RELIABILITY UNLEASHED

THE ENGINEERING PLAYBOOK

From Chaos to Confidence

A Comprehensive Guide to Site Reliability Engineering

9 Parts | 34 One-Pagers | ~8.5 Hours | Technical Track

THE JOURNEY AHEAD

Part 1
Foundation & Vision

Part 2
Observability Mastery

Part 3
Resilience Patterns

Part 4
Incident Excellence

Part 5
Release, Testing & Capacity

Part 6
Cloud & Infrastructure

Part 7
AI/ML & Agentic Ops

Part 8
People & Culture

Part 9
Industry & Roadmap

PART 1 OF 9

FOUNDATION & VISION

SRE Fundamentals, SLIs/SLOs, DORA Metrics, Maturity Assessment

- reliability-unleashed
- sre-foundations
- dora-24-capabilities
- sre-maturity-assessment

WHY SRE? WHY NOW?

4+

AI Agents
in production

24/7

Bot operations
never sleep

100s

Daily commits
across worktrees

Bots don't get tired. But they can fail.

And when they do, who responds at 3 AM?

DEVOPS VS SRE

```
class SRE implements
interface DevOps { }
```

DEVOPS

→ Philosophy, culture, movement

→ "Break down silos"

→ Continuous delivery mindset

→ Automation everywhere

SRE

→ Specific implementation

→ Error budgets, SLOs

→ Toil reduction targets

→ On-call engineering

"DevOps is the philosophy; SRE is the implementation." — Google

THREE PILLARS OF OPERATIONS

REACTIVE

- Alert triage & response
- Runbook execution
- Incident management
- Escalation protocols

PROACTIVE

- SLO monitoring
- Capacity planning
- Change management
- Toil reduction

PREDICTIVE

- Anomaly detection
- Chaos engineering
- AIOps & ML
- Self-healing systems

THE VISION: AUTONOMOUS RELIABILITY

"Bots that monitor, diagnose, remediate, and learn — with humans for strategy and novel challenges."

70%
Auto-resolved at L1

<15min
MTTR target

99.95%
Availability goal

PART 2 OF 9

OBSERVABILITY MASTERY

Three Pillars, OpenTelemetry, Alerting Strategy, High-Cardinality Events

- observability-mastery
- multi-window-alerting
- use-method-performance
- observability-2.0
- alert-tuning-playbook

LEARNING FROM INDUSTRY LEADERS

GOOGLE SRE

Error budgets, 50% cap

NETFLIX

Chaos Monkey

AWS

Well-Architected

META

SEV culture

SPOTIFY

Golden paths

TOYOTA

Kaizen

HIGH-RELIABILITY ORGANIZATIONS

Lessons from Aviation, Nuclear, Healthcare, Military

1	**Preoccupation with Failure**	— Never ignore small failures
2	**Reluctance to Simplify**	— Embrace complexity
3	**Sensitivity to Operations**	— Real-time awareness
4	**Commitment to Resilience**	— Detect, contain, recover
5	**Deference to Expertise**	— Empower frontline decisions

AVIATION: CREW RESOURCE MANAGEMENT

Origin: 1978 United Flight 173 — crew ran out of fuel while troubleshooting

70-80% of accidents from **human error**, not mechanical failure

> *"Up until 1980, we worked on the concept that the captain was THE authority. What he said, goes. And we lost a few airplanes because of that."*
>
> **— Captain Al Haynes, United 232**

Bot Application: Actively seek input from other bots; hierarchical authority yields to expertise

NETFLIX: CHAOS ENGINEERING

Philosophy: "Avoid failure by failing constantly"

Chaos Monkey

Latency Monkey

Chaos Gorilla

0 Impact

When AWS lost 10% of servers (Sept 2014), Netflix kept running

Bot Application: Regular game days, failure injection testing, resilience as cultural value

SCALING RELIABILITY: INDUSTRY EXAMPLES

STRIPE
99.999% uptime
Defensive design

UBER
Millions RPS
Jaeger tracing

SHOPIFY
57.3 PB BFCM
9-mo prep cycle

DISCORD
30M msg/sec
Elixir + ScyllaDB

ROBLOX
145K machines
Cell architecture

CLOUDFLARE
320+ cities
Follow-the-sun

LATENCY TIERS: RIGHT-SIZING RELIABILITY

LESSONS FROM MISSION-CRITICAL INDUSTRIES

SPACE

- Triplex redundancy
- 7K+ engine tests
- Formal verification

MILITARY

- Disciplined initiative
- Decentralized exec
- Pre-deployment sim

NUCLEAR

- Defense in depth (5)
- Diverse redundancy
- Safety isolation

DEEP SEA

- 3 battery buses
- 180+ monitored
- Galvanic failsafe

JUST CULTURE: BLAMELESS POST-MORTEMS

"Blame closes off avenues for understanding how and why something happened."

