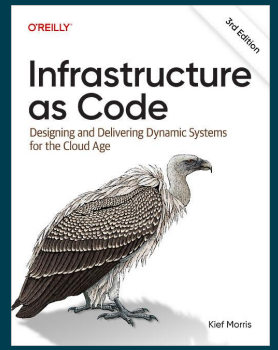
# Pulling Continuous Delivery inside the agentic loop




**Kief Morris**

Distinguished Engineer

**Technology Advisory Services**

How do we use GenAI to build software to last?

/thoughtworks

https://bit.ly/4dS28pT


I see value in hands-off, "all-in" agentic workflows.
thoughtworks
Photo by Homa Appliances on Unsplash

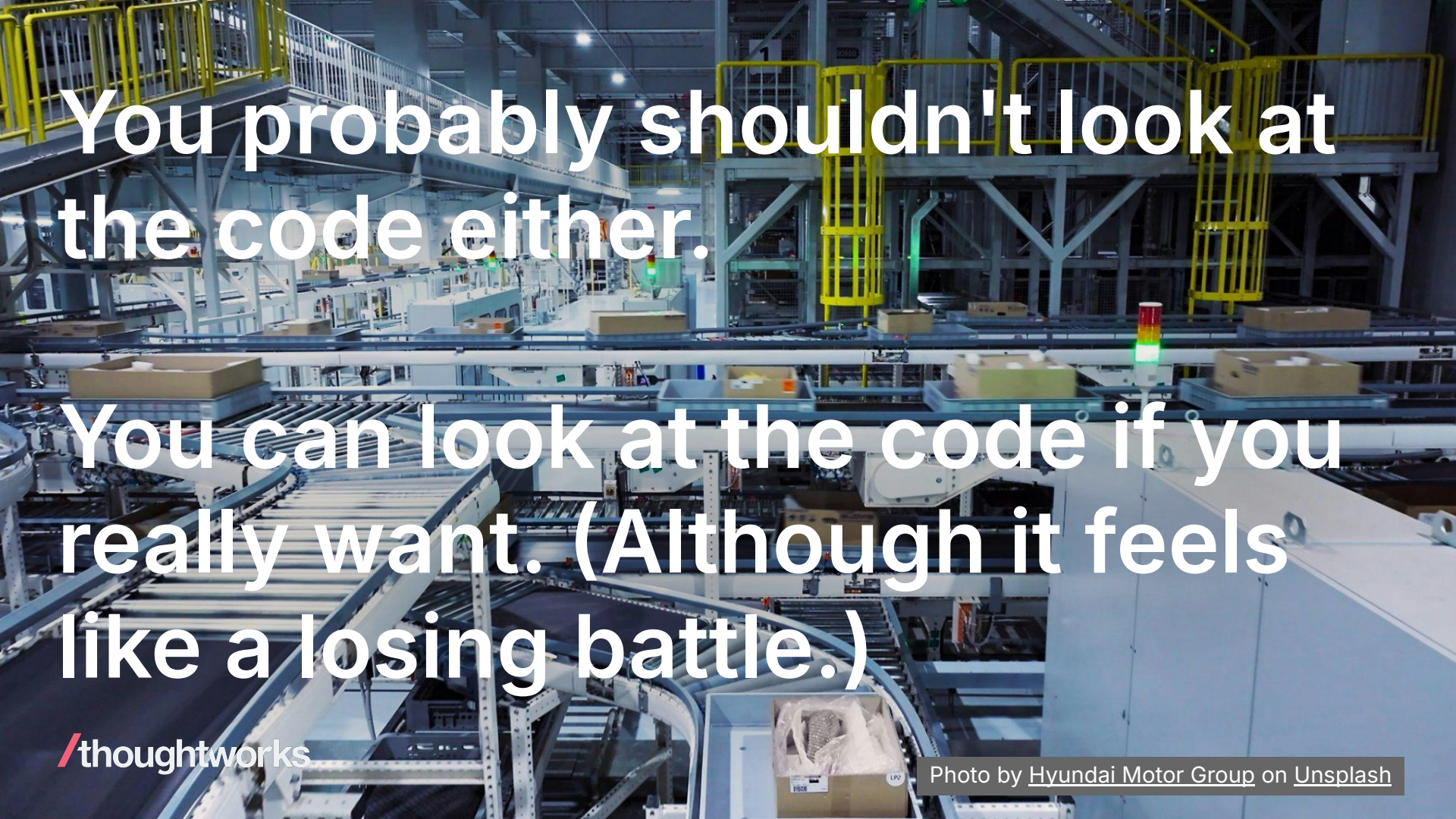

But it means
getting better at
ensuring "good".
thoughtworks
Photo by Homa Appliances on Unsplash


Which means being clear about how we define "good":
thoughtworks
Photo by Homa Appliances on Unsplash

I believe using CD pipelines within agentic workflows can help build better software.


All-in, "hands-off" agentic workflows.
/thoughtworks
Photo by Hyundai Motor Group on Unsplash


You don't touch code.
Even when you don't like the code that that the agents extrude.
/thoughtworks
Photo by Hyundai Motor Group on Unsplash


You probably shouldn't look at the code either.
You can look at the code if you really want. (Although it feels like a losing battle.)
/thoughtworks
Photo by Hyundai Motor Group on Unsplash

