

Visualizing the Systems That Drive Your Team

Everyone knows what the problem is. They just all know something different.

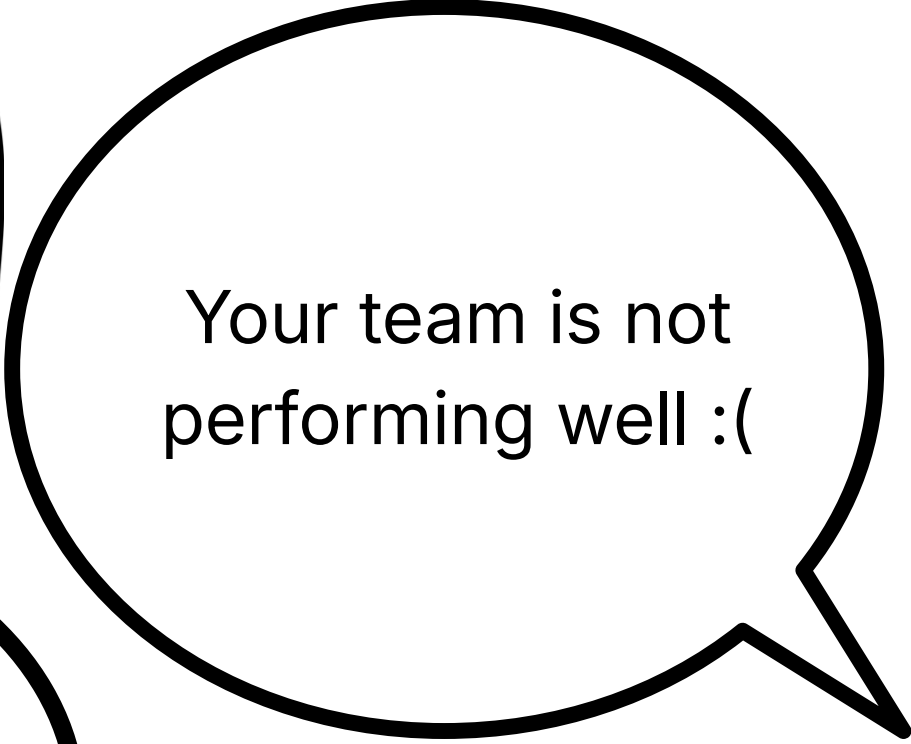
Visualizing the Systems That Drive Your Team

Everyone knows what the problem is. They just all know something different.

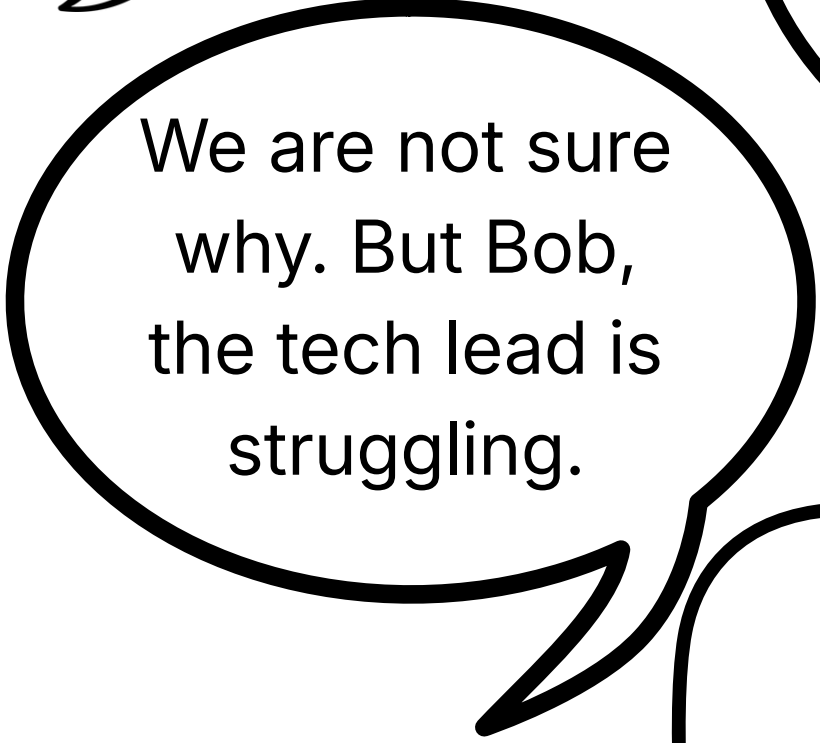
A (common) first week as the new manager



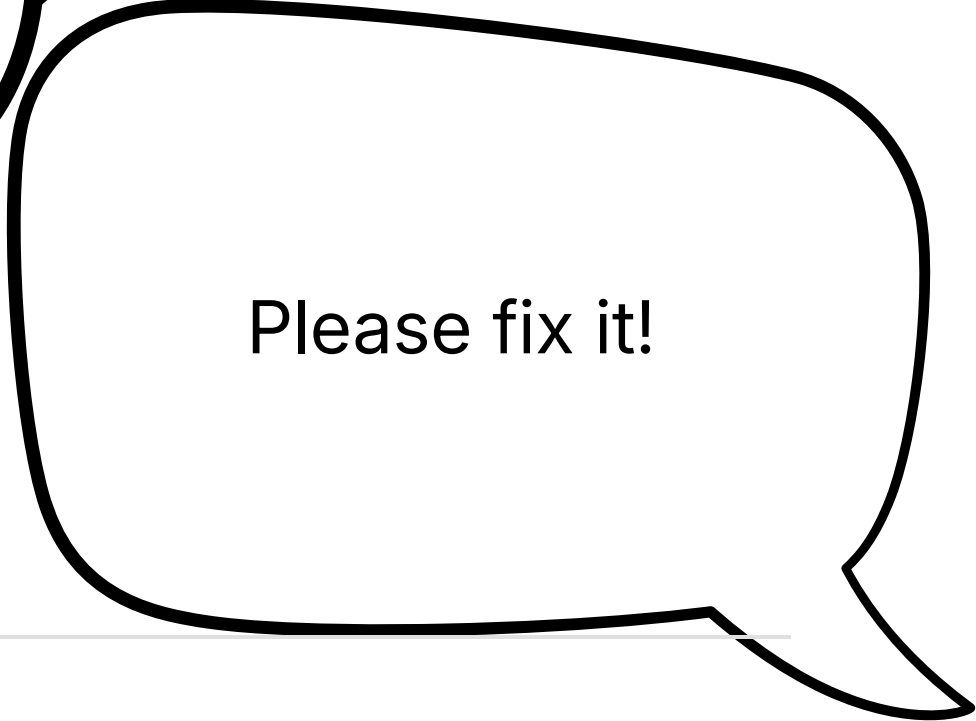
Welcome to
your new role
as manager!



Your team is not
performing well :(



We are not sure
why. But Bob,
the tech lead is
struggling.



Please fix it!

A project delivery tale

[Leaders]
Why is this
taking so long?

[Product Mgt]
Design is
blocking
progress

[Designer]
Requirements
are still in flux

[Engineers]
We don't have
anything to do

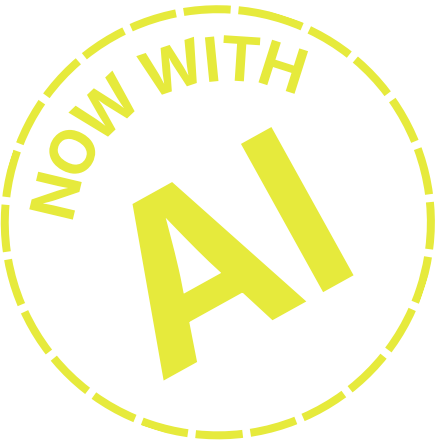
IT'S NOT BECOMING SIMPLE

**Delivering a software
product is
not becoming simple**

IT'S NOT BECOMING SIMPLE

**It became easier in
many ways,**

OUTPUT



IT'S NOT BECOMING SIMPLE

**It became easier in
many ways, and
more complicated
in others**



“Here is my perspective,
in a twenty-page document
(A)I just wrote”

“Here is my perspective,
in a twenty-page document
(A)I just wrote”

We are building a lot of machinery to align agents:
harnesses, loops, factories.

We are not talking as much about **how to align the
humans.**

The humans who are driving the agents.

“Here is my perspective,
in a twenty-page document
(A)I just wrote”

Leadership is making the invisible, visible.

When everyone has a different perspective, the
challenge is to **visualize how the system is actually
working.**

Let's talk about how.

Vice President of Engineering at Braze

~100 people across 13 teams

Previously:

- Product Engineering in different companies
- Consulting at Thoughtworks
- Founded two companies

Hi, I'm
Francisco
Trindade

Welcome to
your new role
as manager!

Your team is not
performing well :(

We are not sure
why. But Bob,
the tech lead is
struggling.

Please fix it!

01

**A (common)
first week as the
new manager**

THE NEW MANAGER CHALLENGE



Welcome to
your new role
as manager!

Your team is not
performing well :(

We are not sure
why. But Bob,
the tech lead is
struggling.

Please fix it!

The team was not delivering consistently, missing their estimates.

It needed to be “**fixed**”.

Fixing it for me meant understanding how the team worked.

- Observe the team.
 - Talk to team members and get their perspectives.
 - Analyze the data. Where was the time going?
-

THE NEW MANAGER CHALLENGE



Welcome to
your new role
as manager!

Your team is not
performing well :(

We are not sure
why. But Bob,
the tech lead is
struggling.

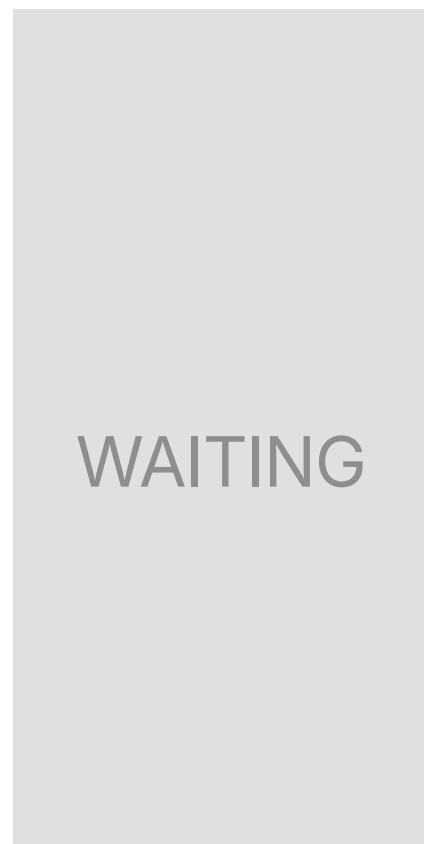
Please fix it!

