DevOps Institute

ADVANCING THE HUMANS OF DEVOPS

SRE and Incident Management

MODULE 7

Workshop Series Overview

Choose a module to get started

In each module you will join an instructor-led presentation from AWS and an ambassador of the DevOps Institute. Following the presentation you'll be able to choose a hands-on lab to complete from a selection of the best tools in DevOps. **Pick a module to get started** and **subscribe to email updates** to learn when new content is available.

Module 1

Presentation: **Available On-Demand**

Practicing DevOps

In this first presentation you'll get an overview of the workshop series and receive practical instruction on how to build the right foundation for a successful DevOps practice in AWS.

Watch now >

Module 2

Presentation: **Available On-Demand**

CI/CD Pipelines

In this module you'll learn how to implement a well-engineered CI/CD pipeline that considers governance and provides traceability from idea to production.

Watch now >

Module 3

Presentation: **Available On-Demand**

Evolving to Continuous Deployment

Deploying code changes live into production is still a terrifying prospect for many organizations. We'll dive deep into how using the right processes and tools can make this safe and advantageous.

Watch now >

Module 4

Presentation: **July 29, 2021**

Infrastructure as Code

Here you'll get the in and outs of how to really automate the Ops in DevOps. Craft templates and automate infrastructure provisioning to safely enable everyone with self-service environments.

Register now >

Module 5

Presentation: **Aug 26, 2021**

Continuous Testing

Testing throughout every stage of the pipeline is critical to ensure quality for end users. In this session we'll dig into best practices for developers and architects, covering functional, integration, unit testing and more.

Register now >

Module 6

Presentation: **Sept 15, 2021**

Observability and Monitoring

This session will dive into strategies for knowing how elements of your applications interact and perform, when and where issues arise, and how to fix and prevent them.

Get updates >

Module 7

Presentation: **Oct 13, 2021**

SRE and Incident Management

As reliable as we design our applications, there will always be incidents. In this session you'll learn how to make life easier when things go wrong and get immediate feedback to teams.

Get updates >

Module 8

Presentation: **Nov 18, 2021**

DevSecOps

Organizations looking to achieve fast deployments need to do so safely. Participate in this module to learn how to achieve early, automated, and continuous remediation of security events.

