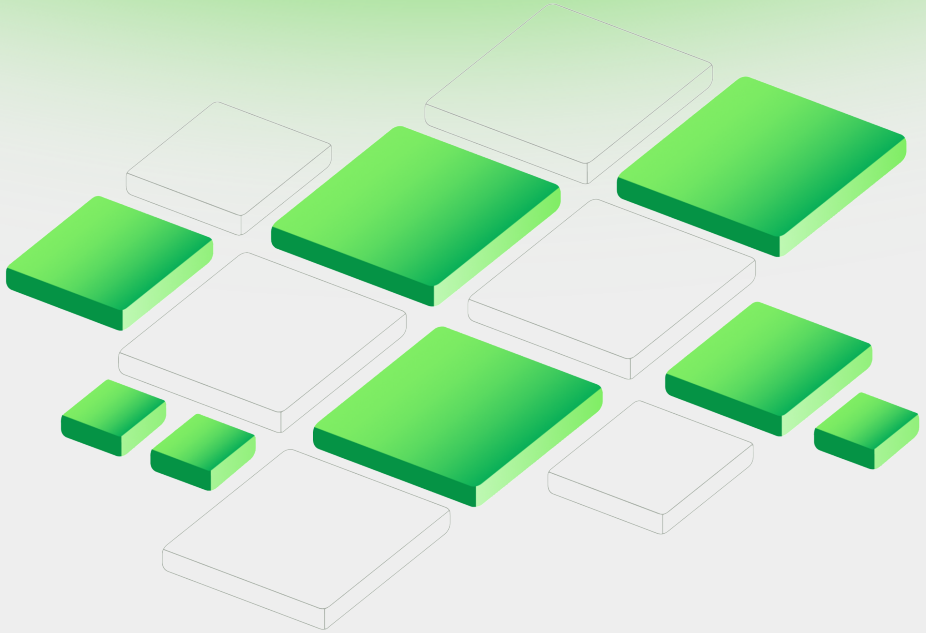
Do startups have time for STAMP's controlling ways?



usenix SRECON

Jonathon Belotti
Modal Labs, MTS



01

What startup?

What startup?



Modal

01

Modal

The platform for production AI



Inference

Serve LLMs, image, video, and audio models.



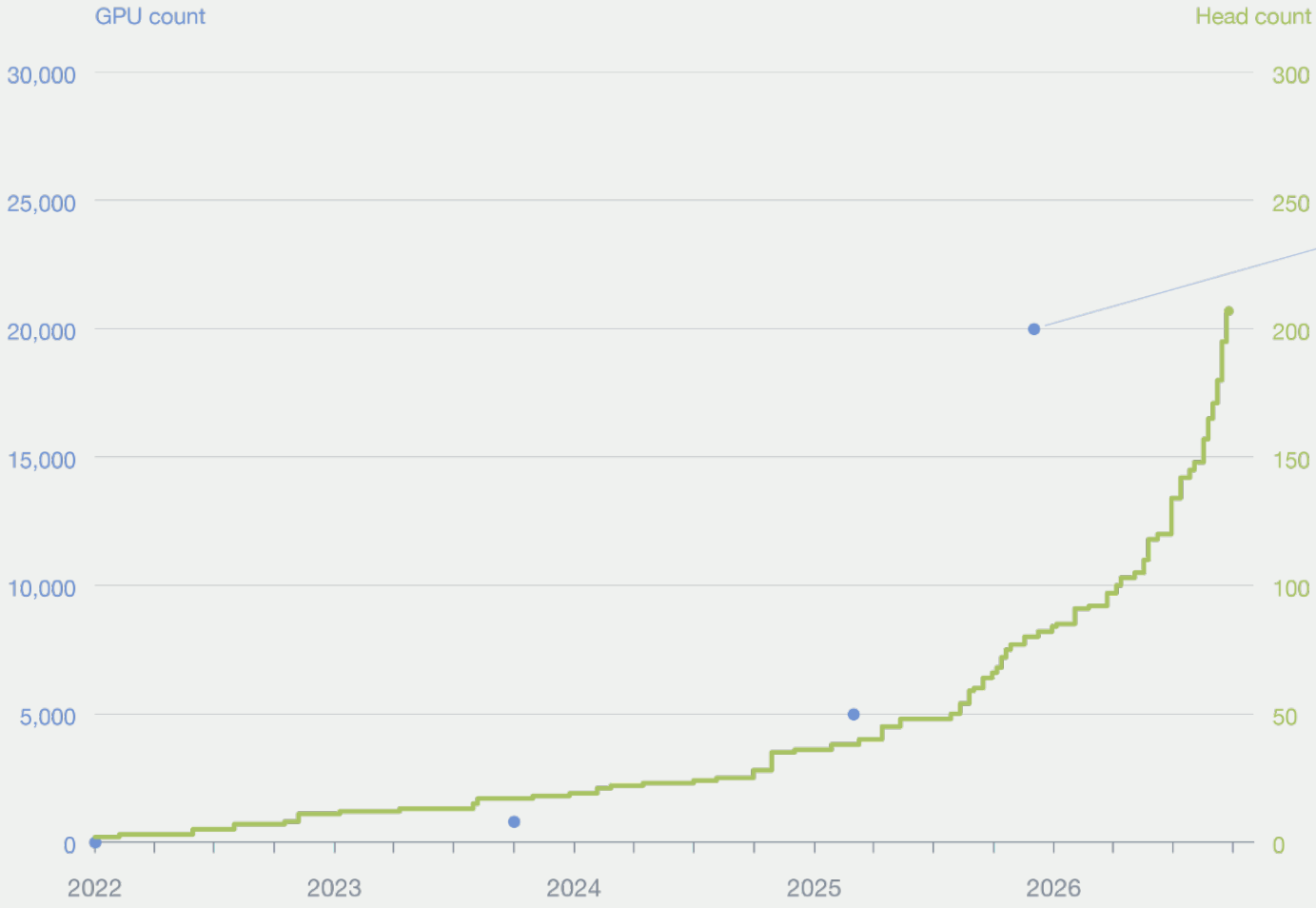
Training

Fine-tune, run RL, and evaluate models on GPUs.



