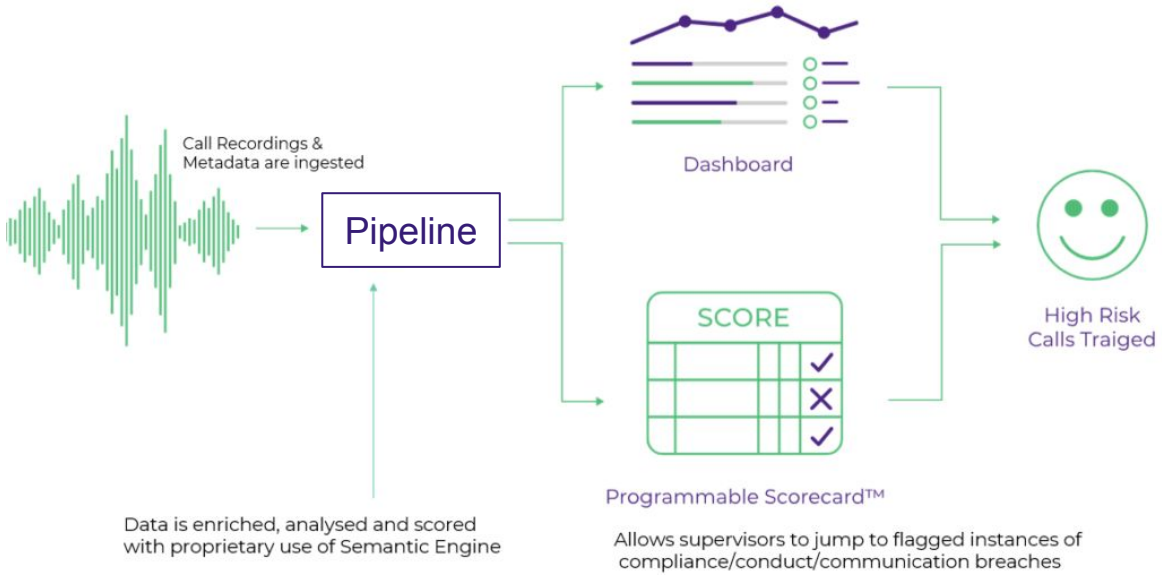
~2022.12.07

Observability is not Analytics!

Andrew Frederick Cowie
Head of Platform Engineering
Tlon Corporation

The End

On the violations of Brooks's Law



- Understand what production is doing
- Feedback between system and developers
- Simplify build system
- Increase deployment frequency
- Control costs
- …

Good News

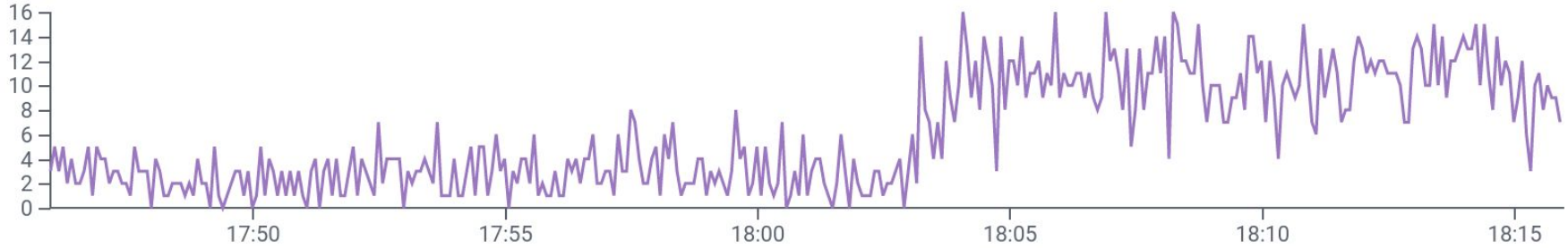
Implementing observability was a game-changer for us.

We dramatically reduced our time to identify problems, isolate causes, and see the effects of changes.

Team can readily get insight into both the correctness of individual services and modules, and the characteristics of the distributed system overall.

Total spans

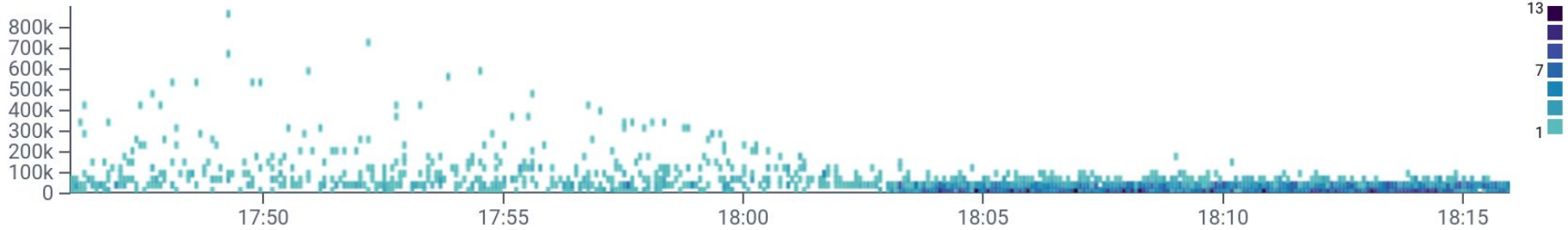
Run Query



Latency

Bubble Up

Run Query



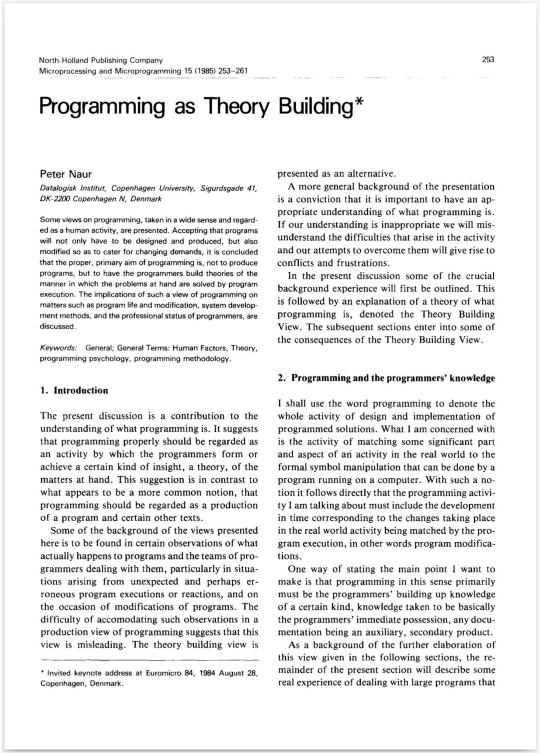
Bad News: It's harder than we'd like to think

“

During the program life a programmer team possessing its theory remains in active control of the program, and in particular retains control over all modifications. **The death of a program happens when the programmer team possessing its theory is dissolved.** A dead program may continue to be used for execution in a computer and to produce useful results. **The actual state of death becomes visible when demands for modifications of the program cannot be intelligently answered.** Revival of a program is the rebuilding of its theory by a new programmer team.

”

—Peter Naur, “Programming as Theory Building”, *Microprocessing and Microprogramming* 15 (1985) pp. 253-261



North-Holland Publishing Company
Microprocessing and Microprogramming 15 (1985) 253–261

253

Programming as Theory Building*

Peter Naur
Datalogisk Institut, Copenhagen University, Sigurdsgade 41, DK-2200 Copenhagen N, Denmark

Some views on programming, taken in a wide sense and regarded as a human activity, are presented. Accepting that programs will not only have to be designed and produced, but also modified so as to cater for changing demands, it is concluded that the proper, primary aim of programming is, not to produce programs, but to have the programmers build theories of the manner in which the problems at hand are solved by program execution. The implications of such a view of programming on matters such as program life and modification, system development methods, and the professional status of programmers, are discussed.

Keywords: General; General Terms: Human Factors, Theory, programming psychology, programming methodology.

