[flagged] Agile sucks for software development and I'm tired of pretending it doesn't

75 points by nabazlbhf1337 10 hours ago | hide | past | favorite | 78 comments

Warning: Unpopular opinion ahead.

Agile extremists love to dance around the broken record of: "but Agile is what 'high-performing' development teams use".

Is it?

According to who?

What study?

What was the sample size?

How long was the study run?

What caliber of companies were included in the study?

Most importantly, what was the percentage increase in deliverability/speed/efficiency/developer happiness/customer happiness when using agile vs without?

These are questions the agile overlords either don't know or refuse to provide the answer to.

Even if said studies exist (hint: quality ones don't), those studies would likely have been conducted by statisticians who have never worked on a development team, thus making their study unqualified right out of the gate.

I don't know what your definition of high-performance is but my definition would be along the lines of "delivering high-quality software quickly and efficiently".

Now please explain to me what part about spending half the day in meetings rather than writing code is actually "high-performing". There is a reason Elon Musk just fired half of Twitter. Sitting around in meetings all day [pretending it's important](https://www.linkedin.com/pulse/sorry-your-job-doesnt-matter-what-matters-wayne-k-spear/) does not provide value. Clearly Elon is calling the bluff of all of the corporate shills who managed to convince company leaders that these meetings truly are important by firing everyone who sits around all day doing something other than writing code. He clearly put 44 billion dollars where his mouth is and told you that your pointless meetings don't matter and they don't need it to build quality software.

My previous company spent 1 hour every Wednesday talking about whatever urgent items needed to be addressed and then we peaced out for the rest of the week unless anything came up (in which case we would reach out to the person ad-hoc in Slack, rather than scheduling un-needed recurring weekly meetings)

My current company spends 4 out of 8 hours every day in meetings. These meetings include:

- Stand up - Sprint planning - Retrospective - Refinement - probably 5 others that I'm forgetting

I'm not an agile extremist so you'll have to excuse me if I've gotten any of the their ridiculous verbiage wrong.

But let's cut to the chase.

Someone explain to me:

Where within the definition of "high-performance" does taking 2 weeks and 4 meetings with 10 developers on each call just to deliver a simple list-filter feature fit in?

I ask because such a feature would have taken my previous non-agile team not more than 1 meeting and not more than 1 day to complete.

If team A takes 2 weeks and 4 meetings to deliver said feature and team B takes 1 day and 1 meetings to deliver said feature then simply put, team A needs to sit down and shut up as they hold no ground to talk about "high-performance".

The time-tested adage of "put your money where your mouth is" holds true no matter how many agile verbiages you want to throw at it or how many agile-non-believers you scream obscenities in the face of.

Below is an open challenge to any person or organization:

Have your agile team deliver high-quality software more quickly and efficiently than my non-agile team and I will print out this post and eat it on camera.

Until then: sit down and shut up.

add comment

reilly3000 9 hours ago | next [–]

I don’t really know anything other than Distraction Driven Development. Between being waterboarded with Slack, email, and noise on both about everything else, and walls of all team meetings… only the loudest, quickest, and shiniest things get done. The carefully groomed backlog is irrelevant in a crisis, and in the realm of devops it seems that’s the norm.

I’m certain that I am part of that problem. My neurotype seeks, even demands novelty. My role requires creative work among a stream of incidents. The not quite platform team thing. I’m finding myself working late and at odd times to get creative work done and it’s not sustainable.

The promise of agile is to fix this. The reality is most people can’t, won’t, or simply don’t say “no, there’s no cutting in line. It goes to triage then the back of its respective line.”

I don’t think Distraction is a good thing per se, but it feels inevitable from where I sit. Agile eschews planning, or embraces not sticking with the plan as state changes. I’ve not really worked in a waterfall environment, but if it’s agile’s antonym then I’m in.

reply

> readthenotes1 8 hours ago | parent | next [–]
>
> Tbh, i stopped reading and upvoted you after "waterboarded with Slack"
>
> reply

Sverigevader 21 minutes ago | prev | next [–]

I think I understand your frustration, and I agree to a certain extent. But first, consider this:

Could it be that your problem isn't with Agile itself but with Scrum (i.e. how you apply Agile)?

In my experience, it goes something like this;

– "We need a better way to work"

– "Oh, let's go with Agile, everyone is doing that"

– "But that's just a set of principles and guidelines though isn't it?"

– "So we need something on top of it then. To keep track of what we're doing and what we've done? And how fast we've done it. How fast we can deliver things. We have to measure the throughput of the features we ship. Oh and we need status tags so that we know which state every feature is in at any given time. And the features are too big of course, we need to split them into smaller \"tasks\", and they need to contain exactly the information required to complete the work without talking to anyone else. I mean I'm a manager and simply must know what people are working on and how they're solving it and, if they're not doing it fast enough. Right?"

– "Sounds good, how do we keep track of the way we keep track of things?"

– "Hmmmm....."

– "Oh wait! Meetings and ceremonies!!"

If you cannot succeed in a situation where you have motivated people, in an environment where learning by failure is encouraged and management that trust their workers I don't think you can blame Agile. But you could perhaps blame the framework that you execute Agile with because they have a way of demotivating even the best of us.