Get updates >

MISSION: Bringing Joy to Work

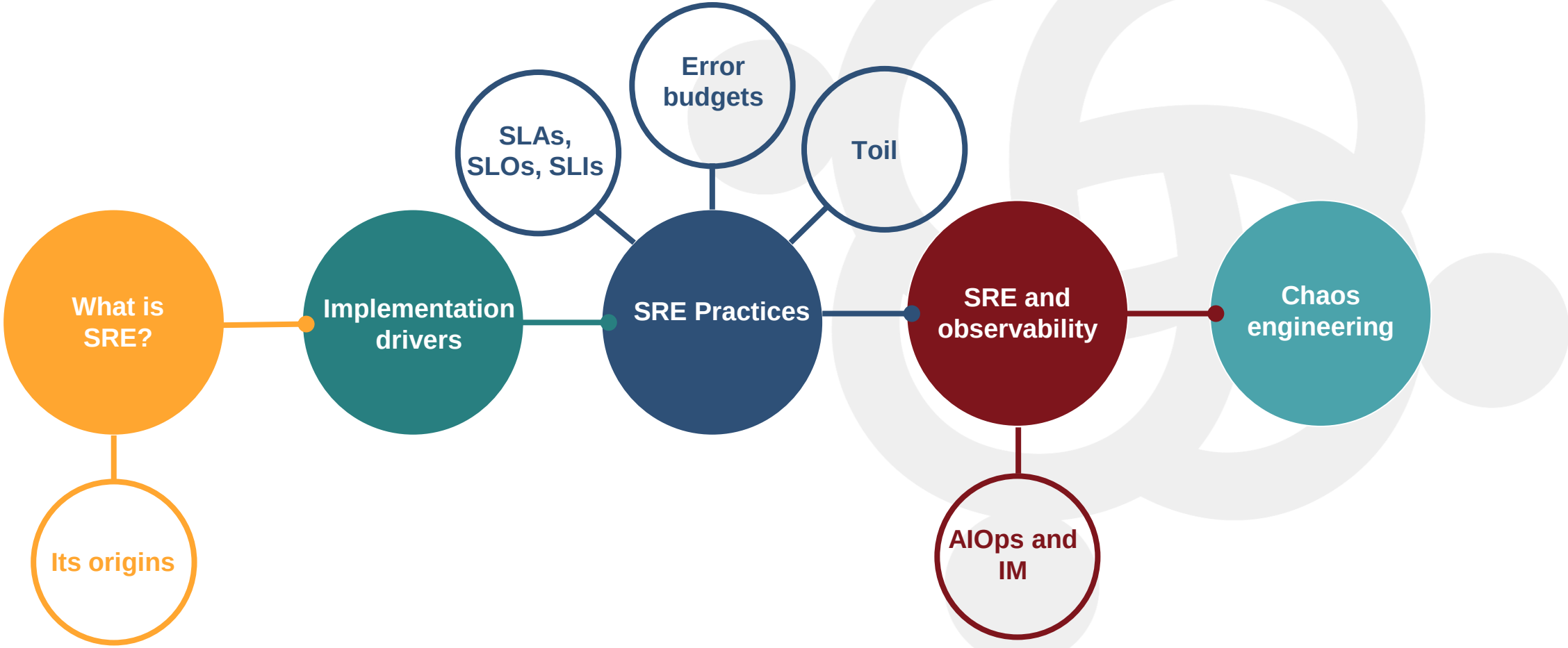
Helen Beal

Herder of Humans

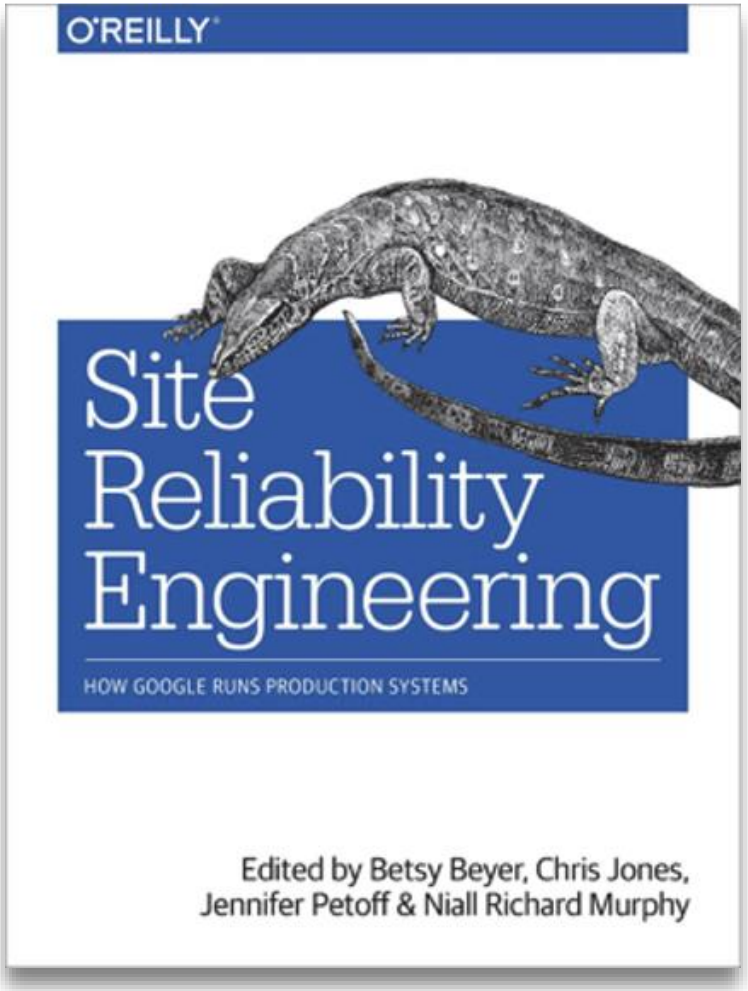
@bealhelen

Helen Beal is a DevOps and Ways of Working coach, Chief Ambassador at DevOps Institute and an ambassador for the Continuous Delivery Foundation. She is the Chair of the Value Stream Management Consortium and provides strategic advisory services to DevOps industry leaders such as Plutora and Moogsoft. She is also an analyst at Accelerated Strategies Group. She hosts the Day-to-Day DevOps webinar series for BrightTalk, speaks regularly on DevOps topics, is a DevOps editor for InfoQ and also writes for a number of other online platforms. She regularly appears in TechBeacon's DevOps Top100 lists and was recognized as the Top DevOps Evangelist 2020 in the DevOps Dozen awards.

Flow: Talk map

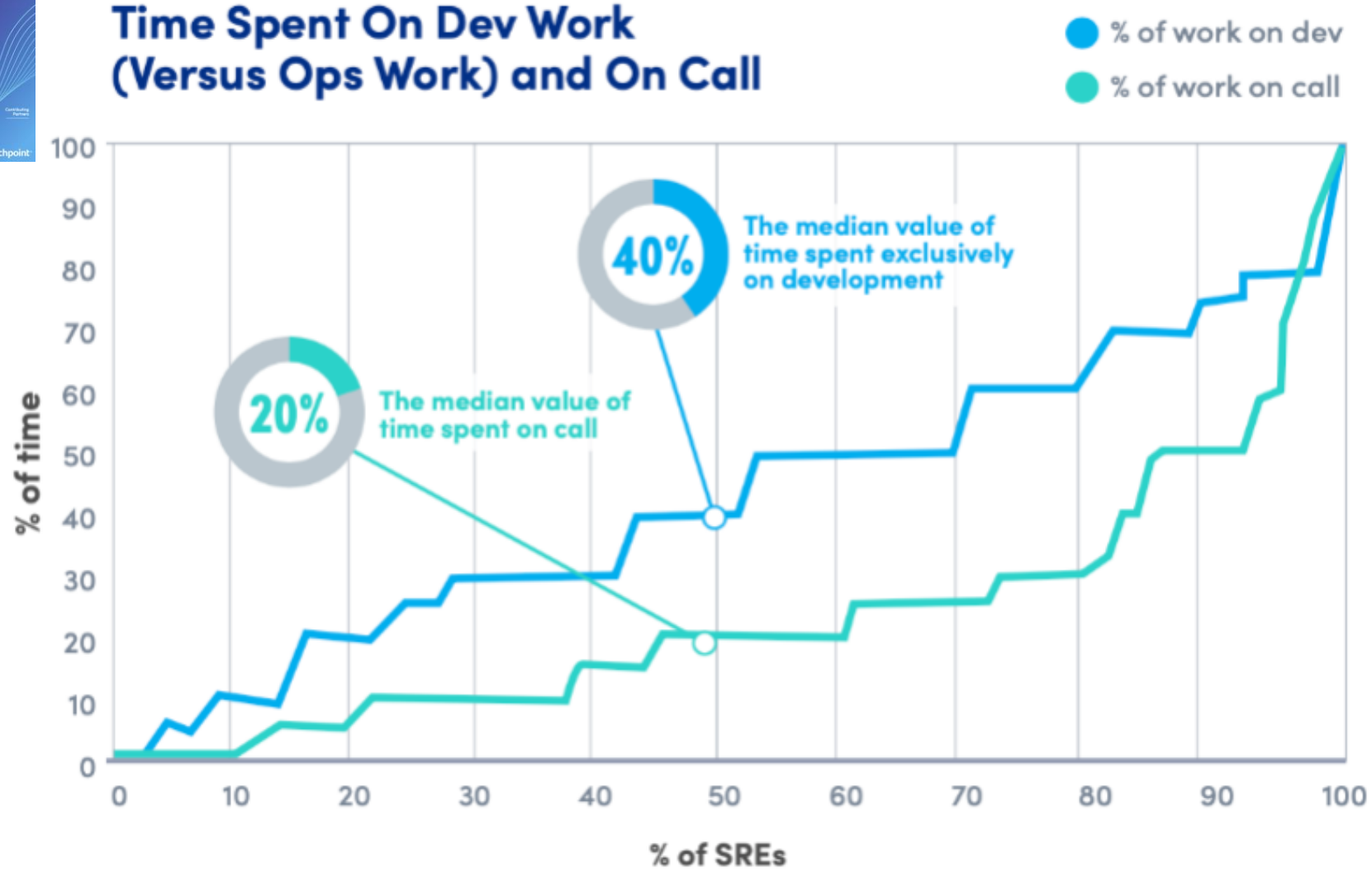


What is SRE?

