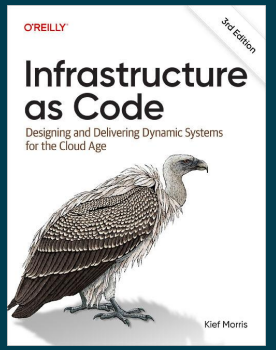


Humans on the Loop

Taking Agentic Engineering End to End

Kief Morris

Distinguished Engineer

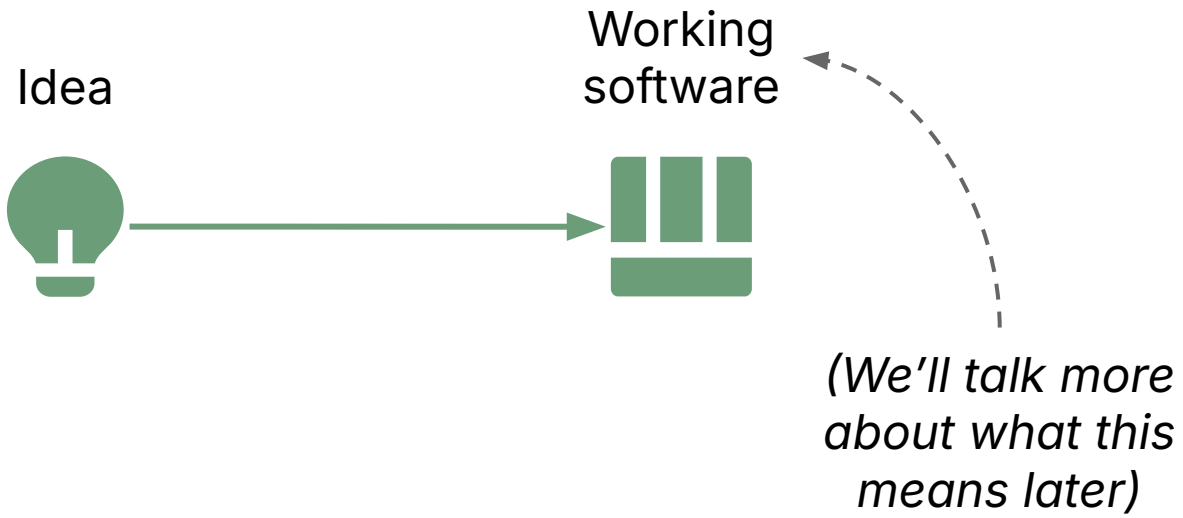
Technology Advisory Services

thoughtworks

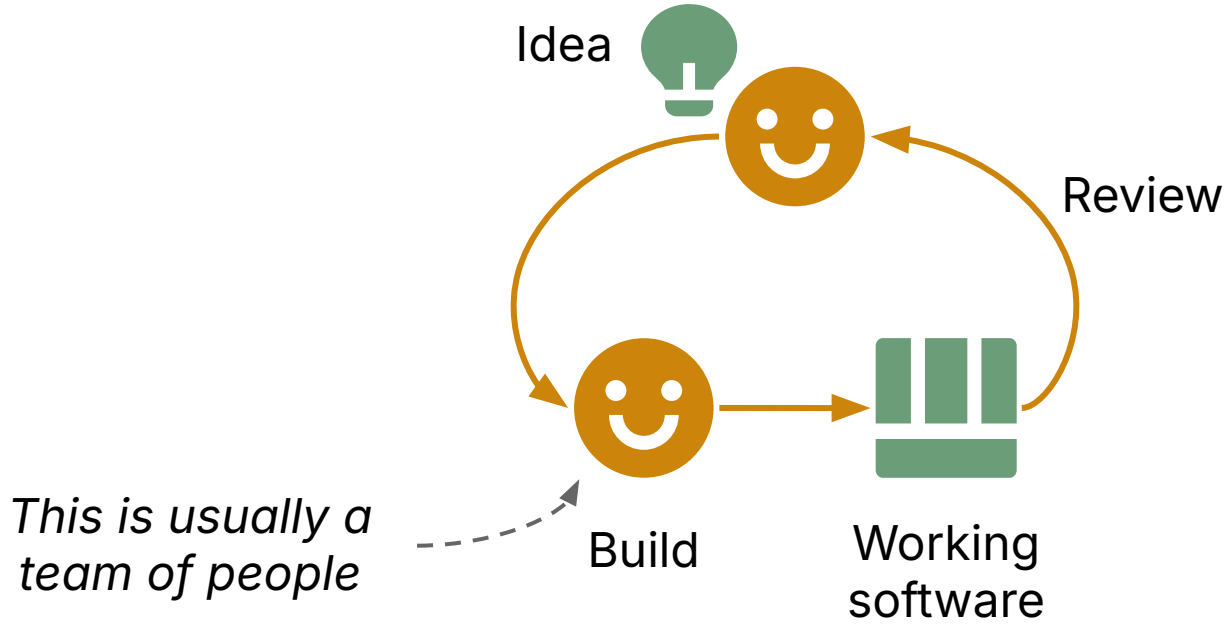
https://bit.ly/4uNbOYj

Where and how should humans review what their agents are building?