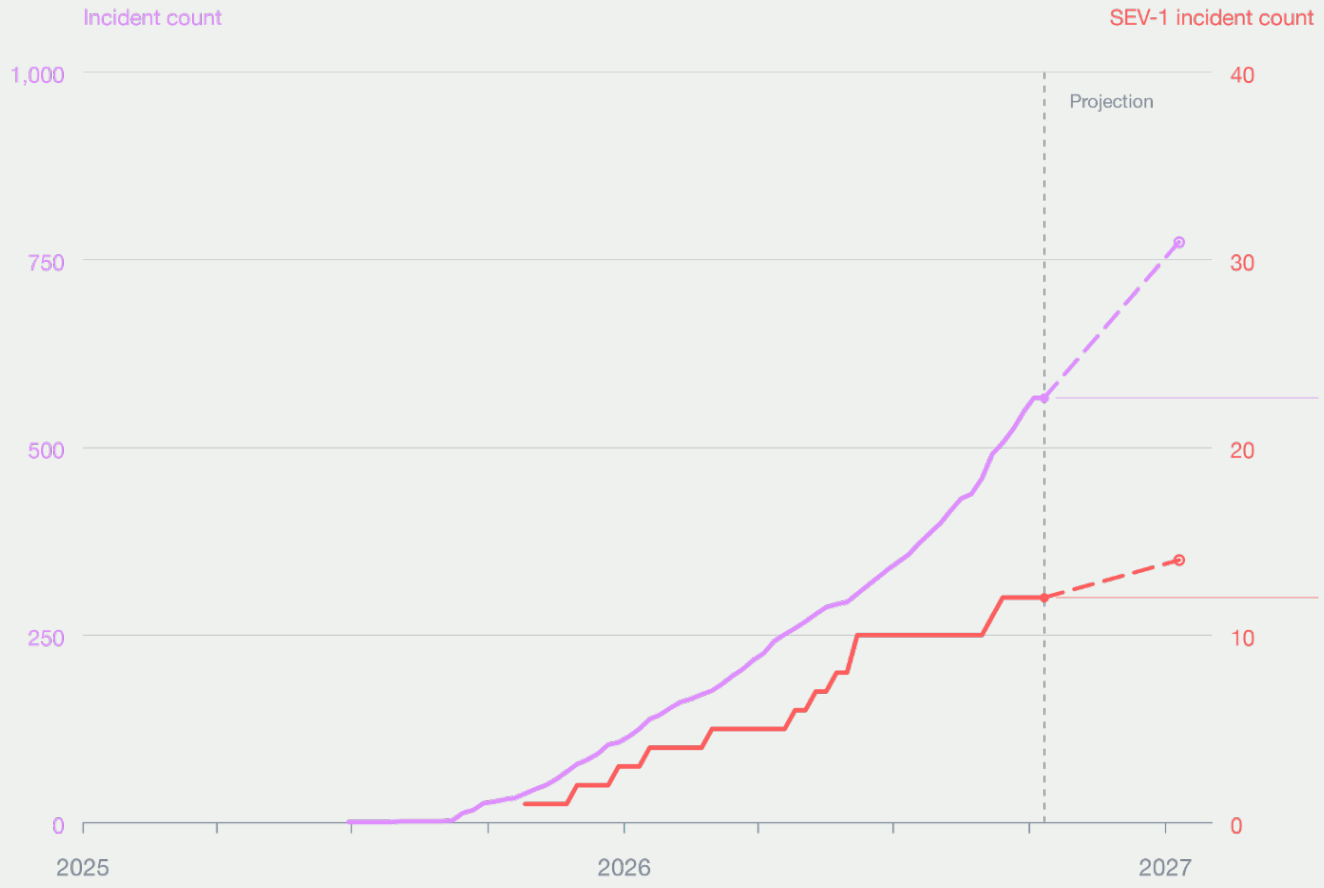
Sandboxes

Isolated environments for coding agents, RL rollouts, and untrusted code.



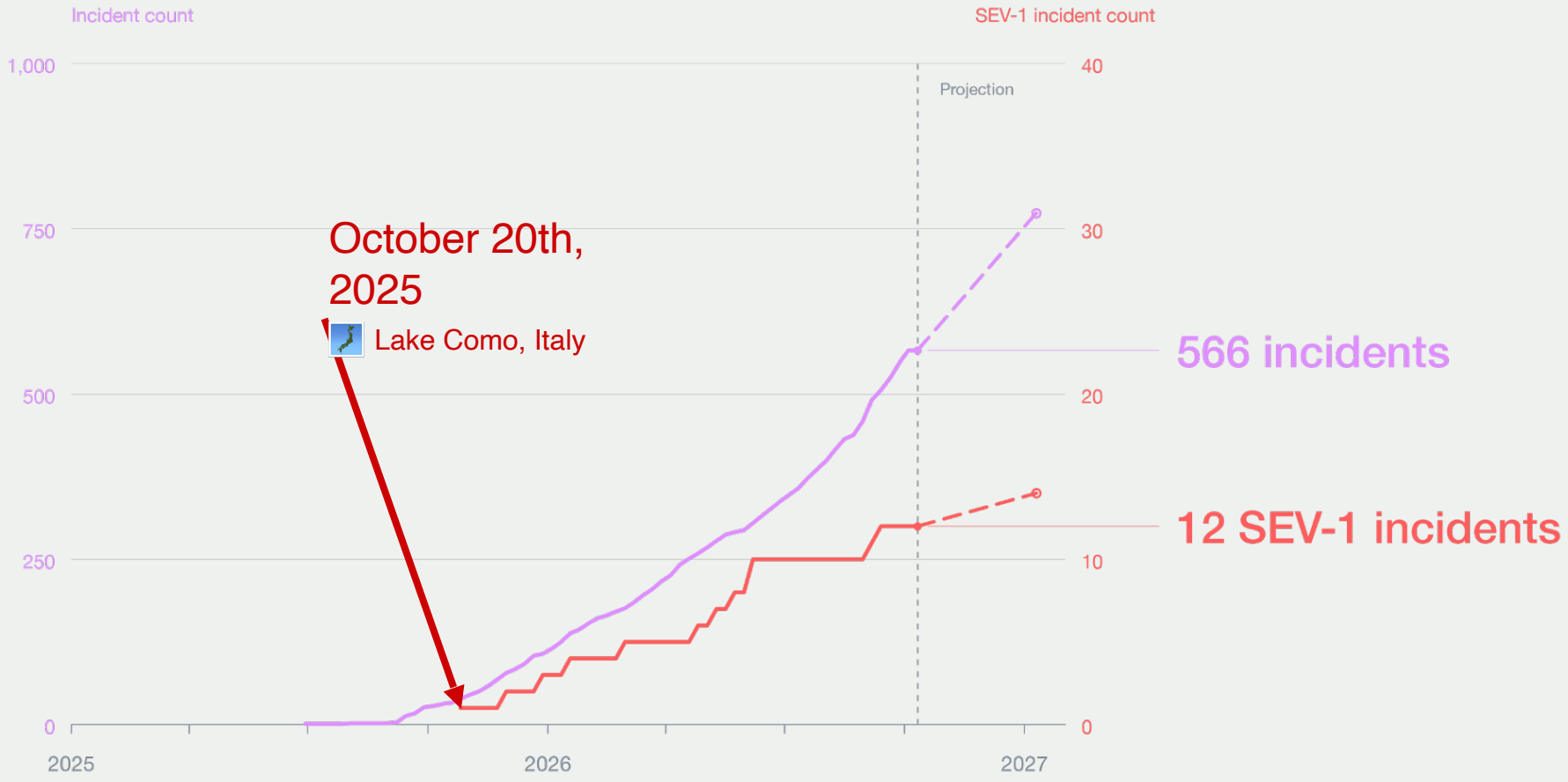
20,000 GPUs

207 employees



566 incidents

12 SEV-1 incidents


Incident count
SEV-1 incident count
1,000
750
500
250
0
40
30
20
10
0
Projection
October 20th, 2025
Lake Como, Italy
566 incidents
12 SEV-1 incidents
2025
2026
2027

