

Making your Automation a Better Team Player

Laura Nolan, SREcon EMEA 2023

O'REILLY



Edited by Betsy Beyer, Chris Jones,
Jennifer Petoff & Niall Murphy



Seeking
SRE

CONVERSATIONS ABOUT RUNNING PRODUCTION SYSTEMS AT SCALE

Curated and edited by
David N. Blank-Edelman



usenix
THE ADVANCED
COMPUTING SYSTEMS
ASSOCIATION

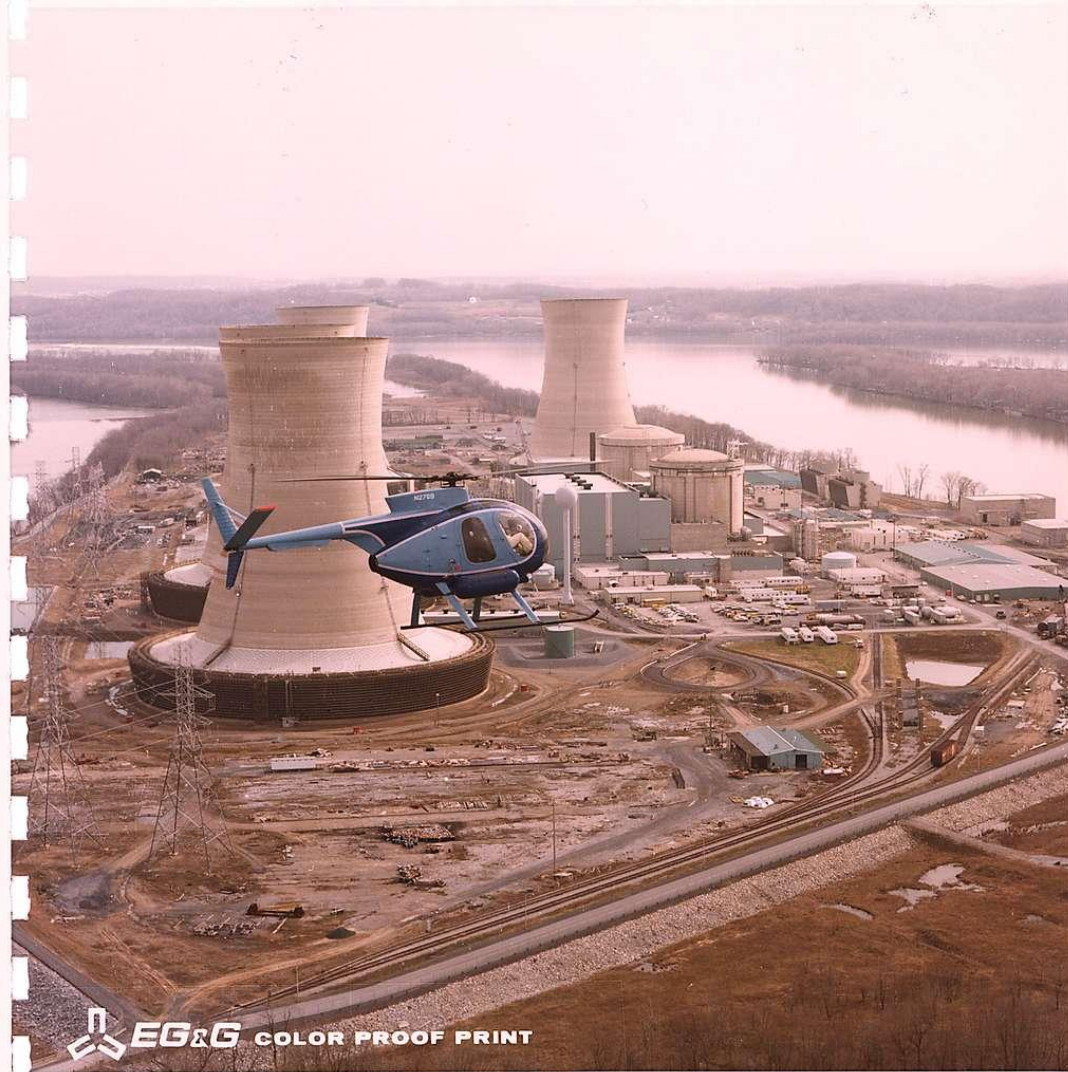


<CODE>YOUR
FUTURE
CAMPAIGN TO STOP
KILLER ROBOTS



LUNDS
UNIVERSITET





COLUMBIA

ACCIDENT INVESTIGATION BOARD