The goal

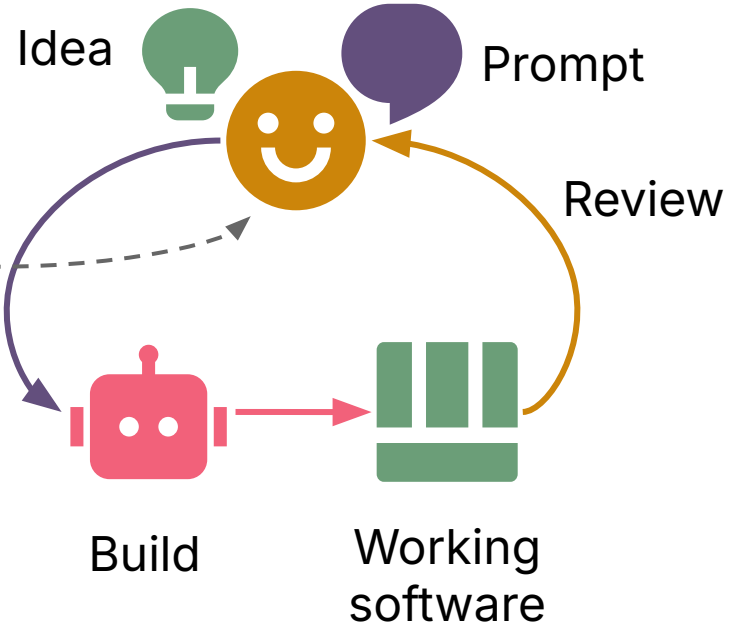


Legacy approach: Humans are the loop



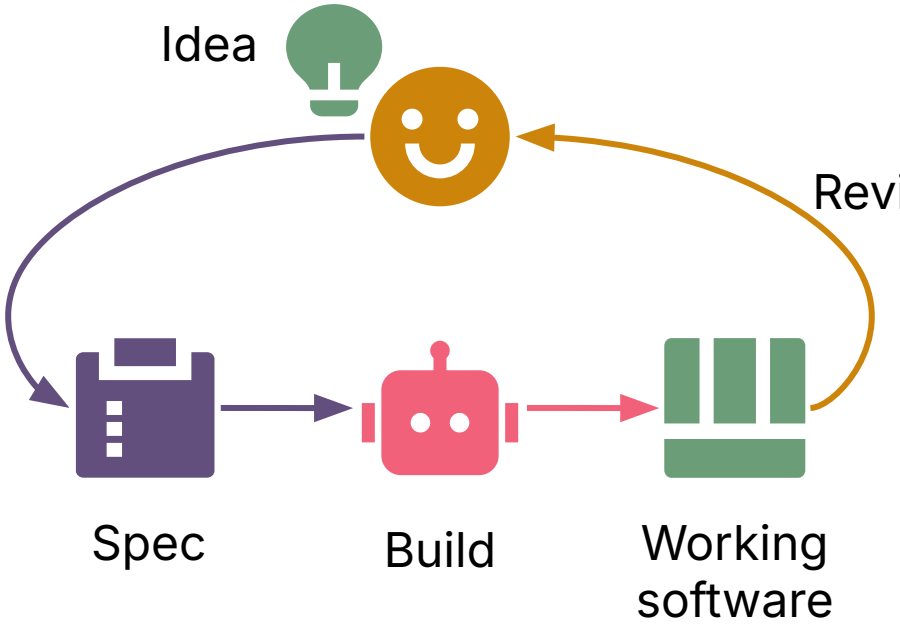
Vibe coding (basically)

This could be a team of people



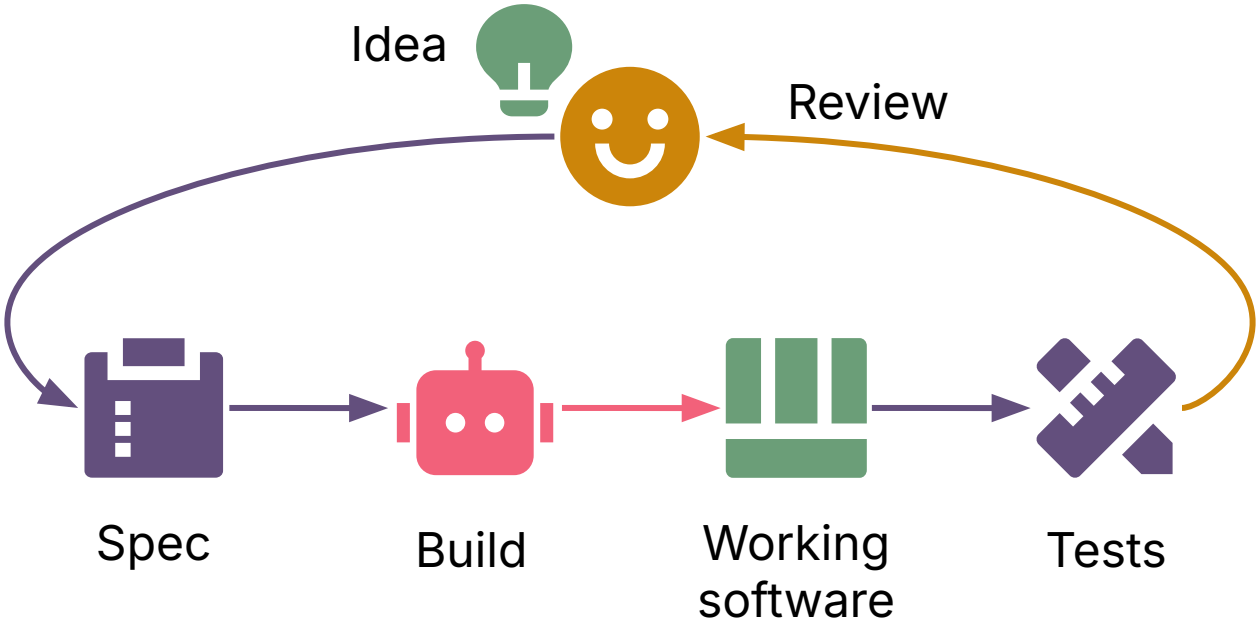
It depends on what "working" software means for the solution being built