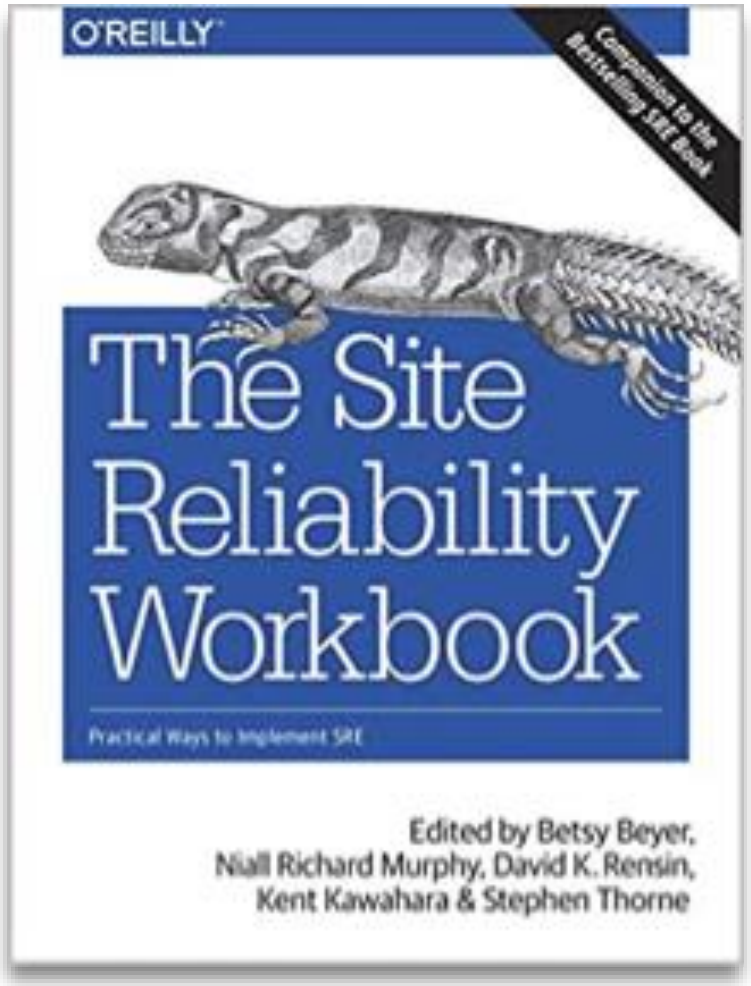


- The goal is to create ultra-scalable and highly reliable distributed software systems
- SRE's spend **50% of their time doing "ops" related work** such as issue resolution, on-call, and manual interventions
- SRE's spend **50% of their time on development tasks** such as new features, scaling or automation
- Observability, monitoring, alerting and automation are a large part of SRE

Time Spent On Dev Work (Versus Ops Work) and On Call



Where has the concept come from?



- Site Reliability Engineering (SRE) is a discipline that incorporates aspects of software engineering and applies them to infrastructure and operations problems
- Created at Google around 2003 and publicized via SRE books

"What happens when a software engineer is tasked with what used to be called operations."

Ben Treynor, Google

class SRE implements DevOps

- **DevOps**

  is a set of practices, guidelines and culture designed to break down silos in IT development, operations, architecture, networking and security.

- **Site Reliability Engineering**

  is a set of practices we've found to work, some beliefs that animate those practices, and a job role.

Let’s ask Mark

How do you connect SRE and DevOps?

Successful SRE Implementation Drivers

60%	How quickly we resolve incidents
43%	The amount of time between failures
41%	How quickly we do root cause incident analysis
40%	How quickly we push product updates
33%	How quickly our business can expand to new markets
22%	How quickly we can understand the cause of social media sentiment

Results in higher performing organizations and sustainable, scalable business

Higher customer engagement leads to increased revenues and profitability

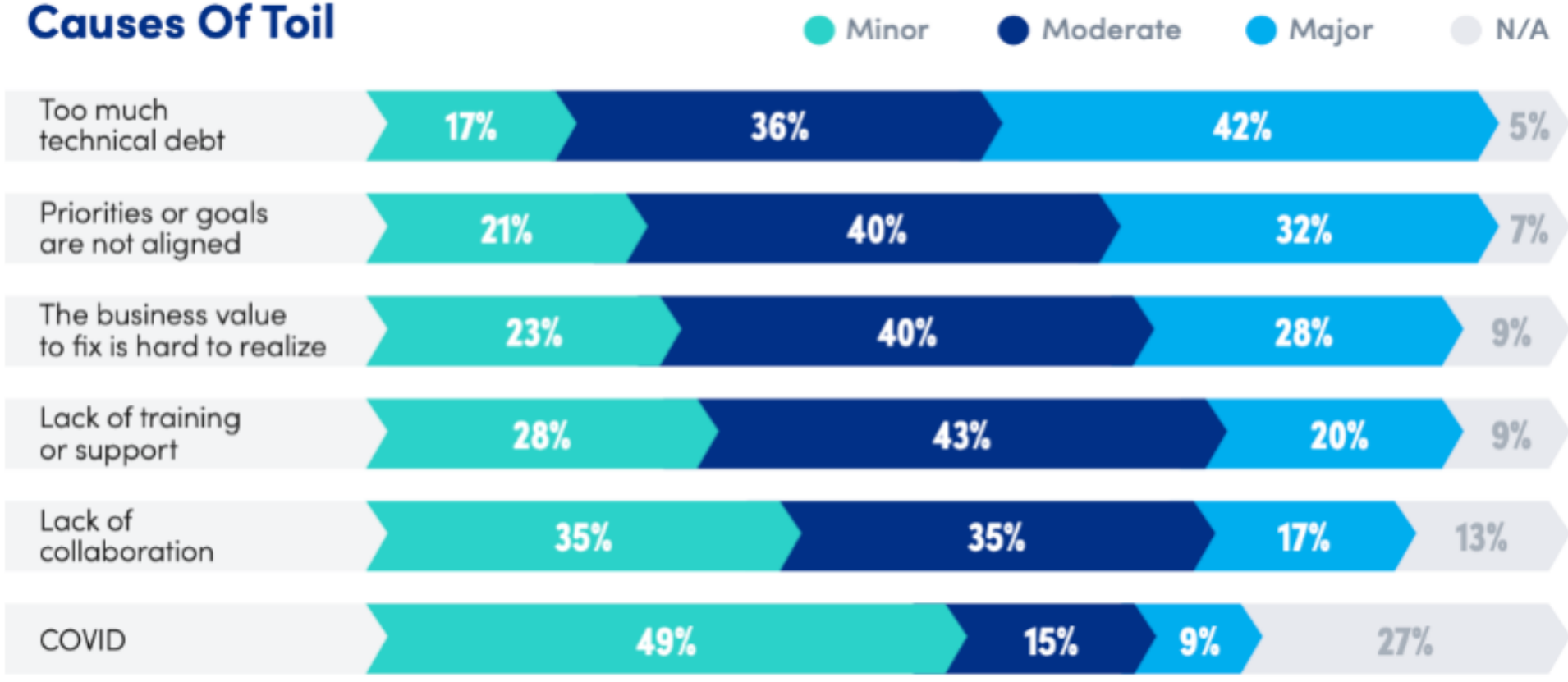
Sublime customer experience leads to more usage, more positive reviews and referrals

Toil and the wisdom of production

- Any manual, mandated operational task is bad
- If a task can be automated, then it should be automated
- Tasks can provide the "wisdom of production" that will inform better system design and behavior