BACK
TO
THE 80'S

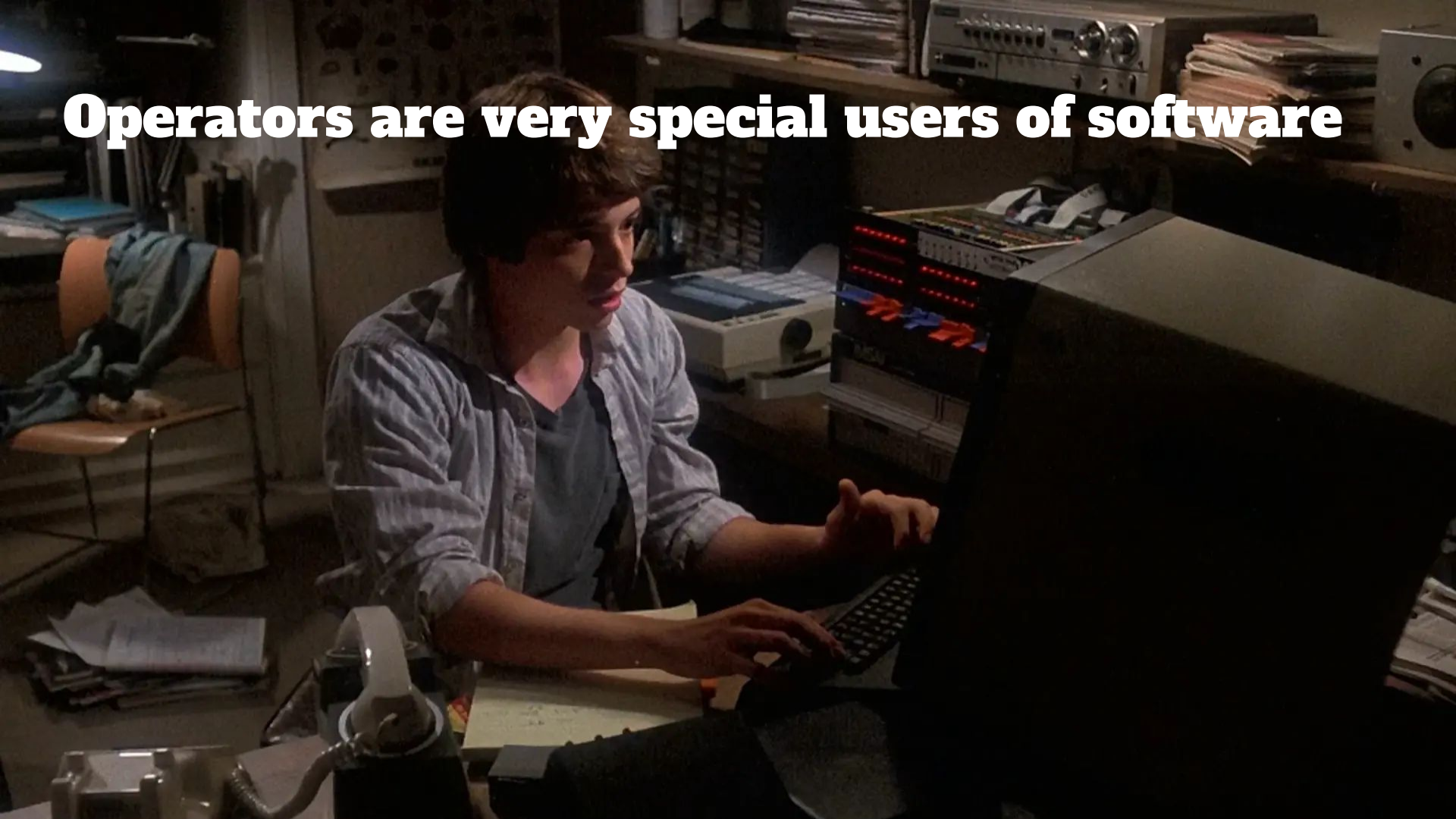
The logo features the text "BACK TO THE 80'S" in a bold, bubbly, multi-colored font with a rainbow gradient from yellow to purple. The word "BACK" is on the top line, "TO" is on the second line, and "THE 80'S" is on the third line. The text is set against a purple star shape that has a black shadow. The letter "K" in "BACK" is replaced by a cassette tape icon, which is also rendered in the same multi-colored gradient style.