In a perfect world (where you're all in the same room all the time) the Agile manifesto *and* the development principles it comes with, a whiteboard and post-its will be all you need. Your problem is what you've replaced the whiteboard and post-its with.

I could of course be wrong here but this is, in my experience what breaks the camels back. Mostly...

reply

adjkant 9 hours ago | prev | next [–]

> My current company spends 4 out of 8 hours every day in meetings

This doesn't have anything to do with agile. You can run "agile" with as little as an hour of meetings a week if you want. Planning, retro, refinement in one weekly, async standup in Slack. You can bring standups in person, have them daily or less frequently, adjust the frequency, change your sprints from 1 to 2 weeks.

Even the heaviest weight version of this I can imagine (daily 30 min standups, 1 hour planning, 30 min retro, weekly sprints) adds up to 4 hours total for the *week*. So what you're describing is 80% something beyond that.

> taking 2 weeks and 4 meetings with 10 developers on each call just to deliver a simple list-filter feature fit in?

This sounds like you've moved from smaller company to bigger company and are noticing things move slower, though correct me if I'm wrong.

Either way, these are the questions: Why does the feature actually take two weeks to build? Are there more factors beyond the team? Larger scale? More testing / QA needed than pushing out to prod? Just plain worse developers? Bad PR practices that delay the feature? These factors again are nothing to do with "agile".

Another person asked this well, but really you've offered no notes on what your old team did differently that was not "agile". What's the alternative that people are missing?

reply

anonymoushn 9 hours ago | parent | next [–]

> Even the heaviest weight version of this I can imagine (daily 30 min standups, 1 hour planning, 30 min retro, weekly sprints) adds up to 4 hours total for the week. So what you're describing is 80% something beyond that.

Pivotal Labs, well known for "doing Agile right," spends the entire Friday doing no work every week.

reply

adjkant 9 hours ago | root | parent | next [–]

Disclaimer: Defining and using a narrow definition agile IMO is a useless rabbit hole.

And the big names in tech also purport to use agile and spend maybe an hour every Friday or Monday doing "agile" type meetings. Which are we talking about here?

If it helps, add to my above post that I'm using "agile" to mean the philosophy of defined sprints with stand ups, planning, and retrospectives to help execute a larger, changing roadmap. There are many many other rules people can choose to add, but my experience across many companies is that this is the shared core in practicality.

When people say agile, 95%+ of the time they don't mean whatever Pivotal Labs is using for a standard. I've practiced "agile development" at F10, big tech, and under 250 person startups, and no one has ever referenced a strict spec definition like that, not even the F10 which basically said "here's some detailed guidelines some use, take what works". So what's the relevance of this strict definition?

reply

anonymoushn 9 hours ago | root | parent | next [–]

I agree that there's no use defining things so narrowly, but I think your experience about the meeting load involved in "doing agile" is atypically low for the industry in general.

reply

adjkant 9 hours ago | root | parent | next [–]

Maybe so, but it's varied across size and industry and matches with my friends experience in other broad areas. So I guess let's reframe: what is my meeting load estimate (the 4 hour a week high side) missing? How does that get to 4 hours a day, and if not there, what is the maximum?

To be clear, that's not to say that other meetings can't be used, but that they are not part of the "agile" process. I can easily imagine a dev ending up with 4 hours a day, but that's more related to company size and process. Things like design reviews, meeting with other teams, not being able to quickly find the right point of contact, using meetings to find out you have the wrong person, not defining clear agendas, inviting too many people to meetings, and so on. I'd bet some of these are affecting OP, but again, this has nothing to do with the style of development planning/process.

reply

anonymoushn 9 hours ago | root | parent | next [–]

It's easy to get to 4 hours a week of only standups.

reply

Volundr 7 hours ago | root | parent | next [–]

If your running strict scrum your stand-ups should be 15 minutes. In my current company as well as my last role that'd be a long stand. They were usually closer to 5.

reply

adjkant 9 hours ago | root | parent | prev | next [–]

Okay, that's not what I said or what OP said though:

> How does that get to 4 hours a *day*

reply

cowtools 9 hours ago | root | parent | prev | next [–]

Agile isn't just a workflow or philosphy. It's a way of life. No one has ever ben sucessful without triaging first.

reply

orwin 9 hours ago | root | parent | prev | next [–]

Honestly, at my last company, in my first team we reserved the fridays afternoons for demo and short formations (the most complex, that also resonated the most with me, explained how and why the network architecture was made, while some were as basic as explaining how to make good PR).

It was way more productive than with my second team where we used one hour every two week for the retro. In big company, the time you actually spend on understanding how everything work is never really lost.

reply

koonsolo 2 hours ago | prev | next [–]

I feel the need to respond here, but I really don't want to. Anyway...