Perspectives also varied across team members.
They were trying to optimize by starting more
work.

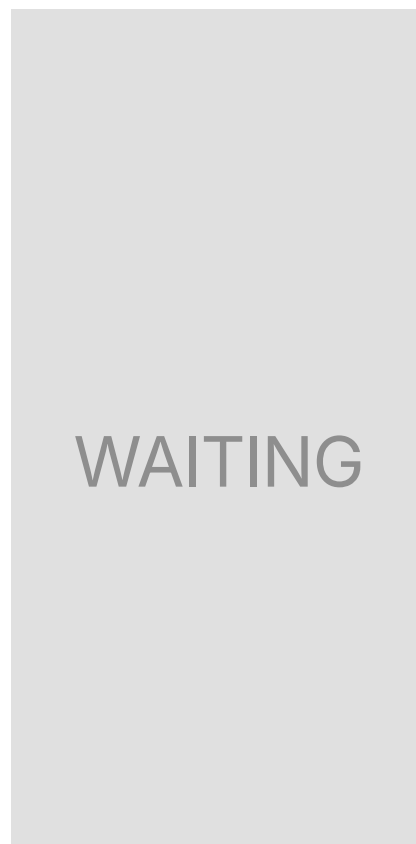
Everyone was working hard, yet nobody could
agree on what to change.

Until we looked at the lifetime of a ticket.

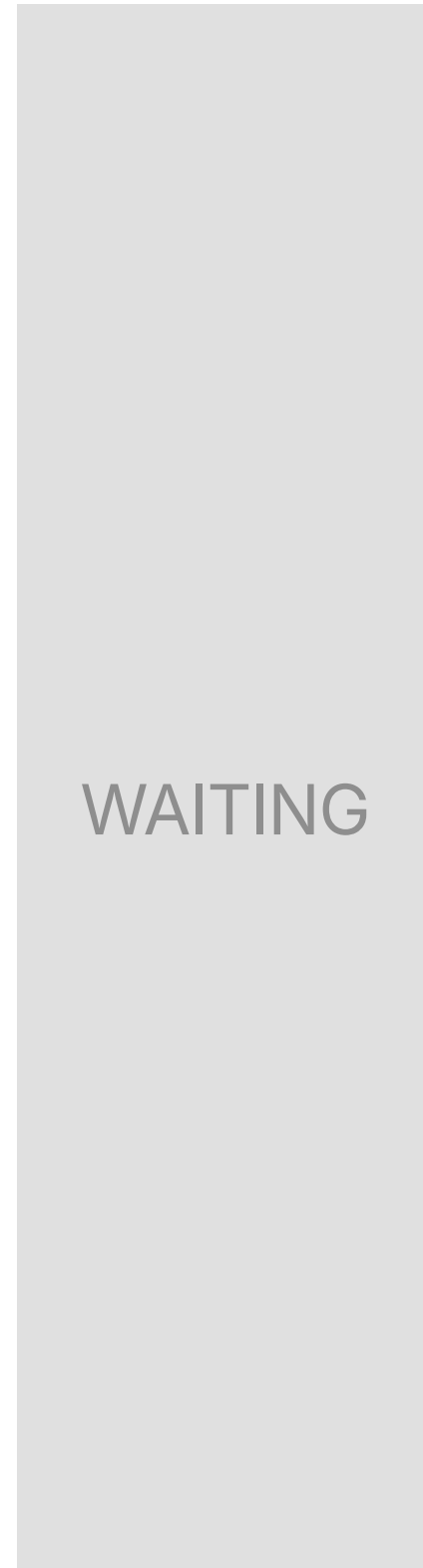
DAY 1



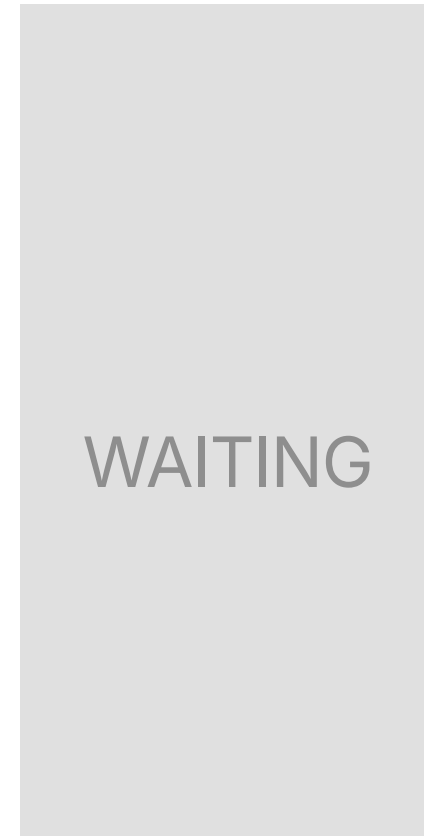
DAY 2



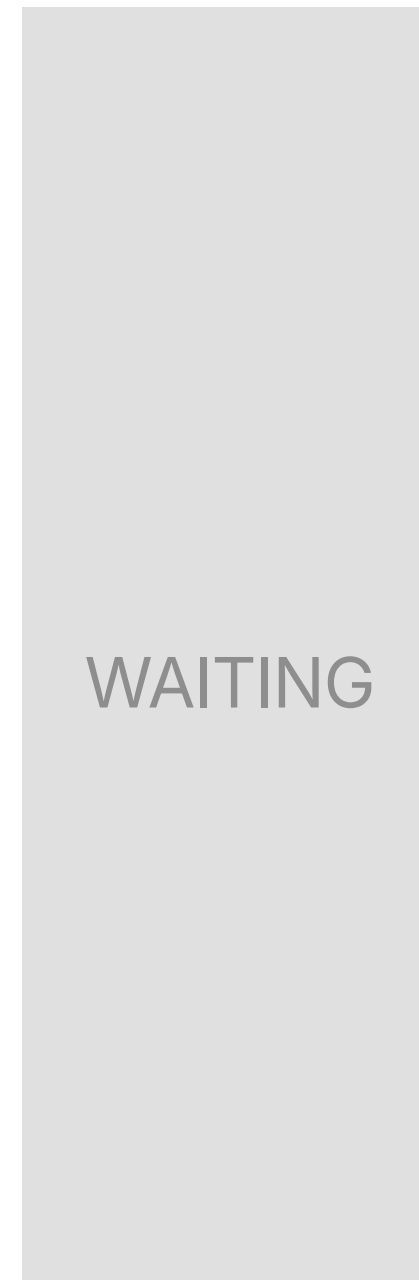
DAY 3



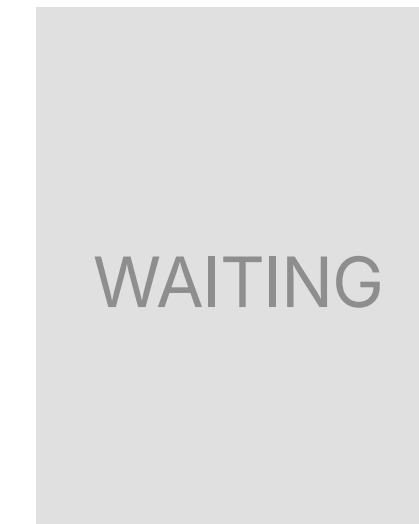
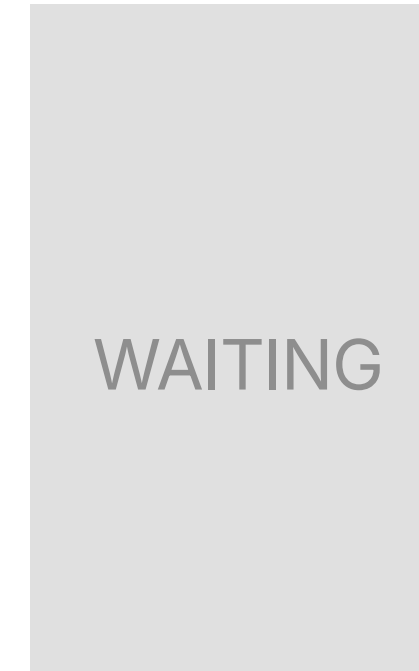
DAY 4



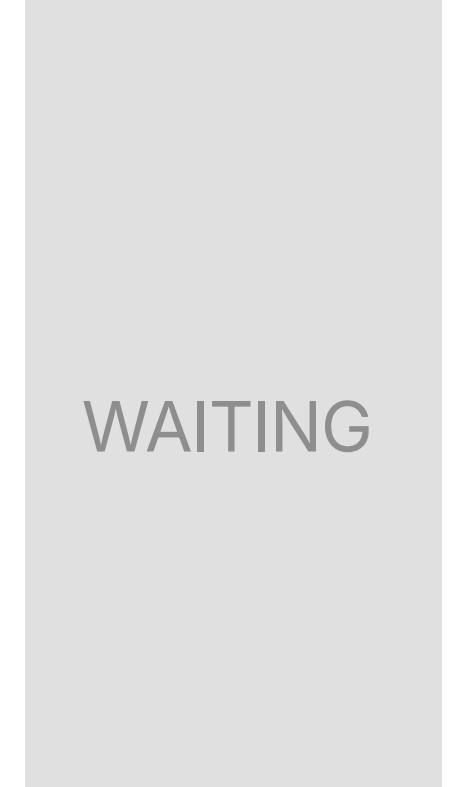
DAY 5



DAY 6



DAY 7



THE NEW MANAGER CHALLENGE



Welcome to
your new role
as manager!

Your team is not
performing well :(

We are not sure
why. But Bob,
the tech lead is
struggling.

Please fix it!

Once we all could see the same problem,
resolution was simple.

Experimented with synchronous reviews. The
team's velocity became stable (and much
higher) within two weeks.

LESSON 01

**Time to complete work is what matters.
Everything else follows from it.**

88%

of a pull request's life is spent waiting — nobody committing, nobody reviewing.