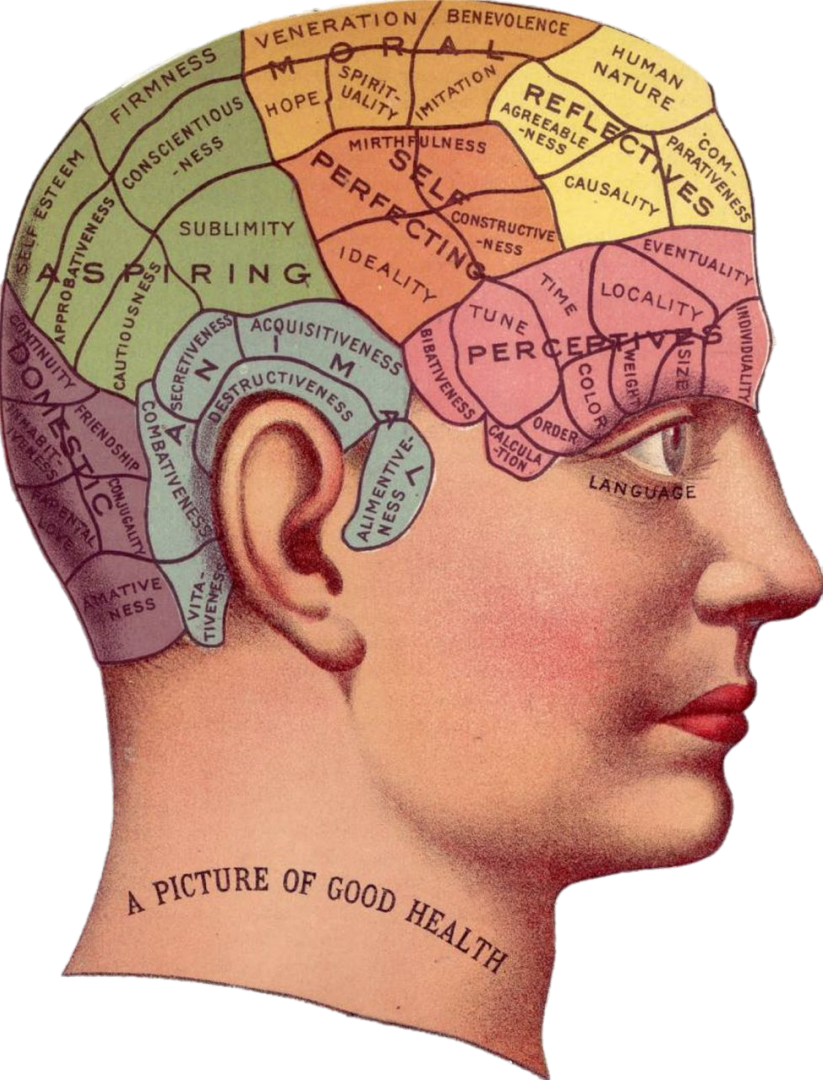






Operators are very special users of software





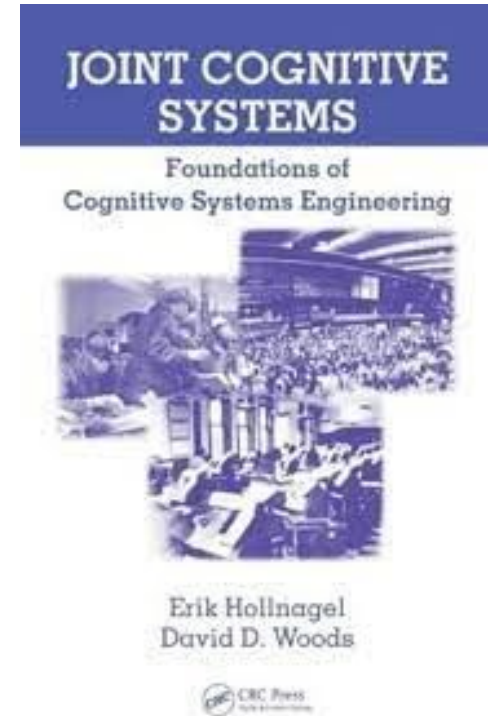
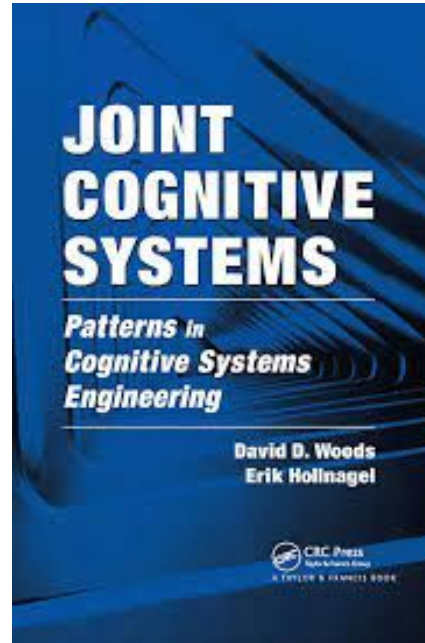
Cognitive Work

- Understanding
- Planning
- Problem solving
- Analysing and Synthesising
- Judging

Joint Cognitive Systems

Systems where computers and humans collaborate on cognitive work.

JCS is all about thinking about cognitive systems as a whole - not as discrete parts done by humans or by computers.



MABA-MABA or Abracadabra?

Progress on human-automation coordination

Abstract

In this paper we argue that substitution-based function allocation methods (such as MABA-MABA, or Men-Are-Better-At/Machines-Are-Better-At lists) cannot provide progress on human-automation coordination. Quantitative “who does what” allocation does not work because the real effects of automation are qualitative: it transforms human practice and forces people to adapt their skills and routines. Rather than re-inventing or refining substitution-based methods, we propose that the more pressing question on human-automation coordination is “how do we make them get along together”.



Sidney W A Dekker

Griffith University

77 PUBLICATIONS 4,025 CITATIONS

SEE PROFILE

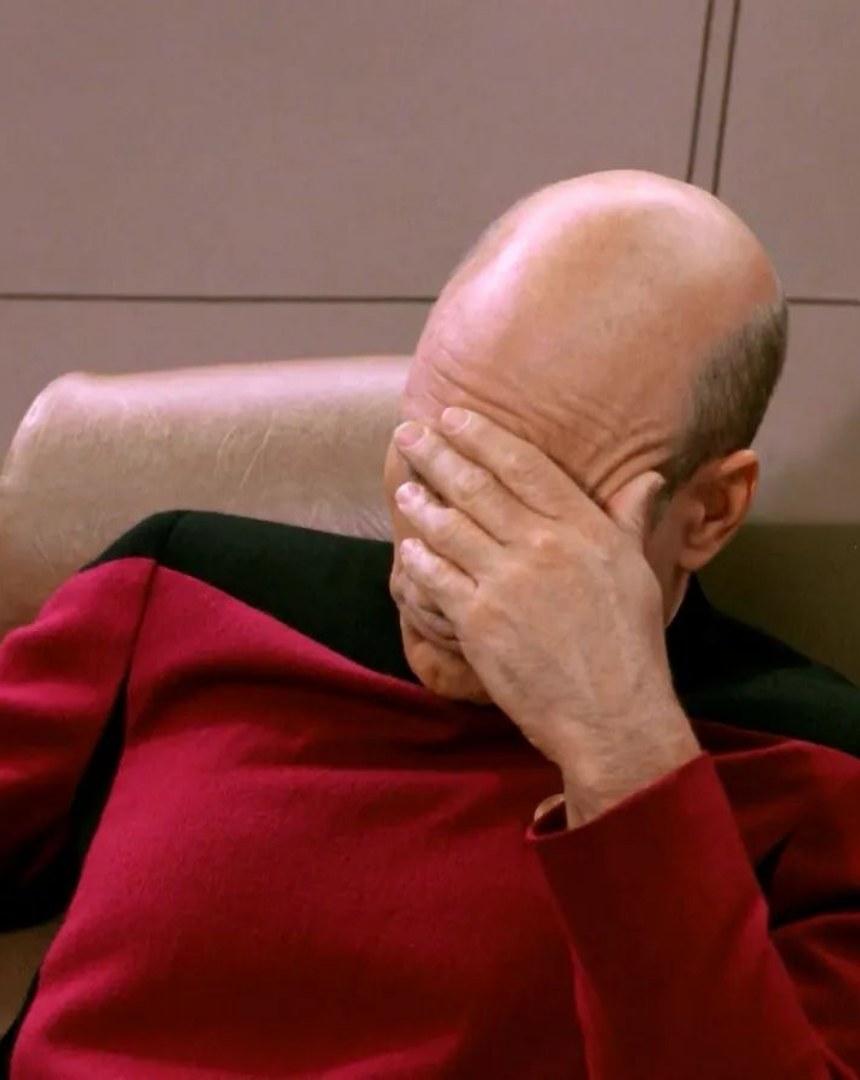


David D Woods

The Ohio State University

433 PUBLICATIONS 27,080 CITATIONS

SEE PROFILE



“I think the key thing is that you’re enabling people to do what people are really, really good at, and you’re enabling machines to do what machines are really, really good at.”

– Trae Stephens, Anduril

“Robots will be able to do everything better than us.”

– Elon Musk



Brief Paper

Ironies of Automation*

LISANNE BAINBRIDGE†

Key Words—Control engineering computer applications; man-machine systems; on-line operation; process control; system failure and recovery.

Abstract—This paper discusses the ways in which automation of industrial processes may expand rather than eliminate problems with the human operator. Some comments will be made on methods of alleviating these problems within the 'classic' approach of leaving the operator with responsibility for abnormal conditions, and on the potential for continued use of the human operator for on-line decision-making within human-computer collaboration.

designer errors can be a major source of operating problems. Unfortunately people who have collected data on this are reluctant to publish them, as the actual figures are difficult to interpret. (Some types of error may be reported more readily than others, and there may be disagreement about their origin.) The second irony is that the designer who tries to eliminate the operator still leaves the operator to do the tasks which the designer cannot think how to automate. It is this approach which causes the problems to be discussed here, as it means that the

Some examples of JCSes in Software

- Monitoring and alerting - Prometheus, Grafana, DataDog, etc
 - And your particular configurations
- CI/CD systems: Jenkins, ArgoCD, etc
- Config management: Chef, Terraform, Kubernetes operators, etc
- Orchestration: Kubernetes, autoscaling groups, systemd, etc
- Consoles and status pages for software infrastructure
- Runbooks and other kinds of operator documentation
- Logs and metrics that your systems emit

JOINT COGNITIVE SYSTEMS



EVERYWHERE

Are we doing a good job?