The core of Agile is that it values:

- Individuals and interactions over processes and tools

- Working software over comprehensive documentation

- Customer collaboration over contract negotiation

- Responding to change over following a plan

Agile also has 12 development principles:

1. Customer satisfaction by early and continuous delivery of valuable software.

2. Welcome changing requirements, even in late development.

3. Deliver working software frequently (weeks rather than months).

4. Close, daily cooperation between business people and developers.

5. Projects are built around motivated individuals, who should be trusted.

6. Face-to-face conversation is the best form of communication (co-location).

7. Working software is the primary measure of progress.

8. Sustainable development, able to maintain a constant pace.

9. Continuous attention to technical excellence and good design.

10. Simplicity—the art of maximizing the amount of work not done—is essential.

11. Best architectures, requirements, and designs emerge from self-organizing teams.

12. Regularly, the team reflects on how to become more effective, and adjusts accordingly.

So now please check if your company is doing this or not.

If you say "We're following most of that, and it really sucks", then we can have a valuable discussion.

But if you say "We're not following that and it really sucks", well then why are you arguing that Agile doesn't work?

reply

amiga-workbench 9 hours ago | prev | next [–]

> My current company spends 4 out of 8 hours every day in meetings.

I hate to pull a "no true scottsman", but absolutely nothing about that is being agile. The manifesto says nothing about cargo culting various ceremonies, in fact quite the opposite.

reply

bamboozled 9 hours ago | parent | next [–]

My experience with Agile has been a little bit meeting heavy though, the rituals etc.

It’s a really hard balance to get right and Agile easily ends up turning into a lot of pointless meetings teams feel like they need to have because that’s part of the manifesto they prescribed too.

I’ve also found lot of counselling and explaining starts happening when sprints aren’t going well or people become fatigued.

We could argue it’s not an Agile problem, like everything it’s susceptible to a lot of problems which we like to gloss over because it’s convenient to do so and those who advocate for agile haven’t bothered to revise and improve on the original manifesto to address these weaknesses.

the Agile manifesto is a bit like a bible in that sense.

reply

WheelsAtLarge 6 hours ago | parent | prev | next [–]

True, there's a problem with the way it's implemented. But developers rarely have a choice on how PMs manage their projects. Agile just doesn't work for long term projects. It leads to junk code that developers hope they will fix later or long hours for them trying to meet deadlines. A lot of the time it's both.

reply

koonsolo 4 hours ago | root | parent | next [–]

If you say "Project Manager", I'm already suspicious that you're actually doing agile. The most popular agile framework scrum doesn't have a project manager role.

reply

joaogfarias 1 hour ago | prev | next [–]

It sounds like someone who only had bad relationships and then proceeds to say that all men/women are trash and go M/WGTWO. The world is very big and each one of us can only experience first-hand a tiny part of it.

As other suggested, to understand the roots of Agile, I suggested looking at the Manifesto itself. Additionally, I suggest looking what people like Bob Martin, Martin Fowley, Allen Houb, Kent Beck, and Dave Farley are saying.

E.g. https://www.youtube.com/watch?v=hxXmTnb3mFU https://www.youtube.com/watch?v=4JihsBOBbdI

For reports on teams' performance, I suggest looking at the DORA's report. There thousand of teams have shown they experience in the last ten years. Additionally, if you want to some academic researches, going to Google Scholar and searching for "Systematic Review PUT_HERE_ONE_AGILE_PRACTICE" generally yields good results.

https://www.sciencedirect.com/science/article/pii/S095058490... https://ieeexplore.ieee.org/abstract/document/5654783/

reply

spamizbad 9 hours ago | prev | next [–]

Yeah it does suck and you're not wrong. I've spent the last 3 years at my company trying to refine our Agile/Scrum process into something that gets out of my engineers way. It basically boils down to the Engineering manager, product manager and project manager or "scrum master" actually putting in a good bit of work to make things run smoothly.

1) keep your stands below 15 minutes. Always. There should also never be more than 15 participants in your stand.

2) The EM/PM/SM or all of the above should be writing high quality tickets that are discrete units of work (they can be tested and have some tangible output). They should never be so large as to take a single developer the entire length of the sprint.

3) Do a single weekly grooming/pointing session. This should take less than 2 hours assuming management did their jobs

4) EM/PM/PM/SM need to actually keep the backlog organized. Don't make devs sit thru meetings where you slowly drive Jira

5) Do demos, retro and planning in a 3 hour time block with breaks. Balance what went well with what speed bumps you encountered and call out exceptional work done by team members

6) Make judicious use of doing "proof of concepts" to help spec out features/APIs/Integrations before running off to plan larger initiatives

7) Appoint an "Epic Captain" to each epic (group of related tickets). This is dev who is the subject matter expert and has some decision making authority. Defaults to the EM but they should appoint other devs to take proactive ownership over things

8) Proactively track and discuss technical debt and raise it during retrospectives; EM follows through by ticketing it up.

