

AI And The Future Of Observability

Peter Souter, Sr Sales Engineer - Datadog

World Summit AI · Track 6: AI Labs



DATADOG

Introductions

Sr. Solutions
Engineer

Based in
London, UK

Started at
Datadog...
March 2021

Previously
HashiCorp, Puppet



**Q: What does observability
look like in an AI world?**

3:07am

[ALERT] checkout p99 latency > 2s



A Common “War Room” Roll-Call Situation...

Database team

"Not us, Data Table's look fine"

Network team

"Not us, no packet loss"

Platform team

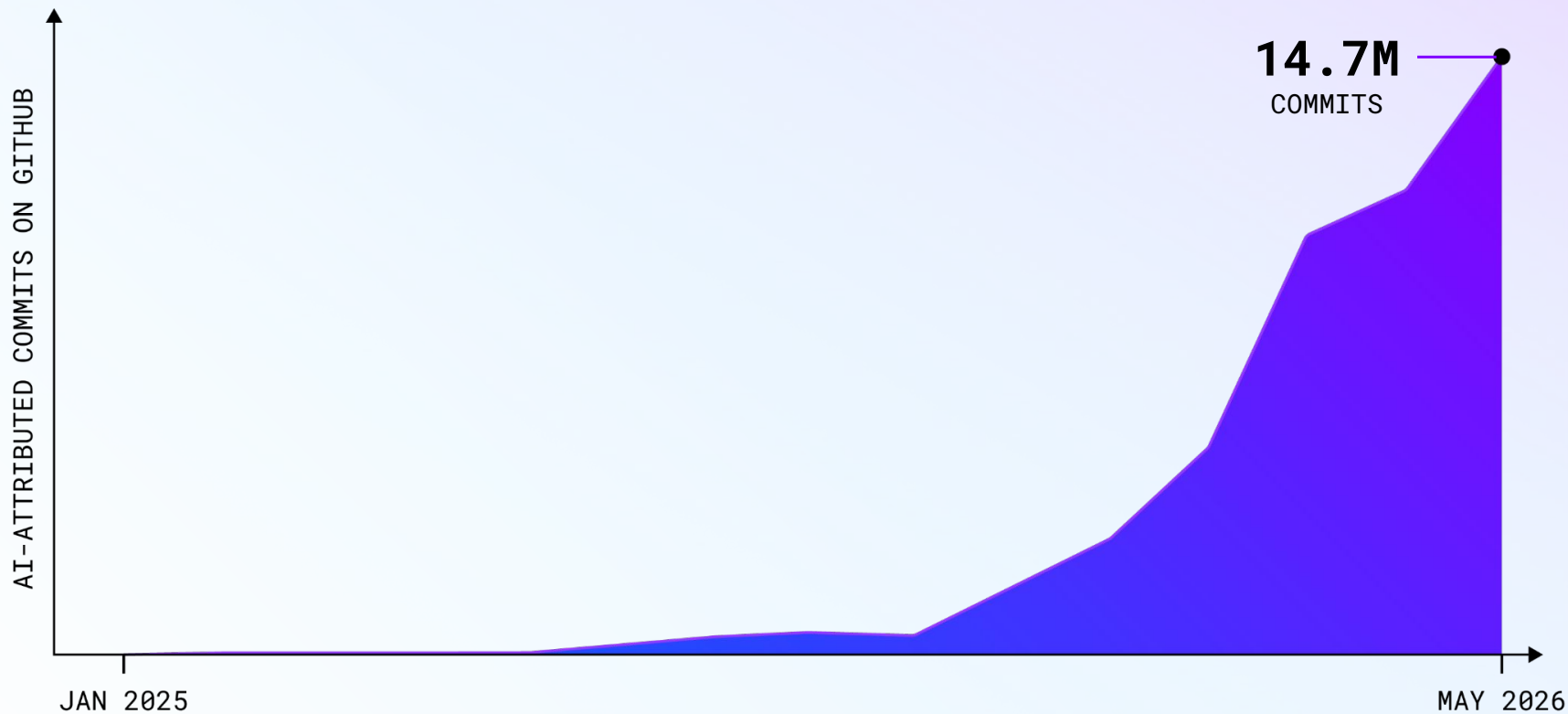
"Not us, cluster looks healthy"

Payments team

"We did a deploy... but that can't be it..."