Lily Cohen 

@lily

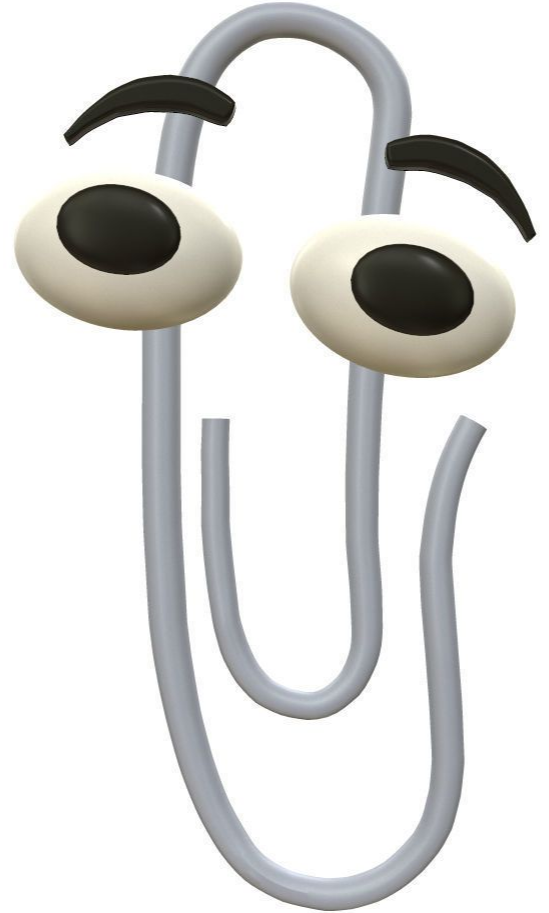
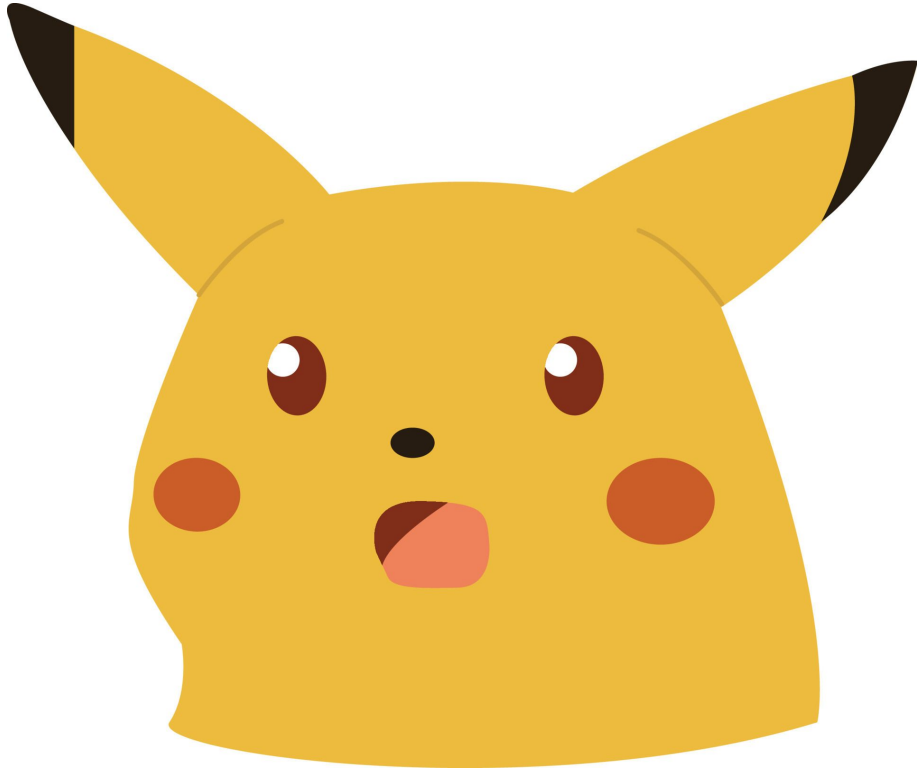
7d ago

Bad news everyone. It is with immense regret that I write to inform you we have suffered a total loss of data for `firefish.lgbt`, `musician.social`, and `outdoors.lgbt`.

How did we get here?

During a routine `#GitOps` repository cleanup a subdirectory containing yml manifests that create our namespaces was moved to a directory not visible by ArgoCD. From Argo's perspective, the directory and yml manifests no longer existed so it went to do its job and clean things up. Had the directory just contained the manifests for the Helm deployments, this would have been okay as the Persistent Volume Claims would have persisted, but deleting a namespace deletes everything it contains.

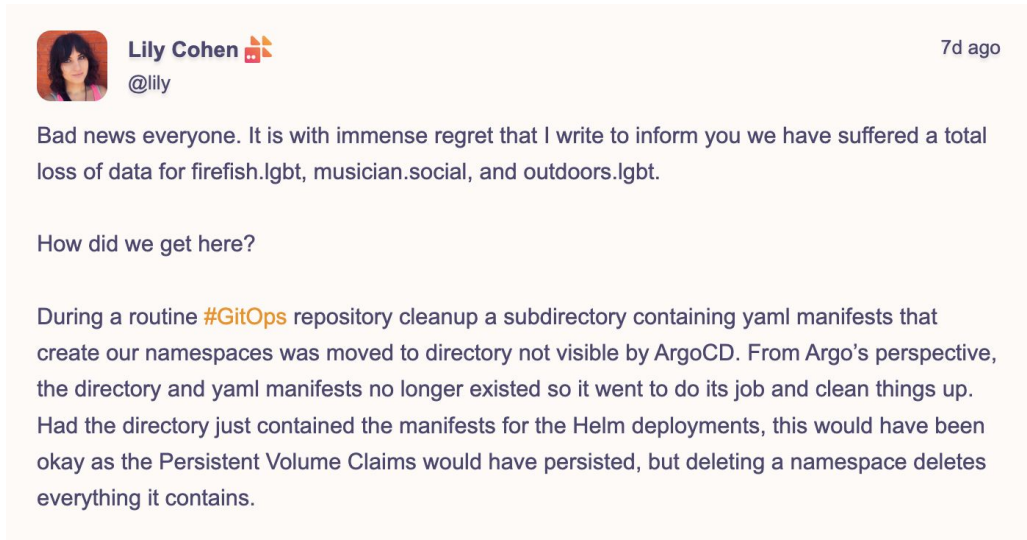
Two Antipatterns to Avoid







Automation Surprise

“Why did it do that!”