Spec-driven development (basically)

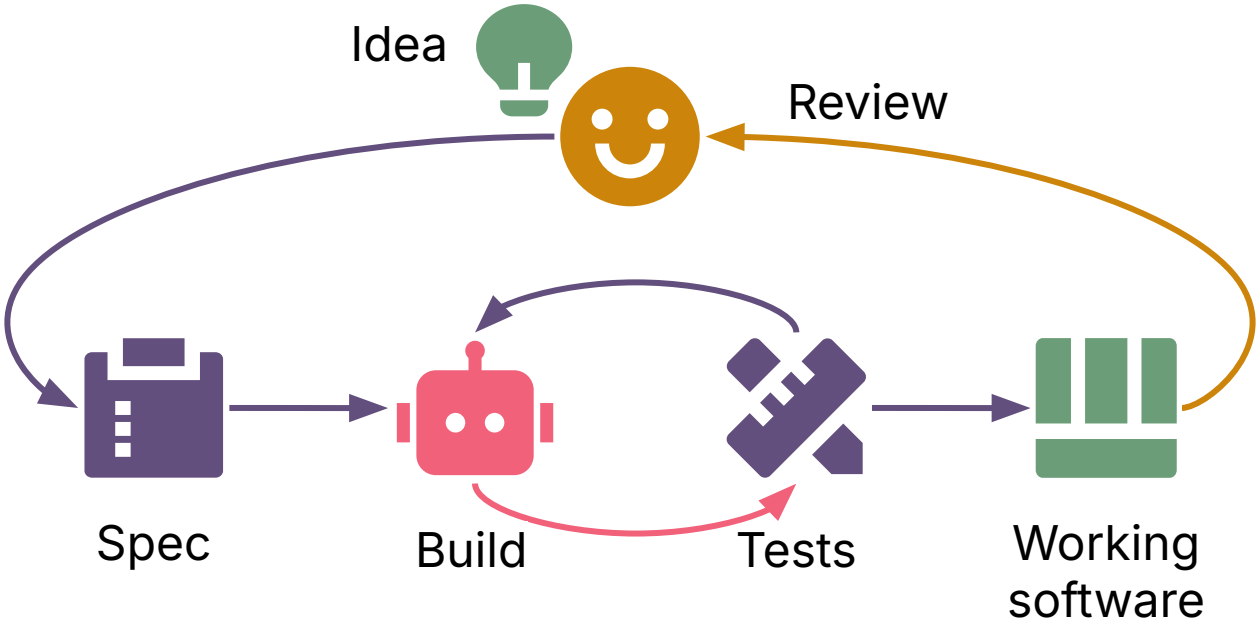


Code reviews can become a bottleneck

Spec-driven development with tests

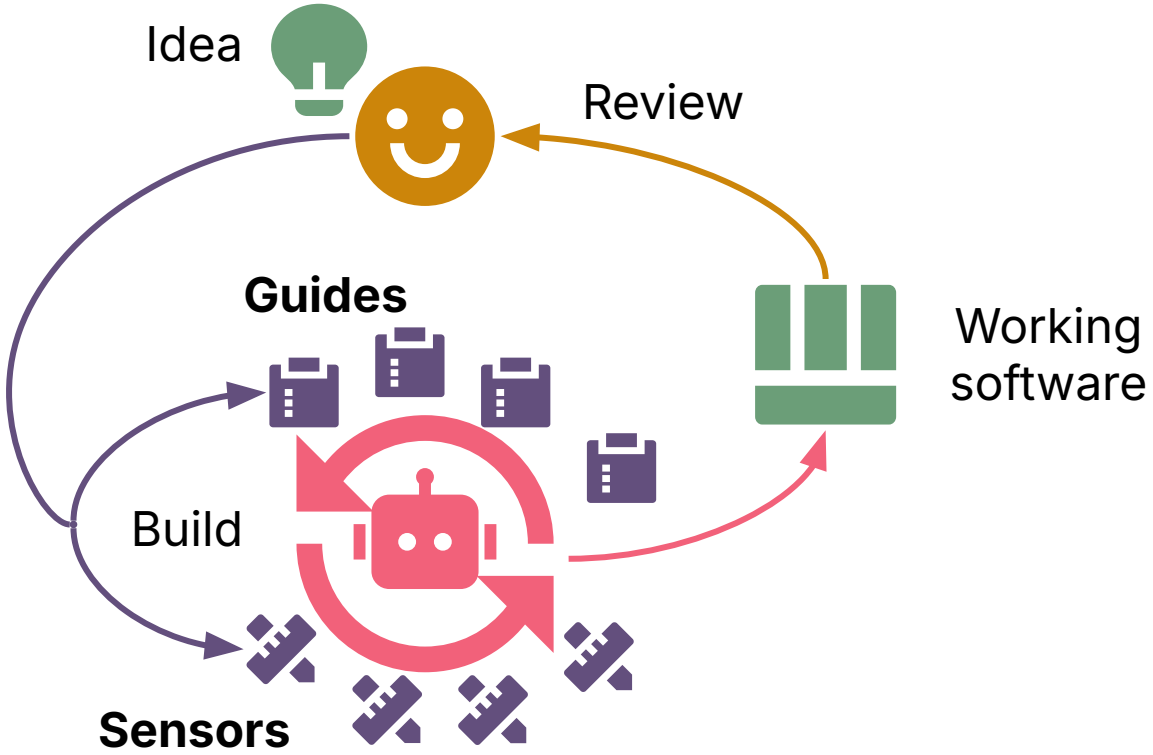