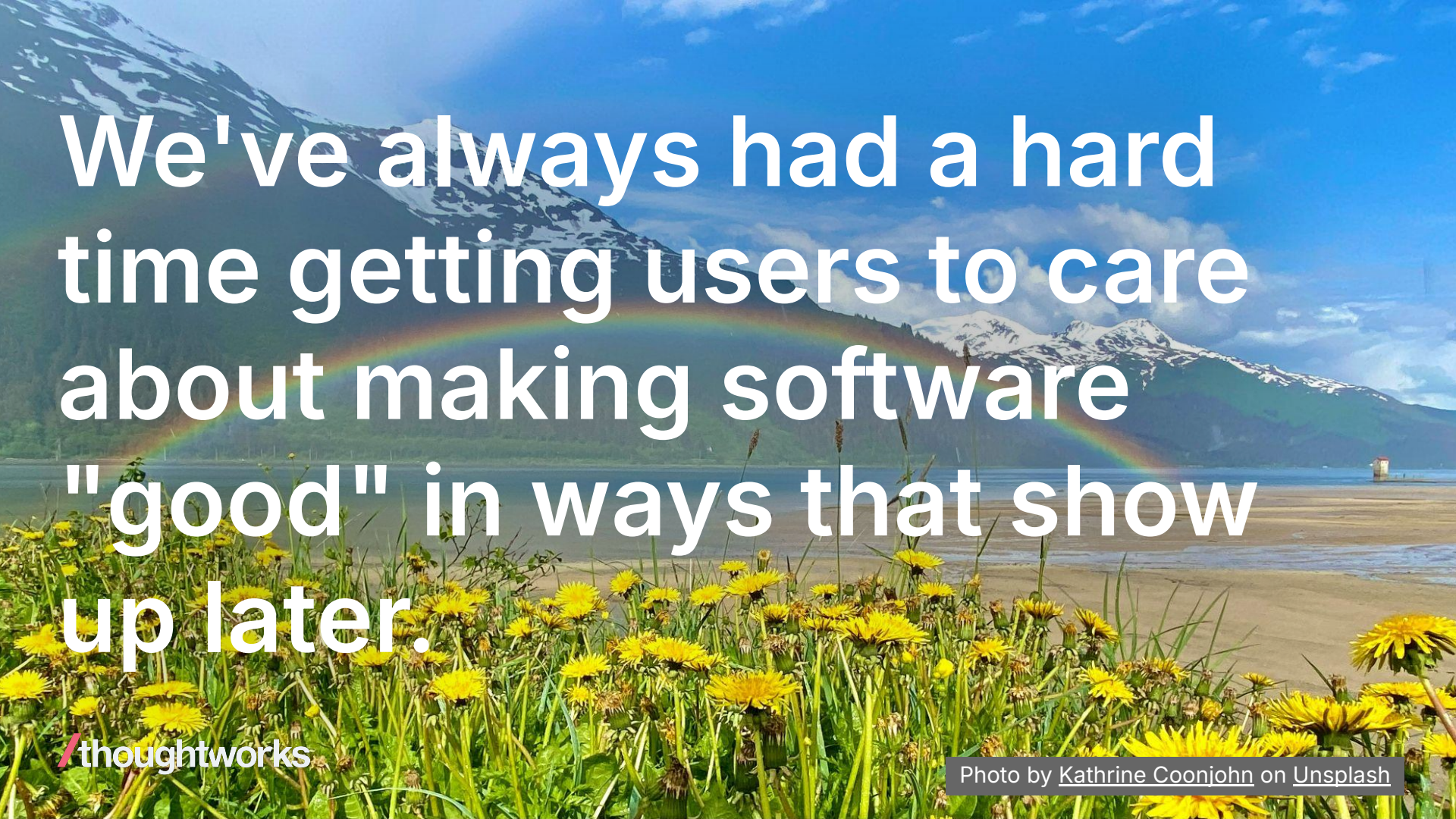

So how do we avoid cognitive debt?
How do we avoid cognitive surrender?
/thoughtworks
Photo by Peter Herrmann on Unsplash


Do we really need our code to be good?
/thoughtworks
Photo by Peter Herrmann on Unsplash


It's the LLM's problem now!
/thoughtworks
Photo by Peter Herrmann on Unsplash


Vibe-oriented approaches build software that is good in the ways that users know they care about.
"It does what I want it to!"
thoughtworks
Photo by Kathrine Coonjohn on Unsplash

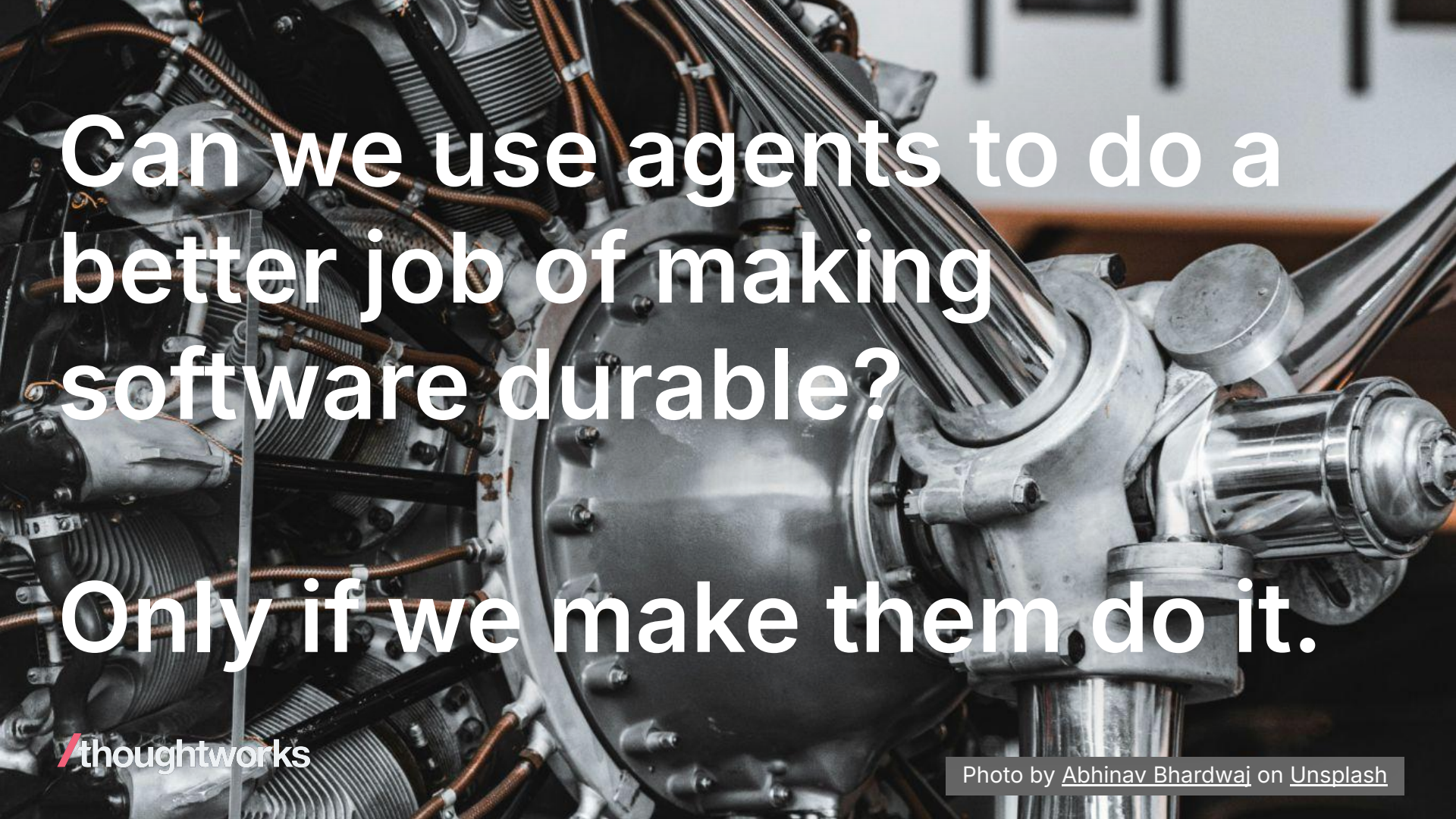

But does GenAI build software that is good in the ways that users will care about later?
Not unless you make it do it.
thoughtworks
Photo by Kathrine Coonjohn on Unsplash


We've always had a hard time getting users to care about making software "good" in ways that show up later.
thoughtworks
Photo by Kathrine Coonjohn on Unsplash

And maybe that's fine for some software.

Maybe for a lot of software made with AI.

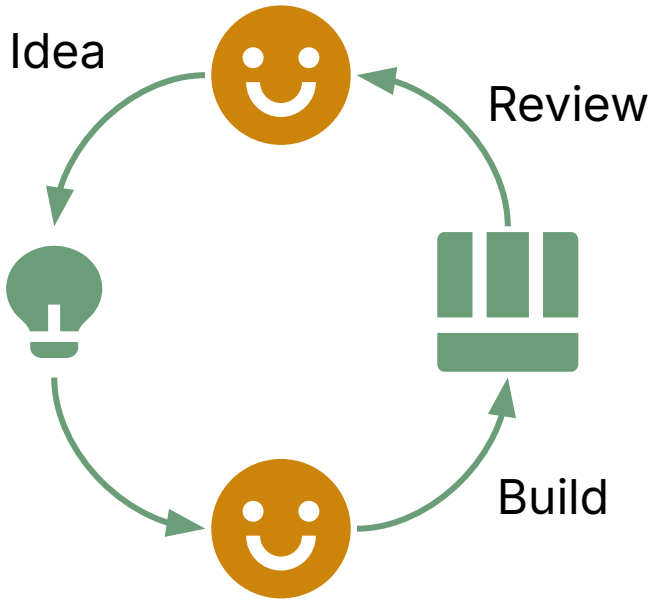

But what if we're building software that needs to last?
thoughtworks
Photo by Abhinav Bhardwaj on Unsplash


Can we use agents to do a better job of making software durable?
Only if we make them do it.
thoughtworks
Photo by Abhinav Bhardwaj on Unsplash

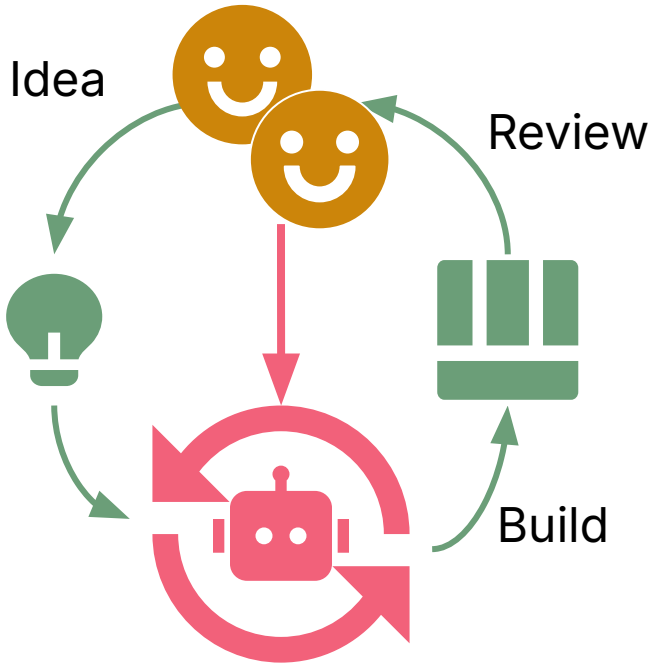

If do we get our agents to build software to last, that can have benefits even for software that we might not think needs it.
thoughtworks
Photo by Abhinav Bhardwaj on Unsplash