Crisis

#incident-41, a.k.a
The Amazon DynamoDB Service Disruption in the Northern Virginia (US-EAST-1) Region

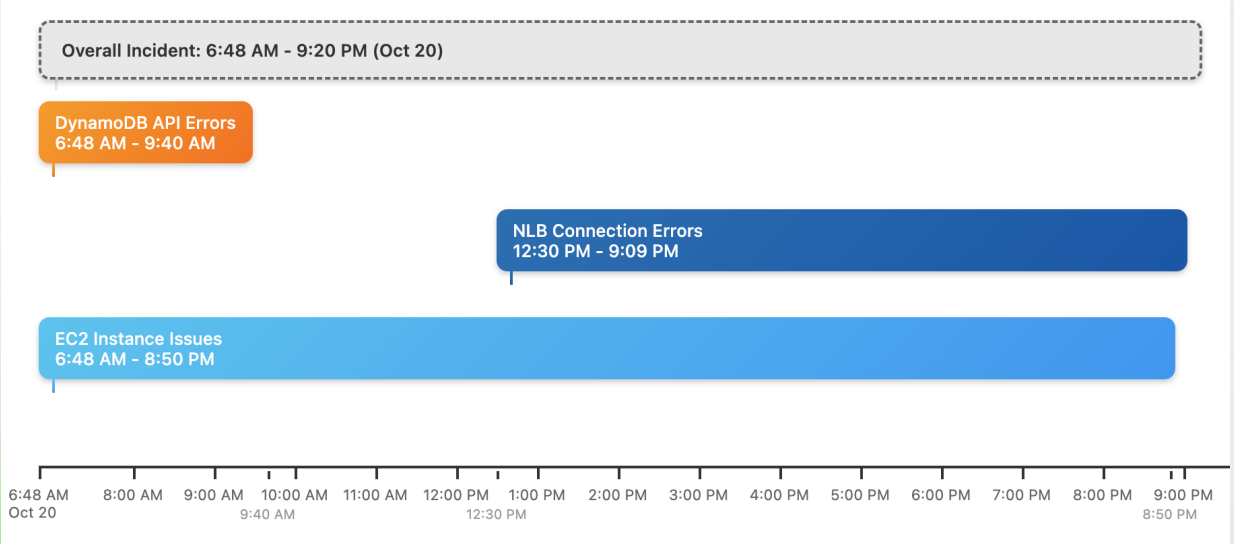




Crisis

#incident-41, a.k.a
The Amazon DynamoDB Service Disruption in the Northern Virginia (US-EAST-1) Region

“The root cause of this issue was a latent race condition in the DynamoDB DNS management system that resulted in an incorrect empty DNS record for the service’s regional endpoint that the automation failed to repair.”



Crisis

“Only a crisis - actual or perceived - produces real change. When that crisis occurs, the actions that are taken depend on the ideas that are lying around.”





Opportunity

The Evolution of SRE at Google
USENIX, Dec 2024

Engineering a Safer World
Systems Thinking Applied to Safety



Nancy G. Leveson, MIT



02

Pre-STAMP SRE practices



02

Pre-STAMP SRE practices

- Root cause analysis (RCA) based postmortems
- Post mortem meetings sometimes held
- Never held premortem meetings
- Action items sometimes completed
- Everyone is working fast

Some RCAs

“Root cause: a few things….”

“Root cause 1 — GPU race … Root cause 2 — Failure classification …”

“Root cause: The task table had too many long-running transactions.”

“Root cause: We were not pinning our `runc` version.” (That’s all I wrote!)

And so on

03

STAMP

est. Oct 2025 at Modal

STAMP as control theory

In order to control a process, four conditions are required:

Goal Condition: The controller must have a goal or goals

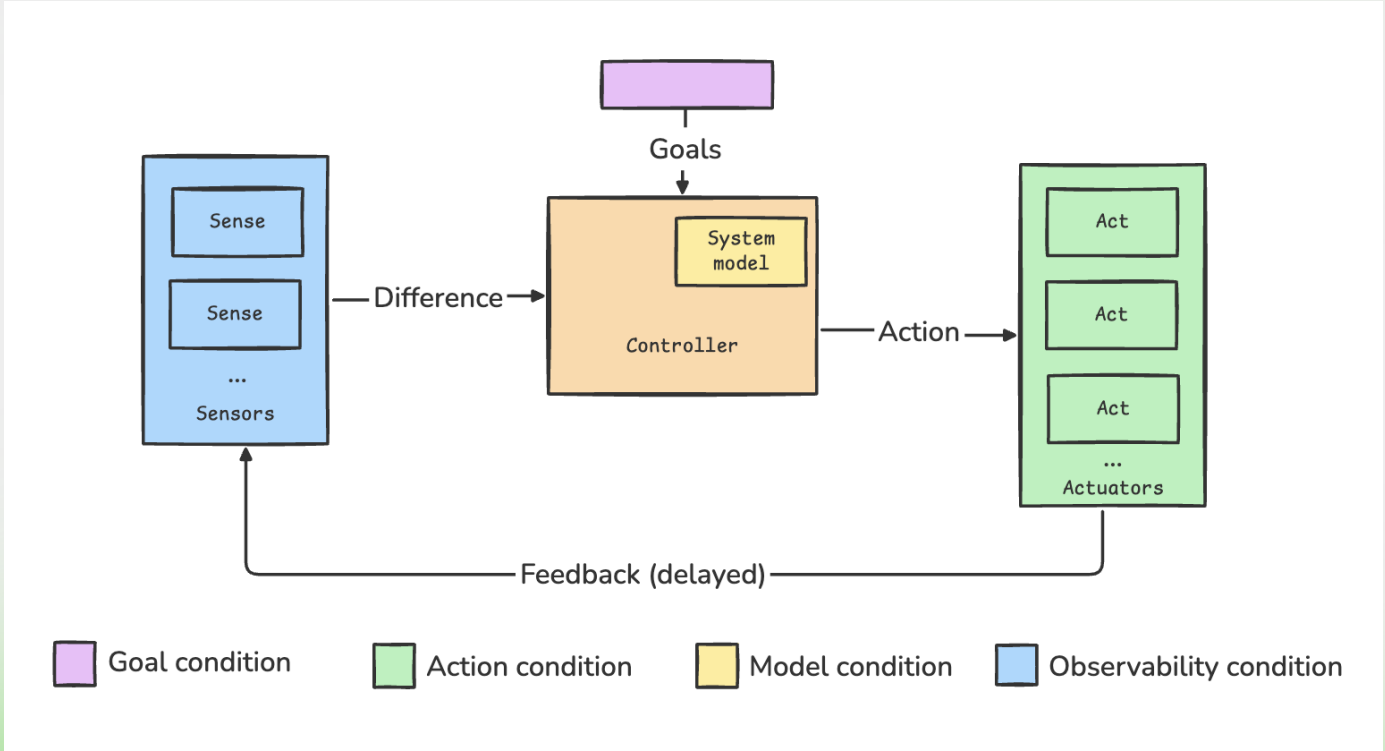
Action Condition: The controller must be able to affect the state of the system. In engineering, actuators implement control actions.