Agentic code / test loop



Agentic harness

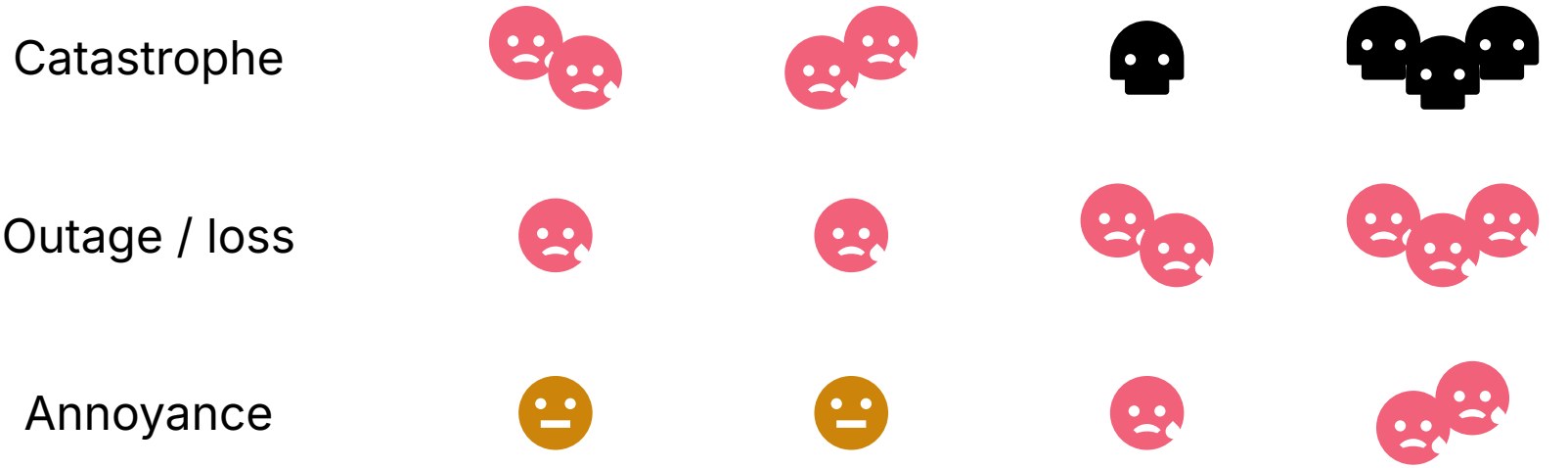
"Building the system that builds the software"



What do we mean by “working” software?