1. Introduction

The present discussion is a contribution to the understanding of what programming is. It suggests that programming properly should be regarded as an activity by which the programmers form or achieve a certain kind of insight, a theory, of the matters at hand. This suggestion is in contrast to what appears to be a more common notion, that programming should be regarded as a production of a program and certain other texts.

Some of the background of the views presented here is to be found in certain observations of what actually happens to programs and the teams of programmers dealing with them, particularly in situations arising from unexpected and perhaps erroneous program executions or reactions, and on the occasion of modifications of programs. The difficulty of accomodating such observations in a production view of programming suggests that this view is misleading. The theory building view is presented as an alternative.

A more general background of the presentation is a conviction that it is important to have an appropriate understanding of what programming is. If our understanding is inappropriate we will misunderstand the difficulties that arise in the activity and our attempts to overcome them will give rise to conflicts and frustrations.

In the present discussion some of the crucial background experience will first be outlined. This is followed by an explanation of a theory of what programming is, denoted the Theory Building View. The subsequent sections enter into some of the consequences of the Theory Building View.

2. Programming and the programmers' knowledge

I shall use the word programming to denote the whole activity of design and implementation of programmed solutions. What I am concerned with is the activity of matching some significant part and aspect of an activity in the real world to the formal symbol manipulation that can be done by a program running on a computer. With such a notion it follows directly that the programming activity I am talking about must include the development in time corresponding to the changes taking place in the real world activity being matched by the program execution, in other words program modifications.

One way of stating the main point I want to make is that programming in this sense primarily must be the programmers' building up knowledge of a certain kind, knowledge taken to be basically the programmers' immediate possession, any documentation being an auxiliary, secondary product.

As a background of the further elaboration of this view given in the following sections, the remainder of the present section will describe some real experience of dealing with large programs that

* Invited keynote address at Euromicro 84, 1984 August 28, Copenhagen, Denmark.

Respect Error Budgets

or

Rewrite Entire System

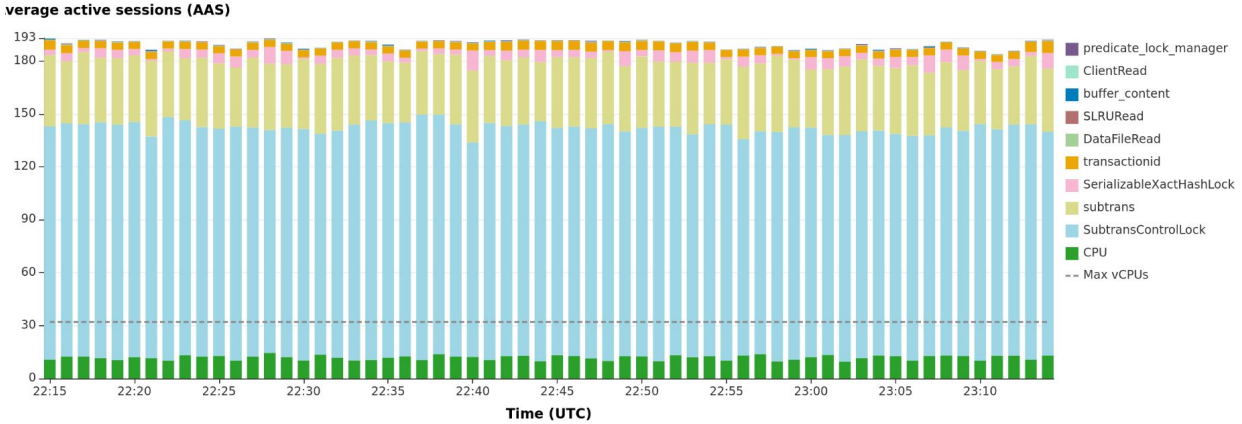
Bad News: It's harder than we'd like to think

We found ourselves with a legacy system where we were unable to safely making even basic changes.

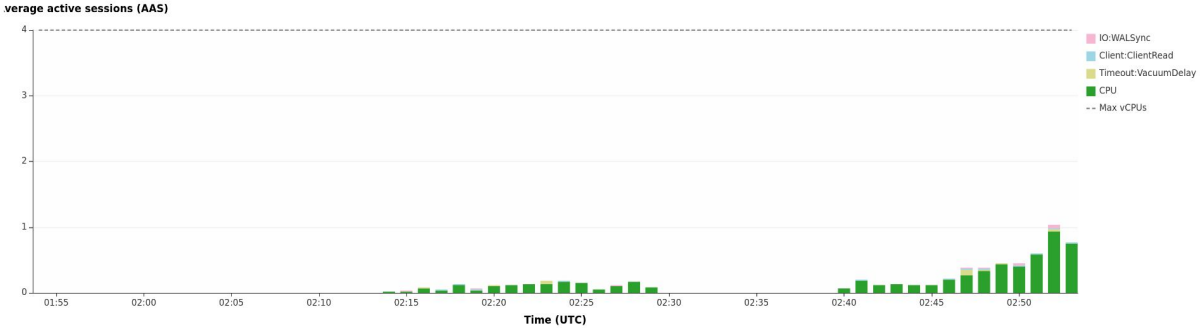
I'm in awe of people who manage to bolt observability tooling into existing systems; we just couldn't do it. Fred Brooks taught us to be wary of doing things over, but we reached the point where we just didn't understand the behaviour of the existing codebase. Being able to do observability *at all* was a major motivation for what turned out to be a massive rewrite of almost the entire system.

The effort represented over half a year's effort by the entire engineering team.

96 CPUs



4 CPUs



Success!

Jumped our iteration rate from ~10 code changes per month to over 300

Averaging over 50 deployments per month, up from 4-8.

+1

Have repeatability, observability, and stability. New features are coming out at a rapid rate; users are thrilled.

Reduced our AWS infrastructure bill by over 70%


Stability is
Hyper Local


T+ 0y, 0d, 00:00:45
MET
012546m
ATMOSPHERE
VERT SPD
Mechjeb
Orbit Info
Orbital speed 621.7 m/s
Apoapsis 30.224 km
Periapsis -596.440 km
Orbital period 9m 56.43s
Time to apoapsis 1m 03.1s
Time to periapsis 6m 01.3s
Inclination 0.658 °
Eccentricity 0.989
Angle to prograde 182.32 °
Maneuver Planner
Create a new maneuver node to:
↓ circularize ↓
Schedule the burn
↓ at the next apoapsis ↓
Create node
Create and execute
Remove ALL nodes
Auto-warp Tolerance:
Lead time:
Delta-V Stats
ΔV include cosine losses
Stage ΔV (atmo, vac) 0, 0 m/s
Total ΔV (atmo, vac) 2770, 3023 m/s
Stage stats
Short stats
Live SLT
Stage TWR SLT Atmo ΔV Vac ΔV Time
0 1.81 1.66 2770 m/s 3023 m/s 1m 52.8s
1 0.00 0.00 0 m/s 0 m/s 00.0s
2 0.00 0.00 0 m/s 0 m/s 00.0s
3 0.00 0.00 0 m/s 0 m/s 00.0s
Maneuver Node Editor
No maneuver nodes to edit.
Conics mode
Current mode: RELATIVE
Surface
568.9m/s
RCS
SAS
HDG 087°
STAGE
002
ROLL
PITCH
YAW

Endpoint Roulette

How many ways can we get YOUR details?

request.path	
/api/v3/user/ad4d251c-b643-46e6-a290-ab32f15ae9b3/details	
/api/v3/userDetails/ad4d251c-b643-46e6-a290-ab32f15ae9b3	
/api/v3/userDetails?clientId=ad4d251c-b643-46e6-a290-ab32f15ae9b3	

Endpoint Roulette

How many ways can we get YOUR details? (1)

request.path	
/api/v3/user/ad4d251c-b643-46e6-a290-ab32f15ae9b3**/details**	
/api/v3/user/31617980-50c5-41ba-a109-9d47fe5338f4**/details**	
/api/v3/user/dcd35c1e-a4aa-4bf5-92d2-9b90b80f59a3**/details**	
/api/v3/user/76fa0b5f-0158-4ea1-87af-d6e19f649333**/details**	

Endpoint Roulette

How many ways can we get YOUR details? (2)

request.path	
/api/v3/userDetails/ad4d251c-b643-46e6-a290-ab32f15ae9b3	
/api/v3/userDetails/31617980-50c5-41ba-a109-9d47fe5338f4	
/api/v3/userDetails/dcd35c1e-a4aa-4bf5-92d2-9b90b80f59a3	
/api/v3/userDetails/76fa0b5f-0158-4ea1-87af-d6e19f649333	

Endpoint Roulette

How many ways can we get YOUR details? (3)

request.path	
/api/v3/userDetails?clientId=ad4d251c-b643-46e6-a290-ab32f15ae9b3	
/api/v3/userDetails?clientId=31617980-50c5-41ba-a109-9d47fe5338f4	
/api/v3/userDetails?clientId=dcd35c1e-a4aa-4bf5-92d2-9b90b80f59a3	
/api/v3/userDetails?clientId=76fa0b5f-0158-4ea1-87af-d6e19f649333	

How many hits on `userDetails`?