SREs must have time to make tomorrow better than today

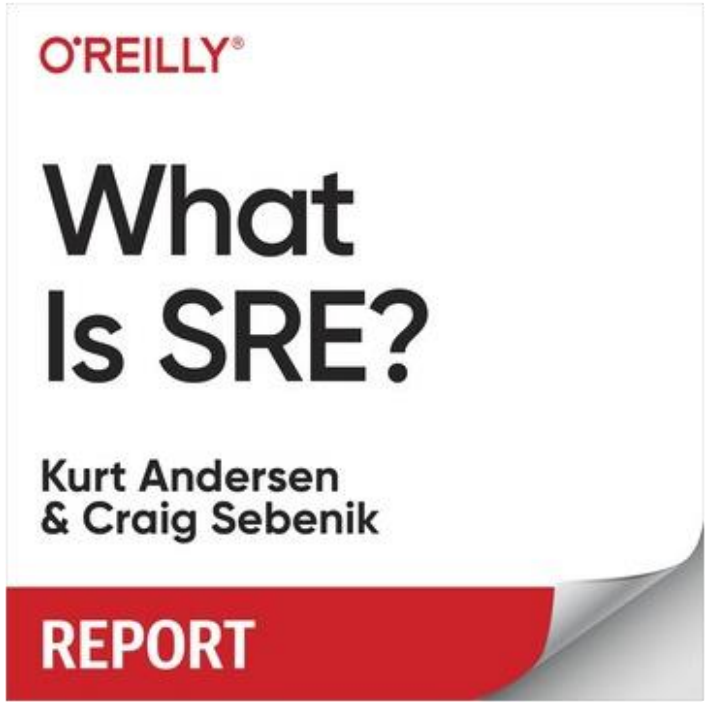
Causes Of Toil



Let’s ask Mark

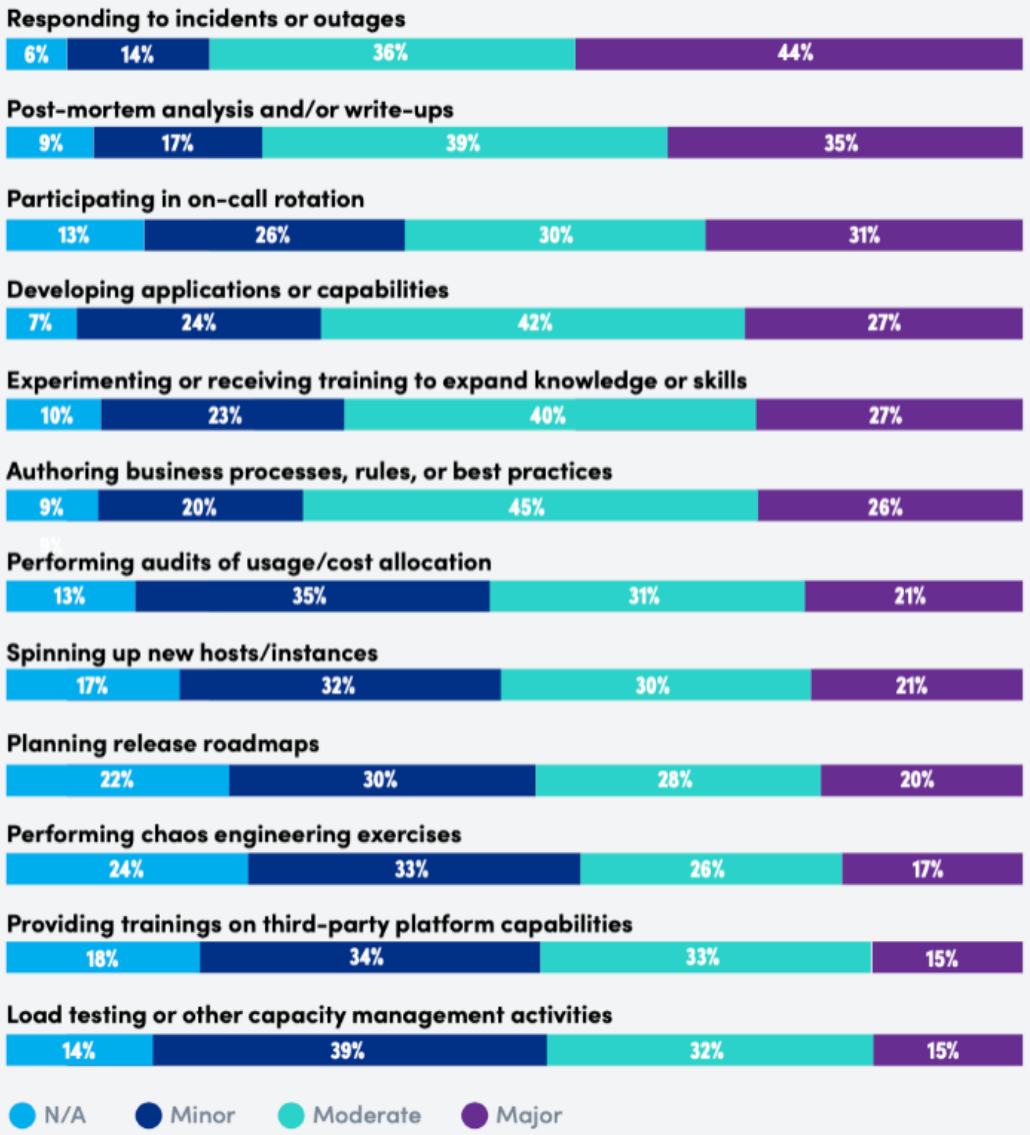
What do you see as the most common causes of toil?

Moving forward to SRE at Slack



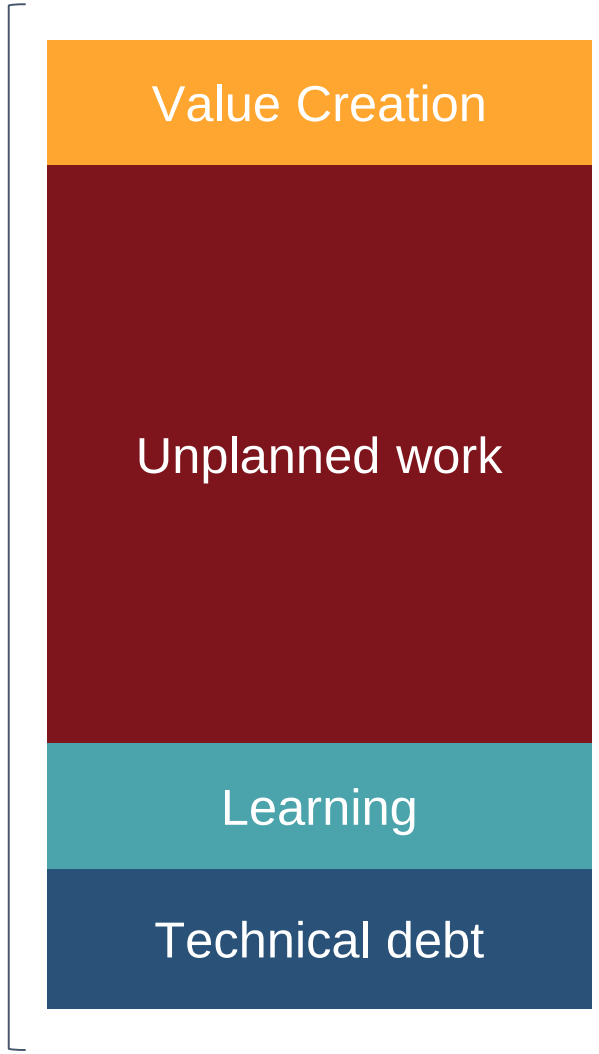
- Slack moved from 100 AWS instances to 15,000 instances over 4 years
- Excessive toil caused by low-quality, noisy alerting
- Ops teams were so consumed by interrupt-driven toil that they were unable to make progress on improving reliability
- Slack **explicitly committed to the importance of reliability over feature velocity**
- Operational ownership of services pushed back into the dev teams resulting in the teams making the code fixes necessary to stop the incident alerts