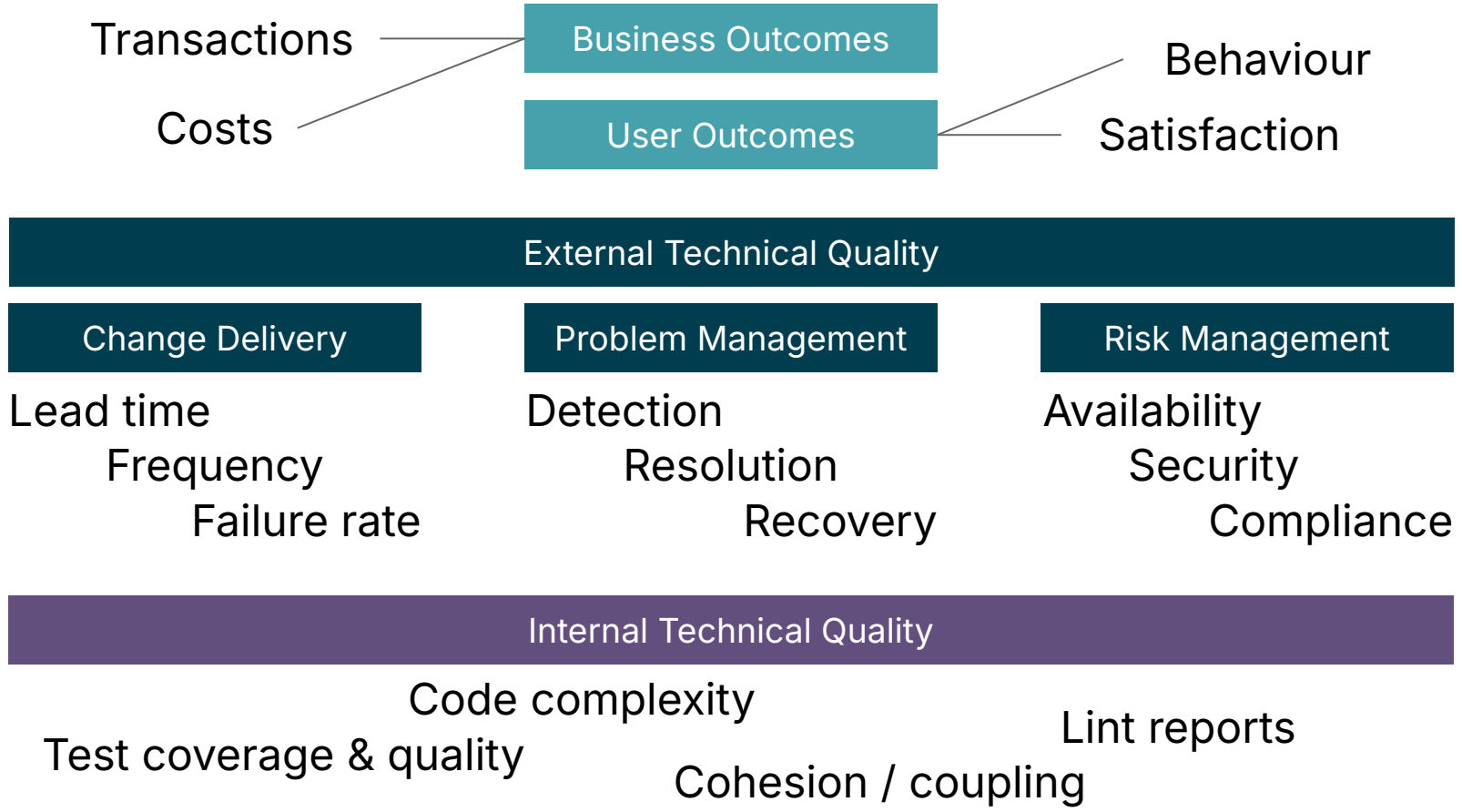
“It depends”

What happens if it goes wrong?



Who uses it?

“Working software”:
Reliably produces good outcomes



Proving the software is “working”

Where we prove that what we're building works

Fast, cheap(ish) *Slow, more expensive* *Late, risky*

Low fidelity	Higher fidelity	Most useful
Simulated functionality	Integrated functionality	Actual outcomes
Code quality	Deployability	
	Operability	

Legacy software delivery flow

“Done” means “code complete”

Seems logical

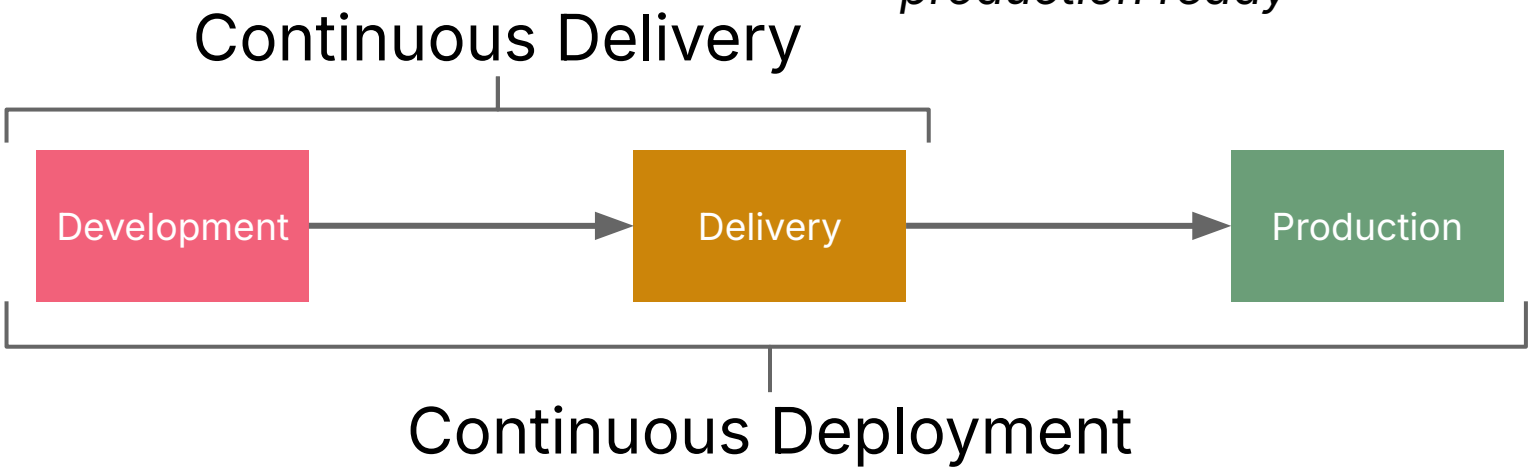
“Production ready” comes at the end

More problems emerge here than expected.