Endpoint Roulette

request.path	COUNT
/api/v3/userDetails?clientId=ad4d251c-b643-46e6-a290-ab32f15ae9b3	1
/api/v3/userDetails?clientId=31617980-50c5-41ba-a109-9d47fe5338f4	1
/api/v3/userDetails?clientId=dcd35c1e-a4aa-4bf5-92d2-9b90b80f59a3	1
/api/v3/userDetails?clientId=76fa0b5f-0158-4ea1-87af-d6e19f649333	1
…	…

Endpoint Roulette

request.path	
/api/v3/userDetails/ad4d251c-b643-46e6-a290-ab32f15ae9b3	
/api/v3/userDetails?clientId=ad4d251c-b643-46e6-a290-ab32f15ae9b3	
/api/v3/user/ad4d251c-b643-46e6-a290-ab32f15ae9b3/details	

request.route	request.query
/api/v3/userDetails	
/api/v3/userDetails	?clientId=ad4d251c-b643-46e6-a290-ab32f15ae9b3
???	

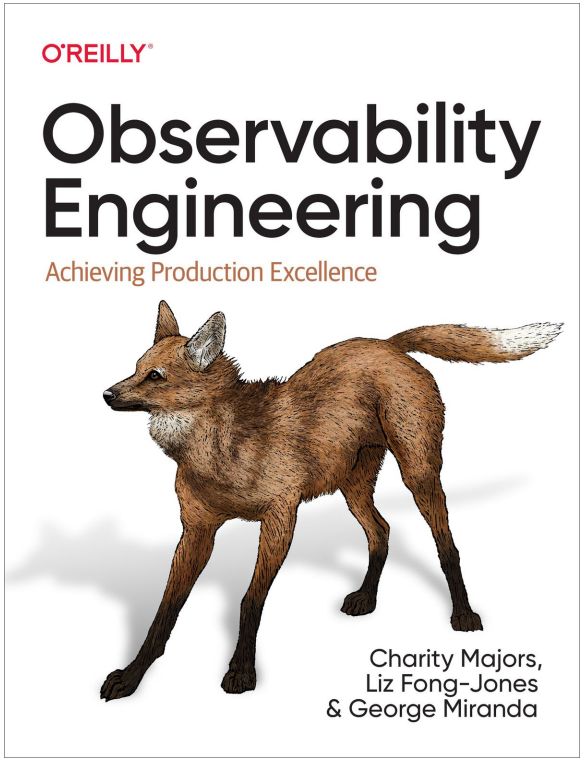
Endpoint Roulette

How many hits on *endpoints*?

request.route	COUNT
/api/v3/playMusic	23,995
/api/v3/rateSong	4,154
/api/v3/userDetails	631
/api/v3/updateProfile	49
…	…

“…without needing to ship new code to production.”

—Charity Majors, Liz Fong-Jones, and George Miranda *Observability Engineering* (2022) pp. 6, 17, 279.



Don’t know in advance everything you need from a given event. Many things *aren’t* useful. `i = 16`

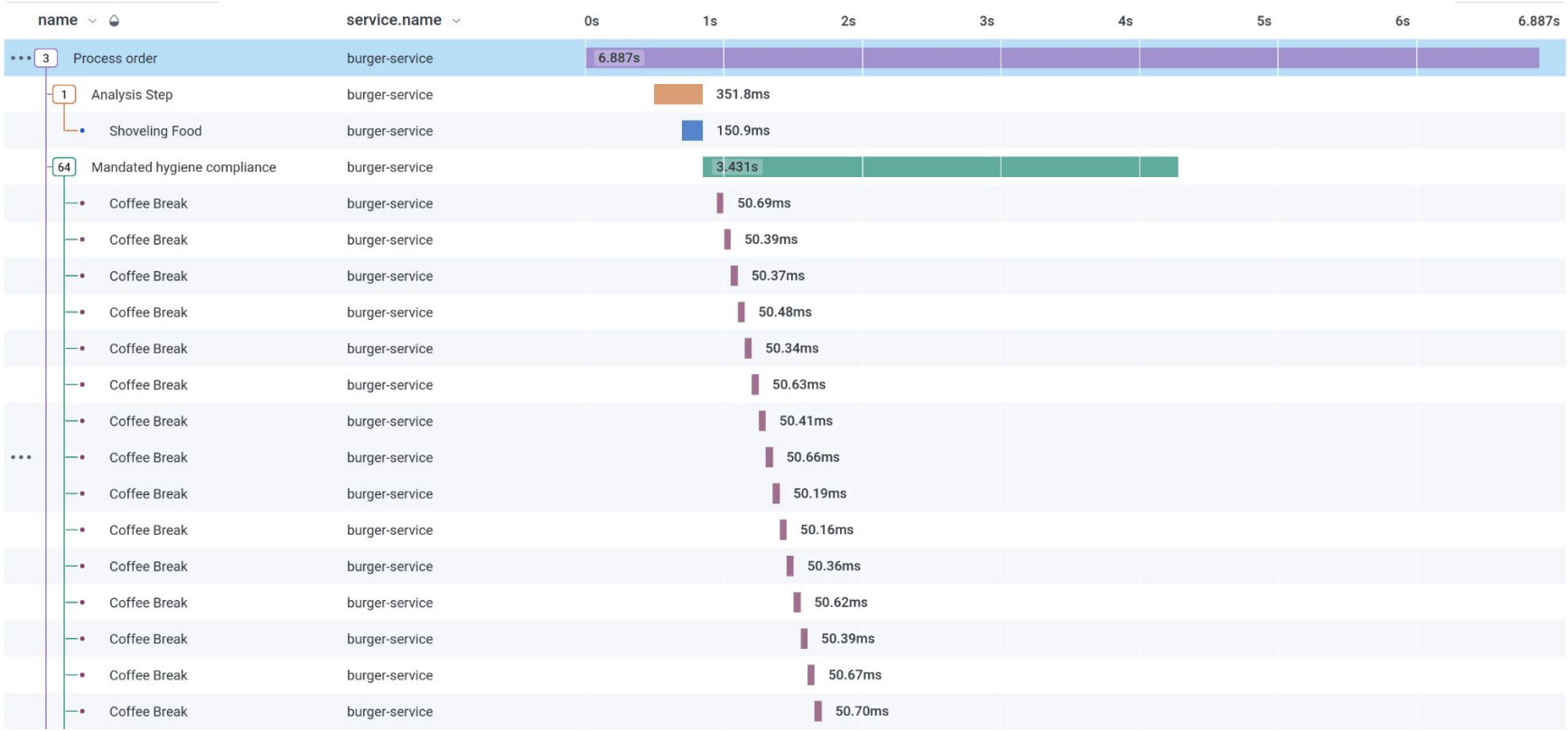
Pollution of field namespace.

Turns out you DO need `i`, except that it’s `retries`, and you need it on the parent span.