— Sidney Dekker

OLD VIEW

People cause failure → Punish

NEW VIEW

Error is symptom → Fix system

Ask "what" and "how", never "why"

PART 3 OF 9

RESILIENCE PATTERNS

Circuit Breakers, Defense in Depth, HRO Principles, Chaos Engineering

resilience-patterns

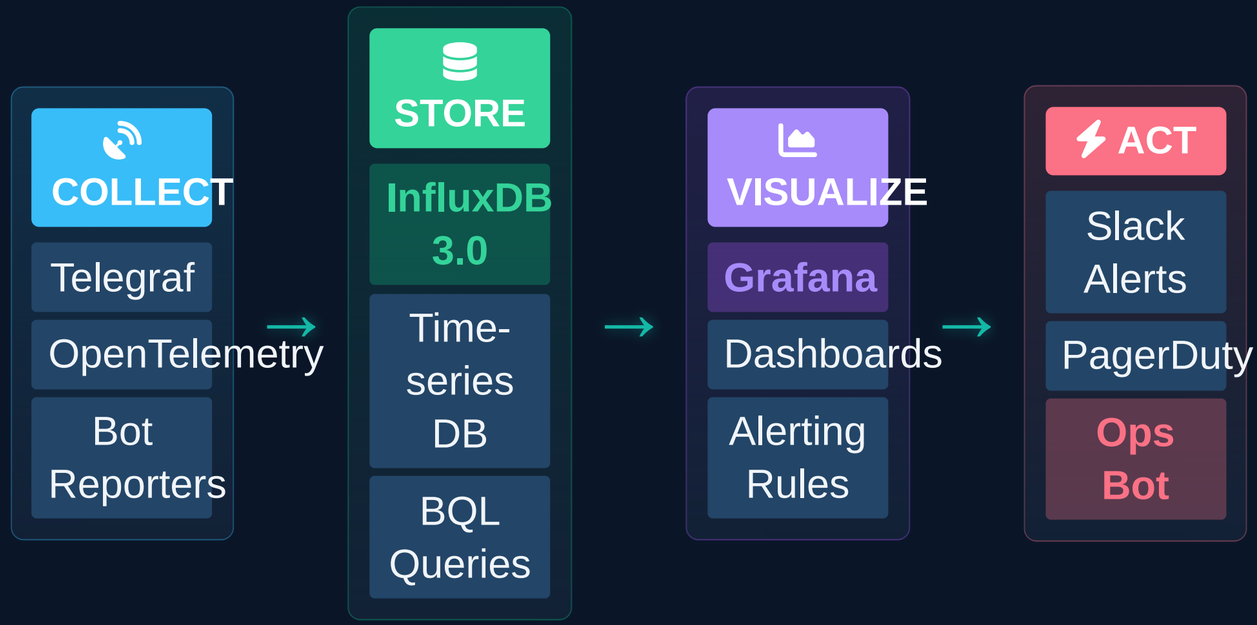
defense-in-depth

hro-pattern-recognition

release-it-patterns

chaos-engineering

OUR OBSERVABILITY STACK



INFLUXDB 3.0 & BQL QUERIES

WHY INFLUXDB?