ACTIVE

228.7h

12.3% of tracked time

WAITING

1635.4h

87.7% of tracked time

MEDIAN WAIT

52.3h

half wait longer than this

LONGEST GAP

193.4h

PR #135313, still open

Every change, first commit to close

Active Waiting | PR opened

Each row is one pull request, positioned in real time. Nights, the 09:00–13:00 UTC dead zone, and Sundays are compressed out of the axis, so bar widths match the hours quoted. Active blocks are often under two pixels at this scale, so they are drawn at a small minimum width to stay visible — exact hours are in the tooltip and the table.

PULL REQUEST

TIMELINE

WAITING

#131260 CLX-1105

0.8h

#131649 CLX-1105

35.5h

#131677 CLX-1105

21.2h

#132167 CLX-1105

71.5h

#132328 CLX-1105

46.9h

#132869 CLX-1107

33.5h

#132930 CLX-1109

75.2h

#133177 CLX-1113

87.6h

02

**The new
bottleneck**

Here is another
problem that,
unless you have
been under a rock,
you had this year:

**Code reviews
becoming a
bottleneck**

“Reviewing [another’s] code is so much harder than writing it.

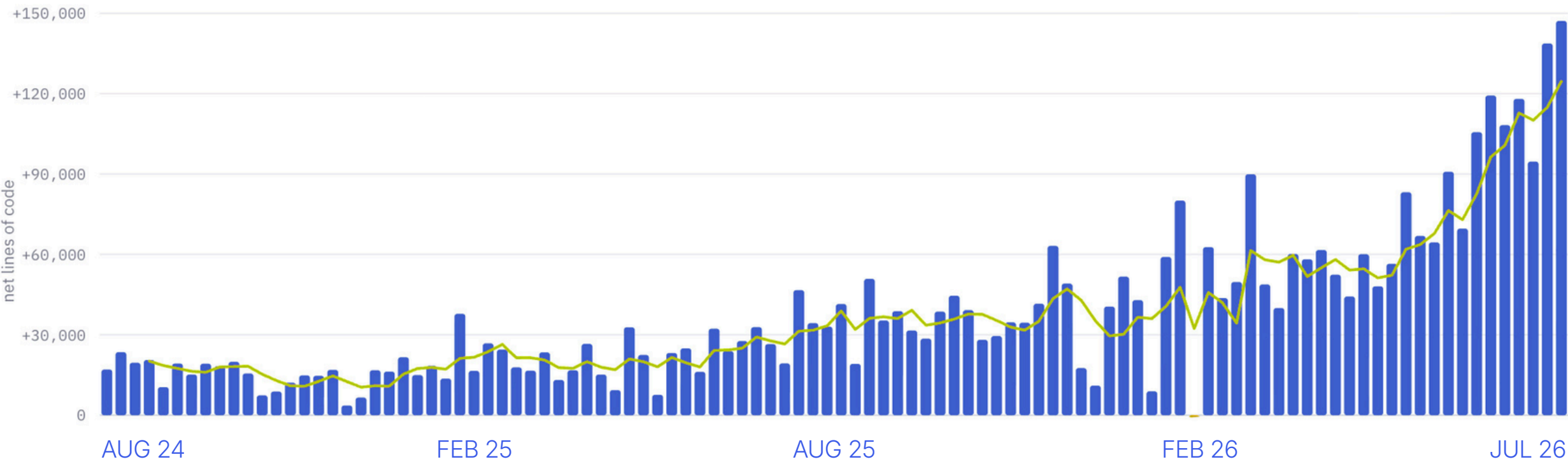
AI tools are increasing the rate at which people can churn out code that needs to be reviewed...”

Here is another problem that, unless you have been under a rock, you had this year:

Code reviews becoming a bottleneck

THE NEW BOTTLENECK

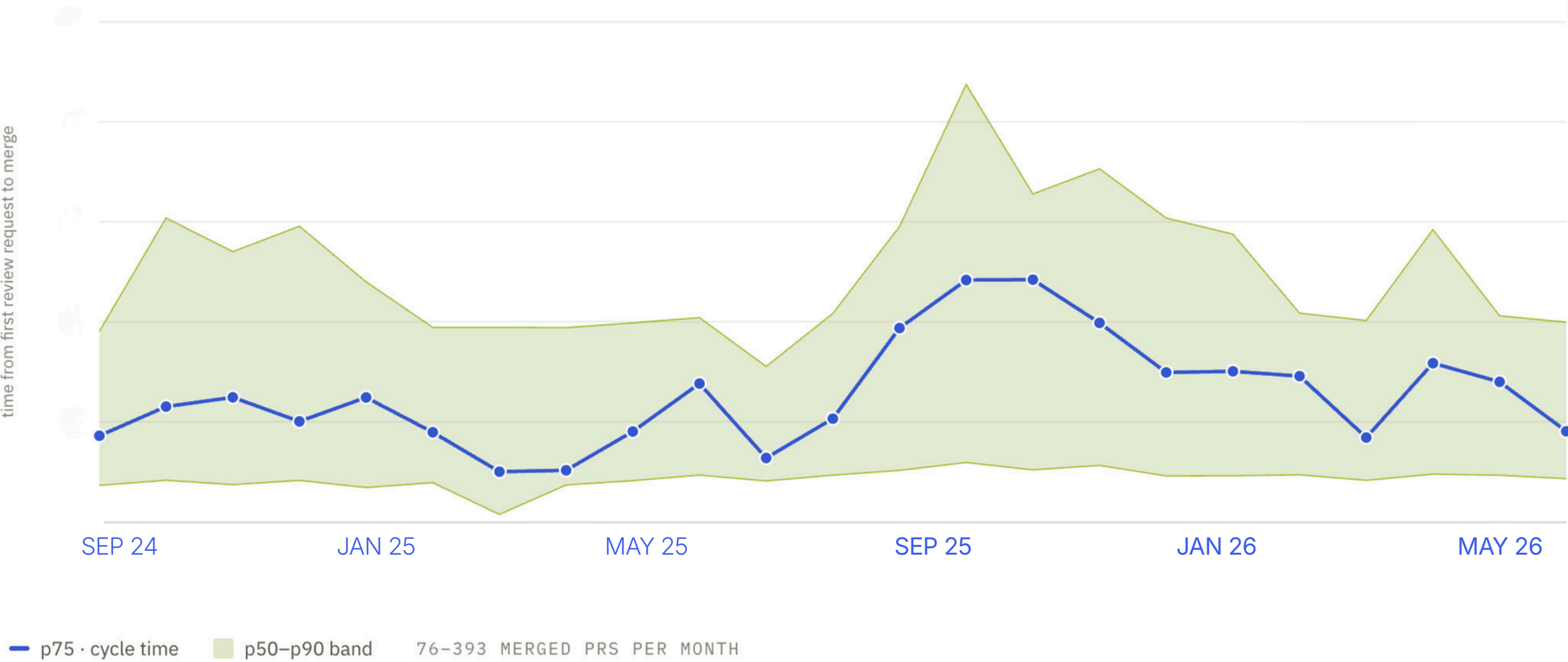
NET LINES OF CODE



As in any other company, our code output grew considerably.

THE NEW BOTTLENECK

PR CYCLE TIME MONTH BY MONTH



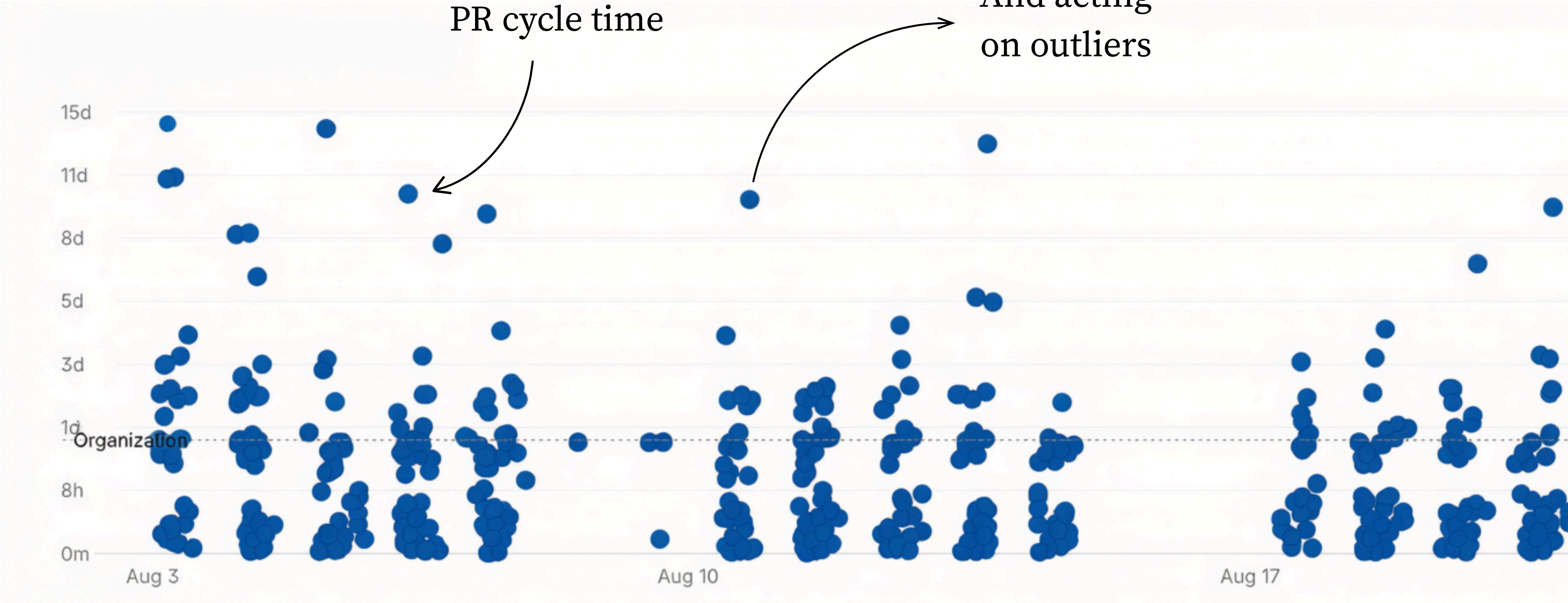
As in any other company, our code output grew considerably.