Always.

DevOps

The codebase is always production ready



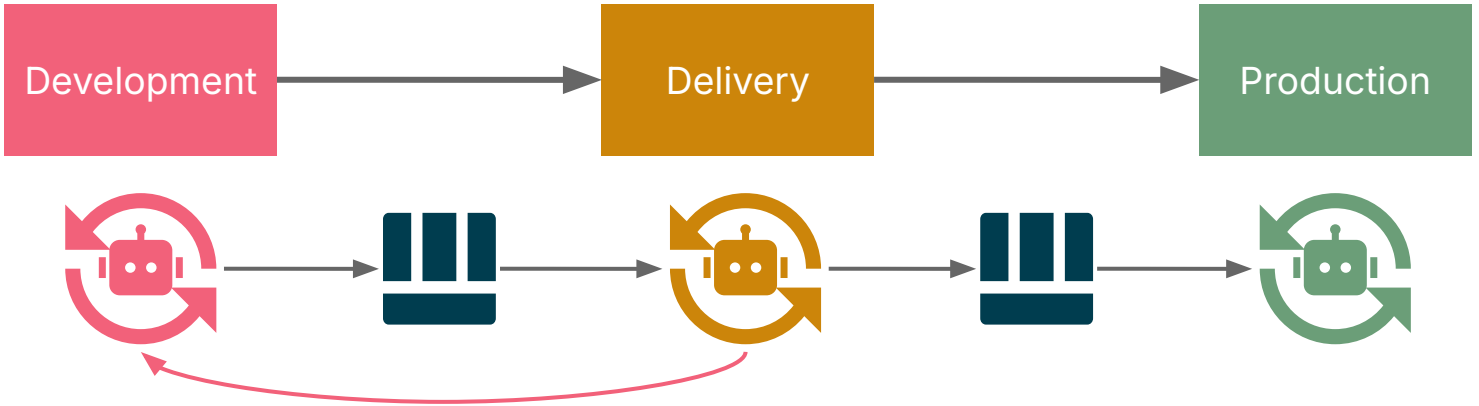
Done means “running in production”

Maybe we don't need CD with agentic engineering?

Agents build the complete solution

Agents make the build production-ready

Agents fix issues in production



"Done" means "now you wait until we do the techie stuff"

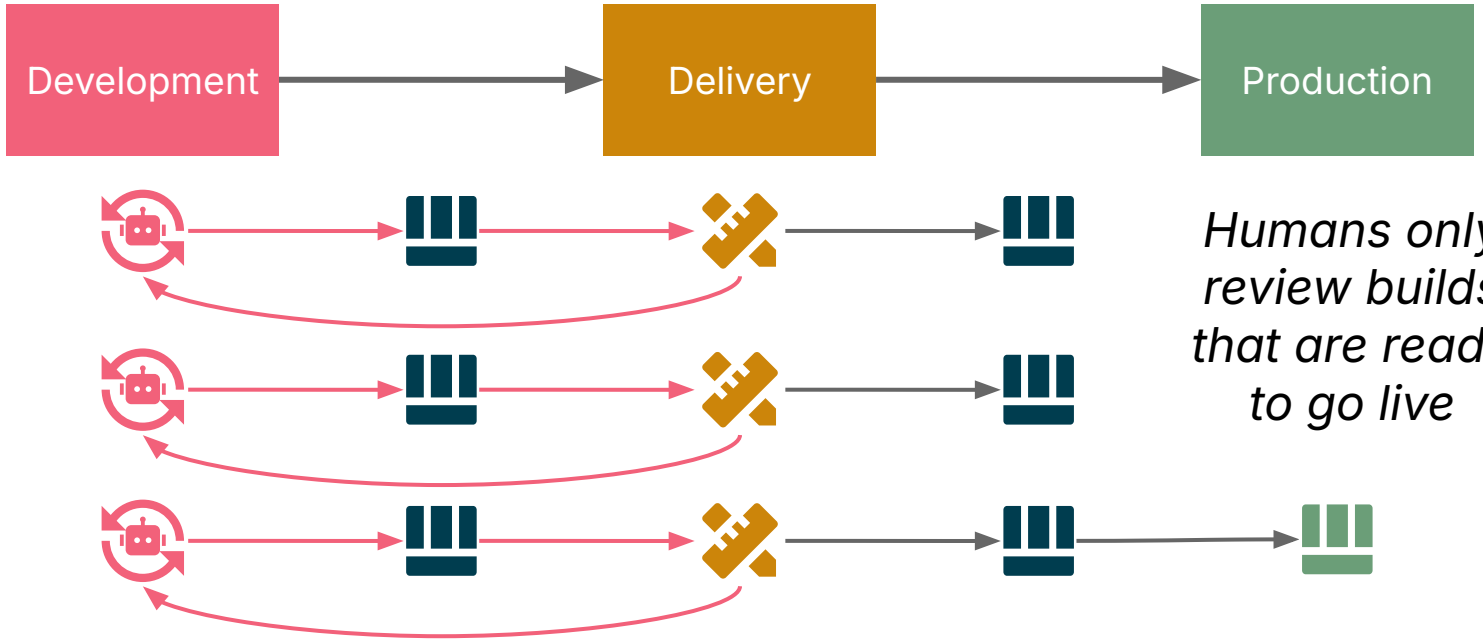
Making the build production-ready often means changing the "final" build