- → Native time-series storage
- → High-cardinality support
- → Columnar compression
- → Sub-second query latency
- → Downsampling & retention

BQL QUERY EXAMPLES

```
-- Session success rate
SELECT mean(success_rate)
FROM bot_sessions
WHERE time > now() - 1h
GROUP BY bot_name

-- Error budget burn
SELECT sum(errors) / sum(total)
FROM api_calls
WHERE time > now() - 30d
```

GRAFANA DASHBOARD STRATEGY

ATHENA SYSTEM

CPU, memory, disk, network

→ SRE Team

BOT ARMY

Sessions, productivity, commits

→ All Engineers

BOT OPERATIONS

SLOs, MCP health, error budgets

→ Ops Bot

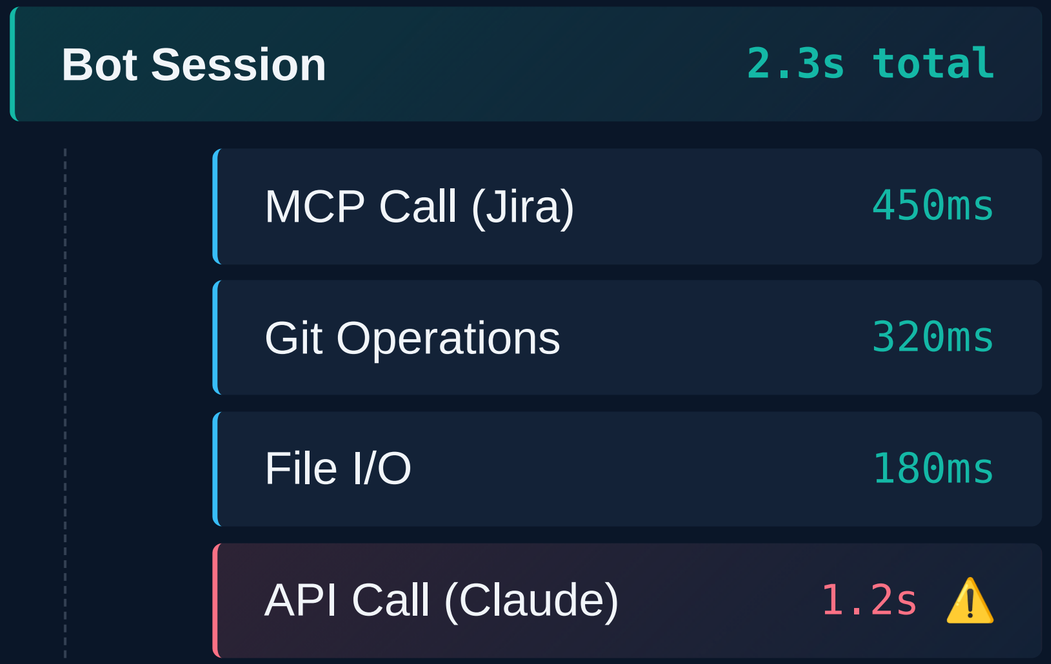
HUMAN EXPERIENCE

Focus metrics, escalations

→ Human CEO

Each dashboard serves a specific audience with relevant context

DISTRIBUTED TRACING WITH JAEGER



Latency breakdown — Where is time spent?

Error propagation — What caused the failure?

Dependency mapping — What calls what?

CORRELATION IDS & AGENT CONTEXT

Session ID `sess_abc123` → Bot Identity `claude-feat` → Task ID `HOME-456` → Trace ID `tr_xyz789`

CROSS-SIGNAL CORRELATION

→ Link metrics → logs → traces

→ Find all activity for one session

→ Reconstruct incident timeline

AUDIT TRAIL

→ Which bot made this change?

→ What JIRA ticket triggered it?

→ Full provenance chain

CENTRALIZED LOGGING STRATEGY

ERROR	WARN	INFO	DEBUG
Failures needing action	Degraded but recovering	Normal operations	Troubleshooting
90 days	30 days	14 days	7 days

- Structured JSON
- Correlation IDs
- Searchable fields
- No secrets

ALERTING PHILOSOPHY: SIGNAL VS. NOISE

> *"Every alert should be actionable. If you can't act on it, it's noise."*

P1 - PAGE

Service down, data loss risk

Immediate response

P2 - NOTIFY

Degraded, SLO at risk

Within 1 hour

P3 - TRACK

Anomaly detected

Business hours

P4 - LOG

Informational

Review weekly

USE METHOD: PERFORMANCE ANALYSIS

Brendan Gregg's systematic approach to resource bottlenecks

UTILIZATION

Average time resource was busy

`CPU: 85%, Memory: 72%`

SATURATION

Extra work queued or denied

`Queue depth, wait time`

ERRORS

Count of error events

`ECC errors, retries, drops`

Apply to every resource: CPU, Memory, Disk I/O, Network, GPUs, API quotas

RED METHOD: SERVICE MONITORING

Tom Wilkie's approach for request-driven services

RATE

Requests per second

`http_requests_total`

ERRORS

Failed requests per second

`5xx responses, exceptions`

DURATION

Time per request (latency)

`P50, P95, P99 histograms`

USE for Resources | **RED for Services** | **Both for Complete Coverage**

MULTI-WINDOW BURN RATE ALERTING

Burn Rate = How fast you're consuming error budget

```
burn_rate = (errors / window) / (budget / period)
```

5 min

Fast Burn

Immediate outage

>10x → P1

1 hour

Medium Burn

Sustained issues

>5x → P2

6 hours

Slow Burn

Degradation trend

>2x → P3

From Liz Fong-Jones & Google SRE Workbook

PART 4 OF 9

INCIDENT EXCELLENCE

Response & Postmortems, Learning from Catastrophe, Runbook Design

incident-excellence

learning-from-catastrophe

runbook-quick-reference

SLOS AND ERROR BUDGETS

SLI	TARGET	ERROR BUDGET
Availability	99.5%	3.6 hrs/month
Success Rate	98.0%	2% failures
Latency P95	<3s	2% slow
MTTR	<15min	Agentic response
Auto-Resolution	70%	L1 handled by Ops Bot

>50%
Ship freely

25-50%
Prioritize reliability

<25%
Feature freeze

OPERATIONAL METRICS: FULL COVERAGE

SYSTEM HEALTH

MCP Availability: 99.9%

Resource Util: <80%

API Headroom: >20%

BOT PRODUCTIVITY

Session Success: >95%

Commits/Session: >3

Stall Rate: <5%

OPERATIONAL TOIL

Manual: <5/wk

Automation: >80%

Alert Noise: <20%

INCIDENT QUALITY

MTTD: <2 min

MTTA: <5 min

Recurrence: <10%

INCIDENT LIFECYCLE (ITIL)

1. Identify → 2. Categorize → 3. Prioritize → 4. Respond → 5. Close

SEV1	Critical — <15 min response
SEV2	Major — <1 hour response
SEV3	Minor — <4 hours response
SEV4	Low — <24 hours response

BOT-FIRST ESCALATION MODEL

L1: Ops Bot — Auto-triage, runbook execution

70%

↓

L2: Bot Team — Bot-to-bot coordination

25%

↓

L3: Human Expert — Complex/novel issues

5%

THE 50% RULE: TOIL REDUCTION

WHAT IS TOIL?

→ Manual, repetitive work

→ No enduring value

→ Scales linearly with growth

→ Automatable

AUTOMATION PRIORITIES

1. Runbook automation
2. Incident triage
3. Deployment pipelines
4. Capacity scaling

TESTING FOR RELIABILITY

UNIT TESTS

Fast, isolated

80%+ coverage

INTEGRATION

Component APIs

Critical paths

CHAOS

Failure injection

Prod-like

E2E

Full workflow

Key journeys

Jane Street: *"Deterministic simulation testing finds bugs random testing cannot"*

CHAOS ENGINEERING & GAMEDAYS

"Avoid failure by failing constantly" — Netflix

1 HYPOTHESIS
Define expected behavior

2 INJECT
Kill process, add latency

3 OBSERVE
Monitor SLOs, alerts

4 LEARN
Fix gaps, document

Chaos Monkey Toxiproxy Gremlin LitmusChaos

ON-CALL SUSTAINABILITY

70%

SREs: on-call → burnout

2,000+

Weekly alerts (3% actionable)

GOOGLE'S SUSTAINABLE LIMITS

- **12h** max shift
- **2** pages/shift
- **25%** time on-call
- **5-8** per rotation

With bot-first response, humans should rarely be paged

BLAMELESS POST-MORTEM PROCESS

1 TIMELINE → **2** ROOT CAUSE → **3** FACTORS → **4** ACTIONS → **5** SHARE

>20% budget | SEV1/SEV2 | Novel failures | Near-misses

THE THREE WAYS OF DEVOPS

From "The Phoenix Project" and "The DevOps Handbook"