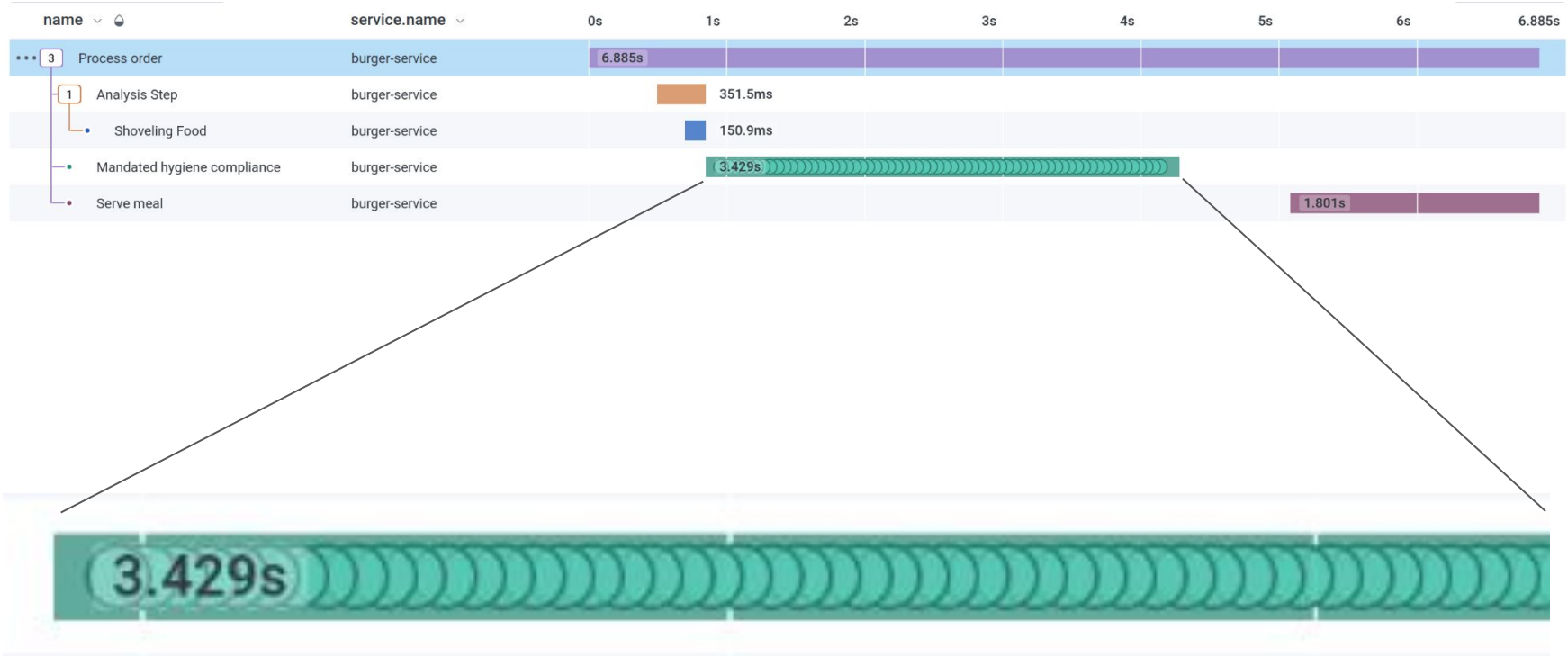
Oh, btw, the spans you have are wrong:

“Yeah, nah.”

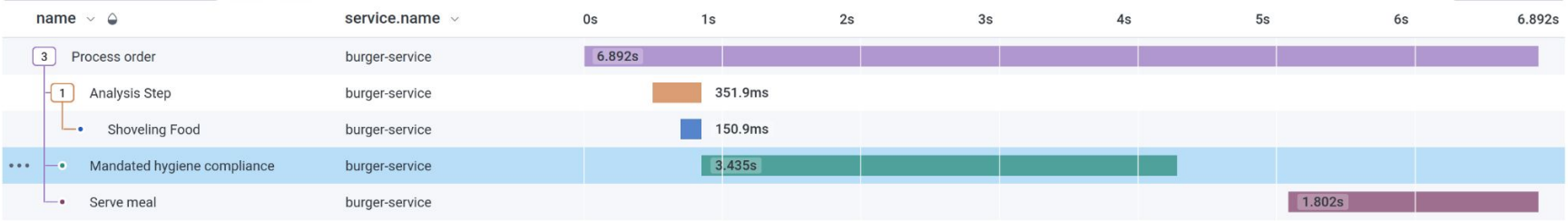
—legendary Australian saying, reliably dated to 1788 when first nations peoples of the continent were presented with the idea of Europeans coming to live with them.



waterfall development methodology ftw



wtf span events



str name

Mandated hygiene compliance

str orderId

25d4d5cd-a8e7-4658-9282-b65e7c89cb47

flt retry

64

str service.name

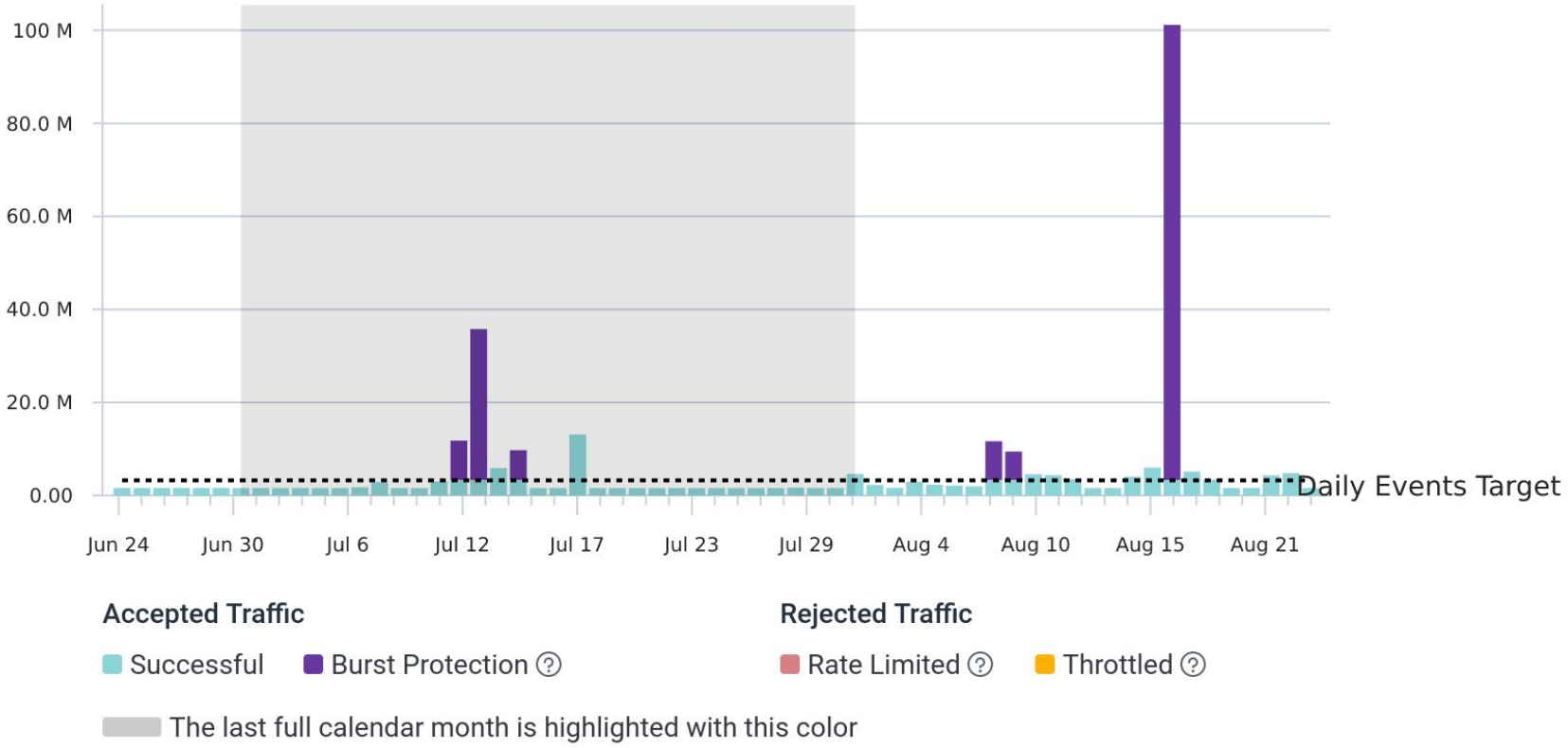
burger-service

Namespace pollution

app.**name**space
k8s.**name**space
kubernetes.app.**name**
app.label
app.**name**
k8s.app.**name**.label
k8s.worker.instance.host
breakfast.order.menuitem.**name**