Model Condition: The controller must be (or contain) a model of the system. (c.f. Conant & Ashby's Good Regulator Theorem)

Observability Condition: The controller must be able to ascertain the state of the system.



STAMP as control theory





4 conditions at a startup

Goal Condition:

CEO: “Do what’s most valuable for the company”

Action Condition:

“Ask for forgiveness not permission.”
“Push to prod. Get on a call. Bias for action”
One monorepo, no codeowners.



4 conditions at a startup

Model Condition:

We wrote all the code ourselves, and understood every line of the 100 ~kLoC codebase. We had few customers and could track everyone in Slack.

Observability Condition:

Live in Slack. Sit in a room the size of hotel room. Lunch together everyday.



STAMP framework tools

The four conditions of control are the core mental framework, but STAMP's 6 domain concepts are the actual tools of analysis.

Losses

A loss is an unacceptable outcome for the system

Hazards

A hazard is a system state or condition that, together with worst-case environmental conditions, will lead to a loss.

Safety constraints

A safety constraint is a constraints that must be enforced by control in order to avoid loss.

Control actions

A control action is something produced by an actuator to change the system state.

Unsafe control actions

An *unsafe* control action is one that moves the system towards a loss.

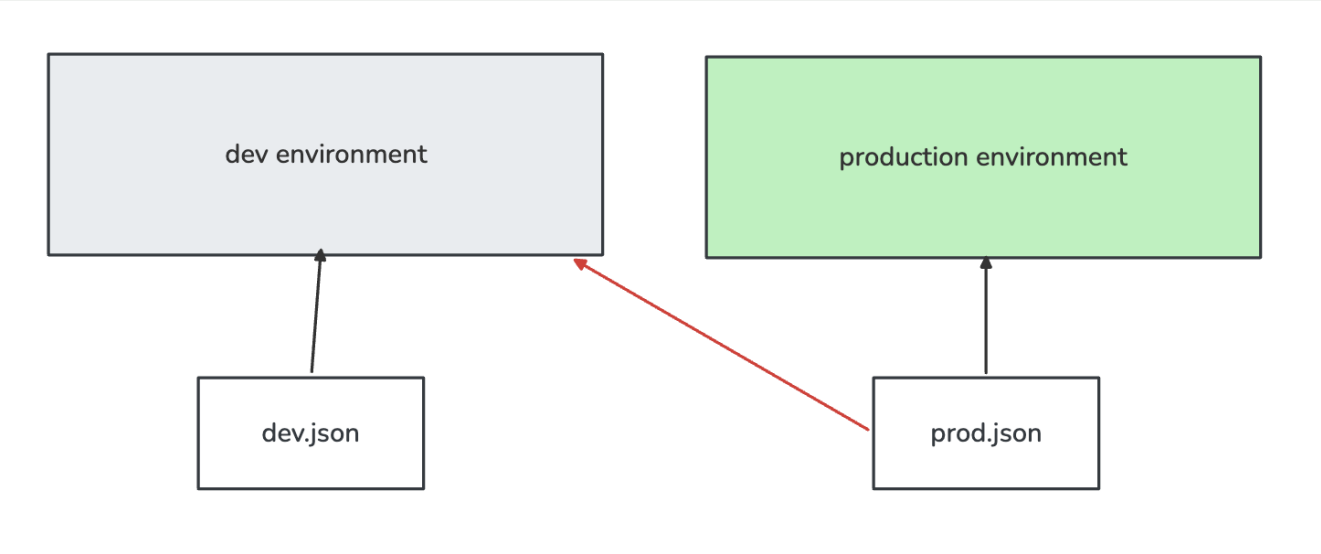
Time

Central and essential in control analysis. Systems thinking is concerned with *dynamics* and *delay*.

STAMP: framework tools example

What happens if you give dev the prod config for 15 minutes?

Incident 2026-01-14: Dev cluster deletes 500+ workers



A **loss** is an **unacceptable outcome** for the system—something stakeholders want to prevent.

A **hazard** is a **system state or condition** that, **together with worst-case environmental conditions**, will lead to a loss.

L1. Production outage / severe degradation due to bad config or wrong secret.

L2. Security incident: secret exposure, secret misuse, or mis-scoped secret access.

L3. Cross-environment contamination: prod uses dev values or dev uses prod secrets.

L4. Undetected misconfiguration: system appears healthy but is running incorrect behavior (silent correctness failure).

L5. Irrecoverable or slow recovery: can't quickly revert to last-known-good config.

H1. Wrong effective config in prod: incorrect merge, missing keys, wrong types/ strings.

H2. Wrong environment selected: prod service reads dev config or wrong S3 object.

H3. Wrong secret resolved: points to wrong secret name/path or wrong AWS account/ region.

…

H9. Config is leaked: security breach gives read access.

04

STruggle

STruggle: branding

Hard to remember, intimidating

STAMP — System-Theoretic Accident Model and Processes

STPA — System-Theoretic Process Analysis

CAST — Causal Analysis based on Systems Theory

Cool and informative

Global Work Scheduler → Borg → **Kubernetes**

Κυβερνήτης → kubernḗtēs (helmsman) → **Kubernetes**

Κυβερνήτης → **cybernetics**

Root cause
Swiss cheese

STruggle: branding

RCA — Root Cause Analysis



SCA — Systems Control Analysis

STruggle: anchor case study

The **Black Hawk friendly-fire shoot down** case study occupies 65 pages, almost double the book space of the next largest case study.