→ FIRST WAY: FLOW

Fast flow from Dev to Ops to Customer

- Small batch sizes
- Reduce WIP
- Eliminate constraints

🔄 SECOND WAY: FEEDBACK

Fast, constant feedback loops

- Telemetry everywhere
- Push quality upstream
- Enable fast recovery

🧠 THIRD WAY: LEARNING

Continuous experimentation & learning

- Take risks, embrace failure
- Build mastery through practice
- Institutionalize improvement

OBSERVABILITY: THREE PILLARS + CONTEXT

METRICS

InfluxDB + Grafana

LOGS

Structured events

TRACES

Jaeger + OpenTelemetry

CONTEXT

MCP + Correlation IDs

"If you can't monitor a service, you don't know what's happening, and if you're blind to what's happening, you can't be reliable."

— Google SRE Book

PART 8 OF 9

PEOPLE & CULTURE

Westrum Culture, Team Topologies, On-Call Excellence, The Three Ways

- people-culture
- oncall-excellence
- three-ways-devops
- team-topologies

BOT ARMY SRE TEAM STRUCTURE

INCIDENT RESPONSE

Ops Bot

Alert triage, runbooks

RELIABILITY ENG

SRE Bot

SLOs, capacity, chaos

OBSERVABILITY

Obs Bot

Dashboards, alerting

SECURITY OPS

Sec Bot

Compliance, audits

PART 5 OF 9

RELEASE, TESTING & CAPACITY

DORA Metrics, Progressive Delivery, NALSD, Testing Automation

- capacity-release
- nalsd-framework
- designing-for-recovery
- slo-design-framework

PAGERDUTY: THE ON-CALL BACKBONE

Grafana Alerts → **PagerDuty** →
- Slack #bot-alerts
- Ops Bot (L1)
- JIRA Incident

PAGERDUTY AI AGENTS (2025)

SRE Agent
Auto-classify, remediate

Shift Agent
Schedule conflicts

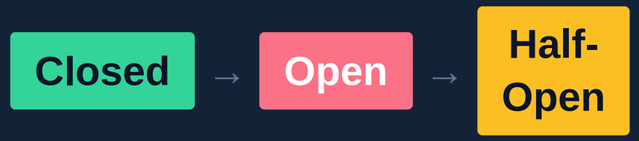
Scribe Agent
Capture insights

Insights Agent
Data analysis

OVERLOAD PROTECTION: CASCADING FAILURE PREVENTION

CIRCUIT BREAKERS

Stop calling failing services



LOAD SHEDDING

Reject requests to protect system

- Priority-based queuing
- Graceful degradation
- Uber's Cinnamon (PID controller)

BACKPRESSURE

Slow down upstream producers

- Rate limiting
- Queue depth limits
- Timeout cascades

DORA METRICS: MEASURING EXCELLENCE

DEPLOY FREQ	
Low	Monthly
Med	Weekly
High	Daily
Elite	**On-demand**

LEAD TIME	
Low	Months
Med	Weeks
High	Days
Elite	**<1 hour**

FAILURE RATE	
Low	>30%
Med	15-30%
High	5-15%
Elite	**<5%**

MTTR	
Low	Weeks
Med	Days
High	<1 day
Elite	**<1 hour**

Elite performers ship faster AND more reliably

INDUSTRY SCALE: FROM STARTUP TO HYPERSCALE

Startup	Growth	Enterprise	Hyperscale
10-100 RPS	1K-100K RPS	100K-1M RPS	1M+ RPS
Monolith • Manual ops	Microservices • On-call	Distributed • Chaos eng	Global • Cell-based
99.5% SLO	**99.9% SLO**	**99.95% SLO**	**99.99%+ SLO**

UNIVERSAL RELIABILITY PRINCIPLES

Applicable to any mission-critical system

1	2	3	4	5
LAYERED DEFENSE	**GRACEFUL DEGRADATION**	**RAPID RECOVERY**	**CONTINUOUS VERIFY**	**AUTO + GUARDRAILS**
Multiple failure barriers	Core function survives	Fast detect-to-resolve	Prove it works	Empower within bounds