**Remember, this is what it
looked like *pre* AI!**

AI Is Changing The Software Ecosystem



**Delivery stability is
particularly important to us
as an observability
company!**

AI Amplified Everything

90%

of tech professionals
use AI at work

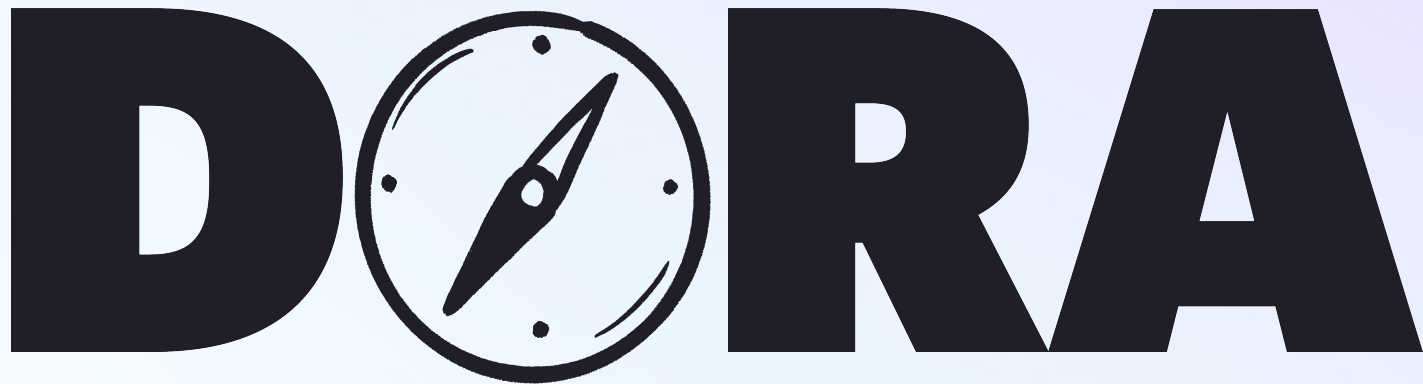
75%

of new code at Google
is AI-generated



AI adoption is linked to
more throughput, and
less stability

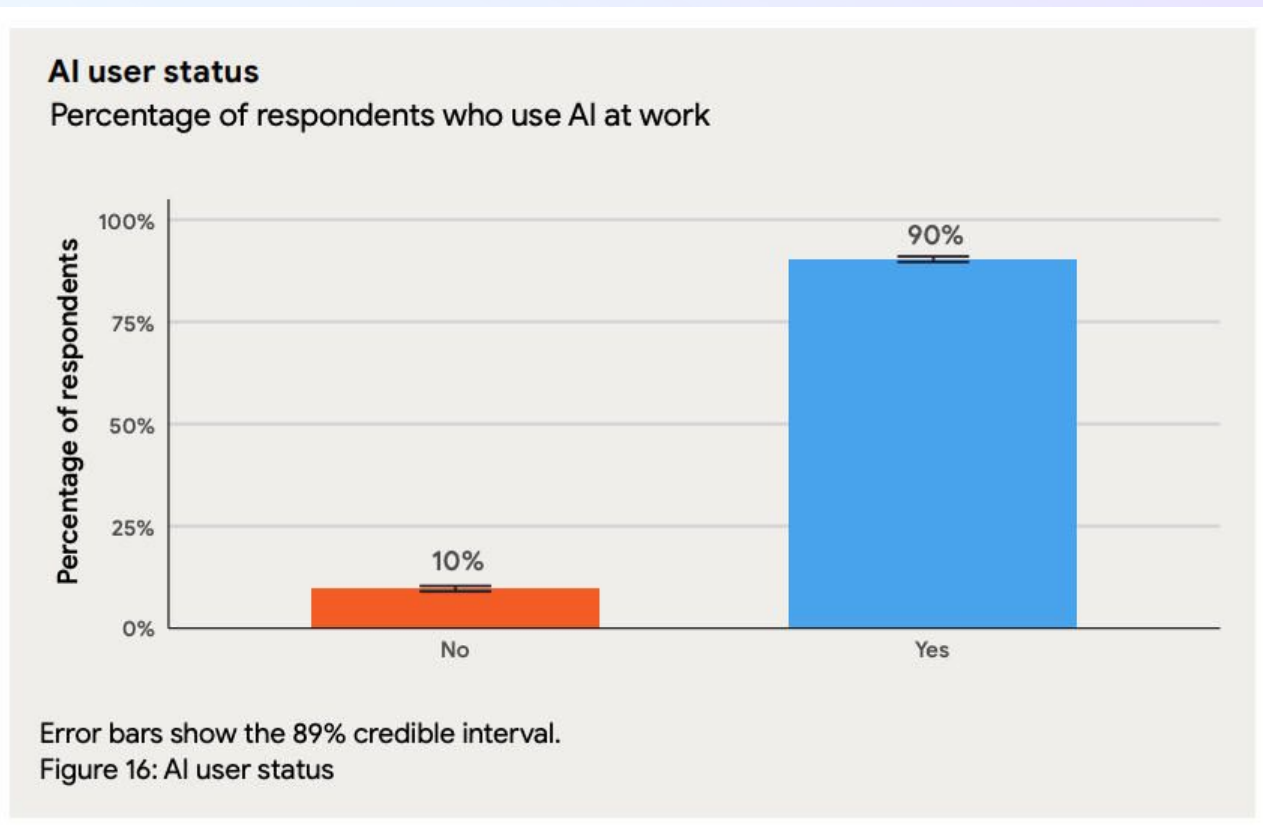
Sources: DORA, State of AI-assisted Software Development (Sep 2025) · Sundar Pichai, Google Cloud Next (Apr 2026)