AAI	Air to Air Interrogation (used with IFF)
ACE	Airborne Command Element (the commander's representative in the AWACS)
ACO	Airspace Control Order (Guidance for all local air operations in OPC)
AFB	Air Force Base
AI	Airborne Intercept (a type of radar on fighter aircraft)
ARF	Aircraft Read Files (supplement to the ACO including the ROE)
ASO	Air Surveillance Officer (one of the positions in the AWACS)
ATO	Air Tasking Order (specific mission guidance for the day)
AWACS	Airborne Warning and Control System (a military air traffic control system in the sky)
BH	Black Hawk
BSD	Battle Staff Directive (late scheduling changes not making it into the ATO)
CTF	Combined Task Force
CFAC	Combined Forces Air Component (tactical control of all OPC aircraft operating in TAOR and operational control of AF aircraft)
DO	Director of Operations
GAO	U.S. Government Accountability Office
HQ-II	Have Quick (frequency hopping) radios
HUD	Heads Up Display
IFF	Identiification Friend or Foe
JOIC	Joint Operations and Intelligence Center
JSOC	Joint Special Operations Component (search and rescue operations inside Iraq)
JTIDS	Joint Tactical Information Distribution Center (provides ground with picture of airspace occupants)
MCC	Military Coordination Center (operational control of the Black Hawk helicopters)
MD	Mission Director (runs the mission from the ground)
Min Comm	Minimal Communications
NCA	National Command Authority (the President and the Secretary of Defense)
NFZ	No-Fly Zone
OPC	Operation Provide Comfort (multi-nation effort to protect Kurdish refugees)
ROE	Rules of Engagement (rules governing actions allowed by the U.S. military forces)
SD	Senior Director (one of the positions in the AWACS)
SITREP	Situation Report
SPINS	Mission-related Special Instructions
TACSAT	Tactical Satellite radios (used by Army helicopter pilots to communicate with MCC
TAOR	Tactical Area of Responsibility (another name for the No-Fly Zone)
USCINCEUR	U.S. Commander in Chief, Europe
VID	Visual Identification
WD	Weapons Director (a position in the AWACS)

Figure 5.2
Acronyms used in this chapter.

Downloaded from http://direct.mit.edu/books/oa-monograph-pdf/2280500/book_9780262298247.pdf by guest on 15 September 2025

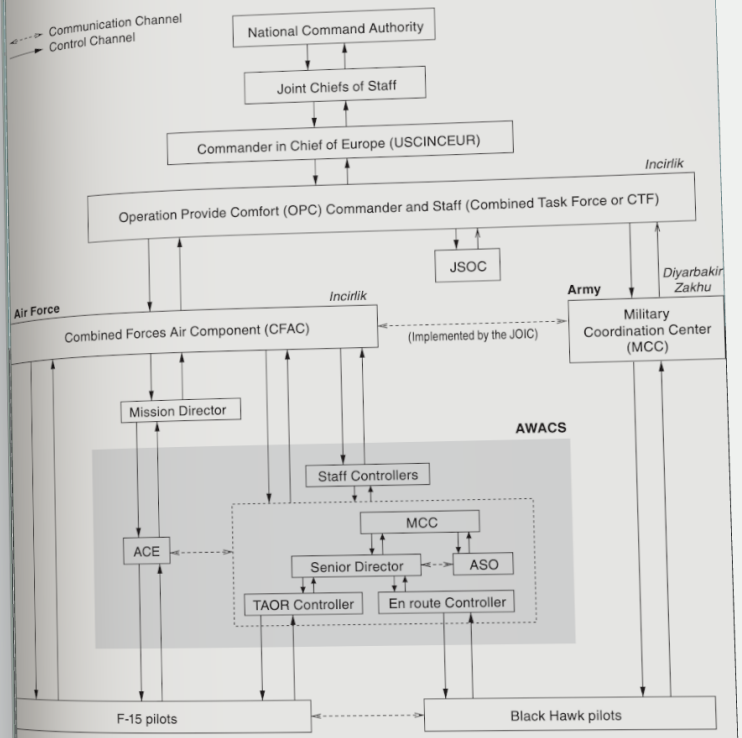


Figure 5.3
Control structure in the Iraqi no-fly zone.

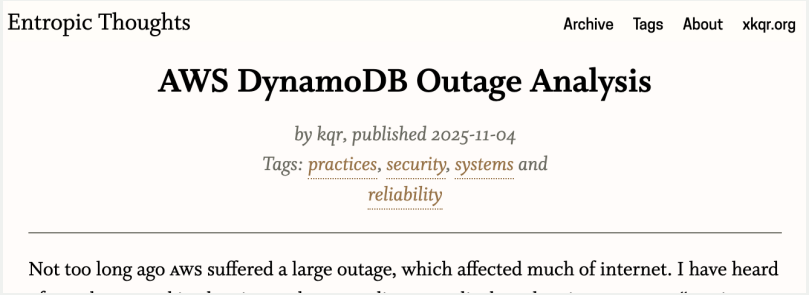
Downloaded from http://direct.mit.edu/books/oa-monograph-pdf/2280500/book_9780262298247.pdf by guest on 15 September 2025

STruggle: incomplete examples

Can you find introductory examples online?

Human (me): I made dozens of searches, read officials docs online, and came away wanting.

Agent (Astra): I did not find a verified match on STPA or CAST that meets your standard: a complete, self-contained analysis of one straightforward software system.



https://entropicthoughts.com/aws-dynamodb-outage-stpa (Nov 2025)



https://www.usenix.org/conference/srecon26americas/presentation/barroso (March 2026)

STAMP

Owner 1	Verification	Tags	Last edited time	Assigned Team	Author
Jonathon Belotti	Empty	Empty	Today 7:02 PM	Empty	Empty

Systems theoretic accident model and process (STAMP)
A fairly bad acronym for a fairly useful software reliability—sorry software *system safety*—framework. If you want background on why this is needed and cool, see Reliability reading list - USENIX The Evolution of SRE at Google (December, 2024).

STAMP Incident analysis

- Jonathon Belotti This is an example of analyzing the incident-148 accident (m...
  2025-12-28: Postgres reaches XID wraparound limit
- 2026-01-14 postmortem: Worker resource leak leads to runc networking outage - STAMP Analysis
- 2026-05-07: AWS us-east-1c overheats - Control failure analysis
- 2026-06-01: Merge queue lag over 1.5hrs - Control failure analysis

STPA

Systems Theoretic Process Analysis (STPA) is a analysis tool that is part of the *Systems Theoretic Accident Model and Processes (STAMP)* approach to software reliability.

Why use these?

For starters, STPA is an arm of STAMP, a Google SRE-endorsed, control-theory based approach to software safety out of MIT. It's likely a way of approaching reliability you haven't tried before,

Systems Theoretic Process Analysis (STPA) is a analysis tool that is part of the *Systems Theoretic Accident Model and Processes (STAMP)* approach to software reliability.

Why use these?

For starters, STPA is an arm of STAMP, a Google SRE-endorsed, control-theory based approach to software safety out of MIT. It's likely a way of approaching reliability you haven't tried before, and on that alone it's interesting.

More specifically, I think creating ranked lists of "losses" and "hazards" is a straightforwardly pragmatic and simple method of discovering reliability problems. If you take nothing away else from STAMP, it should be that you can comprehensively analyze a system's reliability (here called *safety*) by listing all the ways it could cause the business harm (a *loss*) and all the ways the system becomes vulnerable to experiencing harm (a *hazard*).