SRE Activity Breakdown

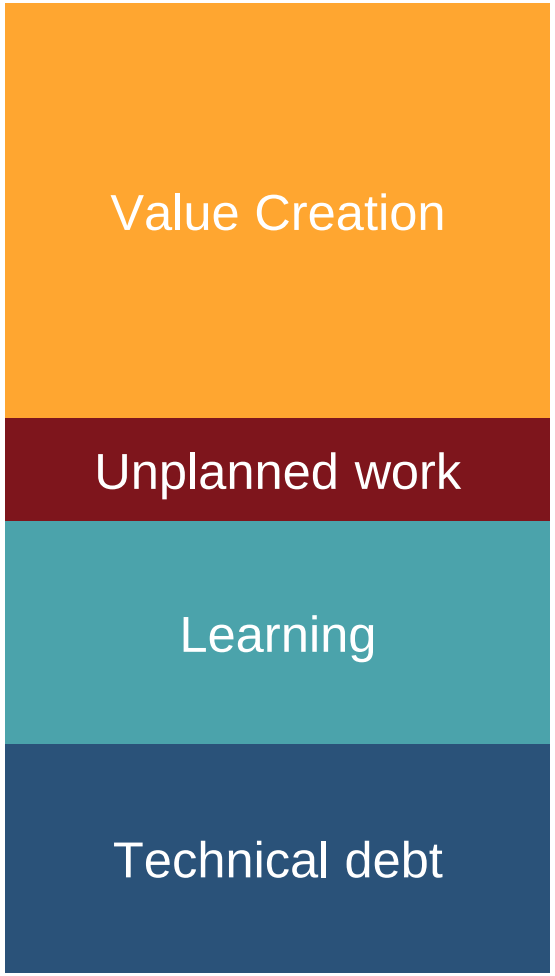


Reducing unplanned work and technical debt

What the team spends their time doing



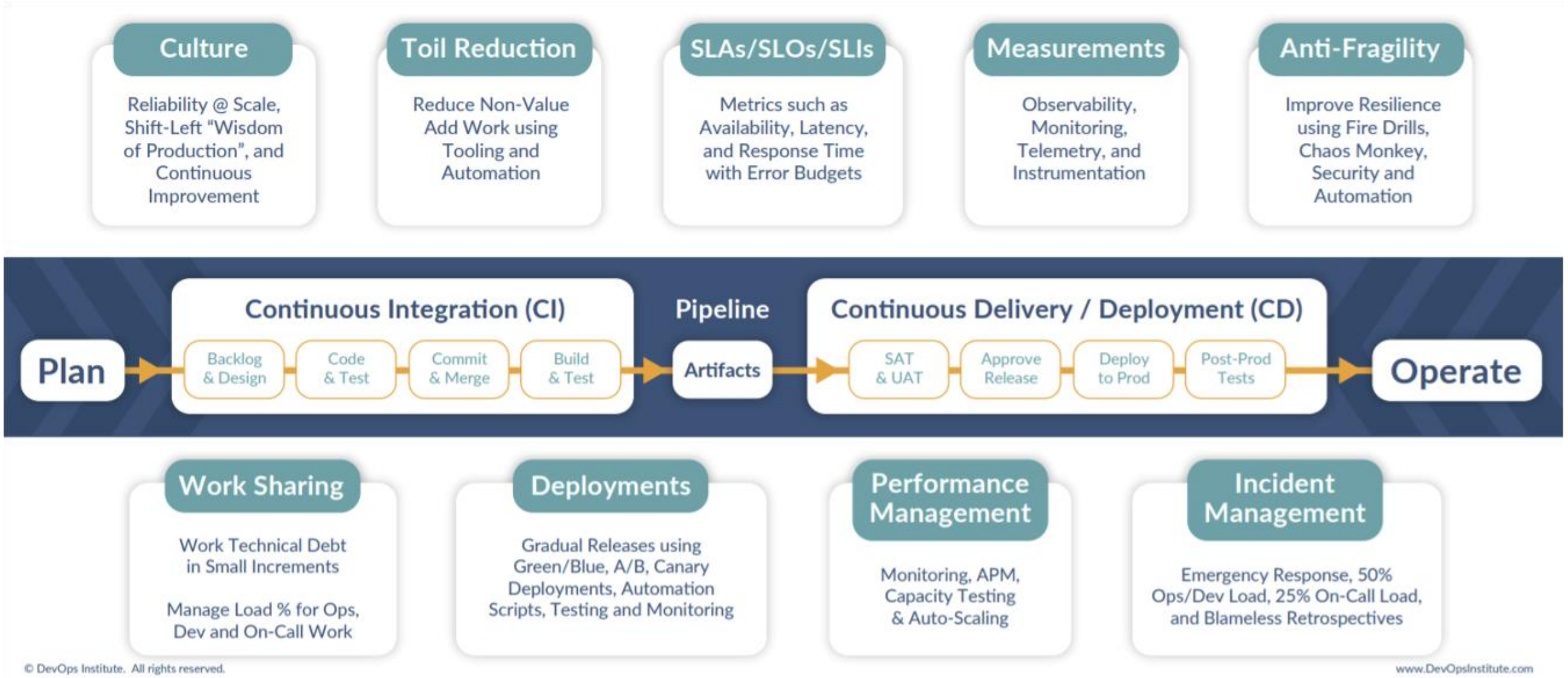
Without SRE



With SRE

How investing in SRE increases innovation

SRE principles and practices



Source: The SRE Blueprint @ DevOps Institute

SLAs, SLOs and SLIs

Service Level...