Send all the events, they said…

Daily Event Traffic, last 60 days



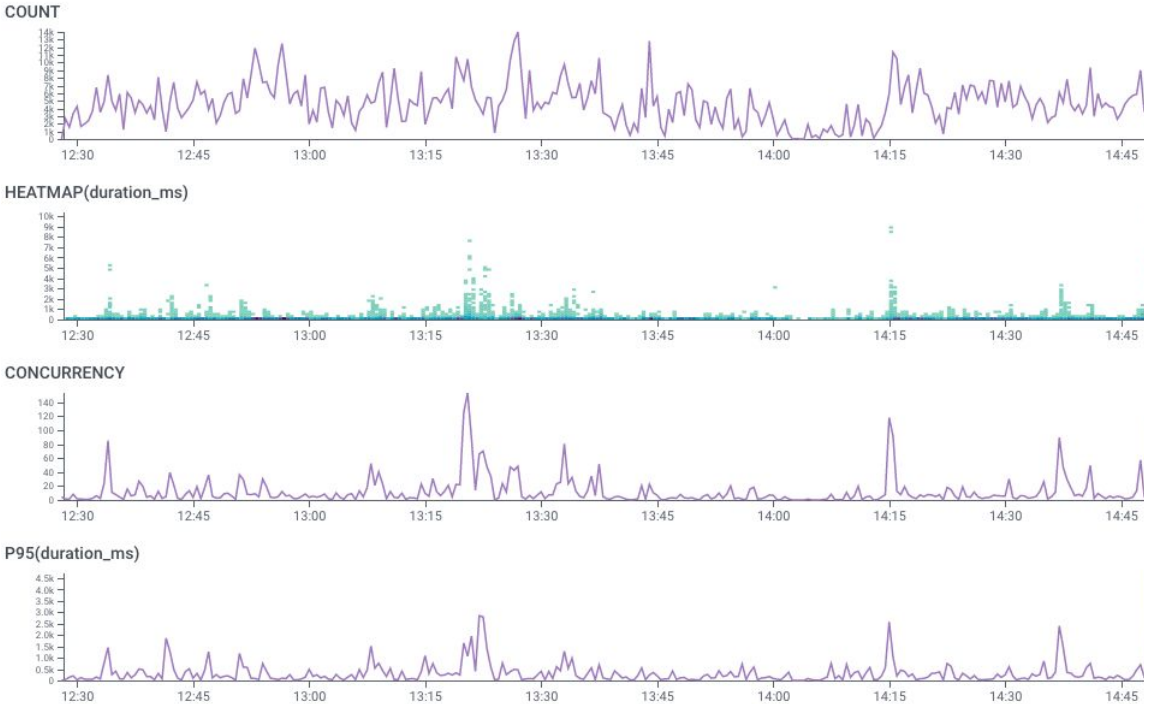
Easy to see problems, right?

HINT: This probably means that some data is corrupted and you will have to use the last backup for recovery.

@timestamp	@message
2022-07-14T11:03:06.…	2022-07-14 01:03:06 UTC:127.0.0.1(65420):rdsadmin@rdsadmin:[379]:FATAL: the database system is starting up
2022-07-14T11:03:02.…	2022-07-14 01:03:02 UTC:127.0.0.1(65390):rdsadmin@rdsadmin:[375]:FATAL: the database system is starting up
2022-07-14T11:03:02.…	2022-07-14 01:03:02 UTC:127.0.0.1(65394):rdsadmin@rdsadmin:[376]:FATAL: the database system is starting up
2022-07-14T11:03:02.…	2022-07-14 01:03:02 UTC:127.0.0.1(65412):rdsadmin@rdsadmin:[377]:FATAL: the database system is starting up
2022-07-14T11:03:02.…	2022-07-14 01:03:02 UTC:127.0.0.1(65414):rdsadmin@rdsadmin:[378]:FATAL: the database system is starting up
2022-07-14T11:02:58.…	2022-07-14 01:02:58 UTC:127.0.0.1(65366):rdsadmin@rdsadmin:[367]:FATAL: the database system is starting up
2022-07-14T11:02:58.…	2022-07-14 01:02:58 UTC:127.0.0.1(65368):rdsadmin@rdsadmin:[368]:FATAL: the database system is starting up
2022-07-14T11:02:58.…	2022-07-14 01:02:58 UTC:127.0.0.1(65372):rdsadmin@rdsadmin:[371]:FATAL: the database system is starting up
2022-07-14T11:02:58.…	2022-07-14 01:02:58 UTC:127.0.0.1(65374):rdsadmin@rdsadmin:[374]:FATAL: the database system is starting up
2022-07-14T11:02:54.…	2022-07-14 01:02:54 UTC:127.0.0.1(65364):rdsadmin@rdsadmin:[366]:FATAL: the database system is starting up
2022-07-14T11:02:53.…	2022-07-14 01:02:53 UTC:127.0.0.1(65356):rdsadmin@rdsadmin:[363]:FATAL: the database system is starting up
2022-07-14T11:02:53.…	2022-07-14 01:02:53 UTC:127.0.0.1(65360):rdsadmin@rdsadmin:[364]:FATAL: the database system is starting up
2022-07-14T11:02:53.…	2022-07-14 01:02:53 UTC:127.0.0.1(65362):rdsadmin@rdsadmin:[365]:FATAL: the database system is starting up
2022-07-14T11:02:52.…	2022-07-14 01:02:52 UTC:127.0.0.1(65354):rdsadmin@rdsadmin:[362]:FATAL: the database system is starting up
2022-07-14T11:02:51.…	2022-07-14 01:02:51 UTC:127.0.0.1(65350):rdsadmin@rdsadmin:[356]:FATAL: the database system is starting up
2022-07-14T11:02:49.…	2022-07-14 01:02:49.795 GMT [351] LOG: skipping missing configuration file "/rdsdbdata/config/recovery.conf"
2022-07-14T11:02:49.…	2022-07-14 01:02:49.795 GMT [351] LOG: skipping missing configuration file "/rdsdbdata/config/recovery.conf"
2022-07-14T11:02:49.…	2022-07-14 01:02:49 UTC::@:[351]:HINT: Future log output will appear in directory "/rdsdbdata/log/error".
2022-07-14T11:02:49.…	2022-07-14 01:02:49 UTC::@:[353]:HINT: This probably means that some data is corrupted and you will have to use the last backup for recovery.
2022-07-14T11:02:47.…	2022-07-14 01:02:47.700 GMT [28124] LOG: skipping missing configuration file "/rdsdbdata/config/recovery.conf"