But we were able to **keep PR cycle time stable.**

THE NEW BOTTLENECK

We did by visualizing
PR cycle time

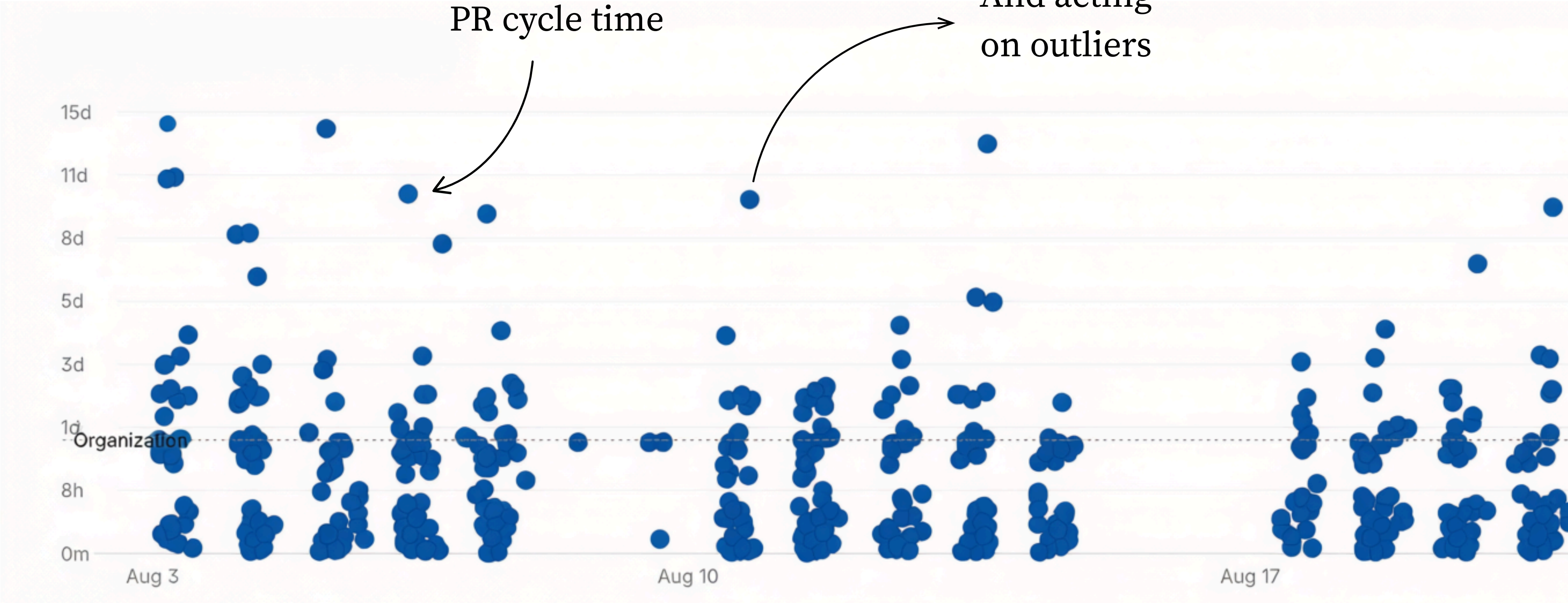
And acting
on outliers



THE NEW BOTTLENECK

We did by visualizing
PR cycle time

And acting
on outliers



Understanding how to make PRs faster is easy if you are looking at the right problems.

Visualizing the system makes it possible to find where to look.

LESSON 02

Stability comes from fixing the right issues (and leaving others alone).

As in any other company, our code output grew considerably.

But we were able to **keep PR cycle time stable.**

WHY IS IT TAKING SO LONG?

03

**The never ending
initiative**

WHY IS IT TAKING SO LONG?

The quintessential
problem for
engineering
leadership:

**Projects take
longer than they
should**

WHY IS IT TAKING SO LONG?

The quintessential
problem for
engineering
leadership:

**Projects take
longer than they
should**

We were running a critical initiative.
One of the most important features for
the company. And progress was slow.

“Why is this taking
so long?”

The quintessential
problem for
engineering
leadership:

**Projects take
longer than they
should**

Continuous delays led to lack of
confidence.

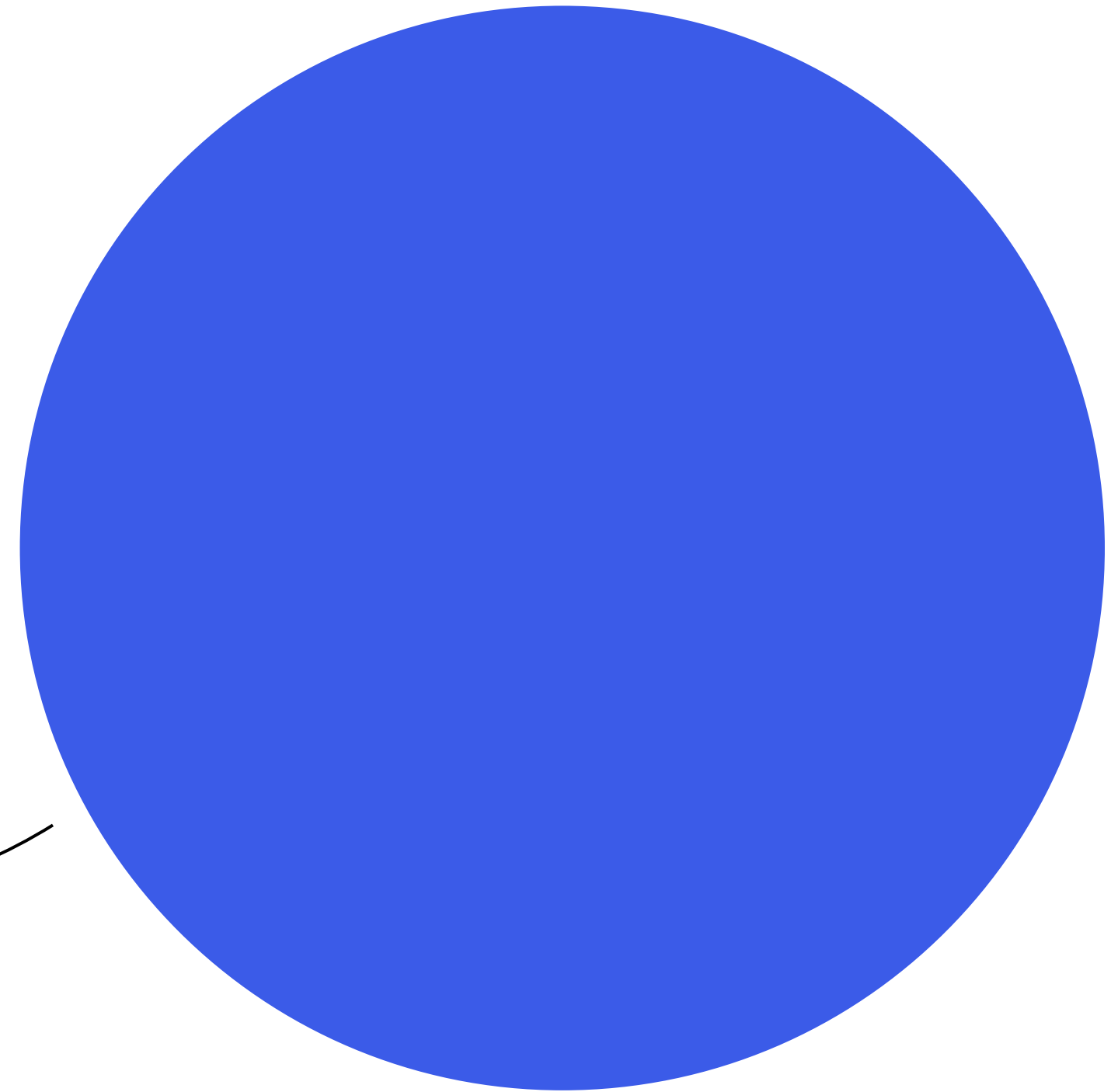
“We are making all this
investment and not getting
the results we expect”.

Or at least we thought so. Until we
visualized where time was going.

WHY IS IT TAKING SO LONG?

The investment we
thought we were
making.

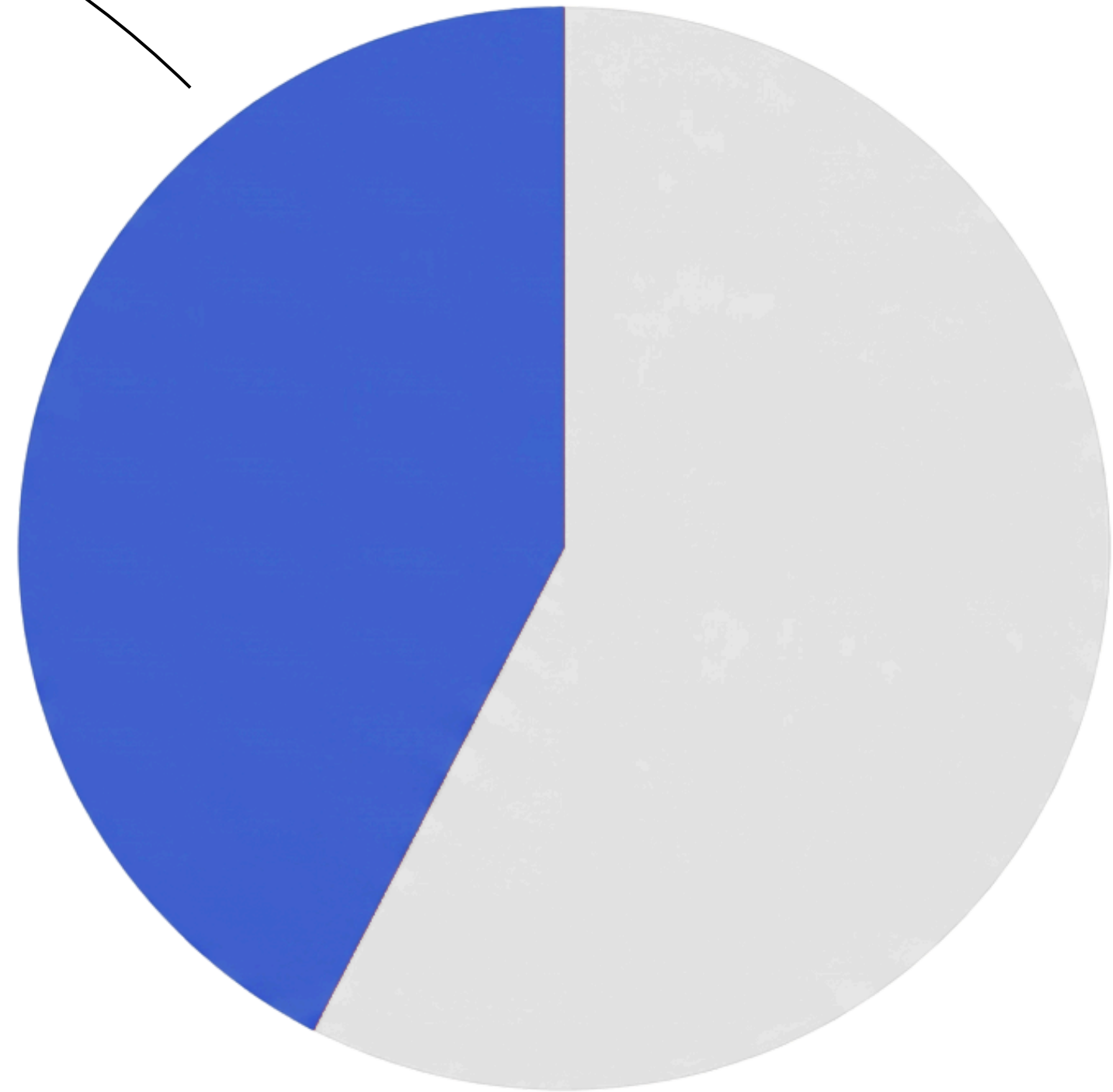
100% capacity on a
team of 5 engineers.