# Agentic coding harness:
# A system that drives the LLM to make good code.

# Legacy approach: Humans are the loop



# Agentic loop. Work on the harness, not the code

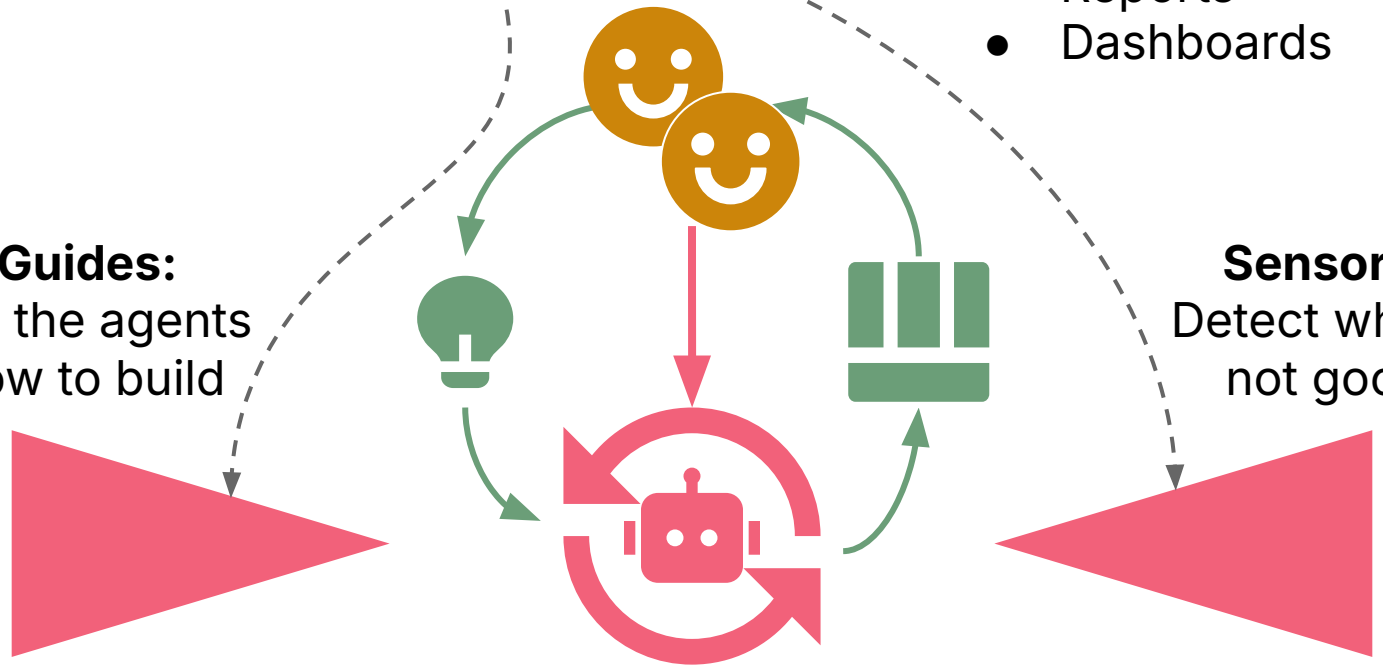



/thoughtworks

**Improve the harness**

**Check:**

- Use it
- Reports
- Dashboards



**Guides:**
Tell the agents how to build

**Sensors:**
Detect what's not good

**Loop:** Repeat until good

/thoughtworks

**Prerequisite:**

We need to clearly and effectively articulate what "good" looks like.

# Code quality?

Who cares!

It's the LLM's problem now!

# Code quality impacts DELIVERABILITY

We need to:

Change it. Configure it. Improve it. Add to it.
Troubleshoot it. Fix it.

/thoughtworks

# How do we measure deliverability?

## Lagging measures

DORA and things like that.

## Leading measures

Linting

Test coverage

Mutation testing, fuzz testing

Code complexity

Vulnerability and supply chain scanning

LLM code evaluations

# Implementation affects OPERABILITY

**We need to:**

Know when it breaks.

Restore it quickly when it does.

**We also need it to:**

Be fast enough.

Have enough capacity.

Be reliable.

Not cost too much to run.

Handle data properly.

Be secure.

Comply with laws and standards.

# How do we measure operability?

## Lagging measures

Deliverability metrics (DORA)

User satisfaction

Business operations metrics (transactions, cost)