9) Actually do planning when launching major initiatives. A few discussions between devs before leaping into several weeks long project won't make you waterfall and will actually reduce meetings in the future

For us, the average time per week an engineer spends in meetings is less than 5 hours total, or <1 hour per day on a team of 8. Kinda high but things run smoothly and nobody is out of the loop

reply

> koonsolo 3 hours ago | parent | next [–]
>
> > Engineering manager, product manager and project manager
>
> What kind of Agile are following that has these roles defined?
>
> reply
>
> throwuwu 8 hours ago | parent | prev | next [–]
>
> This sounds nice. Can you recommend any materials I can forward to my (inexperienced)PM?
>
> reply

throwuwu 9 hours ago | prev | next [–]

Let me guess, the reason you have those meetings is to report to the PM or whoever about what you did and then what they want you to do, plus talking with clients about features and endlessly rehashing things you already discussed, plus some kind of cross communication between groups that don't actually need to share anything with each other right now, plus reshuffling the backlog and re-estimating current work, and so on. If that rings true then welcome to Scrumfall. Get out if you can.

https://www.cio.com/article/309171/scrumfall-when-agile-beco...

reply

add-sub-mul-div 9 hours ago | prev | next [–]

I totally agree. I used to be highly engaged and high performing. Agile and sprints are soul crushing for high performers. They reduce risk for managing underperformers, I get it. But it's such an artificially slow pace and so much mind numbing process. It's like being trapped in Office Space. You can't take true ownership of anything that's distributed among a bunch of different people and spreadsheets. I thrive on ownership.

reply

graiz 9 hours ago | prev | next [–]

Not an Agile extremist. If done effectively agile is a simple framework for aligning on work, having some cadence for updates and accepting or rejecting work based on quality and alignment.

I personally dislike when it's made significantly more complex. High-performing teams can work in a standard Scrum framework or they can customize agile to suite their needs.

If you're spending hours a day in meetings, you're not using the framework correctly. In a fell-functioning team, you'll spend 5-10min max doing some form of standup, in some teams this is even async so no daily meeting. That leaves one planning meeting every two weeks and one

review meeting to see how you've done.

reply

robmccoll 9 hours ago | prev | next [–]

No one will get far trying to have a debate here. Agile is both poorly and conflictingly well-defined and its adherents seem to lean on "no true Scotsman" defences when given an example of massive inefficiencies an organization that claims to do it. Whatever you are doing is "agile" if you are performing well and not if you aren't. Also the idea of some pure waterfall is nearly as much of a strawman as the ethereal agile. Measuring performance is also quite difficult. Is it meeting internal or external deadlines? Providing consistent and accurate estimates of time and effort? Providing software of sufficient quality regardless of those measures? Do what works for your team. Make changes if it isn't working.

reply

> PainfullyNormal 9 hours ago | parent | next [–]
>
> You almost have to admire it from a rhetorical perspective. Kent Beck created a manifesto so vague and flexible that it can literally be anything to anyone.
>
> > Make changes if it isn't working.
>
> If measuring performance is so difficult, then how do you know if it's working or not?
>
> reply

perrygeo 9 hours ago | prev | next [–]

That's a lot to unpack.

But I agree with your general sentiment - As currently practiced, Agile generally involves a lot of ceremony before actually building, with many rounds of collaborative planning and risk management on every decision. It's morphed into something completely different than the original definition...