WHY IS IT TAKING SO LONG?

The investment we were
actually making.

1.7 FTE (not even two
engineers)

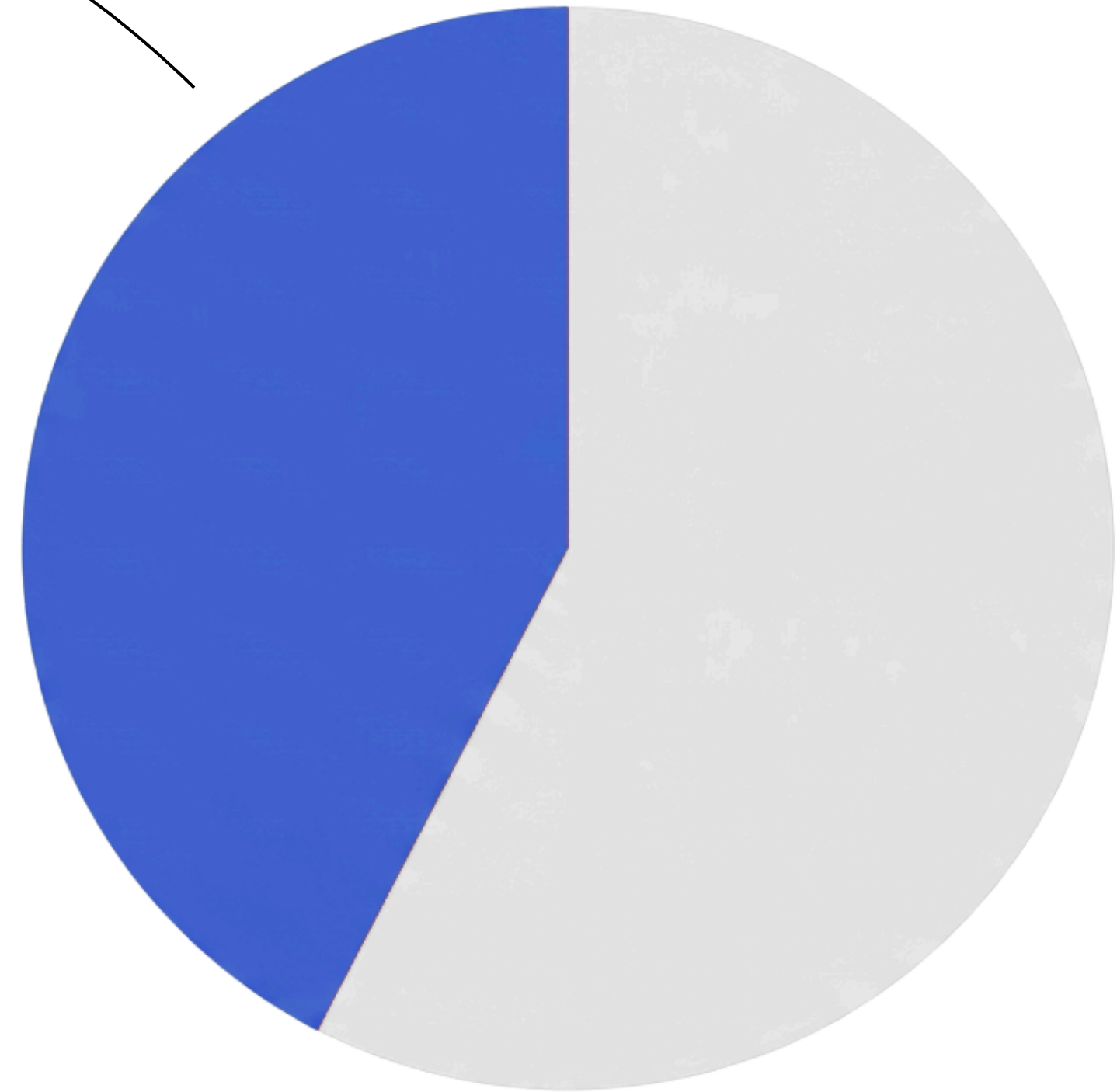


WHY IS IT TAKING SO LONG?

The investment we were
actually making.

1.7 FTE (not even two
engineers)

Most of the team's
time was spent on
support, on-call, and
other side initiatives



WHY IS IT TAKING SO LONG?

The quintessential
problem for
engineering
leadership:

**Projects take
longer than they
should**

Understanding investment made the problem clear.

We aligned on a solution: reduced the scope and provided more staffing. The repeated discussions stopped.

LESSON 03

The best improvement for a team is to ensure it's working on the right priority.

WHOSE FAULT IS IT?

[Leaders]
Why is this
taking so long?

[Product Mgt]
Design is
blocking
progress

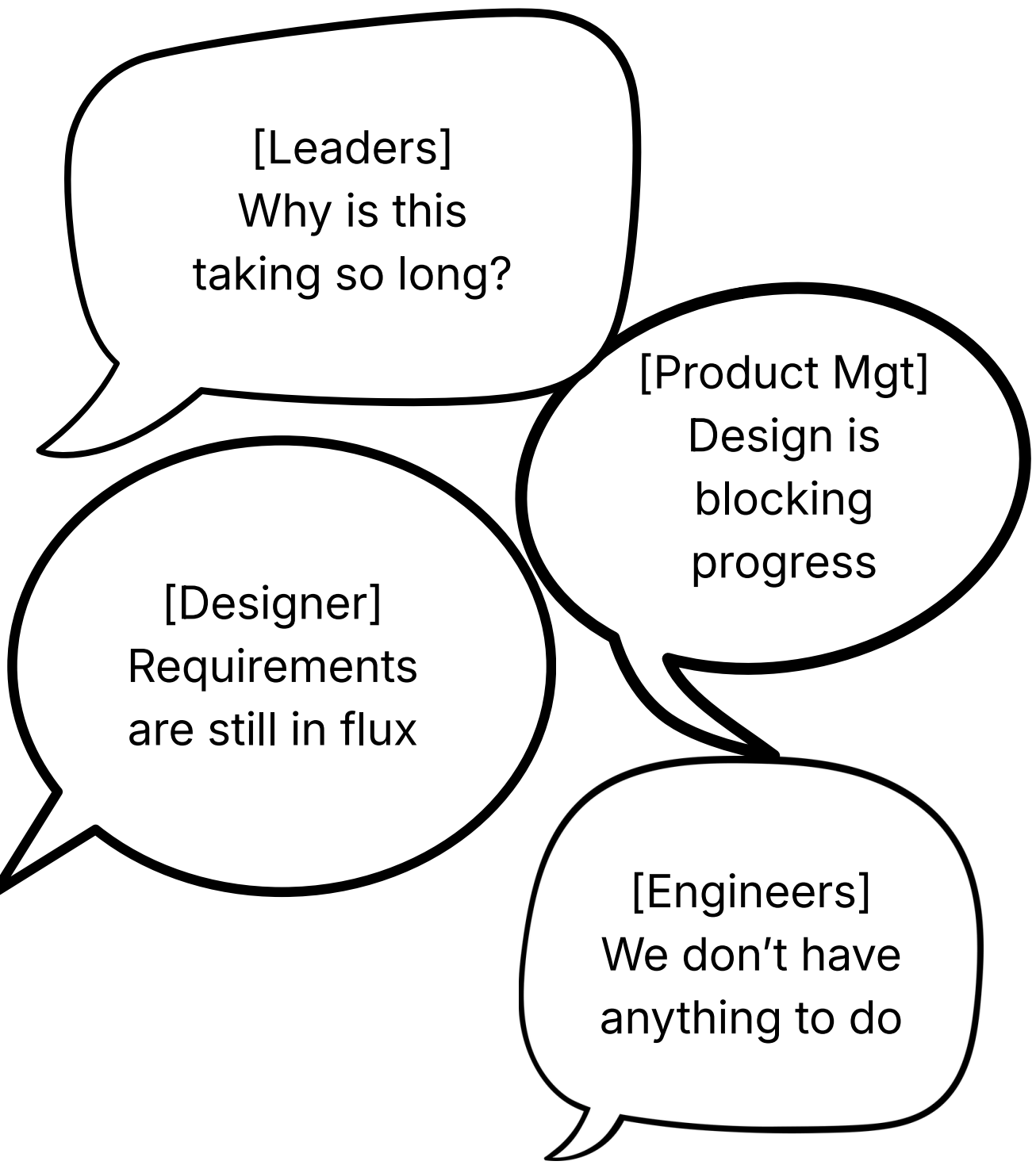
[Designer]
Requirements
are still in flux

[Engineers]
We don't have
anything to do

04

**Broken
collaboration**

WHOSE FAULT IS IT?



[Leaders]
Why is this
taking so long?