Commercial metrics

## Leading measures

Deployability

Performance and scalability testing

Failure scenario testing

Penetration testing

Compliance auditing

# Continuous Delivery

thoughtworks

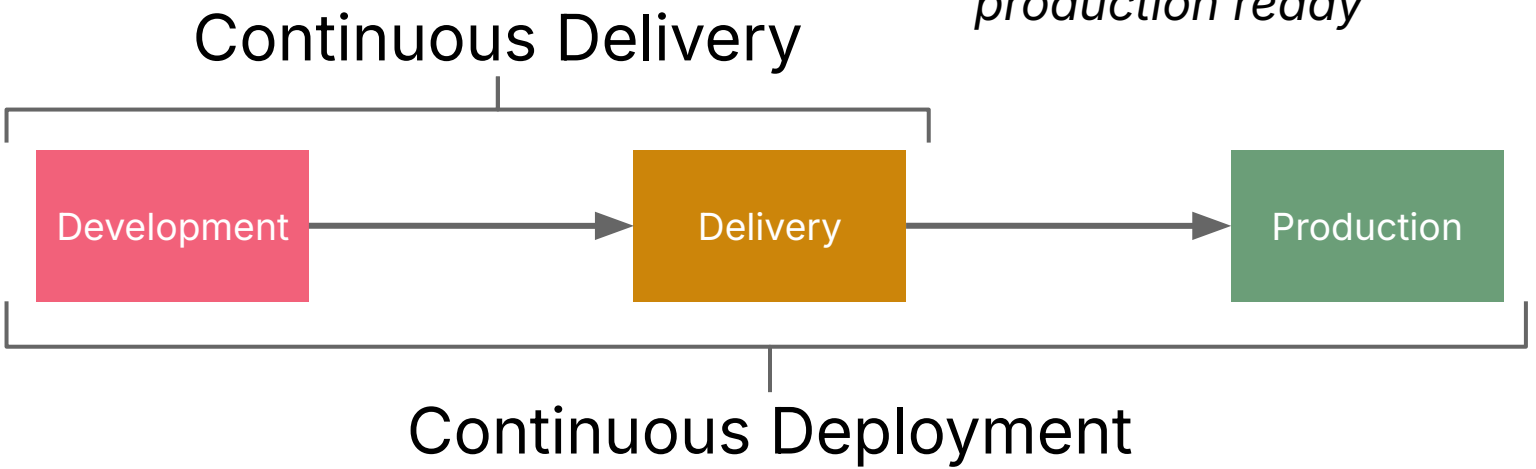
# Legacy software delivery flow


*“Done” means “code complete”*

*Make it “production ready” after it's done.*

# DevOps

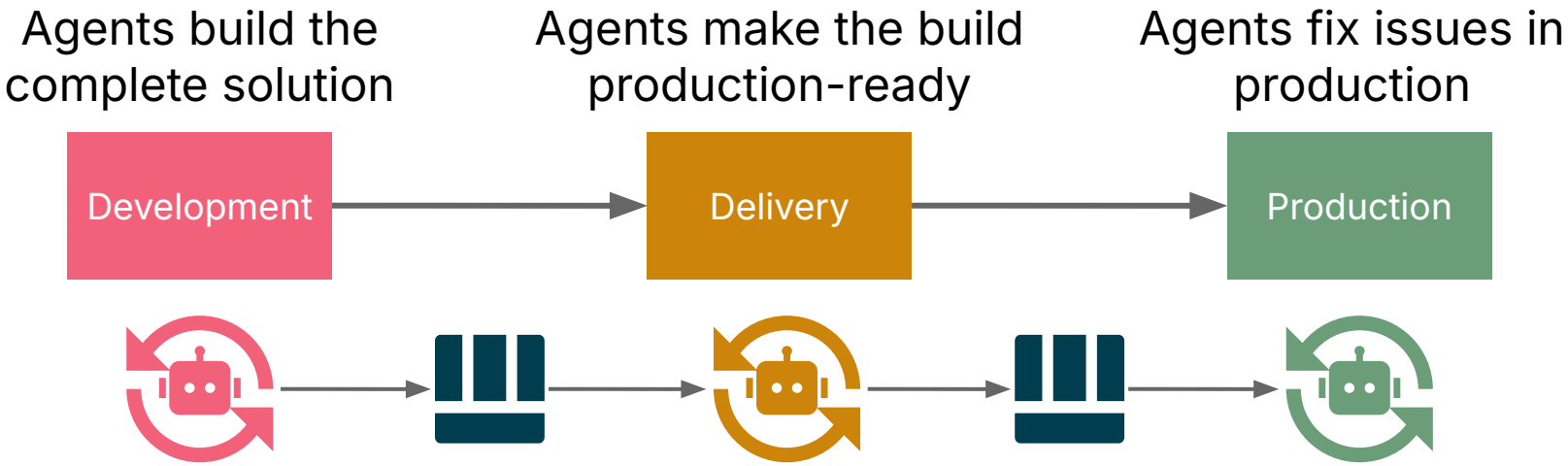
*Confidence that the codebase is always production ready*