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.
```

reply

> a1445c8b 8 hours ago | parent | next [–]
>
> I believe we've moved on from that: https://www.halfarsedagilemanifesto.org/
>
> reply

flerchin 9 hours ago | prev | next [–]

You seem to be conflating excess meetings with agile. I'd consider my team pretty agile, in that we closely collaborate with our customers and deliver early and often. Seems like you do that too.

I think you might be getting pressure to change the way you work by folks that don't perform. That sucks; I've been there. Hopefully the pressure turns into nothing, because that's generally been my experience. Precisely because the folks applying the pressure don't get things done, and that includes changing the way you work.

reply

g9yuayon 7 hours ago | prev | next [–]

> My current company spends 4 out of 8 hours every day in meetings.

When I was in Netflix and Uber, my teams had no status meeting, at all. We came up with a context and a big enough ownership, and just let the team loose. It's something like this: we need to enable services to send their events for analytics without ever talking to the data lake team. Steven, how about you own it? Then, a month or two later, everyone in the company can simply use log4j (at that time) to log an instance of annotated Java class, and in 15 minutes or less, the logged events became available for all kinds of queries in the data lake in the right format with the right metadata. No status meeting. No alignment meeting. No customer outreach. Not unnecessary shit. Things just happened, features just introduced. It's all non-event.

In the end, it is the culture that matters. Everything else is a matter of tactics.

reply

mdmglr 9 hours ago | prev | next [–]

> My current company spends 4 out of 8 hours every day in meetings.

Be the one in your organization to solve this problem. Start with your immediate team. Doesn't have to be this way.

Some 0.02:

1. Sprint planning is slowed down by diving into technical details. Leave that for another time.

2. Retrospective doesn't need to be more than 15 minutes. Sometimes there is nothing to discuss. It may help to have participants populate an issue board before the meeting, so retro can focus on prioritizing and addressing. I use a mattermost channel called "Retro" where devs can add an issue they are having and all the other devs can vote with a thumbs up. Before retro I sort them from most thumbs up to least. Devs can reply to issues with a solution or an offer to help ahead of time so we don't have to discuss the issue in the meeting.

3. Don't let the backlog grow to big. Groom it regularly with tech lead and PO.

4. Maintain an open list where people can populate any "parking lot" items ahead of stand up. I use a mattermost channel called "Parking Lot" and I look at any chat items added on the day of standup. Devs can reply and sometimes solve issues before stand up.

5. Structure your stand up where everyone says: task # from jira/gitlab issues/etc they are working on, any open MRs that need to be merged in, and answer the question: do I have any impediments preventing me from working. The last item is a quick "yes I have impediments" or you don't say anything. Leave the details of the impediment to parking lot. Allow everyone to leave after the last member speaks. Anyone who has a vested interest in a parking lot item or impediment from another team member can stay.

reply

justinator 9 hours ago | prev | next [–]

"Democracy is the worst form of government – except for all the others that have been tried"

I feel the same may be true w/Agile.

reply

projectileboy 9 hours ago | parent | next [–]

Exactly. When people say “agile sucks!” I always want to ask “compared to what?” Engineers who say that they meet for a few minutes once a week and then everyone somehow just knows what to do… frankly I call bullshit. That may work at a 5 person startup, but as soon as you have 100 people, you’re going to have a lot of programmers writing code who are sure they know what’s best but in fact have no idea how their work aligns with the company’s overall goals.

reply

nso95 7 hours ago | root | parent | next [–]

A spec

reply

projectileboy 6 hours ago | root | parent | next [–]

Well… yes. And the spec is perfect, right?

reply

nso95 5 hours ago | root | parent | next [–]

Not perfect, no. But imperfection is no excuse not to try, like agile would have you believe.

reply

SavageBeast 9 hours ago | prev | next [–]

Your current company is doing it wrong - this does not sound like Agile at all. Another point to remember is that Agile is for Product Management and Stakeholders - its not meant to be the fastest, most productive developer pleasing method at all.

It's meant to build cross functional teams of people who can produce predictable amounts of output such that people in Product Management can build roadmaps and concoct release schedules.

Not to mention, once a company has successfully implemented Agile (and its inherent trade-off of dev speed vs. predictable minimums) that company may now replace any person on any team with any other person. Nobody wants to say it but Agile is about hiring low rate contractors instead of expensive FTE resources.

An often overlooked part of Agile is that if you're an experienced dev who's also lazy is that this method provides a lot of cover to hide in. It's about predictable minimums. A clever dev with some talent can work about 4h/day and still outwork some of the less clever cohorts who may work many more hours.

Also, you seem very angry and thats a sign you probably care a great deal about what you're doing at work. Maybe its worth looking for something else to do where things operate more to your liking? Stress really is a killer and few things are as stressful as hating the way your job works every day of your life. Reading your story I think I'd be angry too in your position.

Finally, repeat after me here - "Agile is for MANAGEMENT - Agile is NOT for developers". The Happy Agile Developer is one who's figured out how the game works and uses that understanding to work less.

reply

terramauthe 9 hours ago | prev | next [–]

I'm sorry that has been your experience with agile. That doesn't sound very good. In my experience, if agile rituals take more than three hours per 2-week sprint, something is not aligned with the team or the project. Maybe the necessary skills aren't there. Maybe the team doesn't trust each other. Perhaps the project plan is not aligned with business goals, etc. None of these problems can be solved by adding more agile rituals.

How I prefer to run it:

- 1h per week alternating between estimating and planning

- (optional) 1h per sprint for backlog refinement

I like to keep two sprints worth of work in the backlog. More than that, the spec will change too much, and you will re-estimate. When a project has just started, or the requirements have become understood differently, to get four weeks of backlog work may require additional refinement. I typically do the bulk of this refinement myself (as a tech lead), and I try to avoid asking the team to have any extra meetings beyond what I mentioned above.

It can be good to have retros sometimes, but it's better to have a team that feels safe to share and discuss issues constructively without a mediator.

I'd also like to advocate for 1:1s. We are all humans and social organisms. Ideally, these 1:1s are 80% unrelated to a work task.

reply

mikkergp 9 hours ago | prev | next [–]

How exactly are you defining agile? Scrum is not agile, are you advocating for waterfall? Because the type of project management doesn't dictate the number of meetings, waterfall development would be slow lengthy involve lots of meetings and you May not even start coding until months of requirement gathering. I'm not even saying "you're doing agile wrong" I'm just not sure what you mean by agile.

reply

doctor_eval 9 hours ago | prev | next [–]

To me, the definition of Agile is the Agile Manifesto https://agilemanifesto.org/