[Product Mgt]
Design is
blocking
progress

[Designer]
Requirements
are still in flux

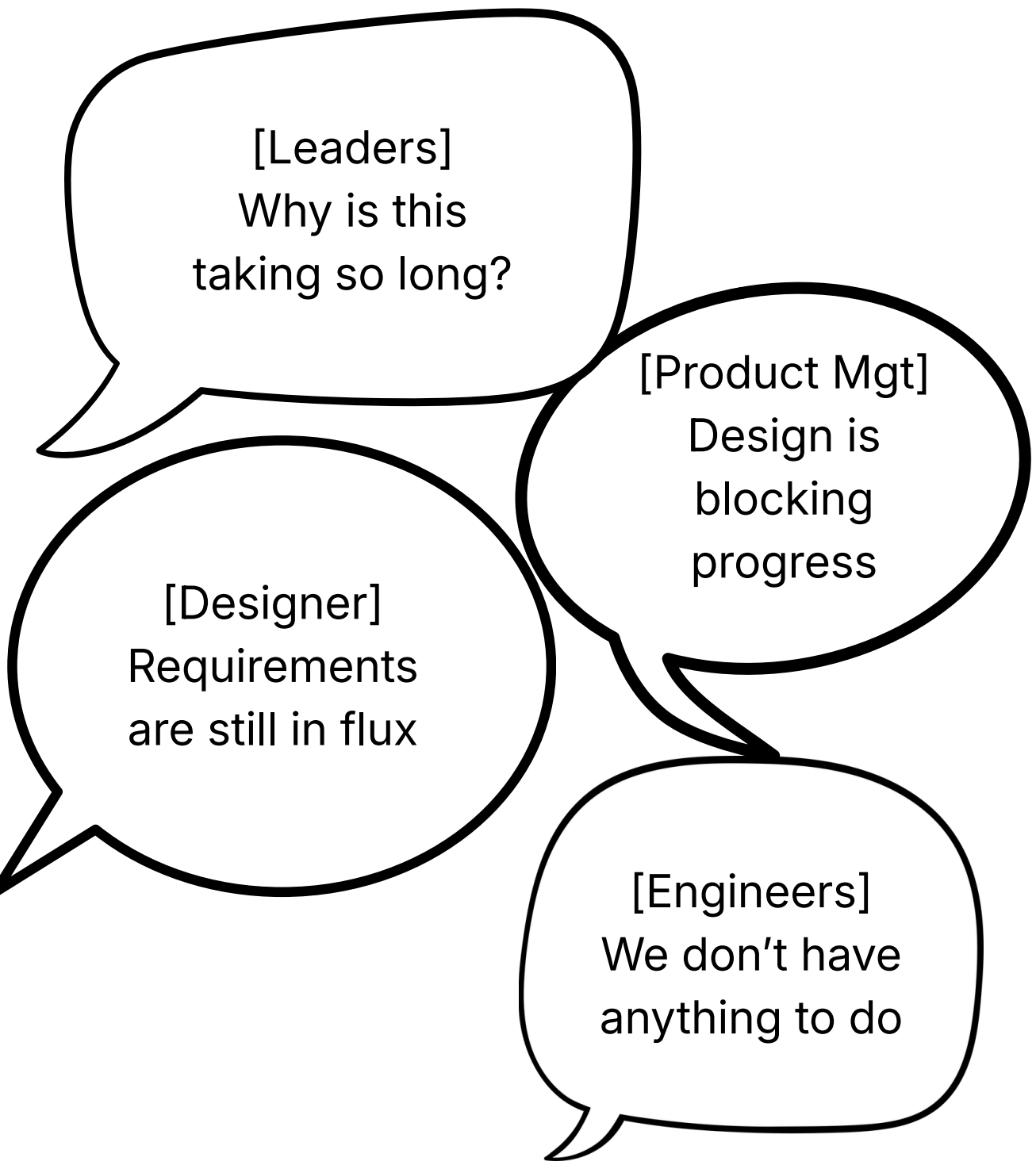
[Engineers]
We don't have
anything to do

A project was causing **constant friction** between team members. Engineers felt blocked, designers felt under pressure.

It was escalated to leaders, who also could not agree on what was wrong.

With AI, everyone was moving faster in different directions.

WHOSE FAULT IS IT?



[Leaders]
Why is this
taking so long?

[Product Mgt]
Design is
blocking
progress

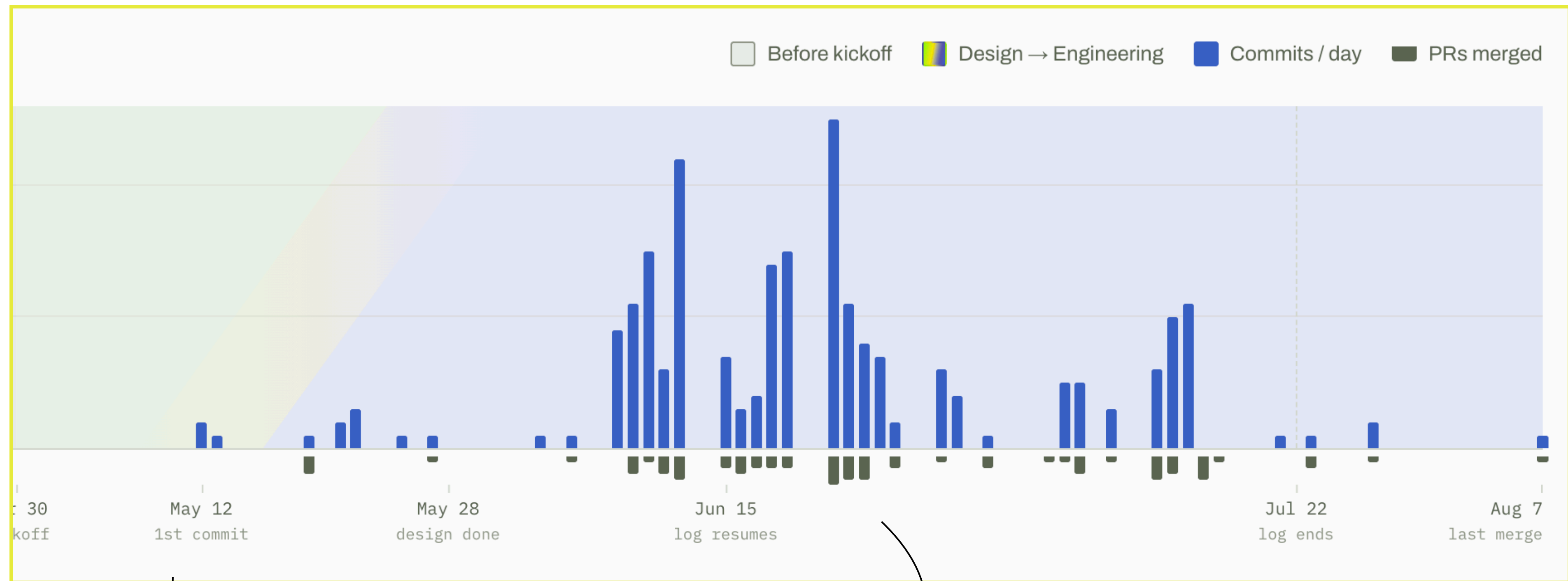
[Designer]
Requirements
are still in flux

[Engineers]
We don't have
anything to do

Were engineers being unreasonable? Were the requirements not correct? Was design taking too long to deliver?

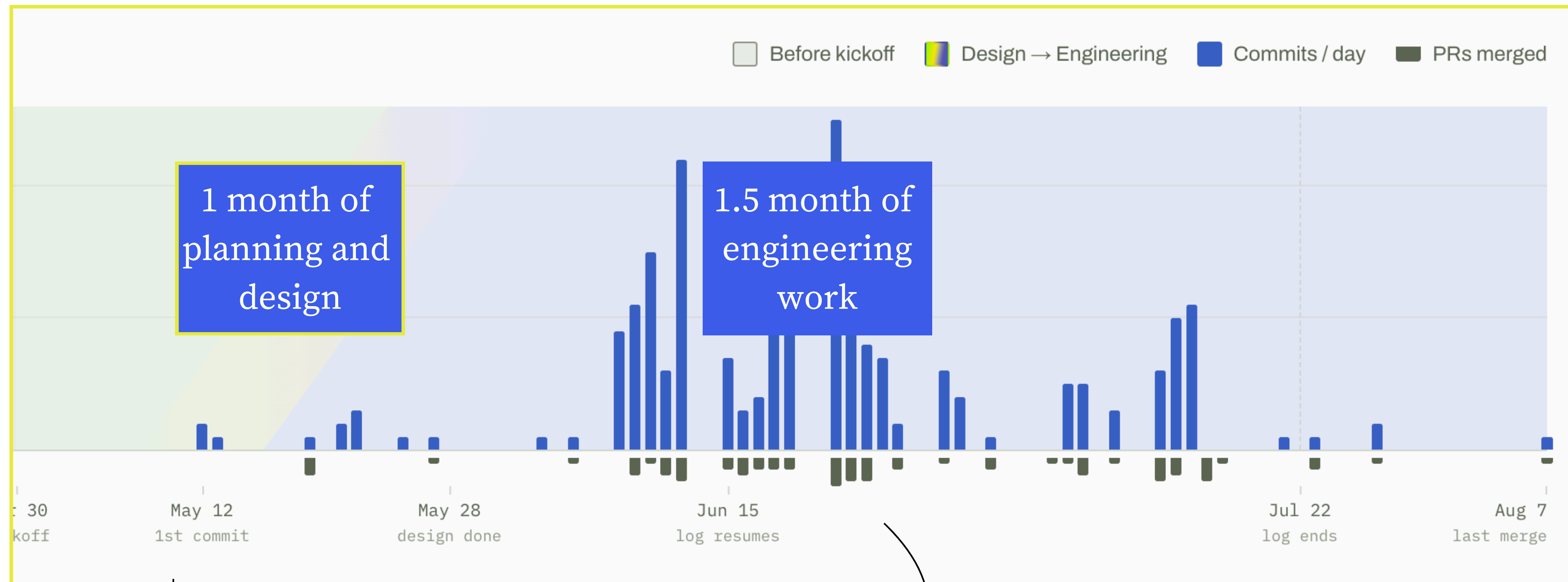
Questions nobody could agree on.

Until we looked at the project timeline.



