



# SRE-MUC Meetup

2023-02-13 / Winter Edition

[muc.sre.xyz](https://muc.sre.xyz) / [muc@sre.xyz](mailto:muc@sre.xyz)

# Agenda

- 19:00 - Welcome!
- 19:15 - Talk 1 - Maximilian Schaugg:
  - ...and now it is yours! Building reliable and maintainable platforms for customers
- 19:45 - Short Break
- 20:00 - Talk 2 - Tal Zwick:
  - Simplifying Development and Testing Infrastructure with mirrord
- 20:30 - Talk 3
  - SREcon 2022 EMEA recap
- 20:45 - Networking + Drinks
- 21:30 - Leave happy and inspired

# New Website

- Christian Rebischke invested some serious time
- Want to publish our slides and talks there
- Check it out at: [muc.sre.xyz](https://muc.sre.xyz)!

# Feedback from the last session

- Quality of talks: 35% really good, 50% good, 15% meh (rating 1-5)
- Length: 71% spot on, 29% a bit too short
- Most folks gave all three talks from the last session a very good or good rating
- 92% agreed to join again

n=14

# Participation!

- Meetups are communities that create engagement!
- We're always looking for talks!
- We're always looking for hosting companies!
- Building a community needs YOU!

Get in touch with us please at [muc@sre.xyz](mailto:muc@sre.xyz)!

Talk 1 - Maximilian Schaugg:

...and now it is yours! Building reliable and maintainable platforms for customers

Talk 2 - Tal Zwick:

Simplifying Development and Testing  
Infrastructure with mirrord

# Talk 3

SREcon 2022 EMEA recap



## Some facts upfront

- SREcon is a gathering of engineers who care deeply about site reliability, systems engineering, and working with complex distributed systems at scale. SREcon strives to challenge both those new to the profession as well as those who have been involved in it for decades. The conference has a culture of critical thought, deep technical insights, continuous improvement, and innovation.
- Americas: 540 attendees (500 in-person, 40 virtual)
- 62% engineers, 18% managers, 13% tech lead

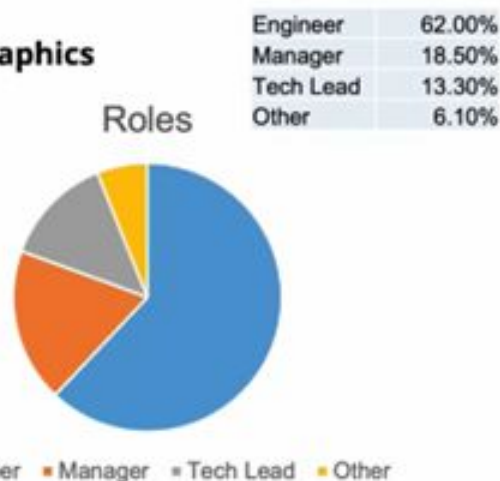
### Attendees

- ~540 attendees (~500 in-person, ~40 virtual)
- 37 countries
- 205 companies

### Talks

- ~200 talks
- 47 accepts
- 58 speakers

### Demographics



## Making the Impossible Impossible: Improving Reliability by Preventing Classes of Problems

- That key strength of SLOs - viewing reliability as a percentage game - can also also be a weakness. Within that framing, there are certain solutions we're likely to overlook.
- This talk explores another lens for reliability - one that's complementary to SLOs: structuring software in a way that rules out entire classes of problem.

- <https://www.usenix.org/conference/srecon22emea/presentation/sinjakli>

### Smug comments:

- State machines
- Memory safety
- Database migrations

Add more  
unit tests

Write  
better C

Just hire  
a DBA

# Over Nine Billion Dollars of SRE Lessons - The James Webb Space Telescope

- After only a few months of activity, the James Webb Space Telescope has already proven to be a spectacular engineering and scientific success. Webb's predecessor, the Hubble Space Telescope, famously required many astronaut visits to repair and upgrade it. But Webb was designed to fly further away from Earth, out of reach of repair by either astronaut or SRE. Any failure would be final and public.
- How did NASA succeed in building the confidence it needed to deploy a completely automated, self-healing, observatory?
- In this session you will learn SRE lessons through the lens of NASA's experiences in developing and launching exploratory probes.
- <https://www.usenix.org/conference/srecon22emea/presentation/barron>

## The Webb "SRE Strategy"

- Aim for 100% availability/success
- Embrace multiple new technologies for a new product
- Invest all efforts in one major deployment for success
- Maximize functional capability/capacity by reducing monitoring/observability load
- Achieve performance and reliability beyond SLA/SLO
- Create redundant systems, as far as possible
- Reduce technical debt / avoid problems detected in previous missions.
- Prioritize NFRs, balanced with functional requirements. Identify single points of failures
- Balance observability requirements (additional load, complexity, costs) with benefits
- Test, test, test and test some more Tests can have business value
- Chaos Engineering as a strategy

