Andrew Harmel-Law @al94781

22 Dec 20 • 58 tweets • al94781/status/1341319003374039040

Software architecture is in crisis, and the way to fix it is a hefty dose of anarchy.

Some lay the blame for this on @boicy with the whole microservices thing.

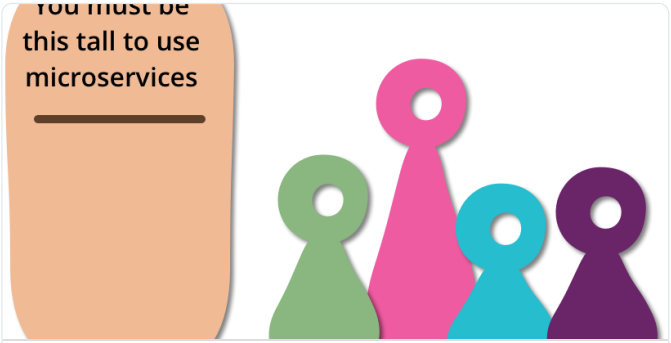
(Admittedly, @nicolefv, @jezhumble and @realgenekim didn't help when they statistically proved that he might have been onto something with all that de-coupling and team-alignment…)

However I don't blame him at all.

I think he saved us; bringing us back to the path of value-delivery and independent services, but now with added independent teams.

But one thing is clear. Microservices need more architecture, not less (as do other forms of #Accelerate-style software organisation).

(See



bliki: MicroservicePrerequisites

There are certain things you need to get sorted out before you can put your first microservices system into production: monitoring, provisioning, and a devops culture.

https://www.martinfowler.com/bliki/MicroservicePrerequisites.html

if you need convincing)

I mean, all those pesky slices we need to carve up our monoliths (or were they big balls of mud?) That's a significant amount of work right there...

The truth is, the vast majority of attempts at - and ensuing aftermaths of - such slicing and continuous delivery have highlighted problems for almost all

organisations, even the ones who were doing great architecture: How to scale their architects.

One of the biggest problems? Suddenly architects (and I include myself in this group) needed to be in far too many places at once, doing all that "architecture".

And so to cope, we architects either kept doing our job and became bottlenecks, or admitted defeat / got circumnavigated, or worse still went back to code and created “frameworks” which helped teams stick to the true path.

Shudder

None of these approaches have worked. And they never will.

And consequently many, many #microservices adoptions failed; with #microservices themselves getting an undeserved bad name in the process.

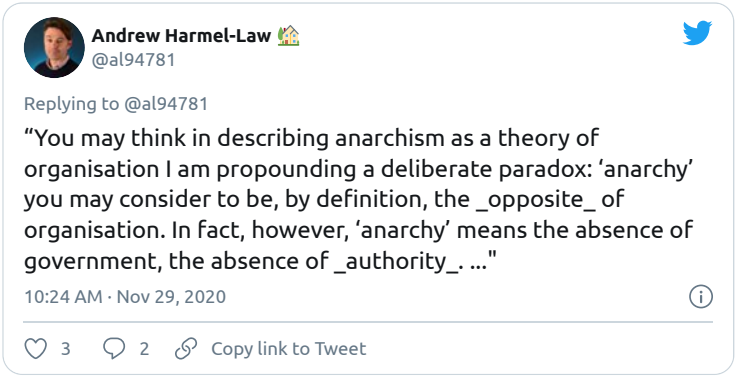
But microservices aren’t a curse on software delivery - and they ought to be a blessing.

What we need is a workable way to approach them, and in the process realise the associated benefits of both team autonomy and improvements in system architecture.

In the remainder of this thread I’m going to introduce the idea of an #AnarchisticArchitecture.

I’ll describe what it is and you could do it. Hopefully you’ll see how it offers the best (only?) way out of this mess.

Let's remind ourselves first what "#anarchy" means: It's the absence of government. The absence of authority.



Straight away that means how we are used to doing architecture, via all-powerful architects taking all the decisions, is going to have to stop.

But decisions still need to get made - that's what architecture is.

(As @Grady_Booch has said "architecture represents the set of significant design decisions that shape the form and the function of a system, where significant is measured by cost of change.")

@martinfowler agrees "software architecture is those decisions which are both important and hard to change".

So the first test is can an #AnarchisticArchitecture deliver on this?

The answer is "Yes".

It's clear that if we want to succeed with an #AnarchisticArchitecture we cannot do away with decision-making.

What's more, our architectural decisions made must still be made deliberately - otherwise we'll be back where we started, or worse.

So how are decisions made in an #AnarchisticArchitecture?

They are "run from below"

How could an #AnarchisticArchitecture approach achieve this? (Remember, to impose something specific wouldn't be very anarchic.)

Personally, I've had great successes using the #AdviceProcess for decision-making. In fact, it's the core anarchic element https://reinventingorganizationswiki.com/theory/decision-making/.

The #AdviceProcess is remarkably simple. There is one rule and one qualifier:

1) Anyone can make an architectural decision (and I mean anyone).

But before making the decision, they must consult two groups of people...

The first group are all those who will be meaningfully affected by the decision.

The second group are all those with expertise in the area where the decision is being taken.

That's it. Thats the #AdviceProcess.

Now, the decision-takers aren't obliged to agree with the advice the folks in these two groups give them, but they MUST seek it out, and they MUST listen to it.

Using the #AdviceProcess for decision-making (or something else which meets the anarchistic principles - it wouldn't be very anarchistic to mandate it) is the most fundamental element of an #AnarchisticArchitecture.

This decentralised decision-making is the core element of any #AnarchisticArchitecture because if it's not adopted, you're simply not anarchistic.

But if you use this core element alone, you are sadly unlikely to realise the benefits of the anarchistic approach.

Why? I'm going to bring in some further famous-architect assistance to help make this point.

There's one more definition of architecture - one which I particularly love - that the power of an #AnarchisticArchitecture approach are crystalised.

It comes from @DanaBredemeyer with commentary @ruthmalan.

'Dana Bredemeyer introduced the ideas of architecture needing to be "(technically) good, right (e.g. fit for context and purpose) and successful (in that it is delivering value in a sustainable way)"'

from https://www.ruthmalan.com/Journal/2016/2016JournalFebruary.htm#Still_Need_Architects by @ruthmalan

Let's return briefly to where we began. Why have #microservices failed in so many places to date? - iI've argued that it's because the intended architecture, if there was one, didn't get shipped to production.

To be successful, an architecture needs to be RUNNING.

@RuthMalan suggests how we might remedy this:

'The last ["Successful"], is speaking to the organisational dimension of the architect's role that is missing from "right system [right], built right [good]"' she writes.

"It is very much about ensuring that conversations that are needed to be happening are happening - not always initiating them, nor always helping to focus or navigate them, but ensuring they do happen [...] and guiding when needed" @RuthMalan concludes.

We can now return to an #AnarchisticArchitecture.

Our adoption of the advice process opened up the space for anyone to make decisions, but this alone does not guarantee success. We've simply removed the possibility of more traditional architectural approaches making us fail.

The remainder of the elements of an #AnarchisticArchitecture are focused on enabling @DanaBredemeyers "success". They are aimed at supporting

@RuthMalan's admonition to "ensure that conversations that are needed to be happening are happening".

There are four supporting elements:

* The first is a forum for conversations;
* The second is a thinking and recording tool;
* The third is a light to illuminate a unified direction;
* And the fourth allows for sensing the current landscape and technical climate.

The first supporting element in an #AnarchisticArchitecture is something to make the conversations supporting all this advice-seeking easier: an Architecture Advisory Forum. (#AAF).

Our current #AAF is a weekly, hour-long meeting were representatives from all the development teams gather and share their active Spikes and in-progress decisions.

Also attending are the typically "also affected" and "expert" group members.

That means we have QAs, UX, Product, leads from other teams elsewhere in the organisation, Chief Architects, ops, line managers, etc. etc.

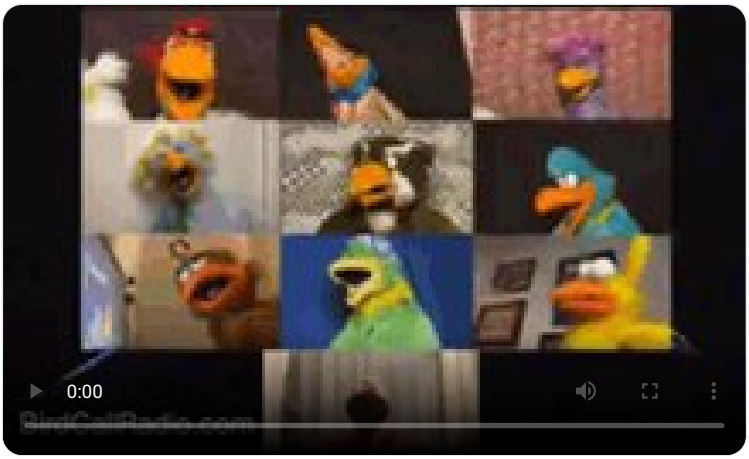
The #AAF is a great forum for having loads of great discussions about all things architecture. It's also great for flagging others who might not be in attendance, but who should be consulted.