A screenshot of a tweet from Lily Cohen (@lily) posted 7 days ago. The tweet discusses a data loss incident caused by an automation error during a repository cleanup.

 **Lily Cohen** 
@lily 7d ago

Bad news everyone. It is with immense regret that I write to inform you we have suffered a total loss of data for `firefish.lgbt`, `musician.social`, and `outdoors.lgbt`.

How did we get here?

During a routine [#GitOps](#) repository cleanup a subdirectory containing yml manifests that create our namespaces was moved to directory not visible by ArgoCD. From Argo's perspective, the directory and yml manifests no longer existed so it went to do its job and clean things up. Had the directory just contained the manifests for the Helm deployments, this would have been okay as the Persistent Volume Claims would have persisted, but deleting a namespace deletes everything it contains.



Two leading causes of automation surprise

- Autonomous processes that don't behave as intended
 - Push a bug in a reported metric, autoscale your systems into the ground automatically!
- Human-triggered processes that do things the user didn't want
 - Example: taking down your systems with a terraform apply run by hand or a CD tool

Avoid scattered autonomous processing

- Autonomous automation scattered through lots of scripts and hooks and crons is very hard to manage
- Out of sight, out of mind is bad for autonomous tooling
 - Running as a full-fledged service
 - Status pages and alerting
- Behaviour should be predictable and as simple as possible
 - Avoid modes and other complexity





Example: Unattended upgrades

- Often enabled by default, and run close to the same time on all hosts
- No central coordination or validation
- Silent, but deadly - zero visibility to operators
- Occasionally breaks things badly
 - for example, [Datadog, March 2023](#)





Clearly display intended actions

- Status displays or uncluttered logs that show past actions and intended/scheduled future actions
 - Full details of execution steps - in order - with configuration values and all parameters
 - Estimated resource usage (e.g. API calls, quota, IP addresses) versus available resources
 - Other relevant context or warnings
- Operators should be able to suspend, cancel, or immediately trigger automated actions



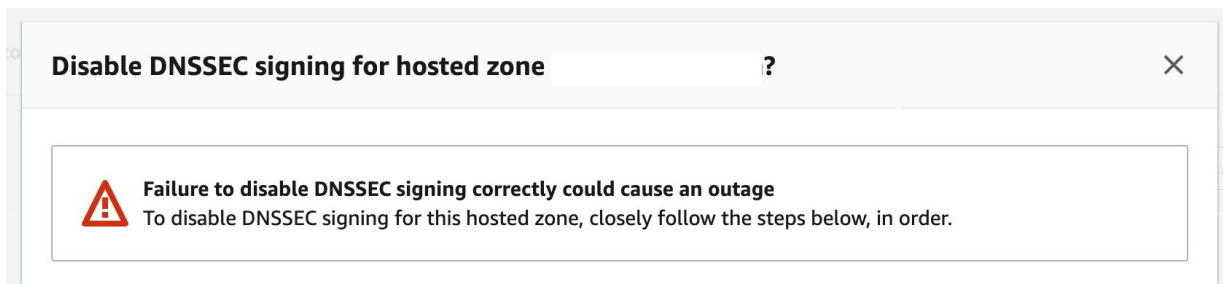
```
237 # module.secret-manager.aws_secretsmanager_secret_version.[redacted] ["json"] will be destroyed
238 - resource "aws_secretsmanager_secret_version" "sm-[redacted]" {
239     - arn          = "arn:aws:secretsmanager:us-east-1:[redacted]:secret:[redacted]" -> null
240     - id          = "[redacted]" -> null
241     - secret_id   = "[redacted]" -> null
242     - secret_string = (sensitive value)
243     - version_id  = "[redacted]" -> null
244     - version_stages = [
245         - "AWSCURRENT",
246     ] -> null
247 }
248 # module.secret-manager.aws_secretsmanager_secret_version.[redacted] will be created
249 + resource "aws_secretsmanager_secret_version" "sm-[redacted]" {
250     + arn          = (known after apply)
251     + id          = (known after apply)
252     + secret_id   = "[redacted]"
253     + secret_string = (sensitive value)
254     + version_id  = (known after apply)
255     + version_stages = (known after apply)
256 }
257 Plan: 2 to add, 0 to change, 4 to destroy.
```

Example: Terraform plan



Example: TF hides warnings

*“All our DNS configuration is managed in [Terraform](#). Terraform is great for managing infrastructure as code, but in this case it made us miss a **critical** warning when trying to disable DNSSEC signing in Route53.”*



Rafael Elvira, Slack

<https://slack.engineering/what-happened-during-slacks-dnssec-rollout/>



Example: Kube pod priorities

“The config for the new Cortex cluster did not include the new Pod Priorities, so all the new Pods were given the default priority, medium. There were not enough resources on the Kubernetes cluster to fit the new Cortex cluster, and the existing production Cortex cluster had not been updated to include the high priority designation for their Ingesters. As the new cluster’s Ingesters had medium priority (the default) and the existing production Pods had no priority, the new cluster’s Ingesters preempted an Ingestor from the existing production Cortex cluster.