SRE CON EUROPE MIDDLE EAST AFRICA



@flyingbarron

## SREs shouldn't do it the Webb way

**Do not!**

**Do!**

- Aim for 100% availability/success
- Embrace multiple new technologies for a new product
- Invest all efforts in one major deployment for success
- Maximize functional capability/capacity by reducing monitoring/observability load
- Achieve performance and reliability beyond SLA/SLO
- Create redundant systems, as far as possible
- Reduce technical debt / avoid problems detected in previous missions.
- Prioritize NFRs, balanced with functional requirements. Identify single points of failures.
- Balance observability requirements (additional load, complexity, costs) with benefits
- Test, test, test and test some more Tests can have business value
- Chaos Engineering as a strategy

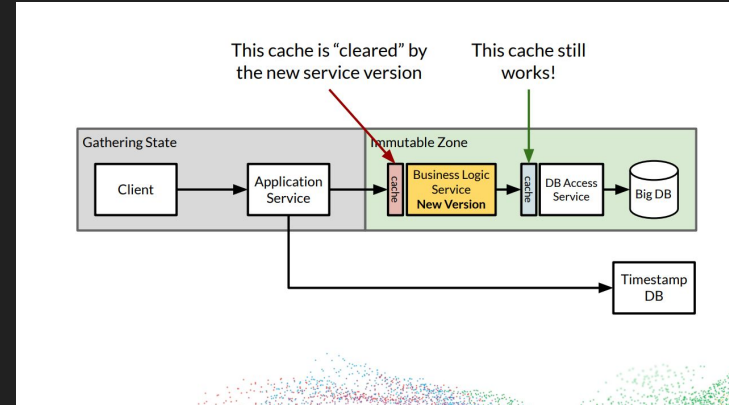
SRE CON EUROPE MIDDLE EAST AFRICA



@flyingbarron

# Caching Entire Systems without Invalidation

- Caching is simplest when dealing with stateless components, so the solutions presented revolve around controlling for states that are sometimes not accounted for, such as user settings, software versions, dependencies, wall time, and database state.
- The "punch line" is that resolving state early in the request flow of your system will allow you to divide your system into two parts, the "state gathering" portion and the "immutable" or stateless portion. The immutable portion would have excellent cache characteristics, such as the ability to transparently cache responses throughout multiple service layers without an active cache invalidation strategy.
- <https://www.usenix.org/conference/srecon22emea/presentation/sperl>



## Benefits of Timestamped Data Storage

Database reads can be cached as well as any service call that depends on it

Point-in-time access is trivially supported

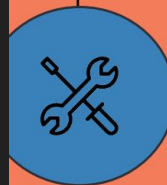
Batch jobs can freeze the timestamp to ensure consistency, while updates continue unaffected

Rollbacks can be performed with a system-wide cap on timestamp

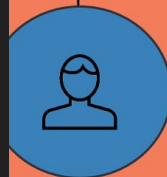
Timed releases are just future-dated timestamps

# Diamonds with Flaws: Examining the Pressures, Realities, and Future of Site Reliability Engineering

- The technology industry moves at an incredible pace. Innovation and change are always at the forefront of everyone's mind. Especially in the Site Reliability Engineering space, people feel pressured more than ever to keep up with all of the newest tools, processes, and philosophies.
- For many organizations, however, chasing all of the shiny things can end up being a detriment as opposed to a benefit. Let's examine these pressures, what the realities of most SRE organizations are, and how we can all best move into the future -- together, thoughtfully, and meaningfully.
- <https://www.usenix.org/conference/srecon22emea/presentation/hidalgo>



***Your*** SRE won't look like my SRE



Don't just do what everyone else is doing. Ignore people like me.

# SRE and ML: Why It Matters

- Machine Learning is an incredibly hyped set of technologies. It seems that ML is becoming an important part of distributed computing. I'll review whether SREs need to know anything about ML yet (probably you do—sorry!).
- And since ML reliability is challenging, I'll suggest some changes required for most SREs and even some significant changes to our profession. Finally, I'll review the state of using ML to automate production with an extremely skeptical eye.
- <https://www.usenix.org/conference/srecon22emea/presentation/underwood>