PART 6 OF 9

CLOUD & INFRASTRUCTURE

Kubernetes, Platform Engineering, Cloud-Native SRE, Multi-Cloud

- infrastructure-reliability
- kubernetes-patterns
- platform-engineering

PART 7 OF 9

AI/ML & AGENTIC OPERATIONS

MLOps, Non-Determinism, Bot Operations, Multi-Agent Systems

ai-ml-operations

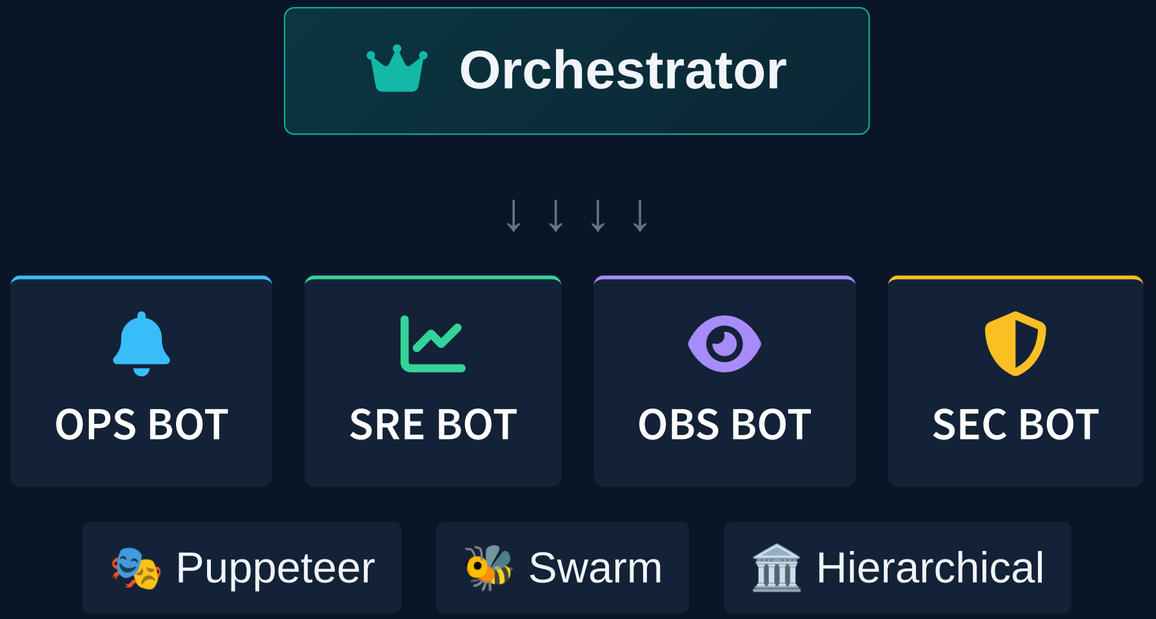
agentic-operations

AGENTIC OPERATIONAL WORKFLOWS

1. **DETECT** — Alert triggered
2. **CORRELATE** — Query signals
3. **DIAGNOSE** — AI analysis
4. **REMEDIATE** — Run playbook
5. **LEARN** — Refine models

The goal: closed-loop autonomous operations

MULTI-AGENT ORCHESTRATION



DATA STRATEGY FOR AUTONOMOUS AGENTS

REAL-TIME

- Last 5 min metrics
- Active alerts
- Deployments

HISTORICAL

- 90-day incidents
- Resolution patterns
- SLO trends

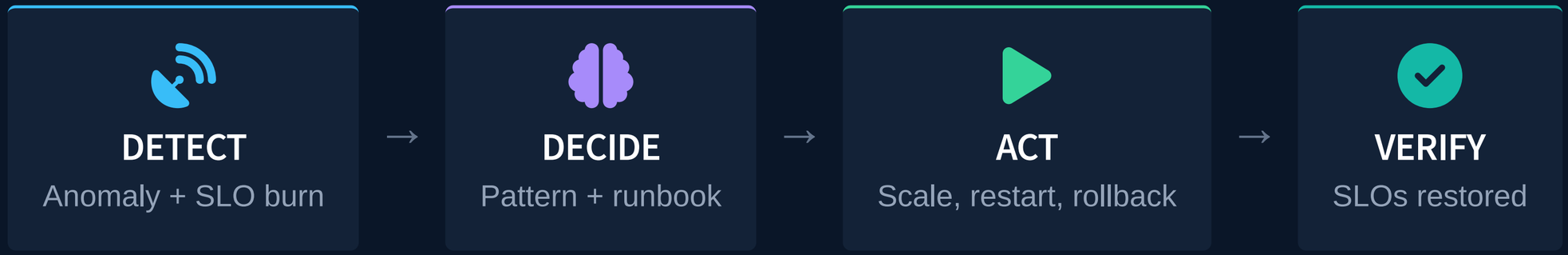
KNOWLEDGE

- Runbooks
- Architecture
- Post-mortems

THE LEARNING LOOP

```
Incidents → Analysis → Patterns → Runbooks → Automation
```