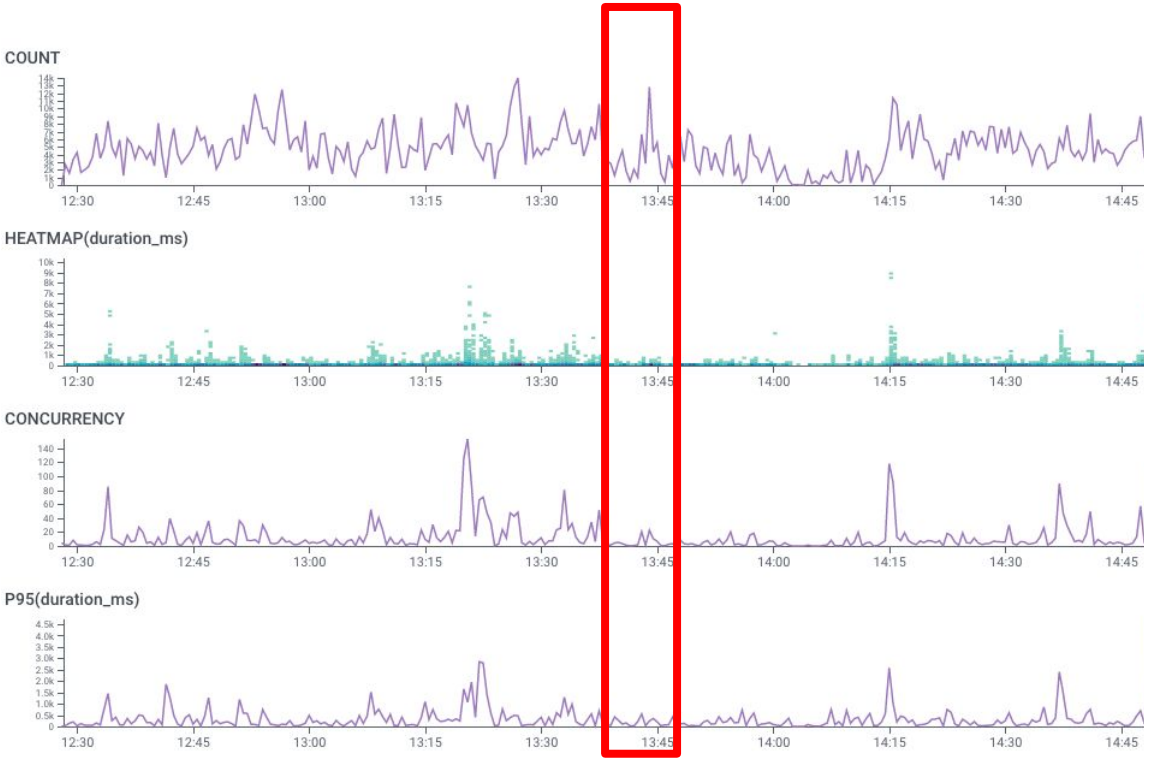
Easy to see problems, right?

Show me the user experiencing 5XX errors:



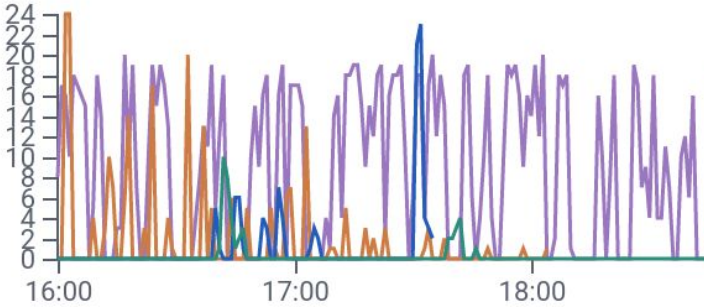
Source: Michael Pistrang via Honeycomb #chart channel, https://honeycombpollinators.slack.com/archives/CJMQWHSBW/p1649448061667239

See it now?

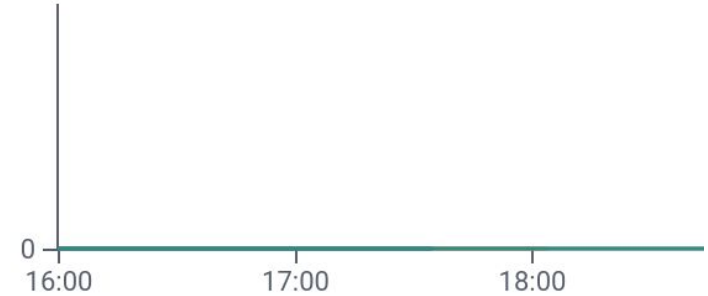

COUNT
12:30
12:45
13:00
13:15
13:30
13:45
14:00
14:15
14:30
14:45
HEATMAP(duration_ms)
CONCURRENCY
P95(duration_ms)

No Carrier

Total spans



Error rate



Your telemetry is not a continuous function.

The observability data set itself is not stable—or won't be if you are continuously improving it.

- evolve a set of rules to guide when to enclose code in a span, when to add a field, and what naming conventions to follow.

Surprise! as we iterate the code we iterate our telemetry, too. After all, once you've learned something and changed the system, it's now a *new* system.

And you can’t report on this.

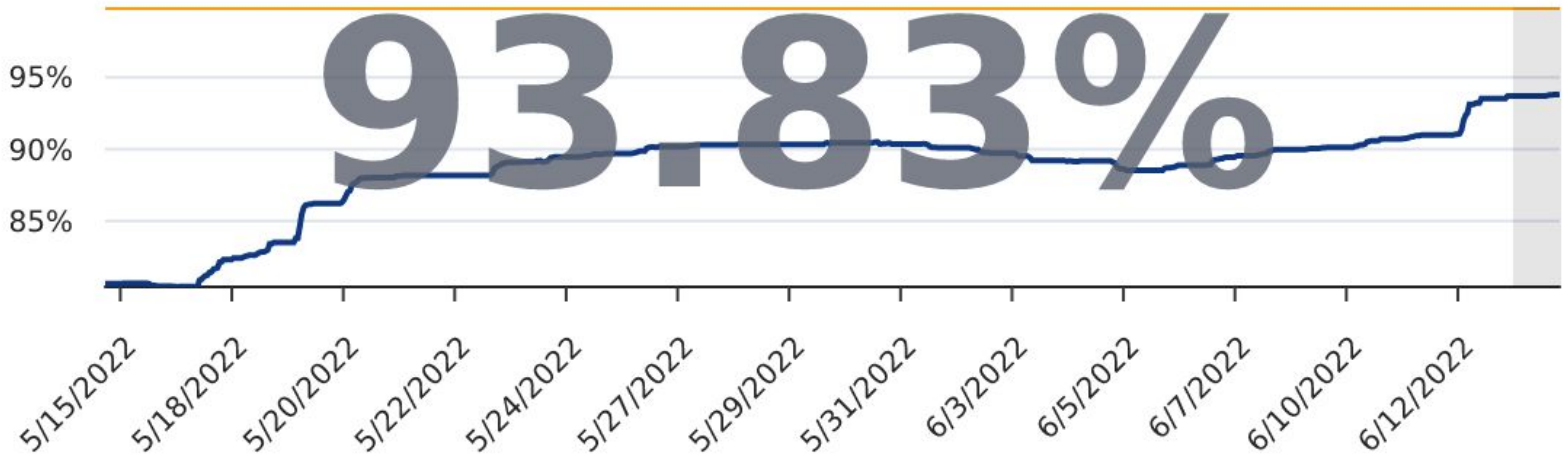
Reporting in the Observability Age

If observability data is no substitute for business metrics,

Can you actually measure your SLOs using SLIs?

Historical SLO Compliance

For each day of the past 30, how often this SLI has succeeded over the preceding 30 days.