10

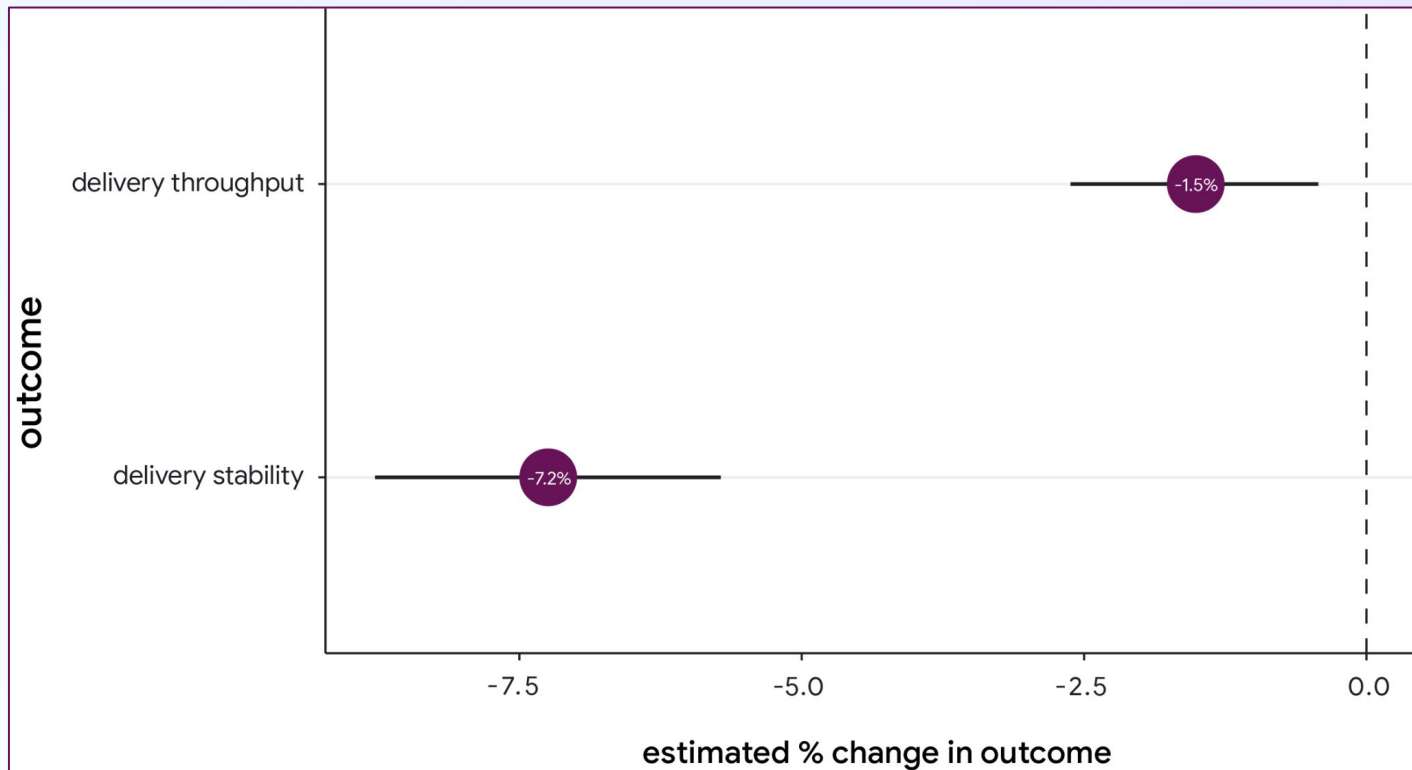
DORA.dev

GenAI: The Productivity Paradox?



The Productivity Paradox: 2024