Finally, and this is speculative, but I think STPA documents may serve as effective tools for agents to analyze our codebase for things that need fixing. STPA documents, and their lists of losses, hazards, and unsafe control actions may prove specification at an appropriate level of abstraction for LLMs. As Marc Brooker recently said, "specification is the future of programming."

Press 'space' for AI or '/' for commands

Analyses

- STPA Hazard analysis: Automated sliding door [Example]
- STPA Hazard analysis: modal-determinator
- STPA Hazard analysis: Modal's config system (Cadre)
- STPA Analysis: Machine Manager
- STPA Analysis: Blobnet
- No more 'Root cause' section
- STPA Analysis: IR-696 (unverified page-copy completion)

05

STAMP adoption at Modal

Thinking in bets, Thinking in systems

Study of failure is niche.

Systems Thinking is niche.

***How Complex Systems Fail* is niche**

Control theory is niche.

PETER NAUR, PROGRAMMING AS THEORY BUILDING

Peter Naur, widely known as one of the authors of the programming language syntax notation "Backus-Naur Form" (BNF), wrote "Programming as Theory Building" in 1985. It was reprinted in his collection of works, *Computing: A Human Activity* (Naur 1992).

This article is, to my mind, the most accurate account of what goes on in designing and coding a program. I refer to it regularly when discussing how much documentation to create, how to pass along tacit knowledge, and the value of the XP's metaphor-setting exercise. It also provides a way to examine a methodology's economic structure.

In the article, which follows, note that the quality of the designing programmer's work is related to the quality of the match between his theory of the problem and his theory of the solution. Note that the quality of a later programm

is related to th

ries and the p

ries.

Using Na

is not to pa

pass along

design. Th

more app

knowled

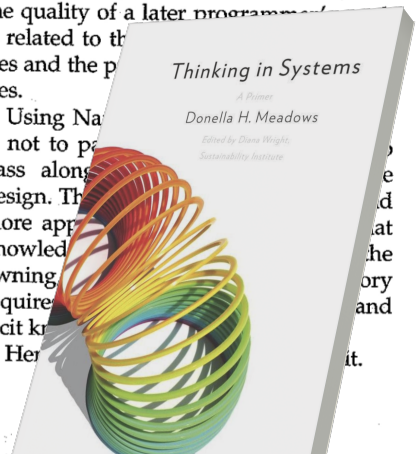
owning

require

tacit kn

Her

it.



"PROGRAMMING AS THEORY BUILDING"

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 accommodating such observations in a production view of programming suggests that this view is m ding. The theory building view i s an alternative.

A more general bac re-
sentation is a convictio t
to have an appropria of
what programming is. d-
ing is inappropriate we w d
the difficulties that arise in
our attempts to overcom
rise to conflicts and

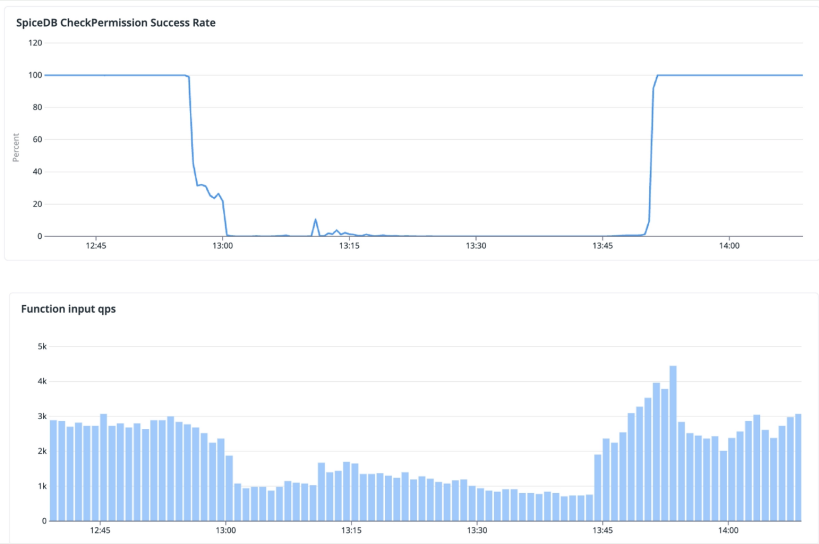




RCA -> SCA: An authz service SEV-1

“All users were unable to use any Modal service due to total failure of our authorization subsystem, SpiceDB.

All Modal services were completely down because nearly every internal RPC failed due to all requests to SpiceDB failing. The dashboard also returned an error for every web request during this time.”



RCA -> SCA: An authz service SEV-1

> At 12:56:10 P50 request latency for CheckPermission began to climb rapidly. It spiked 1000x from 2ms to 2s in 20 seconds. By 12:56:30 it was flatlined at 2s, the gRPC timeout set by our client.
>
> This means at least 50% of requests began timing out nearly instantly.

Original RCA

RCA -> SCA

Zooming out and defining the set of losses, we discover that losses are in *tension* with each other. Avoiding one creates the other.

The two hazard separate the space of losses.

New focus: fail open vs closed mechanism.

The role based authorization control (RBAC) system at Modal exists to prevent these losses.

L1. Unauthorized access by actors *outside* a tenancy (aka. Workspace)

L2. Unauthorized access by actors inside a tenancy (aka. Workspace).

L3. Blocked access to *authorized* actors.

L4. Unavailability of Modal services.

During this incident we experienced L3 and L4 initially and in remediating those we experienced L2 and *potentially* L1. For a brief window we used an ultra permissive, essentially no-op authorization check which effectively disabled authorization and left only authentication.

> A **hazard** is a **system state or condition** that, **together with worst-case environmental conditions**, will lead to a loss.
>
> A hazard is *not the loss itself*, and not an accident—it is a precondition.