T+ 0y, 8d, 04:42:46
MET
Orbit Info
Orbital speed 138.3 m/s
Apoapsis 13.207 km
Periapsis -27.795 km
Orbital period 30m 09.25s
Time to apoapsis 25m 01.7s
Time to periapsis 9m 57.1s
Inclination 168.534 °
Eccentricity 0.389
Angle to prograde 212.97 °
Vessel Info
Max acceleration 8.070 m/s²
Current acceleration 0 m/s²
Max thrust 250.0 kN
Vessel mass 30.979 t
Surface TWR 16.45
Crew capacity 3
How it started
Maneuver Planner
Create a new maneuver node to:
↓ change inclination ↓
New inclination: 0 °
Schedule the burn
↓ at the highest equatorial AN/DN ↓
Create node
Create and execute
Remove ALL nodes
Auto-warp Tolerance: m/s
Lead time: s
Delta-V Stats
ΔV include cosine losses
Stage ΔV (atmo, vac) 1194, 1194 m/s
Total ΔV (atmo, vac) 3253, 3253 m/s
Stage stats
Short stats
Live SLT
Stage TWR SLT Atmo ΔV Vac ΔV Time
0 0.00 0.00 0 m/s 0 m/s 00.0s
1 0.00 0.00 0 m/s 0 m/s 00.0s
2 0.00 0.00 0 m/s 0 m/s 00.0s
3 28.75 28.75 2060 m/s 2060 m/s 1m 49.8s
4 0.00 0.00 0 m/s 0 m/s 00.0s
5 16.45 16.45 1194 m/s 1194 m/s 2m 04.9s
Maneuver Node Editor
No maneuver nodes to edit.
Conics mode: 0 1 2 3 4
Current mode: RELATIVE
Orbit
138.3m/s
RCS
SAS
THROTTLE
G MODE
HDG 079°

How it’s going

What engineering cares about:

- ❏ Uptime & availability
- ❏ Improving performance
- ❏ Deployment frequency
- ❏ Healthy team dynamics
- ❏ Staying within error budget
- ❏ SITUATIONAL AWARENESS
- ❏ Experience for existing customers

What business cares about:

- ❏ Is my feature finished yet?
  (which feature? The one where you're changing the colour of the logo, duh.)
- ❏ How come my mate from Hipster Co can't see the white text on white background?!?
- ❏ Onboarding new customers
- ❏ Why is it slow!
  (which part is slow? Yes.)

What CEO cares about:

- Is it finished yet? Why is it late!
- Getting ARR to the target we promised investors in time.
- Financials. Burn rate. Runway.

  Do we really need engineers? If we outsourced this, it would cost less, right?
- $$$$$

What investors care about:

- $$$$$$

WORK BACKWARDS ↑

Culture

Responsiveness

Delivery

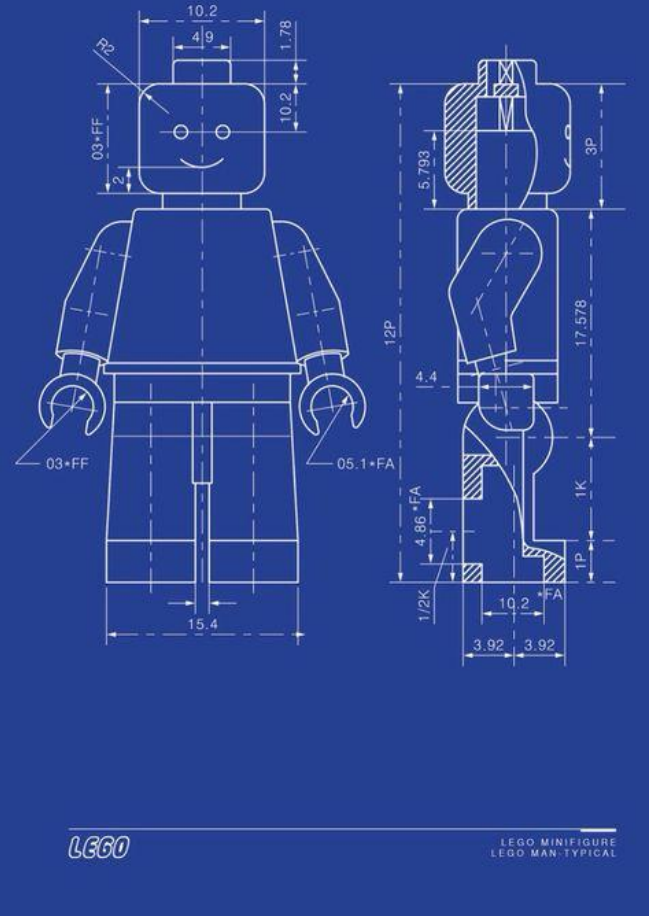
$$$$$$

Observability is a tool.

It’s not the end unto itself.

The End


Coda
Bow Shock Near a Young Star in the Orion Nebula
HUBBLESITE.org



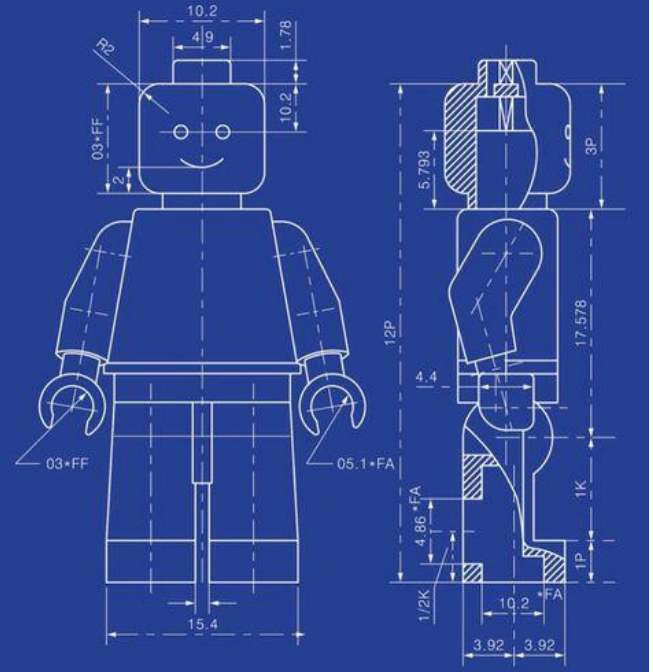
Engineering *(tr. v.)*

(1) Applying science and technology to the **purposes** of an organization;

achieving **outcomes**;

in the face of **constraints**: financial feasibility, social acceptability, and ethical correctness.

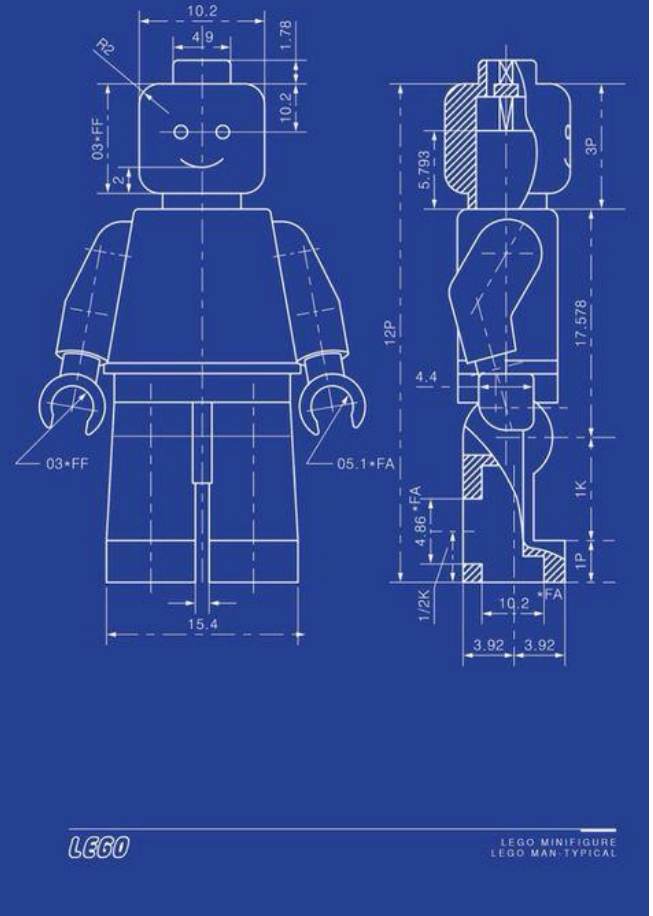
(2) Taking **responsibility** for (1)