In SRE services are managed to the SLO

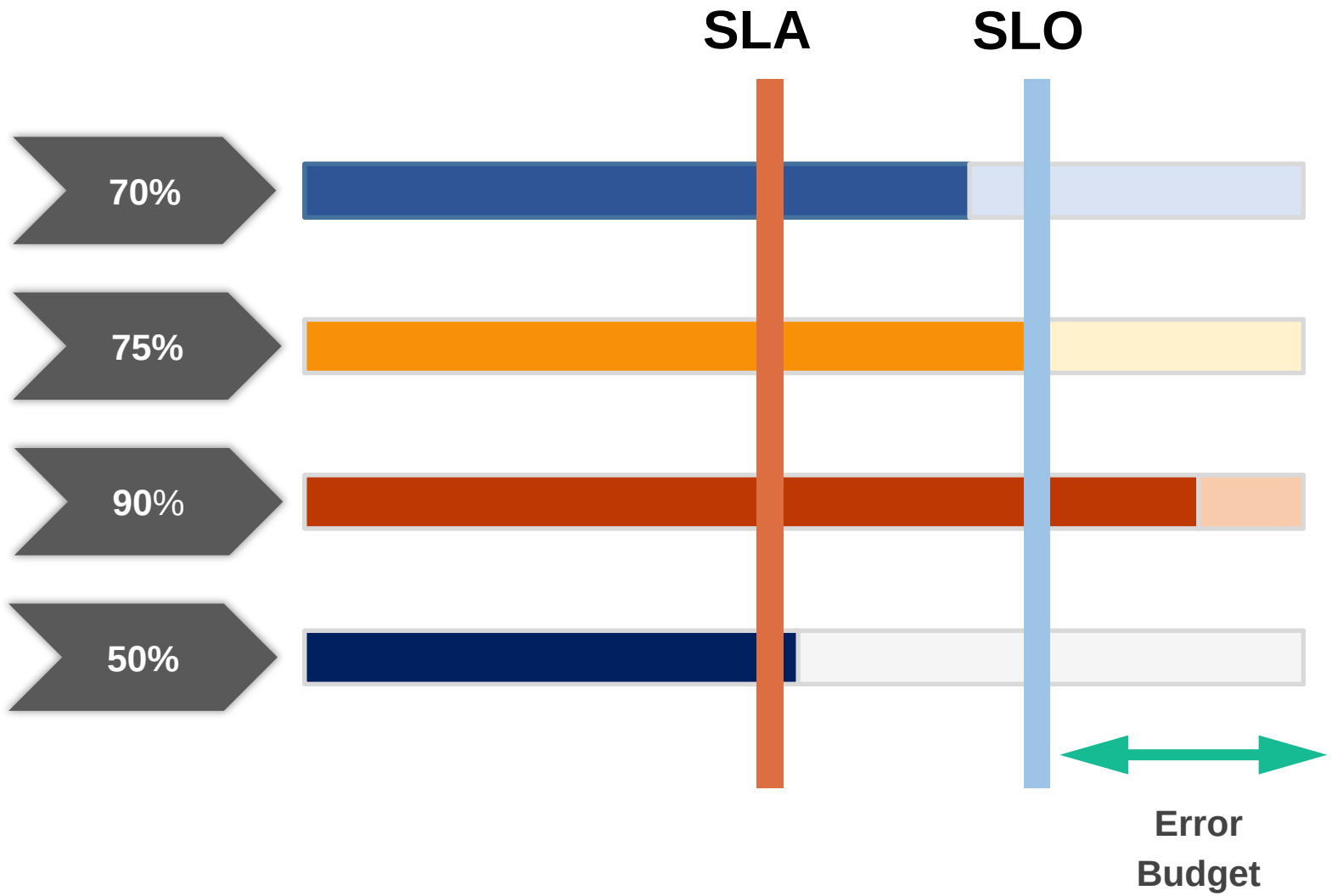
SLA	SLO	SLI
Agreement	Objective	Indicator
A business contract that comes into effect when your users are so unhappy you have to compensate them in some fashion	Specify a target level for the reliability of your service e.g., what the success rate should be 98% (it's never 100%)	An indicator of the level of service that you are providing e.g., http request success rate 99%

SLOs need consequences if they are violated

The VALET dimensions of SLO

	Dimension	SLO	Budget	Policy
V	Volume/traffic	Does the service handle the right volumes of data or traffic?	Budget: 99.99% of HTTP requests per month succeed with 200 OK	Address scalability issues
A	Availability	Is the service available to users when they need it?	Budget: 99.9% availability/uptime	Address downtime issues/outages, zero downtime deployments
L	Latency	Does the service deliver in a user-acceptable period of time?	Payload of 90% of HTTP responses returned in under 300ms	Address performance issues
E	Errors	Is the service delivering the capabilities being requested?	0.01% of HTTP requests return 4xx or 5xx status codes	Analyze and respond to main status codes, new functionality or infrastructure may be required
T	Tickets	Are our support services efficient?	75% of service tickets are automatically resolved	Automate more manual processes