**ML Systems ~~Does~~  
~~Is~~ Are Starting to  
Matter to SREs**



## ML Ops: Making ML Systems Run Reliably

**Status:** Happening **now** and growing fast.

**Future:** Most services will have or use ML components on their critical path. ML is the future of software. (sorry!)

**SRE Goal:** Acquire skills/knowledge to do our (future) jobs well.





# What SRE Could Be: Systems Reliability Engineering

Laura Nolan, Stanza

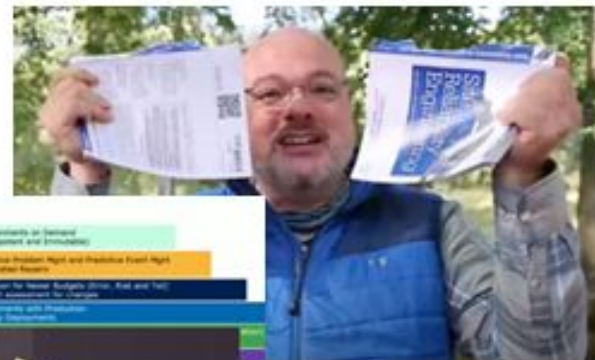
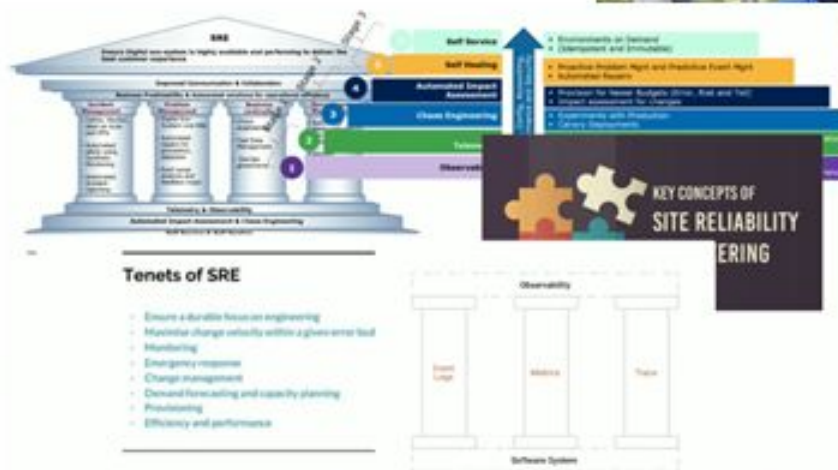
*SRE's Identity Crisis*

## Standardisable Practices

- SLOs
- Incident Management
- Some kinds of Automation

## Non-Standardisable Practices

- System Design
- Anomaly investigation and incident review
- Risk Analysis
- Building safe control planes
- Design service-specific monitoring and altering
- System-specific disaster testing



SRE Philosophy (cf. DevOps Philosophy )

**SRE is: Understanding systems and making appropriate interventions to make them function better.**

**Technical Systems Intervention:** Delaying before re-replicating data; Cellular Design Pattern; Start incrementing counters close to their maximum value to test wraparound

**Socio-systems interventions:** Minimum on-call team size to avoid burn-out; SLOs and error budgets to prioritize work; Blameless IR; Automating Toil

**Systems Thinking:** Describing and understanding systems behaviour; Evaluating Risks; Analyzing Incidents

## Outage Track

*Deep Analysis and Learning from past incidents*

### **Deep Dive: Azure Resource Manager Outage**

Benjamin Pannell and Brendan Burns, Microsoft

### **Commas Save Lives, or at Least LinkedIn**

Todd Palino, LinkedIn

### **Disaster Recovery Testing at Booking.com**

Yoann Fouquet, Booking.com, and Paola Martinucci, Mollie.com

### **Slack's DNSSEC Rollout: Third Time's the Outage**

Rafael Elvira, Slack

### **How We Drained Every Backbone Router Simultaneously**

Francois Richard, Meta

### **Honey, I Broke the Things: Debugging Gray Failures in Production!**

Radha Kumari, Slack Technologies



# Navigating in the Dark

## Nati Cohen, AWS

### *How to get prepared for failure*

Everything can fail, all the time

- Communication Platforms
- Monitoring / Paging
- SDLC / Collaboration

Not just outages

- No data
  - Permissions, network partitions, agent failures
- Partial data
  - Docker awslogs max-buffer-size issues
- Too much data
  - Too many notifications / alerts / logs (log level)

Auxiliary systems

- Shared infrastructure
- Automation

Auxiliary system to Auxiliary system

- Autoscaling -> monitoring
- Gradual deployment / rollback -> monitoring
- ChatBots to Instant messaging
- GitOps -> Source Control Management
- Customer communication -> Collaboration
- Service Health Dashboard -> Internal Networking

### What to do ?

Alarm for "no data"

- Treat missing data as bad
- Absent()

Internal Status Page (for auxiliary systems)

External health-check page that lightly touches our services

Global Playbook (who to call, post-incident procedure)

Bookmark secondary systems (have them ready at hand)

Know your Customers

Prepare CLI commands

Multiple Alarm Targets (or logs), i.e. Docker Dual-Logging

Run Gamedays with "Flying Blind"

# Program / Reflections on SREcon 2022

- Full program with slides and recordings:
  - <https://www.usenix.org/conference/srecon22emea/program>
- Reflections (summary) by Niall Murphy
  - <https://blog.relyabilit.ie/reflections-on-srecon-emea-2022/>
- Next SREcon 2023 Americas
  - 2023-02-21 to 23 in Santa Clara
  - <https://www.usenix.org/conference/srecon23americas>
  - Proposals for lightning talks are still open till later today!

Thanks for coming!

Please give Feedback for  
this session:

