supposed apostates like Jeffries and Thomas still think that.

Jeffries in the article mentioned above, says, “However, the values and principles of the Manifesto for Agile Software Development **still offer the best way I know to build software**, and based on my long and varied experience, I’d follow those values and principles no matter what method the larger organization used.”

I agree.

It’s not hip or cool to talk about Agile now. Agile is boring. **Everybody does Agile, right?** Now is the perfect time to reflect on the last 20 years and ask ourselves a few questions:

- What went right?
- What went wrong?
- What do we want to do differently next time?

A few of us at Simple Thread who lived through the revolution plan to reflect on each of the original [12 Agile Principles](https://agilemanifesto.org/principles.html) over the coming months, contextualize their original meaning, and consider their value in the current milieu of software development.

My hope is that by studying the founding principles, we can learn from the past, and in the words of Dave Thomas, we can retain our **agility** even if we choose to abandon “Agile”.

Loved the article? Hated it? Didn’t even read it?

We’d love to hear from you.

[Reach Out](https://www.simplethread.com/contact)

From:

<https://schnipsl.qgelm.de/> - Qgelm

Permanent link:

https://schnipsl.qgelm.de/doku.php?id=wallabag:wb2agile-at-20_-the-failed-rebellion



Last update: **2025/06/27 11:17**