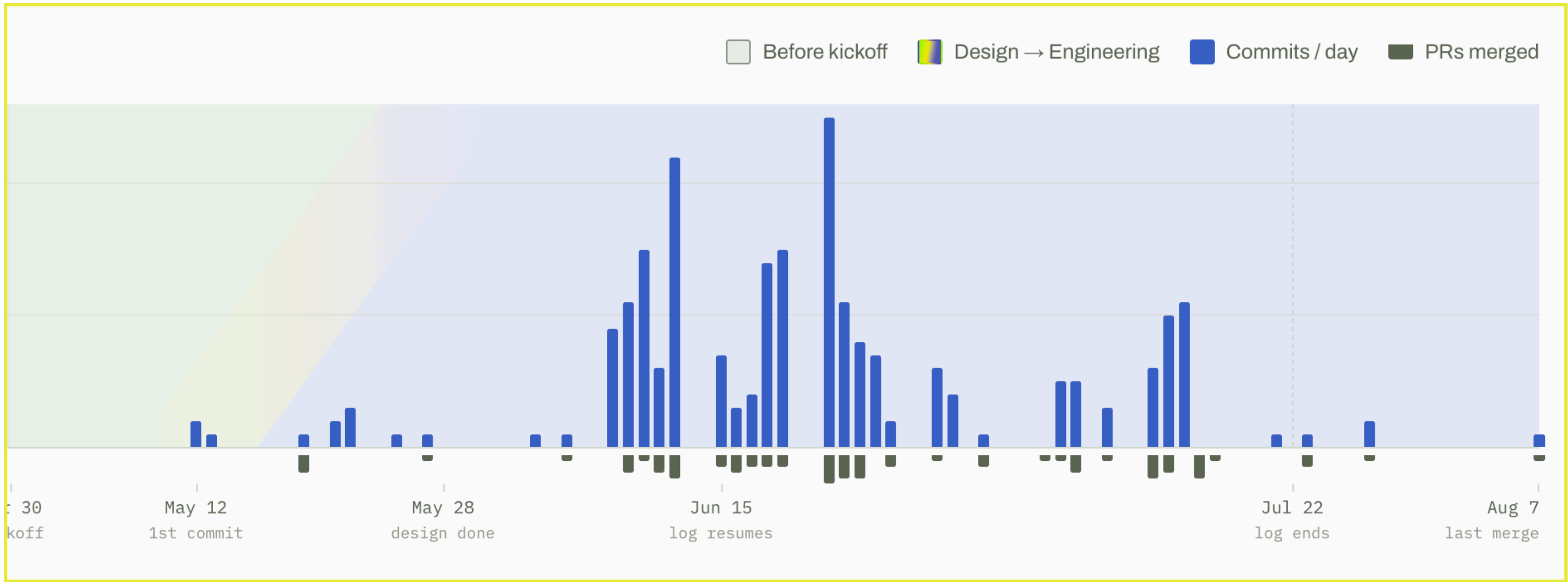
Time between kick-off and starting development was almost as much as development time.

We all **agreed** that was not good enough.



Time between kick-off and starting development was almost as much as development time.

We all **agreed** that was not good enough.



Product, then Design, then Engineering = **Waterfall**

WHOSE FAULT IS IT?

[Leaders]
Why is this
taking so long?

[Product Mgt]
Design is
blocking
progress

[Designer]
Requirements
are still in flux

[Engineers]
We don't have
anything to do

It was hard to get agreement since we were all discussing based on **different perspectives**.

A visualization of the facts on a timeline centered the conversation. It put the blame where it belonged: a broken process.

LESSON 04

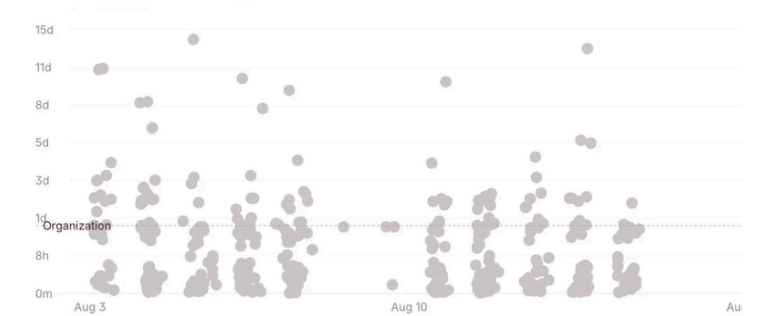
The software delivery system goes beyond engineering.

THE LESSONS

01 Time to complete work is what matters.
Everything else follows from it.



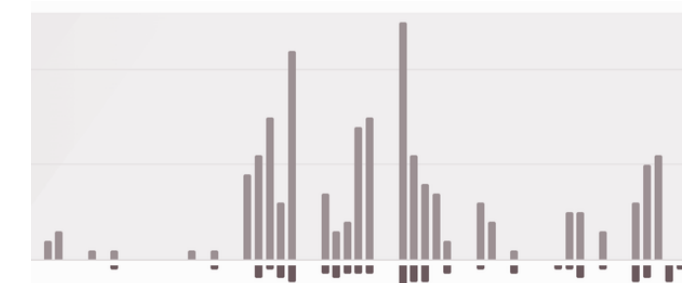
02 Stability comes from fixing the right issues
(and leaving others alone).



03 The best improvement for a team is to
ensure it's working on the right priority.



04 The software delivery system goes
beyond engineering.

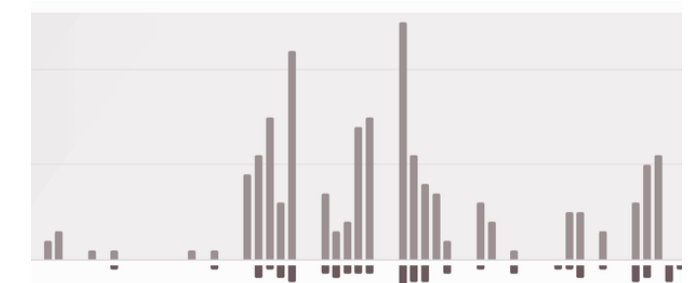
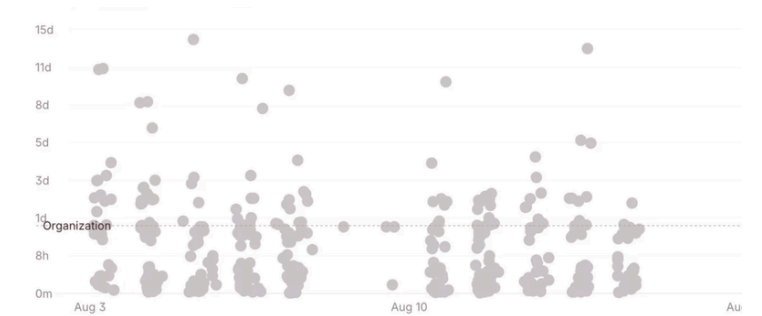


THE LESSONS

These are **not new problems**.

Teams continuously drift and struggle with maintaining effectiveness.

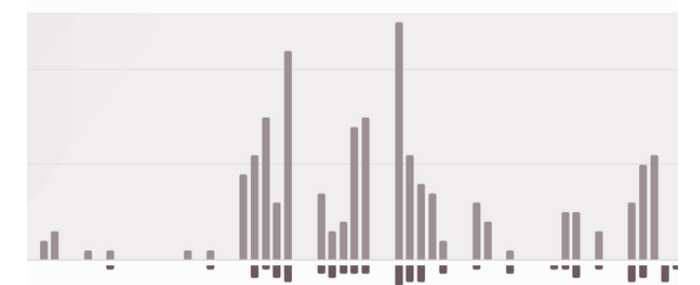
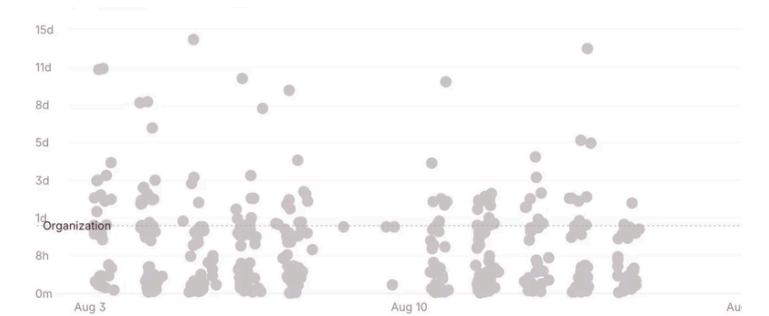
Any group (of humans or agents) will need constant alignment. **AI is making it harder to see.**