*Done means "running in production"*

/thoughtworks

# Maybe we don't need CD with agentic engineering?



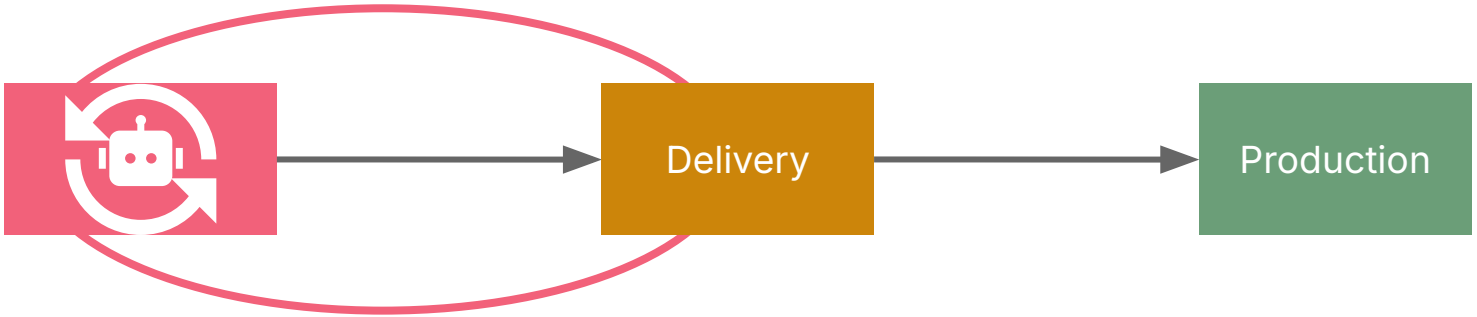

**Operability:** software + infrastructure + processes

/thoughtworks

# We can use a Continuous Delivery pipeline as an operability harness

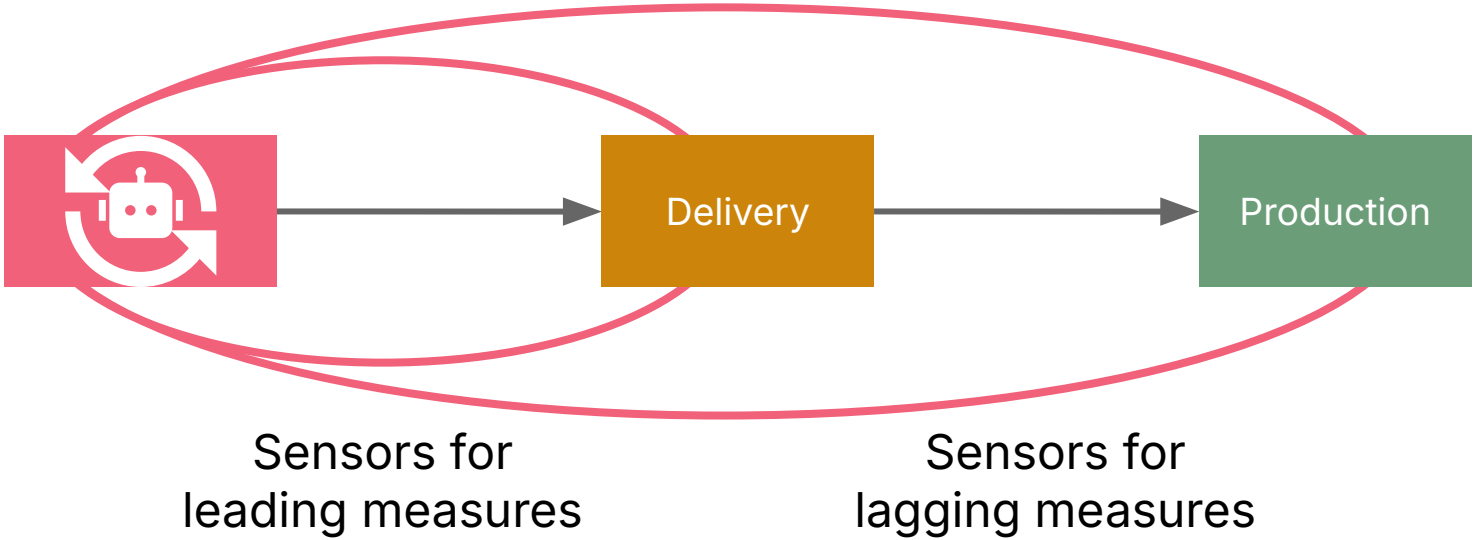
/thoughtworks

# Expanding the agentic loop to include deployment



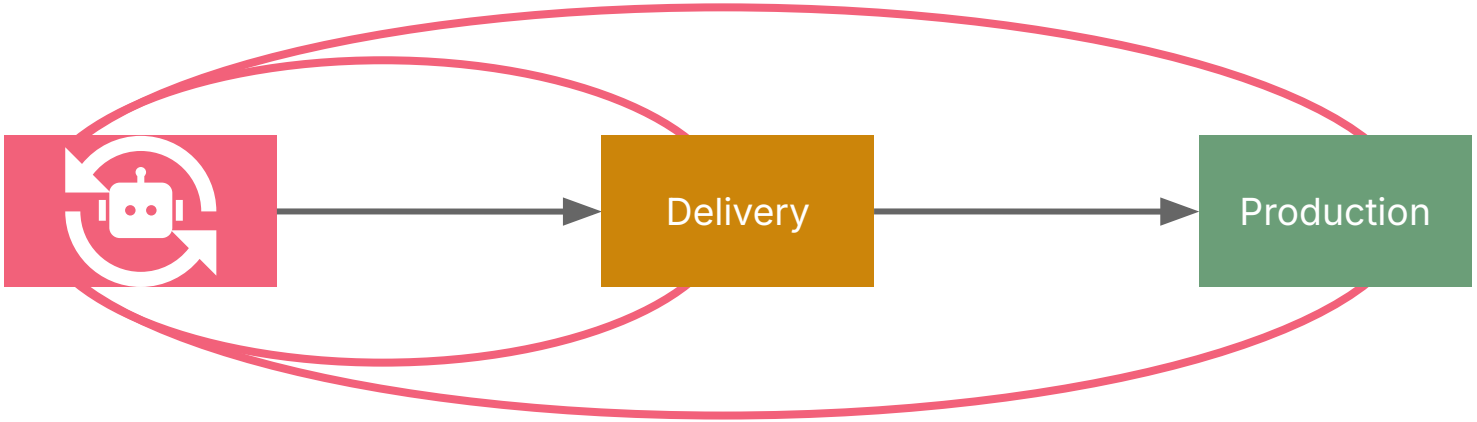

Sensors for
leading measures

# Expanding the agentic loop to include production



# Challenge: Agent makes fixes downstream

**Solution:** Prevent access to the deployed infrastructure




"Watch the job. If it fails, replicate locally, fix the code, push again

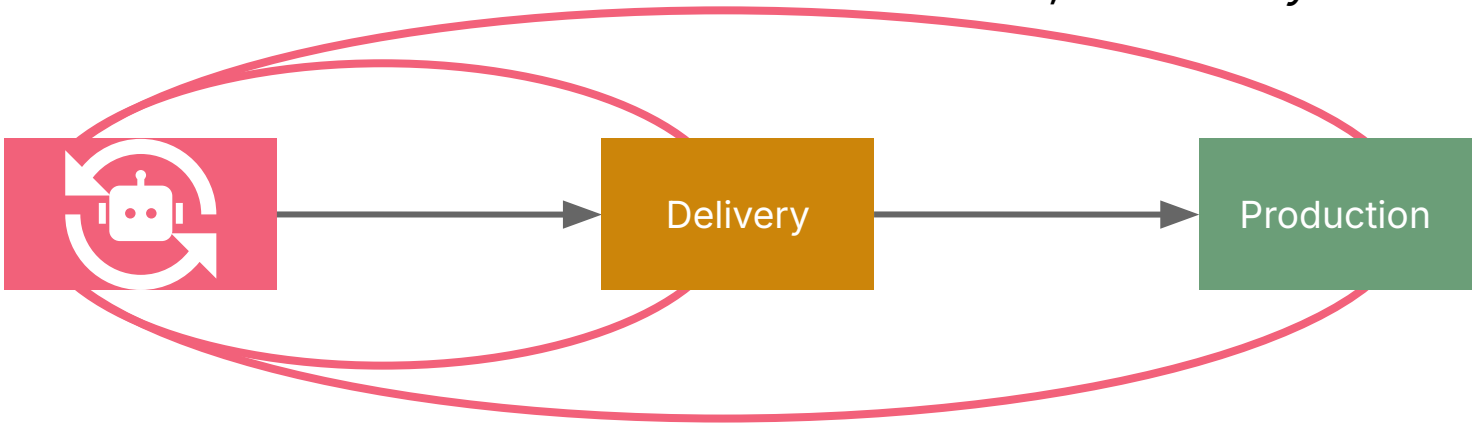
# Challenge: Doesn't take responsibility