```
    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.
```

If this doesn't describe what you're doing, then it's not agile, and it makes no sense to blame agile for what is just a busted process that someone has told you is "agile" without actually understanding what the term was intended to mean.

I've worked in large companies where none of this is true, but they called it agile.

It's not.

reply

mberning 9 hours ago | prev | next [–]

I have met so many developers that have the mistaken notion that doing things other than programming is somehow far beneath them. They are indignant that the business would hire people to check in on what they are doing day to day. But I see it the other way. It's something which developers have a) brought upon themselves through myriad issues ranging from poor communication to outright laziness, and b) have provided no better alternative to, and c) have tolerated in high enough numbers that it is now the defacto workplace standard.

As for myself, I don't like all the agile rituals and ouija board crap, but I also don't mind explaining what I am doing to clueless people a couple hours a week. Things could be much much worse. We could be paid a lot less and have way worse working conditions.

reply

eschaton 8 hours ago | prev | next [–]

Four out of every eight hours a day in process-related meetings doesn't sound very agile to me.

I've done plenty of actually-agile development and it's always involved one meeting at the start of an iteration to select the items to work on from the backlog, a daily 15-minute standup to address any blockers for our immediate work, and then a couple hours at the end of the iteration to demo the completed work and do a retrospective for anything unexpected that came up.

That's always worked great for me, especially combined with sizing work items in points rather than time so estimates come out of measurement of previous work rather than guesses about how long things will take.

reply

duncan-donuts 9 hours ago | prev | next [–]

I'd argue that your previous team was an agile team. My team also has 1 meeting a week where we go over the work for the week, and ask any questions we might have. We do retro like once a month but not always. Depends if we have anything to talk about. We don't do any Agile™ stuff. I'd say we're closer to an extreme programming team which has a lot of influence on capital A Agile. At any rate I agree that the type of top-down Agile forced on teams is not good for teams. The only people I can see benefitting are directors. The style I work in keeps everyone but my team in the dark and unless you have a good track record exceeding expectations you might be forced into Agile™

reply

yodon 9 hours ago | prev | next [–]

I hear that you're angry, and agile certainly has issues, but the things you're angry about aren't agile issues, they're just a broken process that someone has told you is agile.

reply

conk 9 hours ago | prev | next [–]

There's pros and cons to both approaches. At the end of the day the framework you use won't make up for serious deficiencies in the product/organization/tech stack. But you can't ignore benefits with agile over waterfall of delivering small incremental updates quickly, major issues and defects bubble up quicker and can be address before you sink too much time/effor into the product.

This is similar to the lean approach in manufacturing that has been very popular over the past ~20 years.

reply

anderber 9 hours ago | prev | next [–]

From being in various companies, I can tell you that they each had their own definition of what "agile" is. Your company happens to be that there's plenty of meetings to "communicate" what's happening. We are agile, and don't feel at all what you're describing.

I think ideally, agile should be where you put outcome over output, that's it. And I think that's the correct way of thinking when it comes to wanting to deliver the best product.

reply

EamonnMR 9 hours ago | prev | next [–]

> Warning: Unpopular opinion ahead.

Do we do that here? Should we?

> Now please explain to me what part about spending half the day in meetings rather than writing code is actually “high-performing”

I'm not sure what agile methodology you're using that ends up with this many meetings, but Scrum, a popular agile framework, recommends only a few and the daily one is supposed to be relatively short.

> Below is an open-challenge to any person or organization

Define your metrics and make sure your printer has toner.

reply

PainfullyNormal 9 hours ago | prev | next [–]

I couldn't agree more. There is zero incentive to excel in any way. Solve something really cool? No one cares. Take the next task in the sprint. Spot a problem? Add it to the backlog (to be ignored) and do what you’re tol... I mean, take the next task in the sprint. Spot a problem, put together a solution, save or make a bunch of money for the company? Who told you to do that? That wasn’t the next task in the sprint.

reply

unwise-exe 9 hours ago | prev | next [–]

The way your organize work needs to be *appropriate* to the work being done, and to the individual people involved and their relationships.

Sometimes this will look like one of the named methodologies under the "agile" umbrella, sometimes it won't.

Sometimes you get a team that will do fairly well regardless of which methodology they get stuck with. Sometimes you get one that will do badly regardless of methodology.

reply

boxed 8 hours ago | prev | next [–]

Read the manifesto I say. If it seems bad and you aren't making changes then it's by definition not agile.

I worked 10 years in a shop where I could quite literally do zero minutes of meetings per week if I so chose. Which I did from time to time. But the morning meeting was very short and a good time to just say hello to people so I often went.

reply

ricardobeat 9 hours ago | prev | next [–]

> 4 out of 8 hours every day in meetings

Does not sound like agile. Standup is a 15 minute affair, planning & retrospective a once every 2-week (however long your sprints are) event 1-2 hours long. That's all. Roughly 4 hours for every 80 hours of work.

Whatever you have going on there is probably a wild detour off SCRUM inflicted by agile consultants, which seems to be very common these days.

reply

smsm42 9 hours ago | prev | next [–]

> My current company spends 4 out of 8 hours every day in meetings

Here's your problem. The rest is just measuring the symptoms and attaching labels to them, but the disease itself is clear. It's not about "agile". I don't know what is called "agile" in your organization, but whatever it is, the theory is not the problem. The implementation you have is.