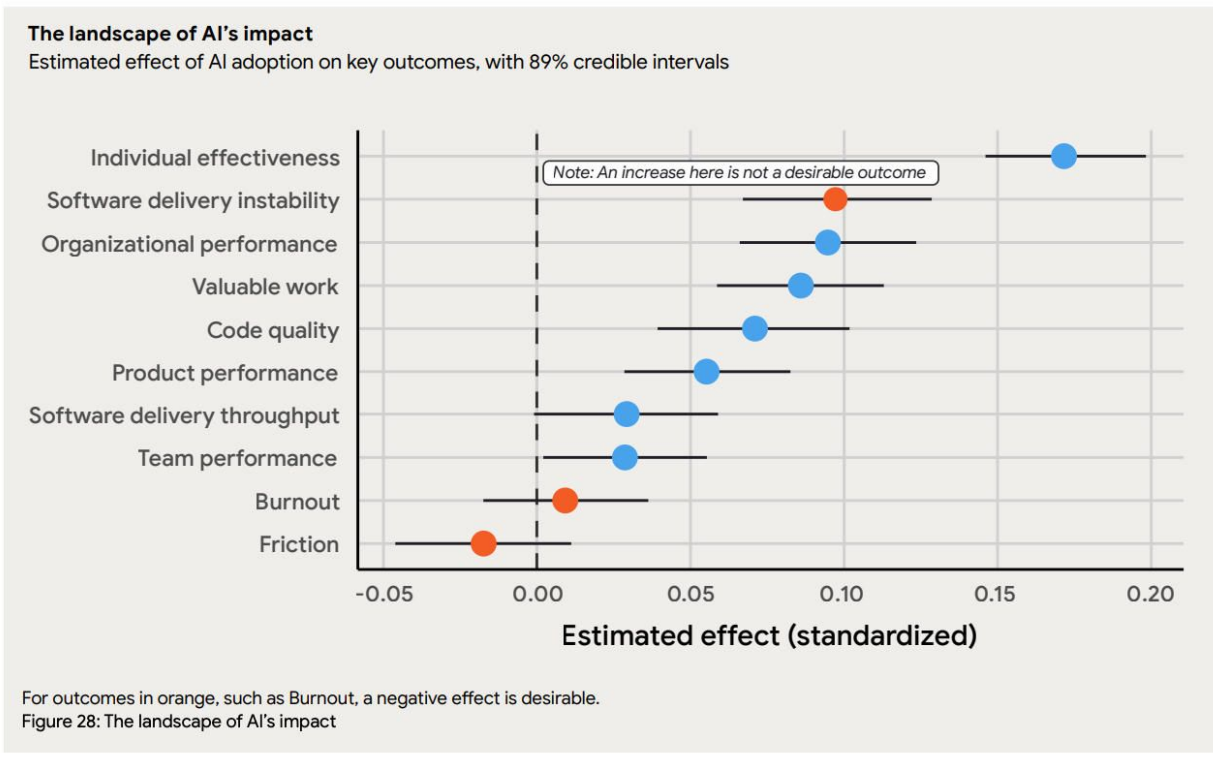
+25% AI adoption → Throughput -1.5% + Stability -7.2%