Maybe we can use agents to do full-blast CD?

Build in increments

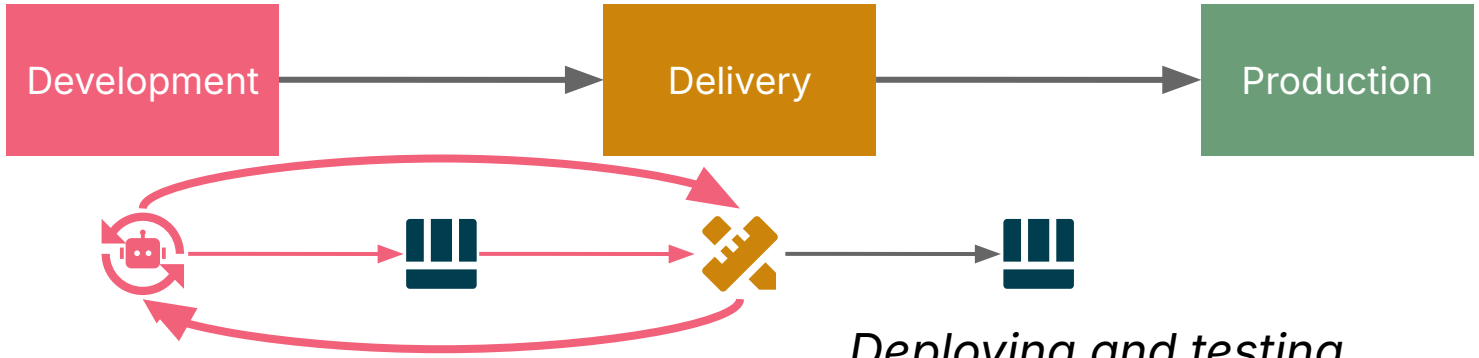
Make each increment production-ready

Deploy on demand



Humans only review builds that are ready to go live

Why I've found this challenging in practice



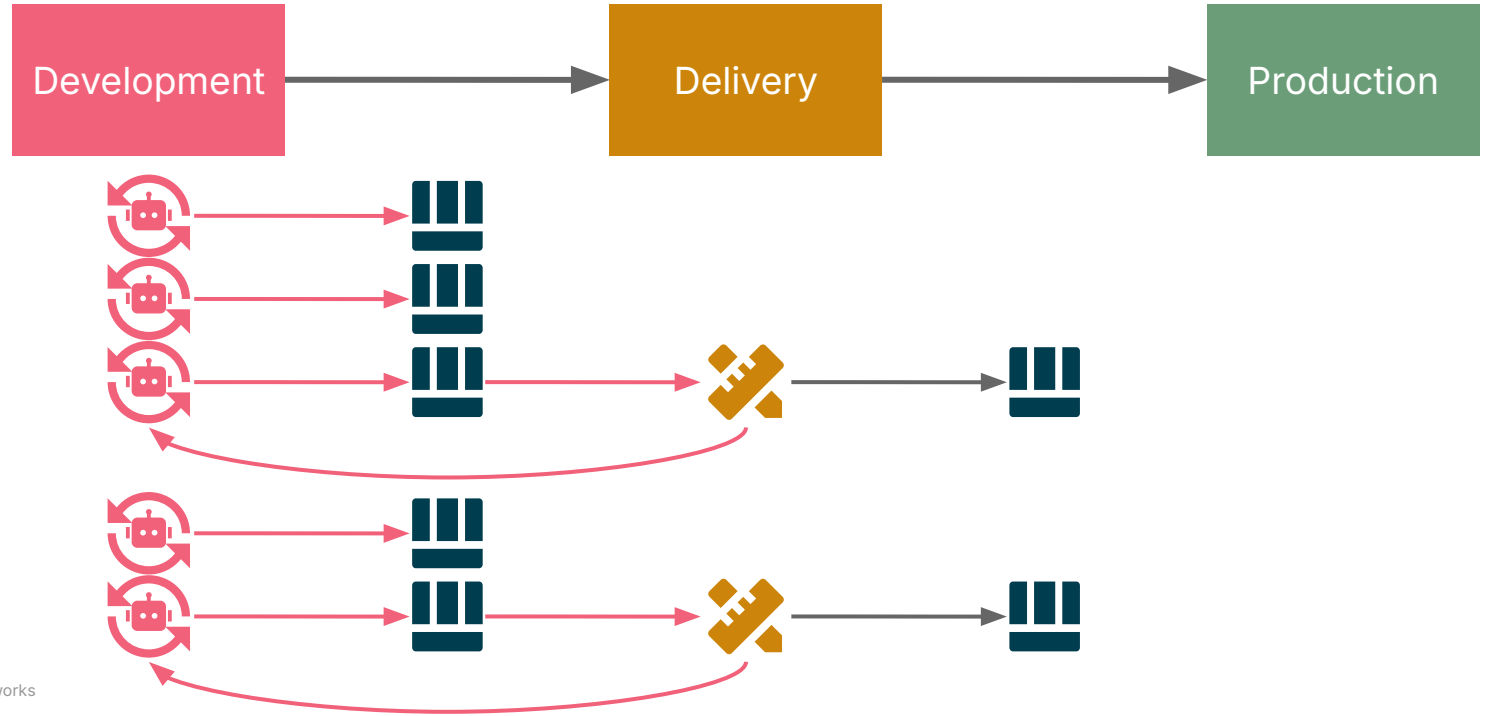
Agents can make it easier to build more thorough operational readiness testing