.... This triggered a cascading failure that eventually caused the preemption of all the Ingestor Pods for the production Cortex clusters.”

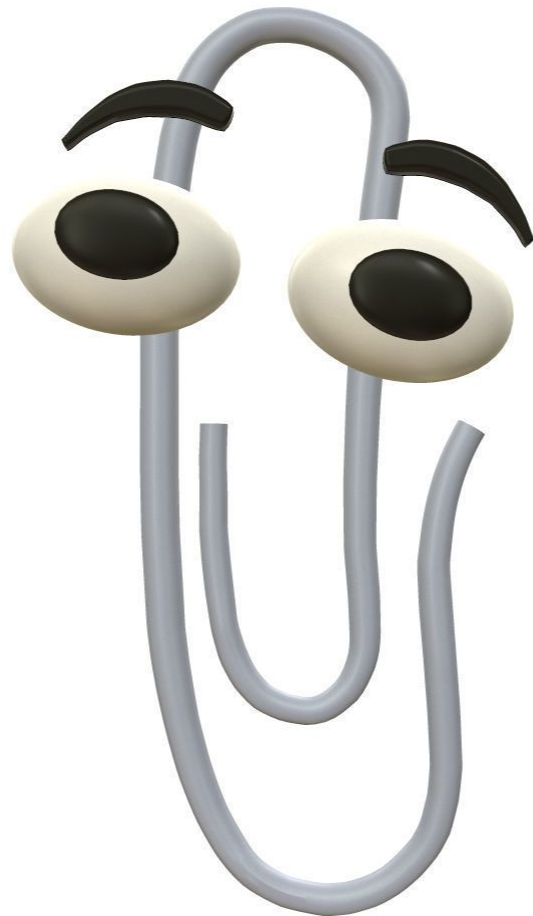
Tom Wilkie, Grafana Labs

<https://grafana.com/blog/2019/07/24/how-a-production-outage-was-caused-using-kubernetes-pod-priorities/>

It looks like you're trying to fix prod - can I roll back the last release for you?

Research shows that making specific recommendations tends to reduce the ability of humans to generate alternative hypotheses.

In troubleshooting, it's important not to get too attached to any hypothesis, but to keep an open mind until the failure mechanism is clear.



Be careful with suggestions and recommendations

- Lots of automated root-cause detection software systems - AIOps hype cycle continues
- Can be useful, but can also be wrong
- This also applies to lower-tech systems - like runbooks!
 - Say 'This has been caused by XYZ in the past and here is how to check' rather than 'This alert is caused by XYZ'
- Related: reflexively blaming the network and not investigating further!





Providing ways to understand system behaviour is often more useful than targeted suggestions

“Participants formed, tested, and abandoned multiple hypotheses during their exploration of the anomaly and search for its sources. This work was quite fast and efficient; participants were quick to seek and use information, especially in the early stages of the response when the nature, extent, and severity of the anomaly was unknown. The earliest activities, however, did not appear to be hypothesis-driven but instead focused on hypothesis generation (Woods and Hollnagel 2006). These efforts were sweeping looks across the environment looking for cues.”

David Woods, STELLA report

<https://snafucatchers.github.io/>



todd underwood 19 hours ago

another one: “we have a lot of internal documentation about troubleshooting and fixing different issues. do you think LLM is the right why to make some smart/gpt-like bot that is trained on this data ?

would this be a valid use-case to train data yourself ?”



todd underwood 19 hours ago

i think that applications like this are one of the most likely applications



Laura Nolan 19 hours ago

I really worry about that kind of usecase, are we going to be here in 2 years talking about the horrible incident because someone ran a command that a LLM told them to run and it made a giant mess

**SRE
CON**[®] —

EUROPE
MIDDLE EAST
AFRICA

2025



Who will eat humble pie?

or



There are two types of tool (in JCS theory)



Prosthesis Example: Kafka Client

“By providing a ready-to-go Kafka client, we ensured teams got up and running quickly, but we also abstracted some core concepts of Kafka a little too much, meaning that small unassuming configuration changes could have a big impact.”

One such example led to partition skew (a large portion of messages being directed towards a single partition, meaning we were not processing messages in real time.)”

Matt Boyle, Cloudflare

<https://blog.cloudflare.com/using-apache-kafka-to-process-1-trillion-messages/>





Building Amplifiers

Don't try to hide complexity: actively try to help operators build a useful mental model of the system

- Summarise to avoid overwhelming operators
 - But avoid strongly suggesting causes for problems
- Provide ability to drilldown and explore all the details (including exact times)
- Try to expose how things really work under the hood
- Explain why all automated actions are taken
- Make constraints and limitations clear



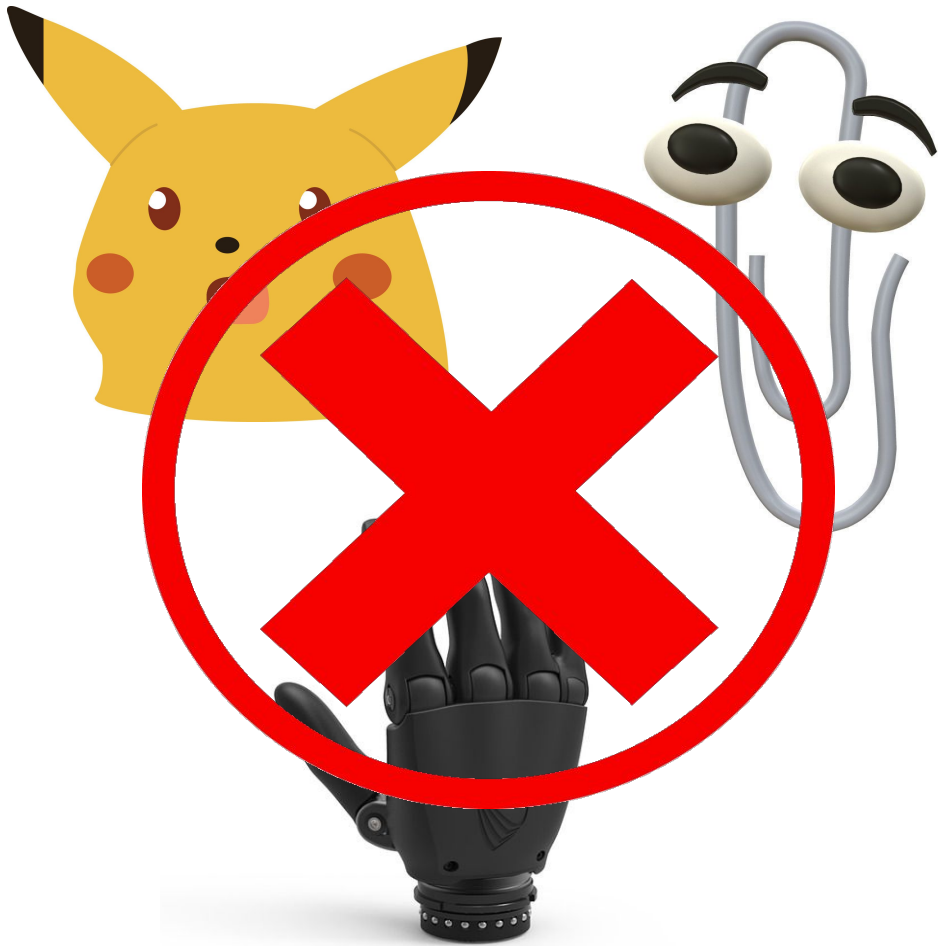
**Use this ONE WEIRD trick when
working on a JCS...**

[CLICK HERE]

**In order to take your SREing to the
NEXT LEVEL!**

[CLICK HERE]

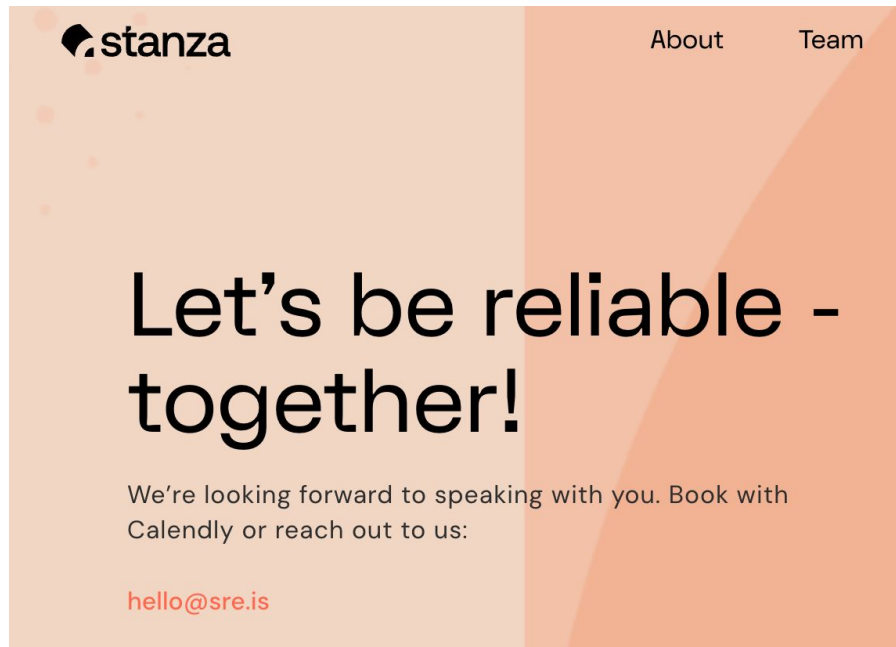
Ask: How does this system help the operator build an accurate mental model of how it works?



Thank you!

Contact me:

- laura@sre.is
- @lauralifts.bsky.social on BlueSky



<https://www.stanza.systems/book-a-demo>