Source: DORA, Accelerate State of DevOps Report (2024)

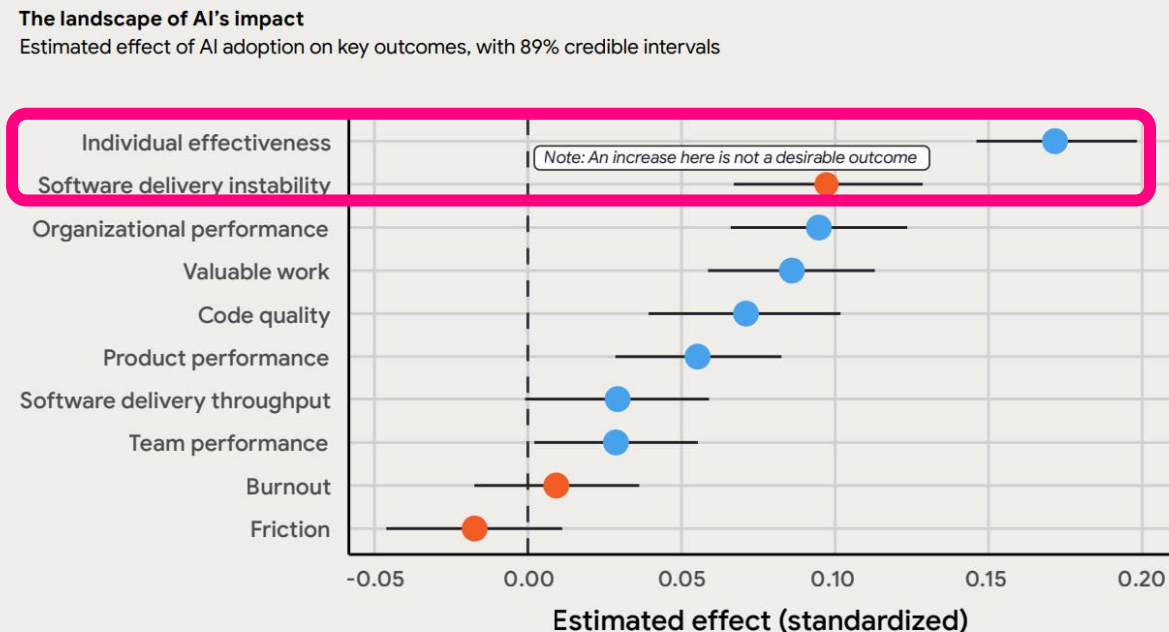
The Productivity Paradox: 2025

Figure 28 visualizes the relationships AI adoption has with these outcomes.²³



The Productivity Paradox: 2025

Figure 28 visualizes the relationships AI adoption has with these outcomes.²³



For outcomes in orange, such as Burnout, a negative effect is desirable.