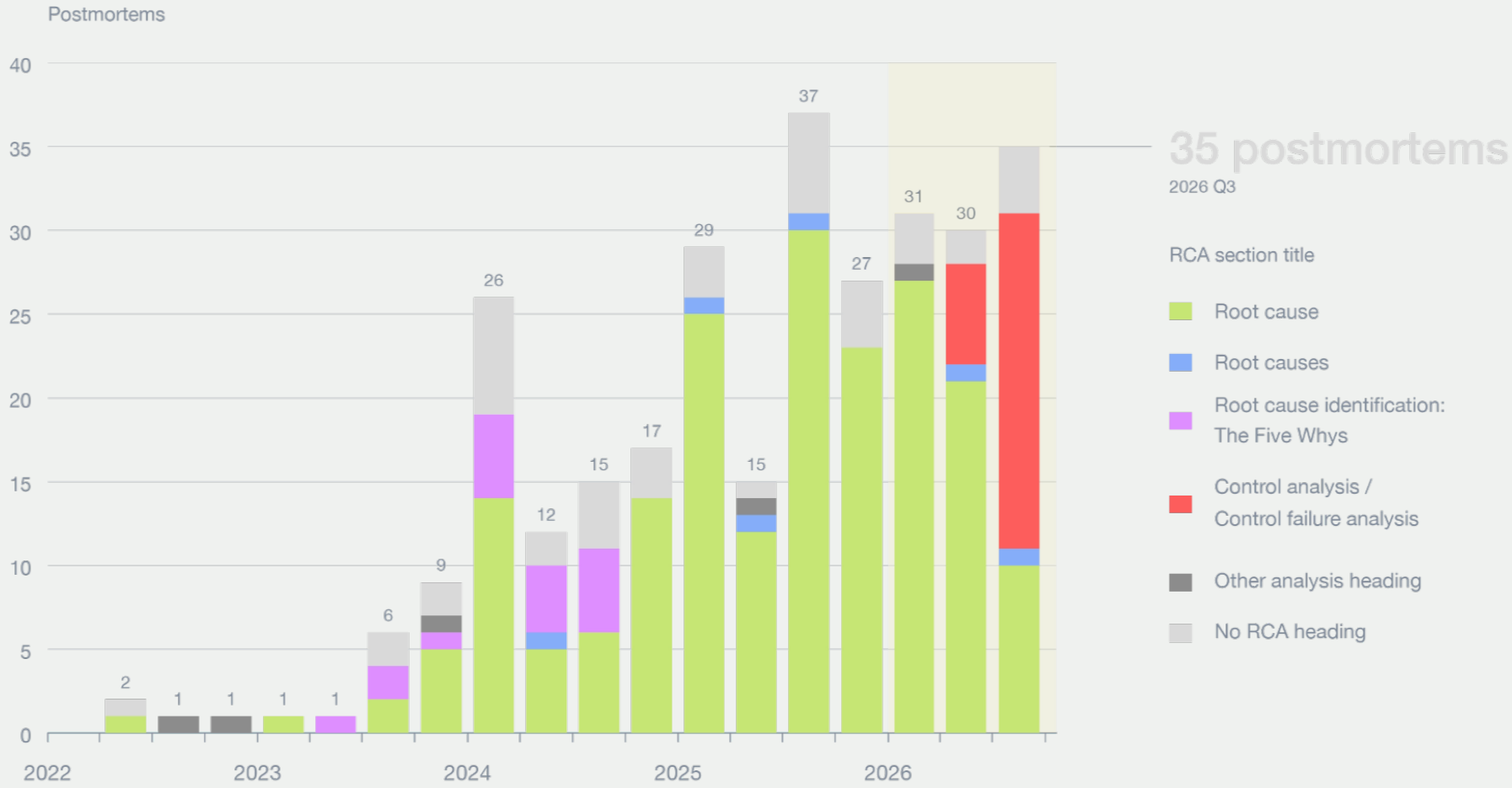
The following hazards combined to produce the losses experienced.

H1. The authorization service becomes unavailable, returning timeouts or errors (leads to L3, L4)

H2. The authorization service incorrectly authorizes requests (leads to L1, L2)

Other hazards exists, for example that the authorization service was absent from parts of our API surface, but they are not relevant to this incident.

It is difficult for this authorization system to 'fail safe'. It can either fail-closed, which prevents L1 and L2 but creates L3, L4. Or it can fail-open, and create at least L2.


Postmortems
40
35
30
25
20
15
10
5
0
2022
2023
2024
2025
2026
2
1
1
1
1
6
9
26
12
15
17
29
15
37
27
31
30
35 postmortems
2026 Q3
RCA section title
Root cause
Root causes
Root cause identification: The Five Whys
Control analysis / Control failure analysis
Other analysis heading
No RCA heading

Putting the ‘socio’ in socio-technical

The **Black Hawk friendly-fire shoot down** case study highlights the importance of sociological dynamics and human communication.

In startups, these dynamics are likely **more important** than any system flaw.

“Human practitioners are the adaptable element of complex systems.”

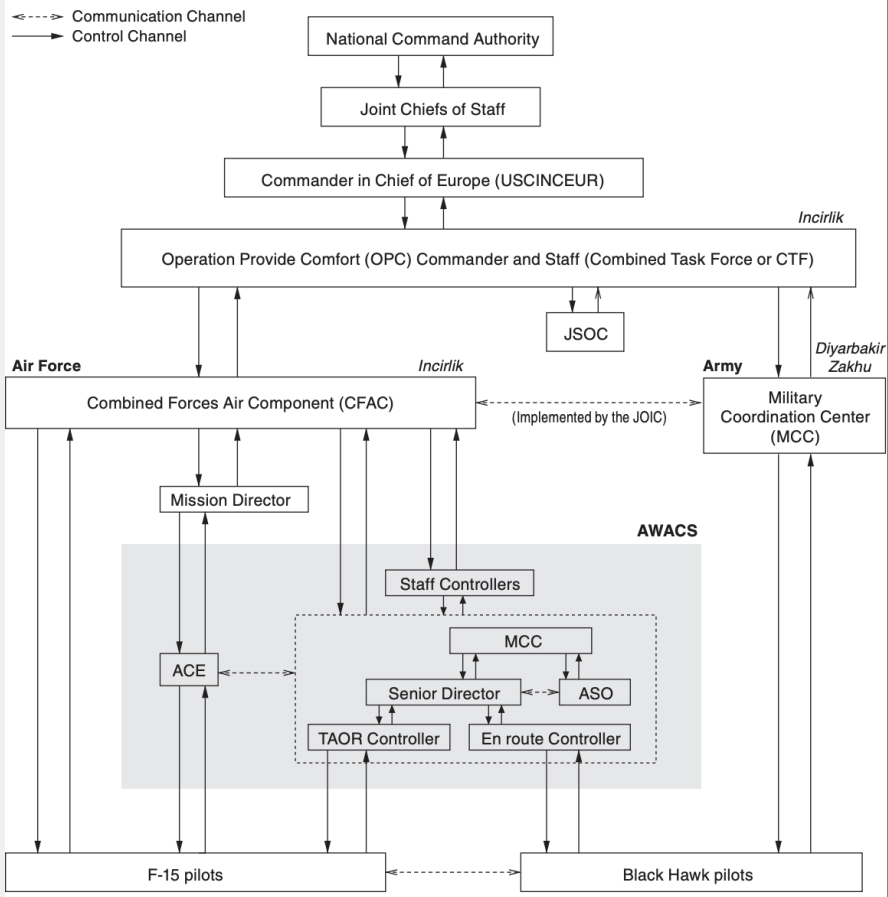
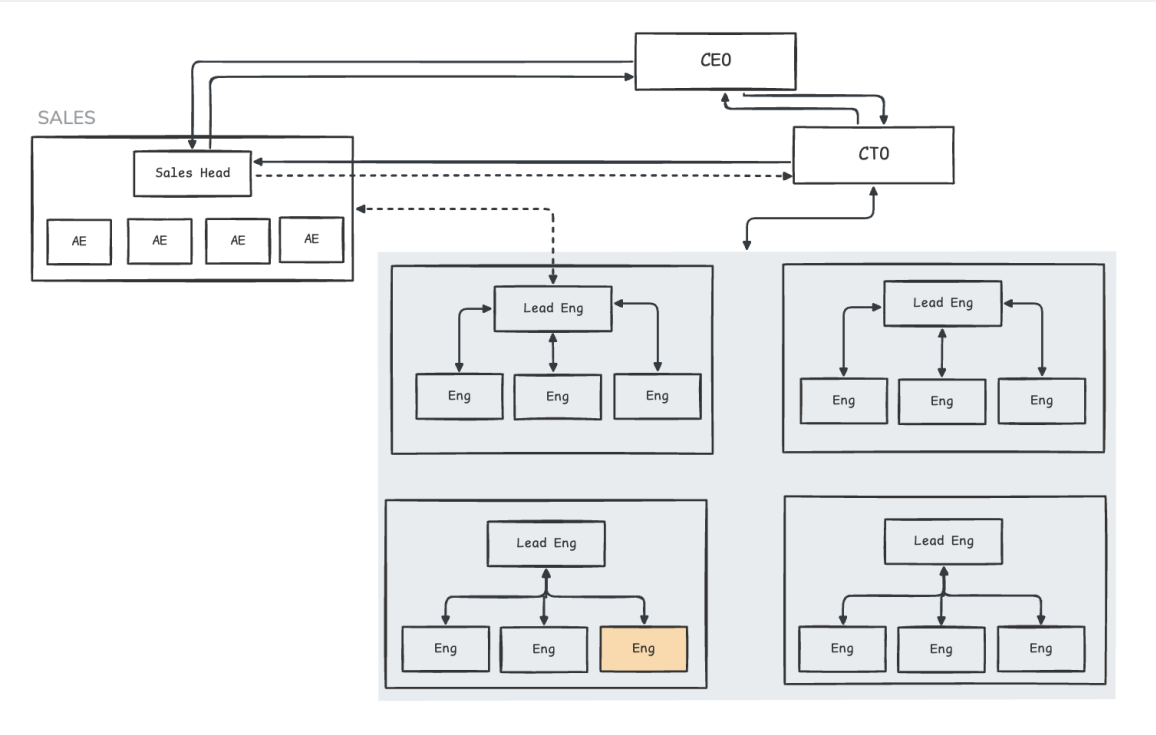