The only rule is that the advice process rules - it's the reason its called the #AAF rather than the AR(review)F or AD(ecision)Forum.

Decision-making still lives with the people needing to make the decisions. It's still the #AdviceProcess.

I've learned a lot about tweaking various elements of an #AAF to make it well attended (in our current one we typically get 20+ attendees with at least half contributing) and focused. But that's a topic for another thread.

It should be pointed out, #AAFs are not only useful, they're fun too. Many people actively look forward to them, because it's where they learn, get great feedback and input on their pending decisions and Spikes, and also show off what they're doing. They're social.



When pending decisions are shared at #AAF they're presented in the shape of the second supporting element of an #AnarchisticArchitecture: Lightweight Architectural Decision Records

Lightweight Architecture Decision Records | Technology Radar | Thou…
This Technology Radar quadrant explores the techniques being used to develop and deliver software
https://www.thoughtworks.com/radar/techniques/lightweight-architecture-decision-reco…

We've found that a lightweight template structure helps folks learning to make architectural decisions. What is more, we've scaled our decision-making and reinforced the #AdviceProcess by encouraging the leaving of comments on #ADR pages.

This means everyone can see the thoughts of people who felt they ought to provide input as a permanent record.

More importantly, the #AdviceProcess can happen asynchronously, speeding everything up, and leaving the #AAF as a forum for the key discussions, and awareness-raising.

The third and fourth supporting elements of an #AnarchisticArchitecture are represented in two key fields on our #ADR template: "Applicable Principles" and "Relevant Radar Blips". I'll deal with them in turn.

Having Architectural Principles is not new. Neither is our approach which is 100% stolen from the excellent "Art of Scalability" by Abbot and Fisher theartofscalability.com.

While their book isn't written from an #AnarchisticArchitecture perspective, the authors realise that for principles to be successful, teams which deliver against them need to feel ownership.

Check out chapter 12 of their book for everything you need to run an Architectural Principles Workshop (it's in the section "Engendering Ownership of Principles").

(And while the JADs and ARB's in the following chapter don't fit the #AnarchisticArchitecture approach we're following here, they do contain some great thoughts on when to update the principles.)

One final point on the principles. They're only principles, and while they come from the teams themselves, sometimes they simply don't apply.

Rather than being a failure of an #AnarchisticArchitecture, it's actually another strength - because being unable to meet a principle is a *great* time to go the #AAF and discuss things, and then detail all the reasons why you're deviating in the resulting document.

That's principles covered - which play the role of a guiding light for everyone to aspire to. But how do we also take note of our surrounding landscape and climate? Architectural decisions are also frequently based on what everyone else is doing, and who has which skills.

Enter the fourth supporting element of an #AnarchisticArchitecture: the Technology Radar.

Technology Radar | An opinionated guide to technology frontiers | Tho…
The Technology Radar is an opinionated guide to technology frontiers. Read the latest here.

https://www.thoughtworks.com/radar

Lots of people have heard of the @ThoughtWorks #TechnologyRadar. Sadly, far fewer know about the fact you can build your own. Check out

Build your Own Radar | ThoughtWorks
Learn how to use our radar creation exercise to have a conversation across all organizational levels and review your entire technology portfolio. Find out more.

https://www.thoughtworks.com/radar/byor

The best way to make your own radar is to kick it off with a session which captures what you have now in your organisation - a baseline sweep or scan if you will.

This too ought to be run as a workshop. I've done them where we keep the same quadrants, but we replace the "trial" ring for one called "retire", allowing us to actively track the lifecycle of tools, techniques, platforms and languages/frameworks through the organisation.

So there we have our five elements of an #AnarchisticArchitecture; one core and four supporting:

1. Advice process
2. Architecture Advisory Forum
3. Lightweight ADRs
4. Team-sourced Principles
5. Your own Tech Radar

Of course it's possible to go further. We've seen that teams working in this way will naturally start to pave their own roads - self-serving their own delivery platform before a Platform Team comes into existence. (See @TeamTopologies for waaaaaay more detail on this.)

I've also seen teams put in place their own #ArchitecturalFitnessFunctions - so that they know when the collective #AnarchisticArchitecture strays outside its intended bounds.

Finally I've seen teams evolve their architectures - putting re-visit dates on #ADRs so decisions can be re-considered in light of additional information, and retiring / superseding decisions which have gone out of date.

That's it. That's the thread. Apologies again its so long (ideas breed other ideas, what can I say.)

All feedback and comments are v v welcome. (apart from "I don't like long threads" - there's @Unrollme for that) Particularly areas where this won't work. (Constraints #FTW!)

• • •