THE LESSONS

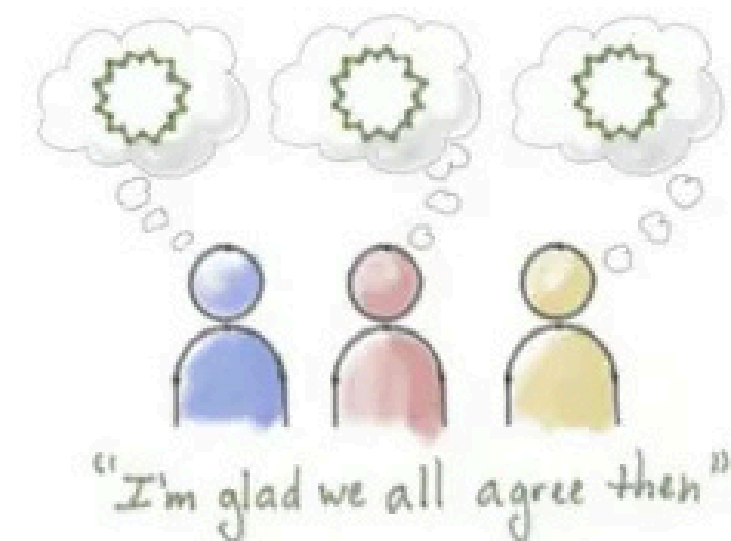
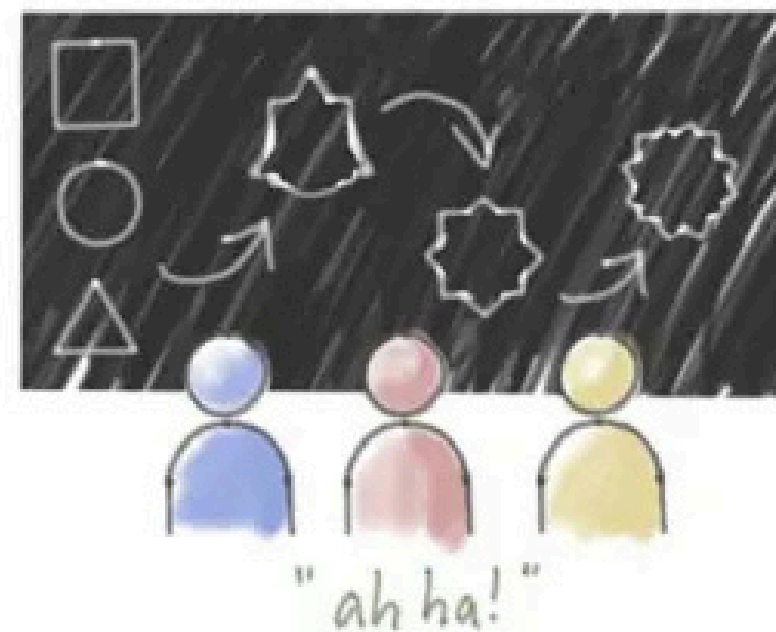
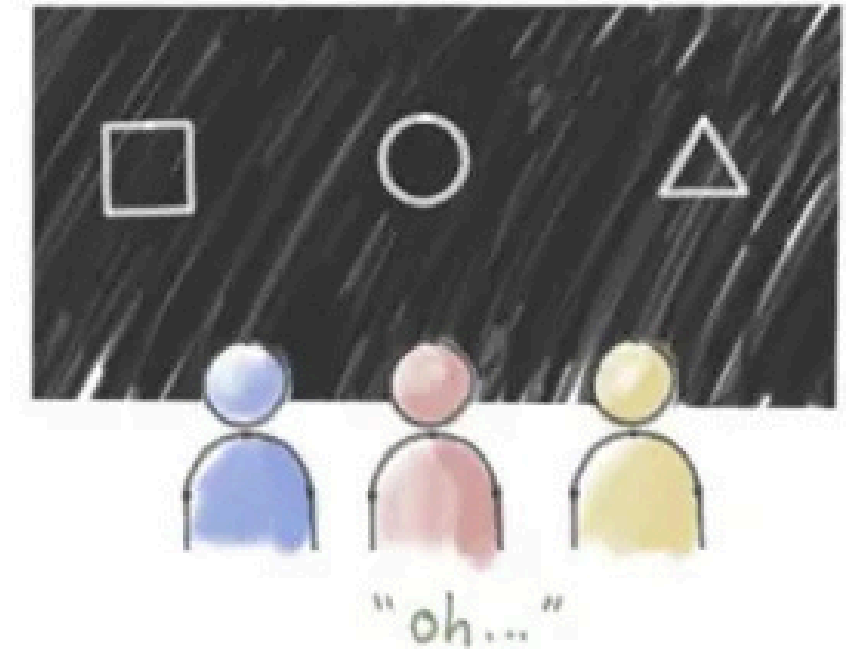
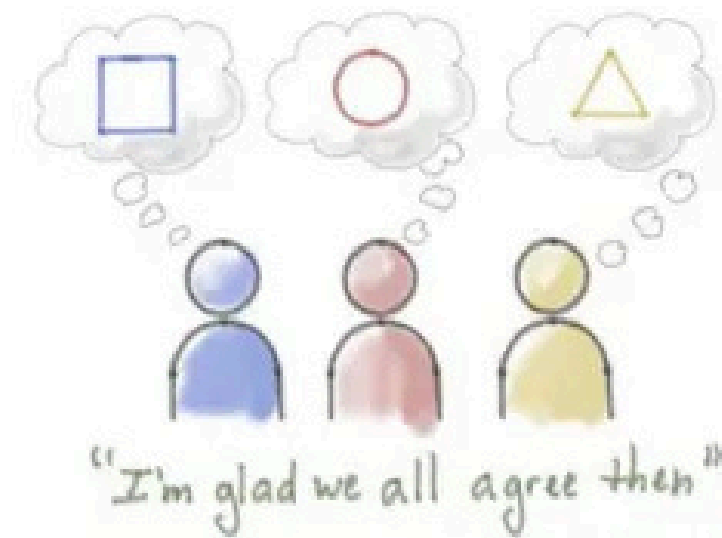
In a moment when **noise is going up.**

Visualizing the system continues to be
the best way to understand the path
forward.



It's not a new
idea.

Information
radiators have
existed for
decades.



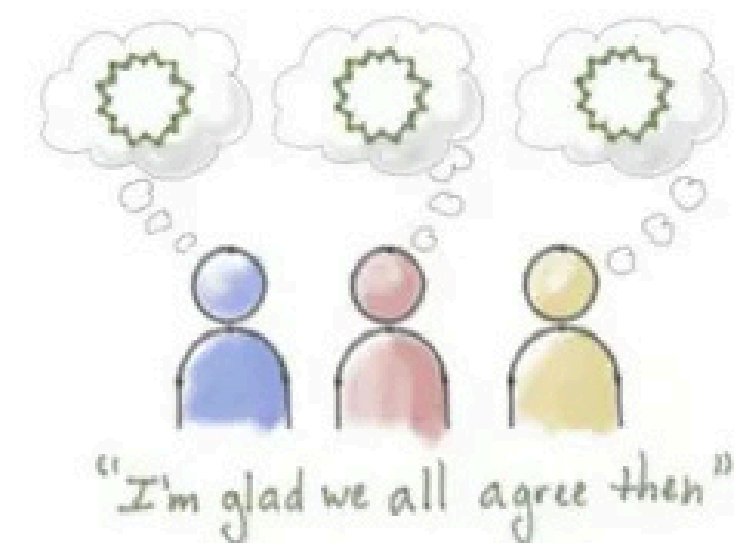
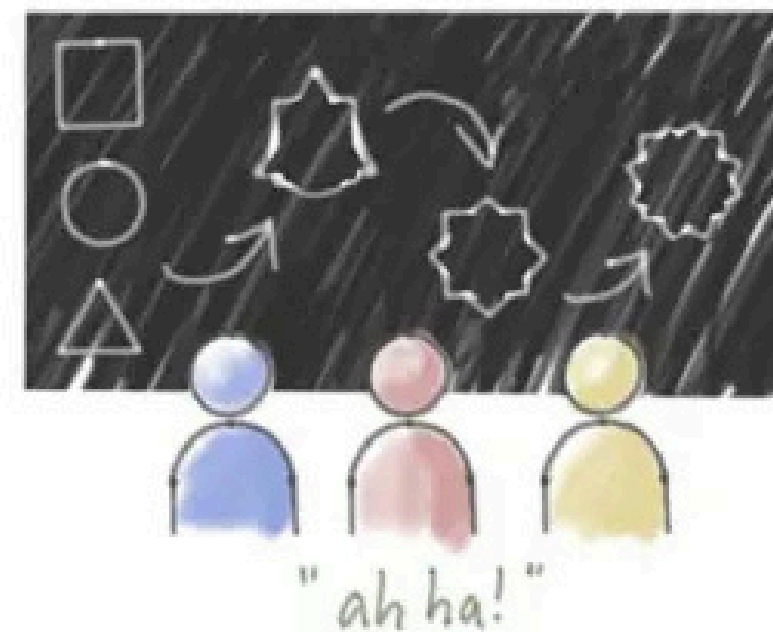
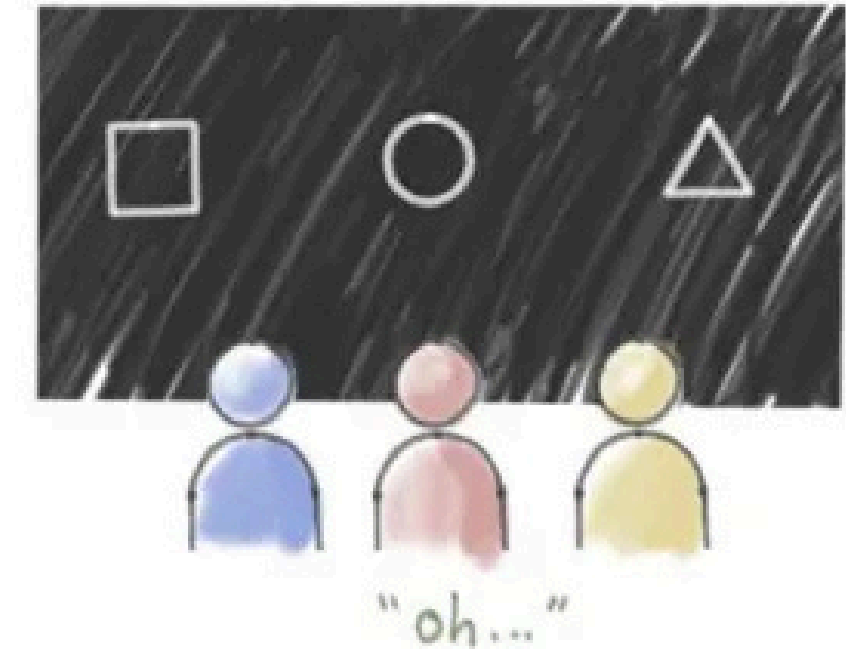
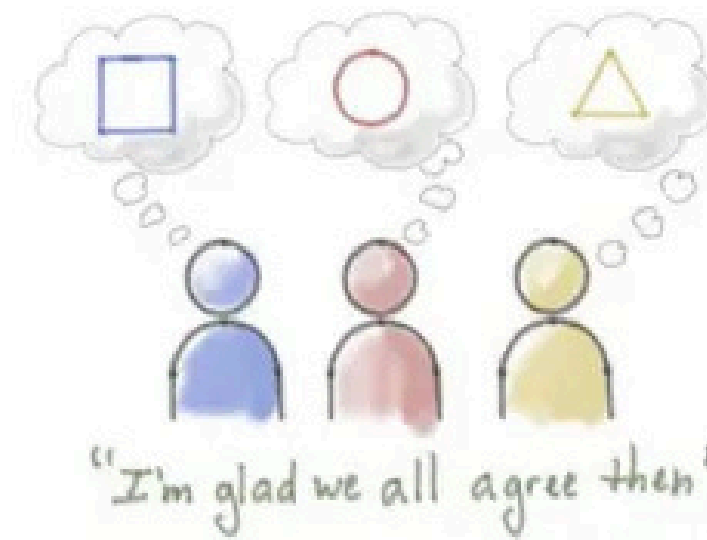
The main lesson:

Observe your team's system.

Understand it. Map the value stream.

Visualize it to make it obvious.

Then act on the right lever.



Thank you!

Leading Effective Software Teams is out today.

Meet the Author

Today at 14:00

O'Reilly booth

Come for a drink, chat, or a signed book!

Francisco Trindade
franciscotrindade.me