Figure 5.3
Control structure in the Iraqi no-fly zone.

Putting the ‘socio’ in socio-technical

Our organization structure was failing to maintain safety/reliability.

Understaffed, isolated mini-SRE division.



Putting the ‘socio’ in socio-technical

Goal Condition: The controller must have a goal or goals

Action Condition: The controller must be able to affect the state of the system.

Model Condition: The controller must be (or contain) a model of the system.

Observability Condition: controller must be able to see the state of the system.

Goal Condition: The controller must have a goal or goals

Action Condition: The controller must be able to affect the state of the system.

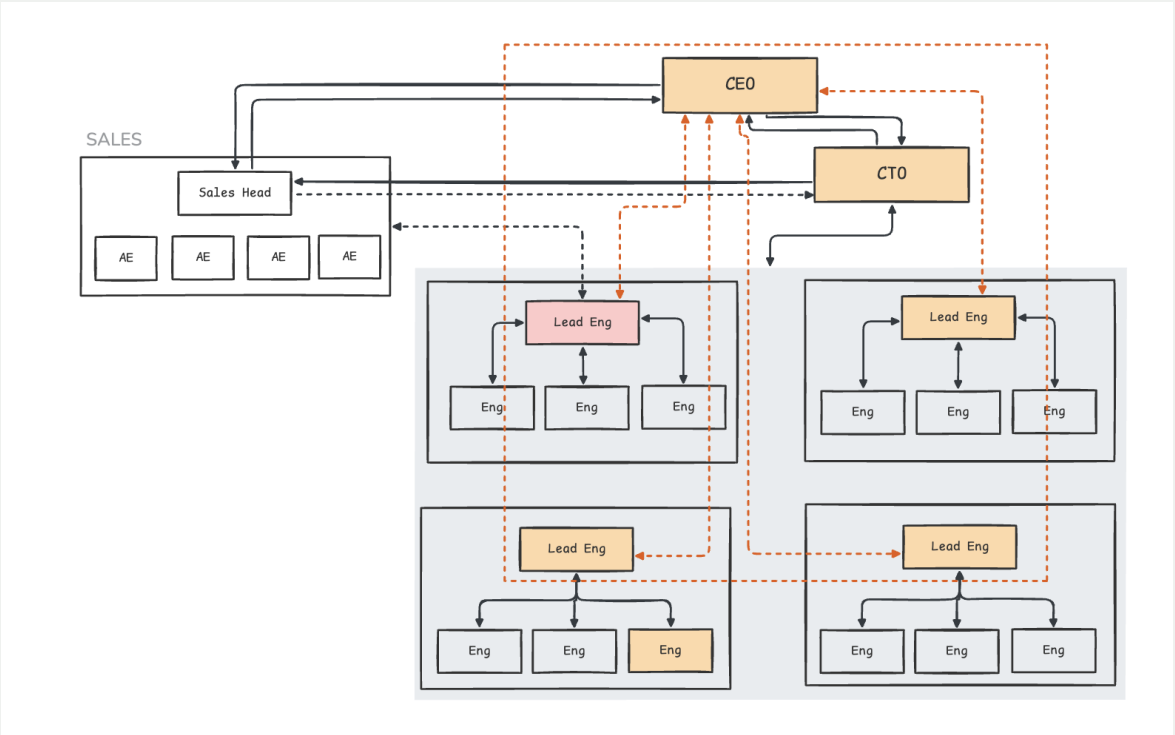
Model Condition: The controller must be (or contain) a model of the system.

Observability Condition: controller must be able to see the state of the system.

Putting the ‘socio’ in socio-technical

Unify all concerns of the business inside core of leads (human controllers).

Sales, customers, engineering, operations, SRE.



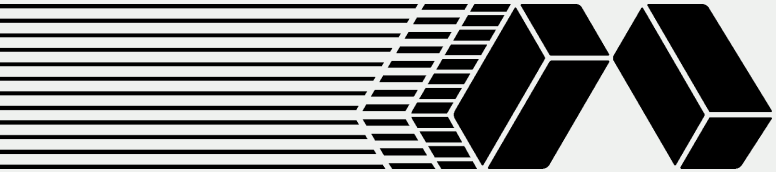


End: Looking ahead

A few questions I have, looking forward into 2027:

- If you like STAMP, should you start studying Cybernetics?
- Does STAMP quickly become more valuable in the agent era?
- Can STAMP be simplified? 80/20 rule.
- Can STAMP analysis be partially automated?

We're hiring!
modal.com/company#careers



Thank You!

Thanks to Tim Falzone, Johan Andersen, and Colin Weld for review.