Agent: *"I fixed it!"*

Me: *“The build is still red?”*

Agent: *“It was red before my change.”*

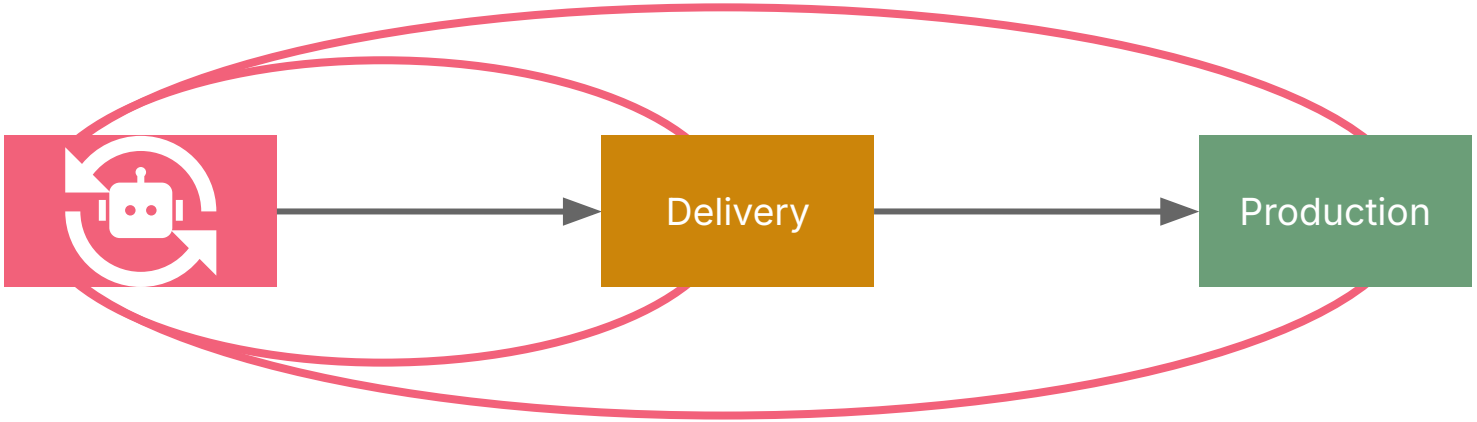
Me: *“Yeah, because you broke it!”*




**Solution:**
Skills for build discipline (?)

# Challenge: Worked on my machine!

**Cause:** Over-fitting

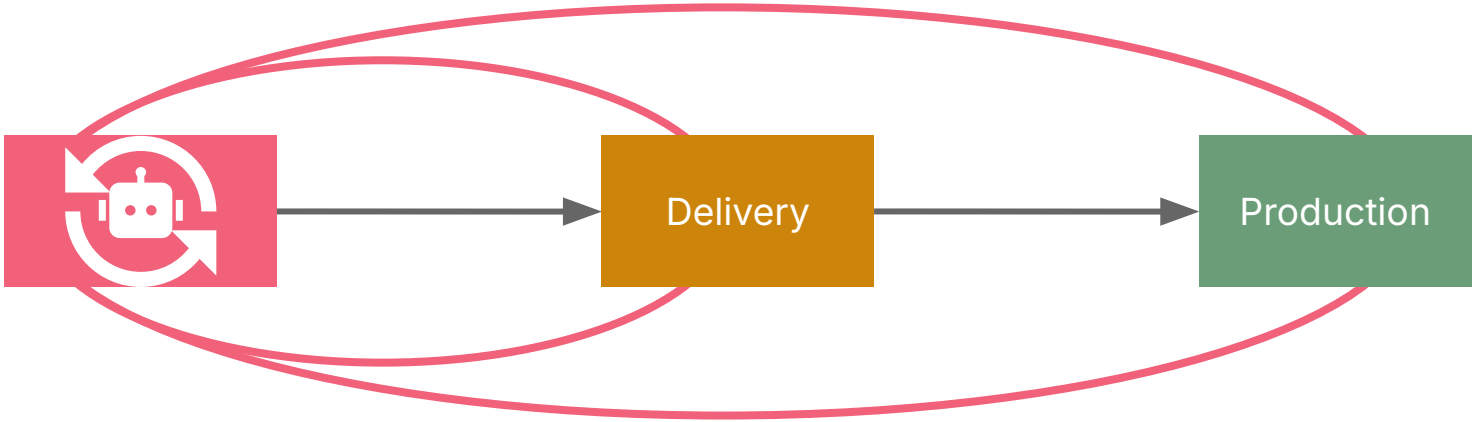



**Solution:**
Test environment parameterization earlier

/thoughtworks

# Challenge: Deploying is expensive

**Plus:** Agents get it wrong. A lot.




**Solution:**
Better local testing

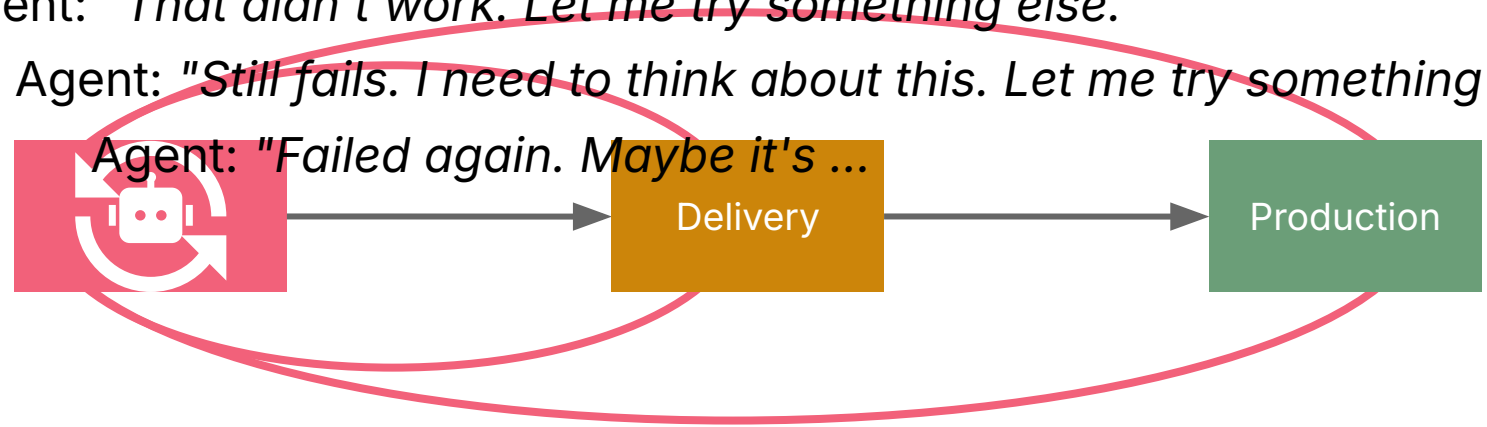
# Challenge: Agents fail a lot

Agent: *"Hmm. The deployment failed. Let me try something else."*

Agent: *"That didn't work. Let me try something else."*

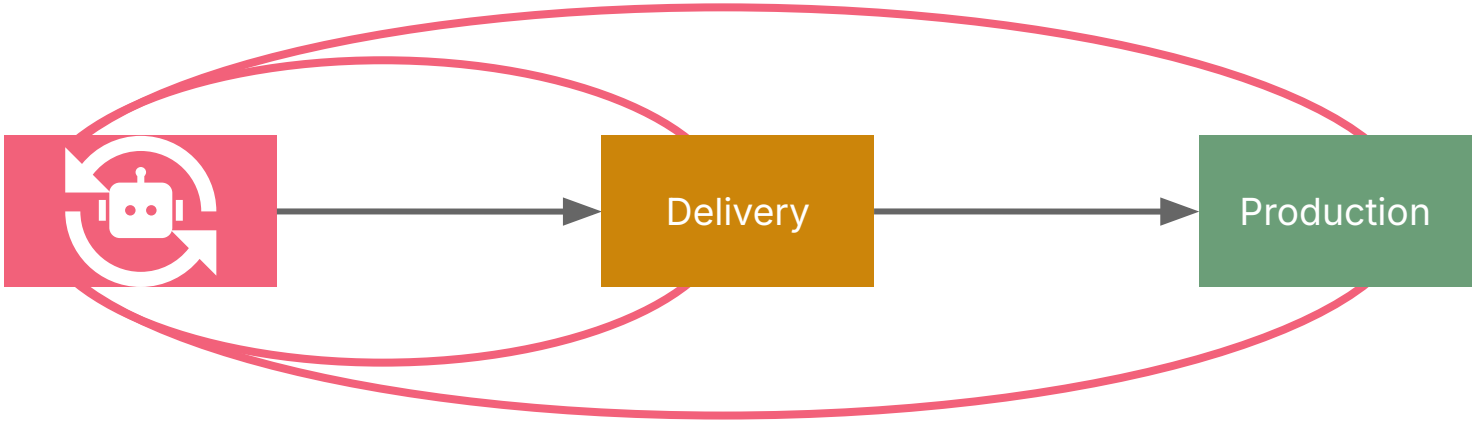
Agent: *"Still fails. I need to think about this. Let me try something else."*

Agent: *"Failed again. Maybe it's ...*



# Challenge: Agents suck at deploying

**Cause:** LLMs don't actually ***know*** how to deploy.




**Solution:** Detailed skills. "Here's how to deploy. Step 1 ..."

**Solution:** Prefer scripts and tools over markdown.

**Solution:** Composability.

**Solution:** Better models. (Or something else?)

# What I think

We can learn a lot with "all-in", hands-off agentic loops

LLMs don't care about building software to last

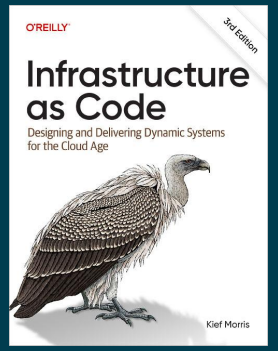
We need to figure out how to make them do it anyway

CD pipelines work pretty well for for humans

Put CD pipelines inside the agentic loop to learn

We will learn better ways

# Pulling Continuous Delivery inside the agentic loop




**Kief Morris**

Distinguished Engineer

**Technology Advisory Services**

How do we use GenAI to build software to last?

thoughtworks

https://bit.ly/4dS28pT