Figure 28: The landscape of AI's impact

Code is now written faster than humans can read it...

“So, are those roll-calls gonna be even worse? More code, more teams and they might not even have written it themselves?”

What Investigating Actually Looks Like

- 1 Get paged, log in, find the right context
- 2 Check what changed: deploys, config, feature flags, traffic
- 3 Form a hunch, query something, rule it out
- 4 Repeat step 3 until something sticks
- 5 **Apply judgement:** Is this the cause, and what do we do?

What Investigating Actually Looks Like

1 Get paged, log in, find the right context

2 Check what changed: deploys, config, feature flags, traffic

3 Form a hunch, query something, rule it out

4 Repeat step 3 until something sticks

5 **Apply judgement:** Is this the cause, and what do we do?

“Gruntwork”

Important, but
automatable

Needs an engineer

**“We’re not looking to
replace the On-Call
Engineers at all!**

**In fact: We’re looking to set
the table for them!”**

So, how could you start out using AI for observability?

Well, the quickest and fastest way you've probably already done...

"Here's my log dump, what's wrong?"

```
$ kubectl logs checkout-7f9c4 --since=15m | pbcopy # paste into chatbot
```

Chatbot: I can see timeouts calling payment-svc. Some things to check:

1. `kubectl describe pod payment-svc` for restarts or OOMKilled
2. Check CPU limits and throttling on `payment-svc`
3. Check DNS resolution and network policies between the two

Not bad!

Sensible, Generic First steps

Missing:

Is this normal? What changed? Who owns it? Has this happened before?

**Not everything
needs an LLM!**

“Claude, replace all the tabs in this document with spaces!”

So, How Do You Model Observability Without an LLM?

First, the model has to know what normal looks like...

Where Other Approaches Work Better vs LLM's

Deterministic Tools and Models

What changed, and when

Forecasting and anomaly detection

Service dependencies and owners

SLO burn rates

LLM

Reading retros and past incidents

Forming hypotheses

Writing and running queries

Summarising the evidence

Humans

Deciding what's actually wrong

Deciding to act

Knowledge nobody wrote down

Accountability



Telemetry != Language

	Text	Observability metrics
Shape	Follows grammar	Spiky, sparse, noisy
Lifespan	Words last centuries	Pods last minutes
Width	One sequence at a time	Thousands of related series at once
History	Mostly "Normal"	Full of past incidents

So We Built One: TOTO

Time series

Optimized

Transformer for

Observability



Trained on Datadog's **own internal telemetry**, not customer data

Open weights (Apache 2.0), plus **BOOM**: an observability benchmark of 350M observations across 2,807 real series

Toto 2.0: five sizes up to 2.5B parameters, trained on **5 trillion** data points, and bigger keeps getting better

Sources: arXiv 2505.14766, This Time is Different (May 2025) · arXiv 2605.20119, Toto 2.0 (May 2026) · github.com/DataDog/toto