reply

tambourine_man 9 hours ago | prev | next [–]

Generalizing methodologies is hard. What works for a 5 people startup might not for a few thousand multi-decade company. Or even for another 5 people startup with different culture.

It’s not that you can’t learn from previous experience, but we have an unhealthy tendency of looking for a one-size-fits-all solution that comes with a bunch of fashionable names attached in this industry.

reply

mmmpetrichor 9 hours ago | prev | next [–]

You're part of the team. have you talked to anyone else about this? Either you can complain online, you can leave the team, or you can start trying to build some consensus with teammates and start making incremental improvements, if management is receptive. It sounds like you needed to let off some steam too! :).

reply

archsurface 8 hours ago | prev | next [–]

Correct. It's so that non-technical and negligibly technical managers can manage junior devs, including 40 year old ex-lawyers with two years of scripting, and boot campers. For the experienced it's a disaster of childishness, obstacles and frustrations.

reply

postalrat 9 hours ago | prev | next [–]

IMO all agile should be about is making sure everything is easy to change. Make code that's easy to change. Make procedures that are easy to change.

And actually make change and find what works.

reply

fiedzia 8 hours ago | prev | next [–]

> My current company spends 4 out of 8 hours every day in meetings.

What your company poor time organisation has anything to do with development methodology?

reply

davidthewatson 7 hours ago | prev | next [–]

Eating print on camera certainly qualifies you as being on-par with Werner Herzog who ate his shoe on camera.

The problem with classifying the need for lightweight methods as an "agile" problem is that agile has become the dogma that it set out to replace twenty years ago. Let that sink in. The word agile divides the room and that's part of the problem with cliched corporate speak.

As a result, there is no more confusing argument in software process today than agile because we do not have consensus on its meaning. At best, defining agile is a choice paradox. This is a result of the spectrum from little a agile (actual agility) to Big A Agile (fake corporate agile) where the pointy haired boss tells you to select from a cargo cult of overly complex choices, all of which are bad. My experience has been that teams that wind up at real agile wind up there because they weren't worried about defining a cornucopia of cliched agile speak prior to moving forward and instead did a lean build-measure-learn cycle and just figured it out as a series of continuous improvement iterations whereas fake agile stems from trying to do what some other unicorn says works for them, failing, and never getting it right because there is no cycle of action learning and reflection. tldr; it doesn't actually matter what words you use to describe the gyrations, what matters is that the gyrations work for you and your team and produce the results the team desires.

FWIW, most of what I've said is supported in the research and publishing of Alistair Cockburn who was there when it all began twenty years ago and wrote Crystal Clear. The other academic I'd reference who has the done the work and arrives at what I believe are the correct conclusions is Allen Holub.

reply

cowtools 9 hours ago | prev | next [–]

seems like a bit of a cope to me...

reply

ddbenyu 8 hours ago | prev | next [–]

It’s useful in situations but it can be a grind if it is constant.

reply

adabaed 9 hours ago | prev | next [–]

Honestly man, maybe your company sucks at applying agile methodologies.

reply

twelvechairs 9 hours ago | prev | next [–]

There's times when 4 hours of meetings a day is the way to go to get to your goal and there's times when no meetings is the way to go.

Different projects/tasks require different solutions and I'm tired of people pretending they don't

reply

rl3 9 hours ago | prev | next [–]

>*There is a reason Elon Musk just fired half of Twitter.*

Unfortunately this heavily detracts from an argument that might otherwise have merit.

>*He clearly put 44 billion dollars where his mouth is and told you that your pointless meetings don’t matter and they don’t need it to build quality software.*

He's made a mockery of himself already and is well on his way to imploding the platform. Moreover, how much software quality do you think is going to come out of Twitter now that employees live in day-to-day fear under arbitrarily extreme deadlines, while understaffed at that?

The situation is a joke at this point and has more to do with Musk's mental instability than it does Agile, as much as I loathe some of its uses and implementations.

reply

jasonjackson 9 hours ago | prev | next [–]

kind of reminded me of this post https://twitter.com/marius/status/1590056610625650688?s=20&t...

agile is for people who prefer 'classical music' working style, but want to simulate start-up mentality

reply

BlueTemplar 9 hours ago | prev | next [–]

Well, I might be pretty inexperienced, but in our project management class it was explained pretty clearly that agile (=lots of iteration) works best for projects with lots of unknowns (especially with a high risk of unknown unknowns), while waterfall (=no iteration) works best for already well-trodden projects in well-known conditions.

(We were too busy for an agile class or project, only got waterfall, since it is simpler.)

reply

Victerius 9 hours ago | prev | next [–]

The number of Agile apologists in this comment section surprises me.

reply

> PainfullyNormal 9 hours ago | parent | next [–]
>
> Why? Agile has been the favored approach to software development among the startup crowd for over a decade now. I don't think I've had a job since 2008 that didn't use it in some fashion.
>
> reply

greenthrow 9 hours ago | prev | next [–]

Agile isn't one thing and isn't gonna fix a bad team, and does nothing to address problems with management. I have had great successes with forms of it, but that doesn't mean that it's the best fit for every team.

reply

ClumsyPilot 9 hours ago | prev | next [–]

> There is a reason Elon Musk just fired half of Twitter.

Incompetence? Vindictiveness? A huge bag of weed?

> Agile team ... non-agile team

What is non-agile team, Waterfall? You sre the one demanding spesific information and studies, so explain clearly, and be spesific, what are you even advocating?

reply

> PainfullyNormal 9 hours ago | parent | next [–]
>
> Have you ever heard of shape-up? https://basecamp.com/shapeup
>