Error budgets

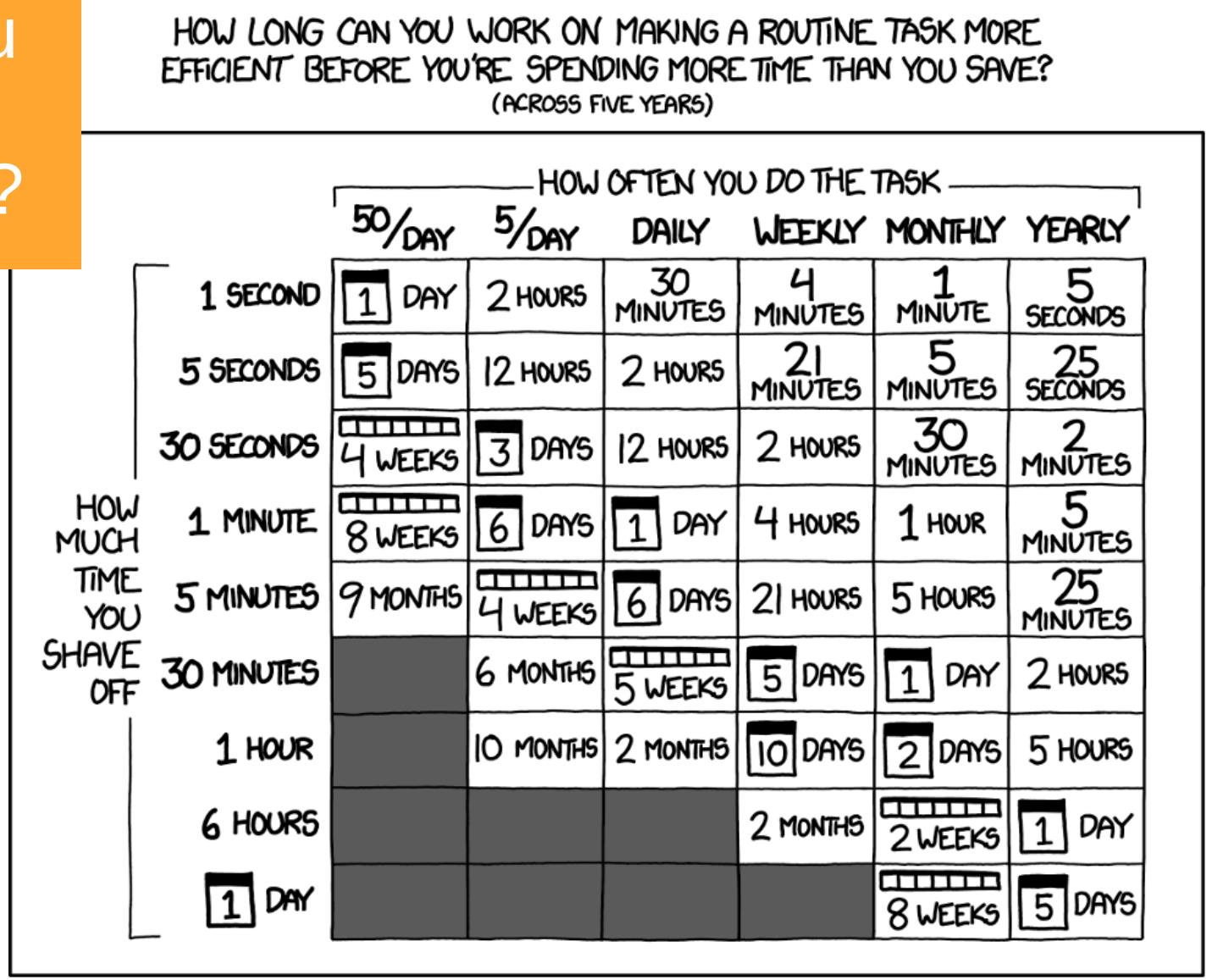


The SLO is a proxy for customer happiness

Error budgets by SLI and SLO

SLI [Metric identifier] [Operator] [Metric]	SLO [Objective] [SLI] [Period]	ERROR BUDGETS [Error Budget] [SLI]
Home page request served in **< 100 ms**	**95% of** home page requests served in < 100ms **over past 24 hours**	**Allow 5% failure** of home page requests served in < 100ms over past 24 hours
95th percentile of Home page latency over 5 mins **< 200ms**	**99% of** 95th percentile of home page latency over 5 mins < 200ms **for the past month**	**Allow 1% failure** of 95% percentile home latency over 5 minutes < 200ms for the past month
Requests should be completed **within 250 ms**	**95% of** requests should be completed within 250 ms **over 24 hours**	**Allow 5% failure** of requests should be completed within 250 ms over 24 hours
Services should be **available for 99.99% of time** (based on **heartbeat events** from bounded system)	95% of Services should be available for 99.99% of time **over 30 days**	**Allow 5% failure** of services availability over 30 days
Book page request response code **!= 5xx**	**99% of** book page request response code !=5xx over the **past 7 days**	**Allow for 1% failure** of book page request response code != 5xx over the last 7 days

Should you automate everything?

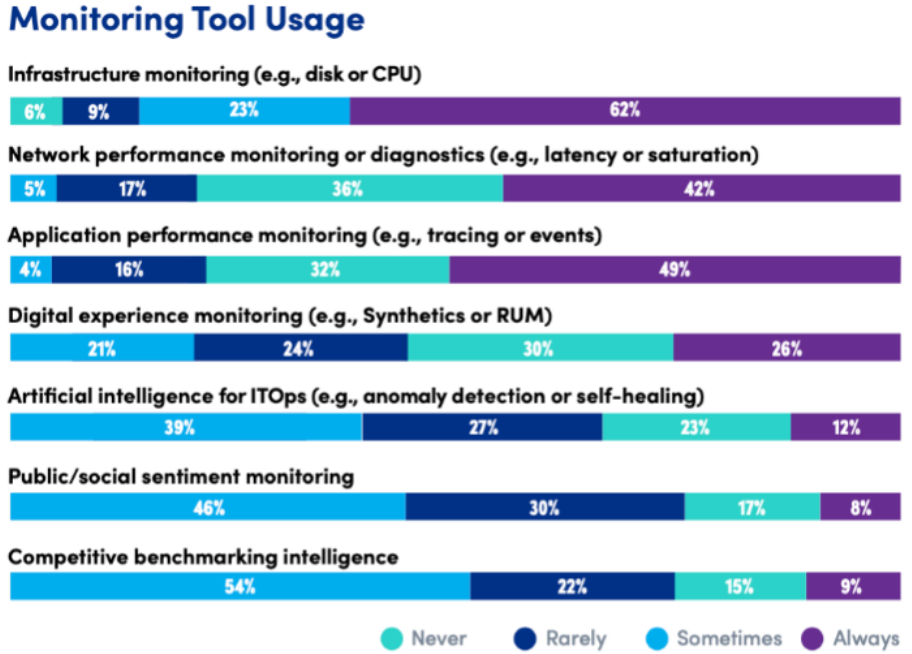


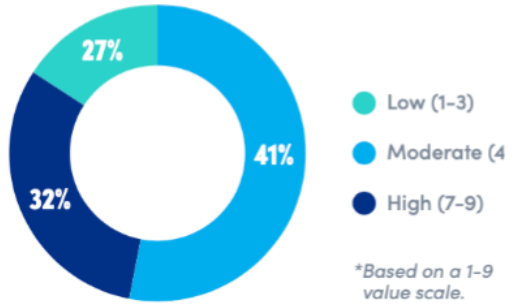
Let’s ask Mark

How do you recommend prioritization of automation opportunities?

SRE and Observability/Monitoring

Observability is a characteristic of systems; that they can be observed. It's closely related to a DevOps tenet: 'telemetry everywhere', meaning that anything we implement is emitting data about its activities. It requires intentional behavior during digital product and platform design and a conducive architecture. It's not monitoring. Monitoring is what we do when we observe our observable systems and the tools category that largely makes this possible.





SRE persona

How Observability Supports SRE's Goals

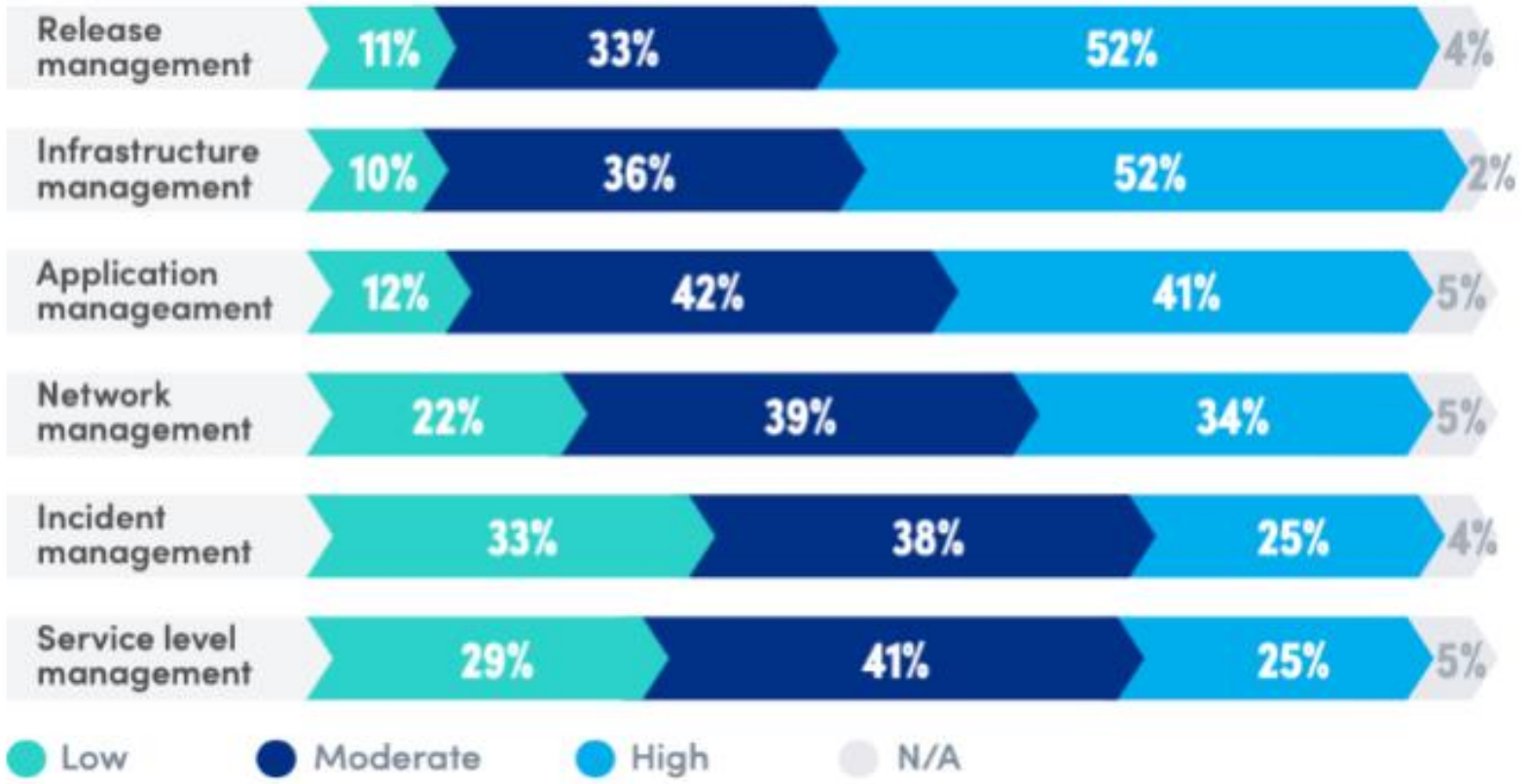
- Reducing the toil associated with incident management – particularly around cause analysis – improving uptime and MTTR
- Providing a platform for inspecting and adapting according to SLOs and ultimately improving teams' ability to meet them
- Offering a potential solution to improve when SLOs are not met, and error budgets are over-spent
- Relieving team cognitive load when dealing with vast amounts of data – reducing burnout
- Releasing humans and teams from toil, improving productivity, innovation and the flow and delivery of value
- Supporting multifunctional, autonomous teams and the "we build it, we own it" DevOps mantra
- Completing the value stream cycle by providing insights around value outcomes that can be fed back into the innovation phase