SELF-HEALING SYSTEMS



Memory: Auto-restart

Latency: Scale up

Deploy fail: Rollback

PLATFORM ENGINEERING: GOLDEN PATHS

"A golden path is a paved road to a well-architected production deployment" — Spotify

NEW SERVICE

Template → CI/CD → Observability → Alerts → Docs

10 minutes to production-ready

BOT ONBOARDING

Identity → Worktree → MCP → Permissions → SLOs

Self-service, automated

INCIDENT RESPONSE

Alert → Runbook → Resolution → Post-mortem

Guided workflow, minimal toil

Make the right thing the easy thing

ATHENA → CLOUD: ENVIRONMENT PORTABILITY

ATHENA (ON-PREM)

- InfluxDB + Grafana
- Jaeger local
- Low latency
- Full control

PUBLIC CLOUD (AWS/GCP)

- Managed services
- Auto-scaling
- Global edge
- Shared responsibility

OpenTelemetry
Abstraction

GitOps + IaC
Deployment

Grafana Cloud
Observability

Env-agnostic
Config

DEPLOYMENT AUTOMATION: BLEEDING EDGE

GITOPS PIPELINE

- Declarative IaC (Terraform)
- ArgoCD / Flux sync
- PR-based deployments

PROGRESSIVE DELIVERY

- Canary releases (1-5%)
- SLO-gated rollouts
- Auto-rollback on error

FEATURE FLAGS

- Decouple deploy/release
- A/B testing built-in
- Instant kill switches

OBSERVABILITY CI

- Pre-deploy SLO checks
- Synthetic monitoring
- Chaos validation

Target: Zero-touch deployments with bot-driven validation and rollback

PART 9 OF 9

INDUSTRY & ROADMAP

Google, Netflix, NASA, Automation Paradoxes, SRE Evolution, Getting Started

industry-leaders

implementation-roadmap

automation-paradoxes

sre-evolution-timeline

IMPLEMENTATION ROADMAP

1

FOUNDATION

Alerting, playbooks

2

RELIABILITY

SLOs, GameDays

3

AUTOMATION

Self-healing

4

INTELLIGENCE

ML, prediction

5

EXCELLENCE

Cloud, 99.95%

AUTOMATION PARADOXES

Bainbridge's "Ironies of Automation" (1983)

SKILL DECAY

Operators lose skills. Can't step in when automation fails.

COMPLACENCY

Reduced vigilance. Failures become catastrophic.

CLUMSY AUTO

Workload increases during high-stress moments.

MITIGATION

Regular drills, transparent automation, graceful degradation.

"The more advanced the automation, the more crucial the human contribution"

SRE EVOLUTION TIMELINE

Past

Manual ops → Automation

Present

Platform Engineering

Future

Autonomous Reliability

KEY TAKEAWAYS

1

Speed & Stability Reinforce

DORA proves elite orgs do both

2

Error Budgets Balance

Quantified risk tolerance for innovation

3

Build for Failure

Resilience is designed, not accidental

4

Automate Toil

<50% cap frees humans for engineering

5

Incidents Are Investments

Every failure makes systems stronger

6

Observability > Monitoring

Understand systems, not just alert

RELIABILITY UNLEASHED

QUESTIONS?

34 One-Pagers Available: Comprehensive reference material for each topic

SRE Foundations | Observability | Resilience | Incidents | Release & Capacity
Cloud & Infrastructure | AI/ML & Agentic | People & Culture | Industry Leaders

Essential Reading

Google SRE Book
Netflix Tech Blog
Dekker's Just Culture

Next Steps

Review one-pagers
Assess maturity level
Build your roadmap