> One of the issues I have with how agile is all things to all people is that nobody's trying new things. They're trying to fit what they're already doing inside the definition of agile instead.
>
> reply

lordkrandel 8 hours ago | prev | next [–]

What a rant.

reply

AnimalMuppet 9 hours ago | prev | next [–]

At the risk of going "no true Agile" on you: There is *nothing* about agile that requires spending 4 hours a day in meetings. You're suffering (I believe that is the correct word) under an agile-in-name-only bureaucracy.

Agile can be done well. I've seen it, and it was amazing. I've also seen "agile" implemented where the number of meetings (and the number of forms) keep going up. But the existence of the fake does not disprove the existence of the real. It just means that you have to be very wary of anything bearing the name.

reply

gedy 9 hours ago | prev | next [–]

I feel for you, but this is really "not Agile". The point is shipping working software every sprint, and during the sprint you are not supposed to be interrupted until Sprint demo.

It's on the so-called Product Owner to prioritize what's next sprint, projections, etc, and a Scrum Master is supposed to help protect you from distractions.

I've done this style of agile before, and it's great. However the issue is many mgmt types still only caring about when "all of this will be done?", and also teams and people which can't think incrementally.

reply

tootie 9 hours ago | prev | next [–]

Well, for one you're definitely being hyperbolic. Half of every day in meetings is worse than the worst agile shop I've ever seen by factor of 10. Also, agile doesn't require any meetings ever. Scrum recommends a cadence of specific meetings which should add up to maybe 2-4 hours per week, but I consider that approach only necessary for immature teams. Also, agile doesn't promise faster delivery. There's no way to project manage your way to faster coding. The only critical principal is iterative improvement meaning that you deliver features to full quality in priority order. The idea is to always have a stable main branch and be able to accurately measure progress even when priorities change. The value is for the team knowing exactly what's expected of them at any given time and for stakeholders to see exactly what is and isn't done at any point. Teams and developers will always run at different paces for different projects. Good agile process is about visibility.

reply

echelon 9 hours ago | prev | next [–]

> My previous company spent 1 hour every Wednesday talking about whatever urgent items needed to be addressed and then we peaced out for the rest of the week unless anything came up

This is agile.

> My current company spends 4 out of 8 hours every day in meetings. These meetings include: - Stand up - Sprint planning - Retrospective - Refinement - probably 5 others that I'm forgetting

This is simply dysfunction.

You're contrasting process heavy vs process light, not agile vs. an alternative methodology.

When people contrast agile against something, it's typically the waterfall method of design. And you'd know that if you saw it. (Interviews, design docs, stakeholder sign offs, etc. before any code happens.)

reply

anonymoushn 9 hours ago | prev [–]

Hello,

This is a low-quality comment due to circumstances such as SBF having stolen a lot of my funds and my lack of experience at higher levels than senior IC in industry (other than startup CEO, I guess). I hope that some people who have worked in management, technical/management, or staff IC roles can give their thoughts soon.

In a small team with not many outside commitments, or with outside commitments that do not take up all of our time, we have enjoyed outsized productivity compared to my experience in industry, because we basically know what everyone is good at and we can get the right work to the right people, and because for the most part everyone feels some sense of professional responsibility, steps up, owns the whole problem, and gets things done. This may be rare even among startups though, because you know, even if you don't deliver you still get six figures of VC funding in your checking account.

In larger companies, it seems like several things happen:

- Many people are hired who basically cannot do some important parts of the process of designing, writing, and deploying software. That's not morally bad or anything! This stuff is hard. They don't teach it in school. Ideally we would provide them with mentorship. Basically no company provides them with mentorship.

- Processes are created to attempt to hold these people's hands and ensure software can be delivered according to any schedule at all, which ultimately prevent people who are actually able to ship things from shipping things, since they must instead create 20 page google docs about 100 LOC subcomponents, then go to several meetings about these google docs, then go to meetings with other teams about how to integrate the components between teams (this part is technically superfluous but perf and promo criteria say if you're senior and you want to get promoted instead of PIPed you have to make everything involve other teams even if it doesn't need to)

- Commitments are made between teams, to customers, or to management that exceed the capacity of teams to deliver, especially when it is impossible to create a staging-like environment, when problems from production come with truncated stack traces because the document size for elasticsearch is much smaller than the length of a Java stack trace, when none of anybody's time can ever be allocated to fixing these things because that would not deliver any features, when all of one's time must be spent in meetings or updating JIRA tickets or talking to PMs who pretend to be customers but have never talked to a customer in their life, etc.

- Because of the difficulty meeting these commitments, more processes must be implemented in an attempt to standardize the output of the team.

- Maybe continue by reading The Gervais Principle?

- It's possible that this isn't "Agile" and it's just "Micromanagement" but it certainly seems like 99% of people who say they are doing "Agile" are doing this stuff, and at some point one has to give up on linguistic prescriptivism.

Aside about Elon Musk and Twitter: It's correct that Twitter was run in a grossly incompetent manner up until now, and that Parag Agrawal is probably the worst tech CEO ever, but a lot of Elon Musk's changes so far have been to destroy the institutional knowledge needed to keep the site running at all by firing the ICs who have that knowledge. This does not seems like a great approach to say the least.

reply