Software Development → **Software Engineering**

↓

Systems Administration → **Systems Engineering**



Software Development → **Software Engineering**

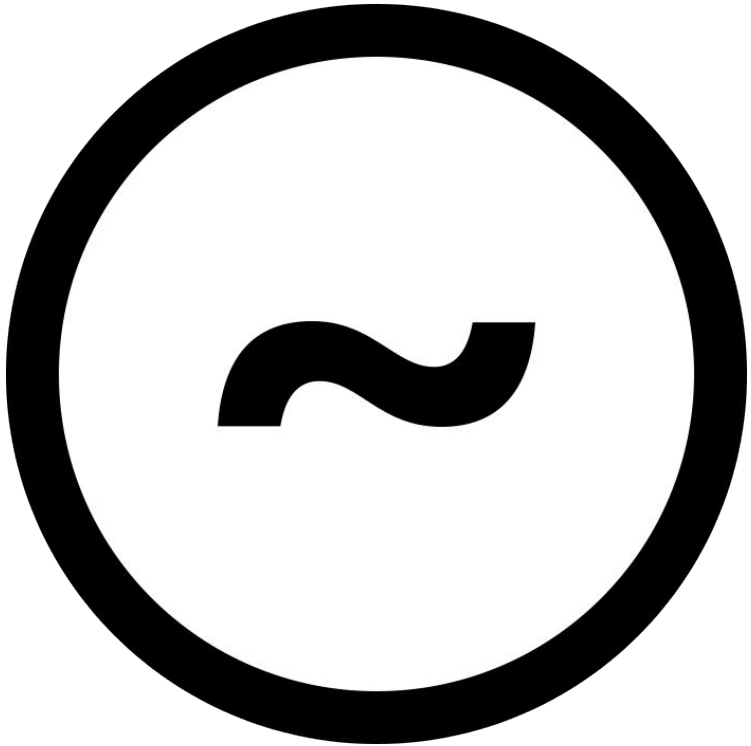
↓

Systems Administration → **Platform Engineering**

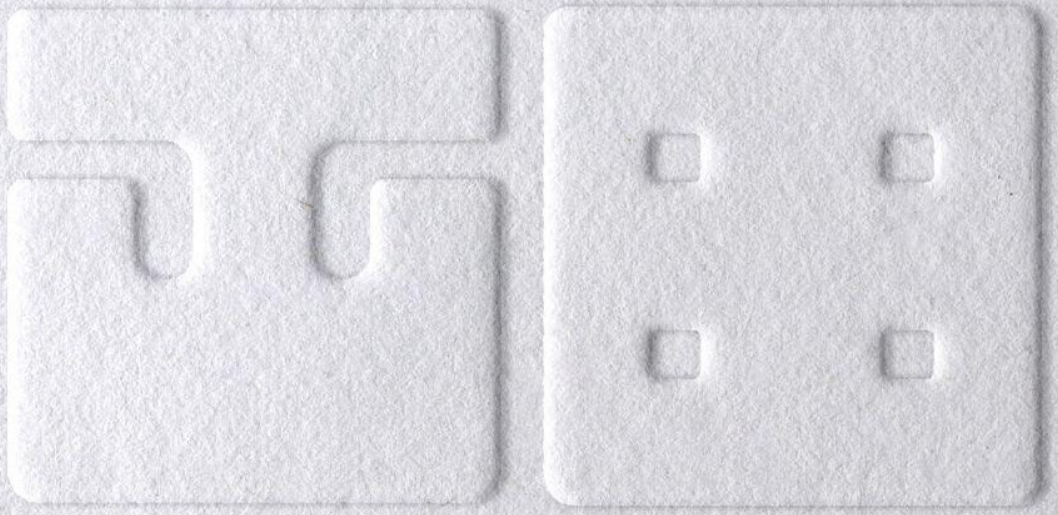
Urbit

Urbit is a new OS and peer-to-peer network that’s simple by design, built to last forever, and 100% owned by its users.

Under the hood, Urbit is a clean-slate software stack compact enough that an individual developer can understand and control it completely.



Computing for Us, not Them.



- Understand what production is doing
- Feedback between system and developers
- Simplify build system
- Increase deployment frequency
- Control costs
- …

Tlon Hosting

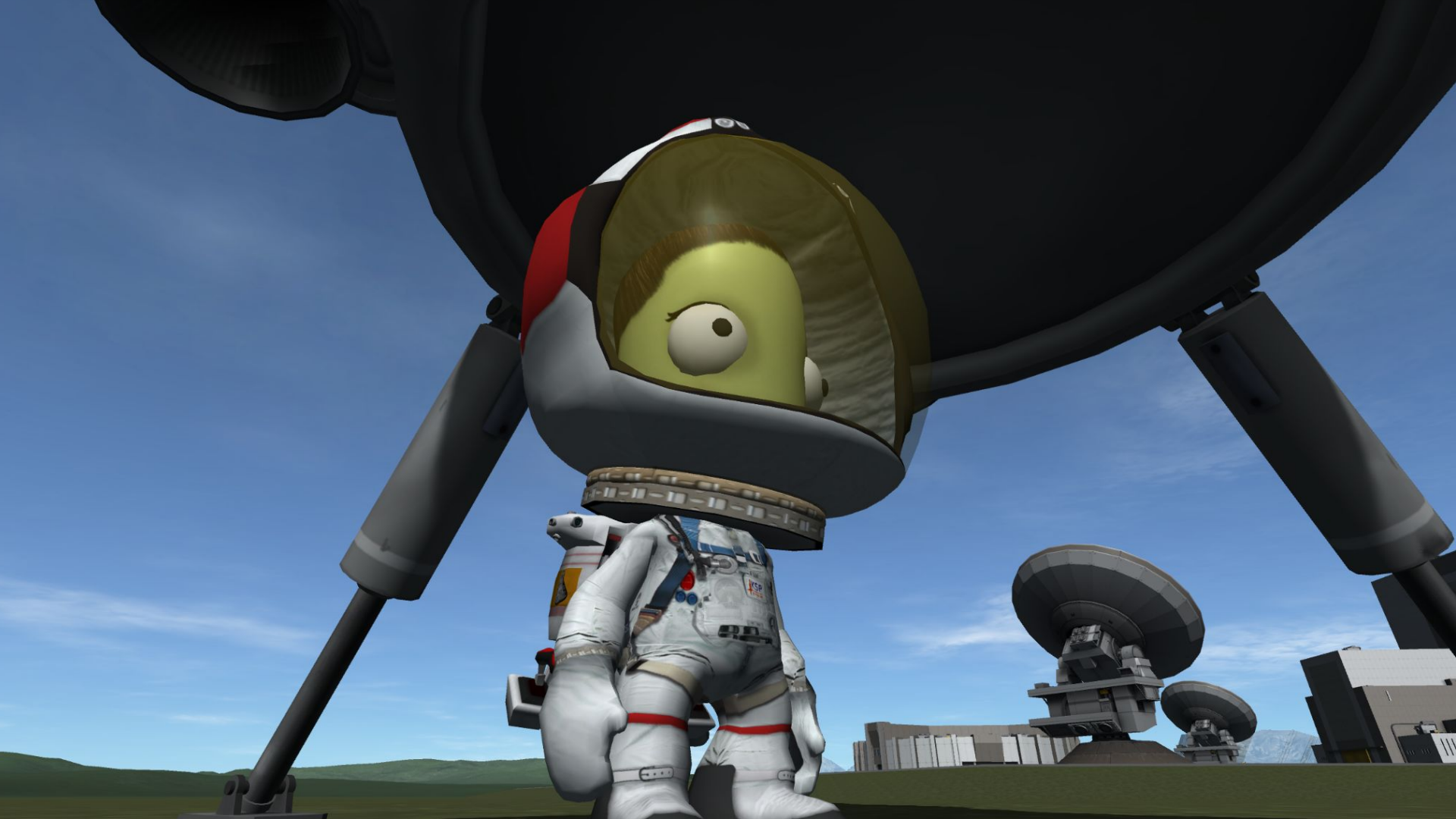
➔ Implement Observability!

- Understand what production is doing
- Feedback between system and developers
- Simplify build system
- Increase deployment frequency
- Control costs
- …

Head Of Platform Engineering

Tlon Hosting


HOPE



The End

(really)


Mastodon
@istathar@tilde.zone
GitHub
istathar
Urbit
~maclyr-nimnyl
Reach out!