Deploying and testing an increment is not faster with agents

And it's often slower (currently)

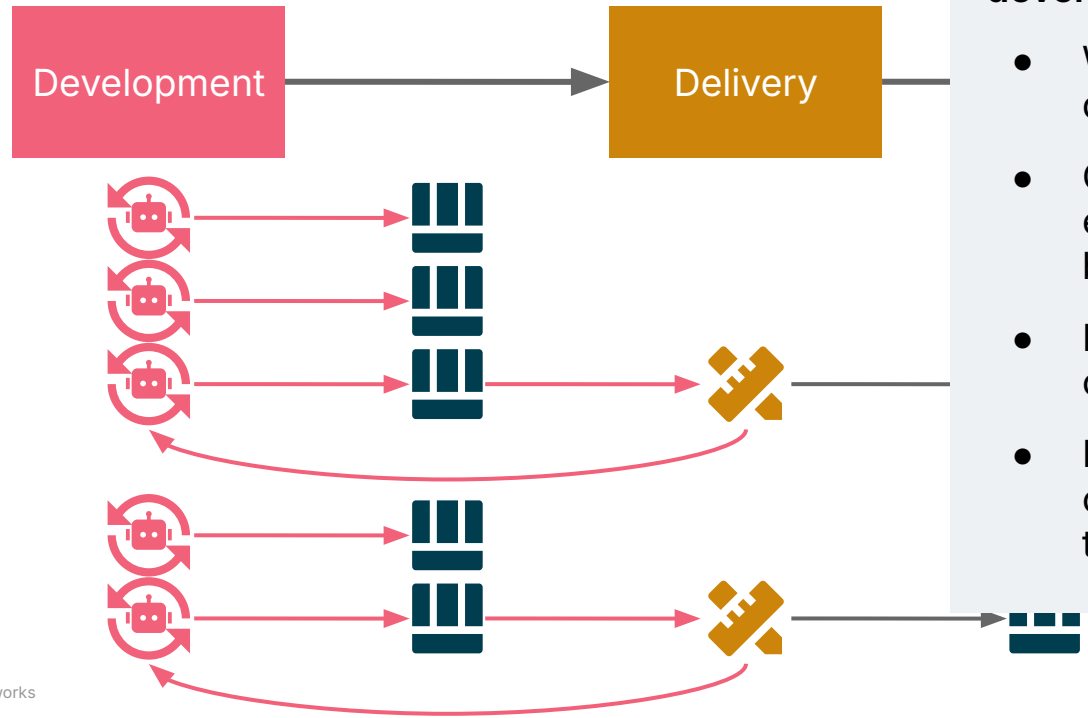
Incrementally testing operational readiness

We should test operational readiness continuously and incrementally



Incrementally testing operational readiness

We should test operational readiness continuously and incrementally



Given a change has passed development tests:

- Will it need a change to the deployment environment?
- Could deployment environment affect its behaviour, implementation?
- Is acceptance sensitive to operational performance?
- Is there value in putting this change live sooner rather than later?

Taking Agentic Engineering End to End

Influences

Harness engineering for coding agent users
Birgitta Böckeler

Harness engineering beyond skills: Using sensors to keep your coding agent in check
(Video) Birgitta Böckeler, Chris Ford

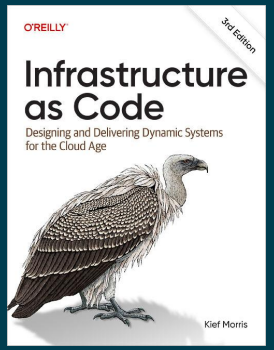
How I Use AI to Code
Chris Parsons

As we build, so we believe
Adam Jacob

Skills Are Context, and Context Needs Tests
Paul Stack

Kief Morris
Distinguished Engineer
Technology Advisory Services

/thoughtworks



https://bit.ly/4uNbOYj