Knowing What Good Looks Like

The Figures From A Model

A forecast of what this metric *should* be doing right now

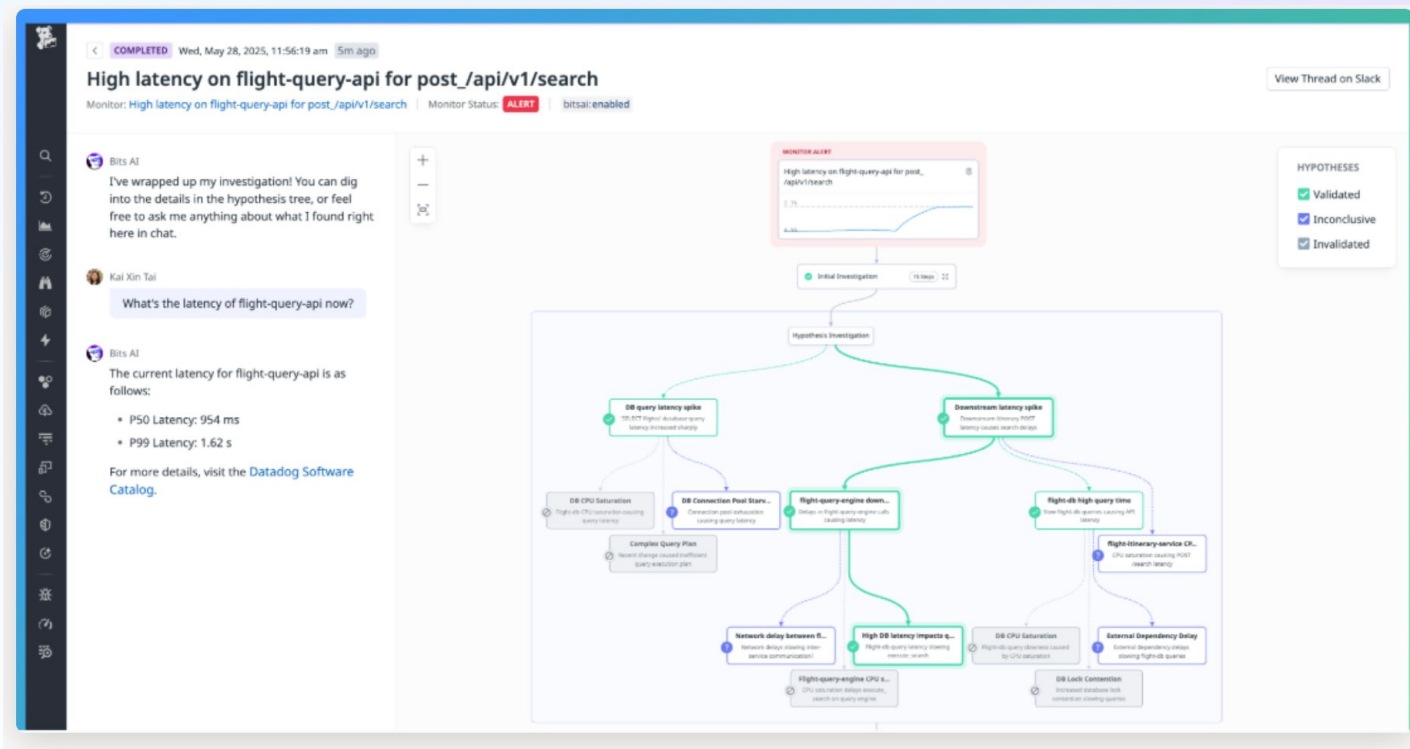
The Organisation Itself

Retros, runbooks, owners, what happened last time

```
# Retro: INC-2291, checkout latency
symptom:  p99 latency up, CPU usage flat
cause:    payment-svc CPU throttled after limits halved (PR #4410)
lesson:   check throttling, not utilisation, after limit changes
```

Retrospectives and Post-Mortems Are Now Context For Agents And Humans!

Right Answer, Right People, Show Your Working



Observability Has A New Audience:

Reading

Agents are now **reading** your telemetry, not just humans

Writing

Agents are now **writing** the code your telemetry describes

Observing

So someone has to **observe the agents** too

Observability Hands The Torch To Autonomy

Where We Want Everyone to Be

Monitoring

Availability
Performance

- Legacy
- Static
- Reactive

Observability

Infra, Log, APM User Experience
Developer Experience

- Cloud-native
- Dynamic
- Spans Dev and Ops

Autonomy

Validation Security & Safety
Alignment Control

- AI-Native
- Dynamic and Stochastic
- Spans Dev, Ops, Security, and Business Impact
- Highly automated



That's the shift to
Autonomous
Operations

The LONG-TERM VISION

Enabling
Autonomy
across Dev, Ops and Security

Thanks!

Questions?

PS. Want to see that hypothesis tree live and in action?
Come and find us at Stand G60