Monitoring Data Usage Drivers

	Minor	Moderate	Major	N/A
Augment troubleshooting and root cause analysis	8%	23%	66%	3%
Ensure service level objectives are met	14%	27%	51%	8%
Enhance the customer experience	10%	36%	49%	5%
Improve employee productivity	23%	38%	31%	8%
Provide analytics to our business teams	27%	35%	31%	7%
Accelerate our ability to innovate automation	29%	40%	25%	6%
Ability to support collaboration efforts	31%	34%	25%	10%

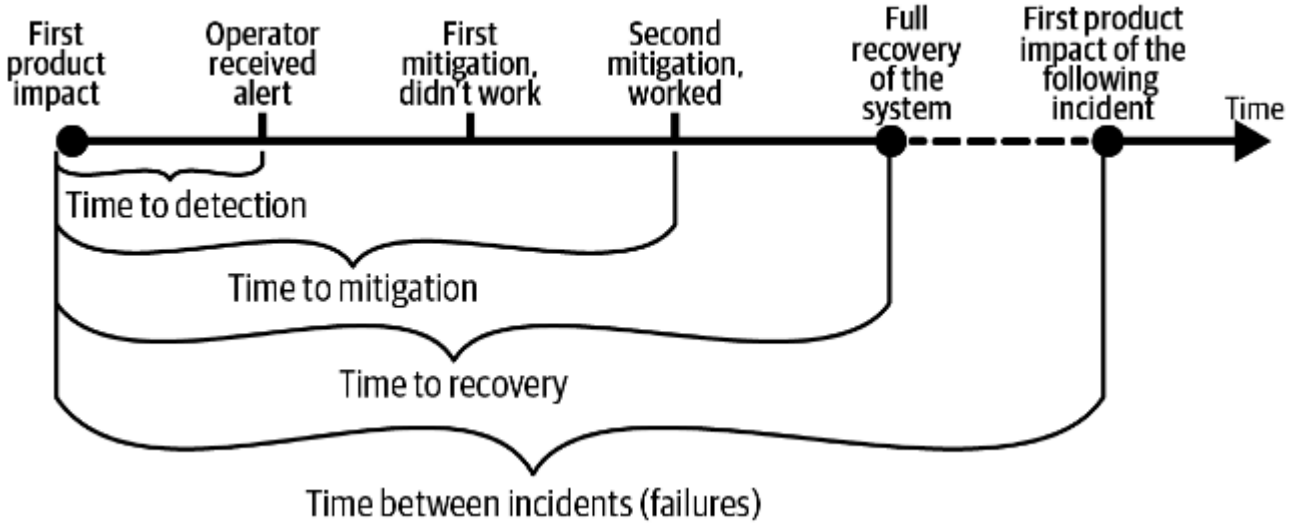
DevOps INSTITUTE

Use Case Automation Levels

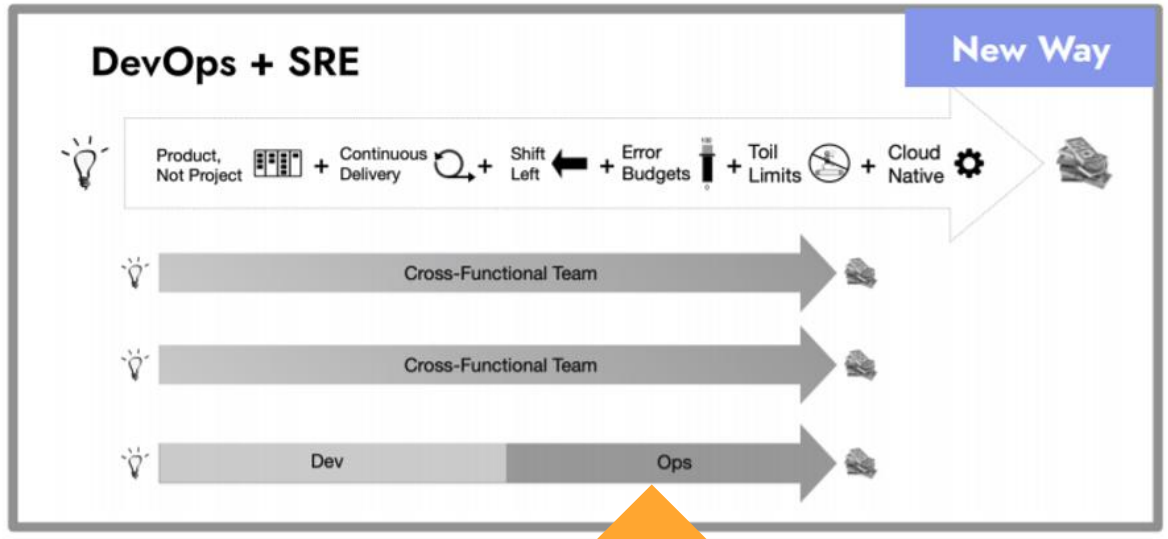


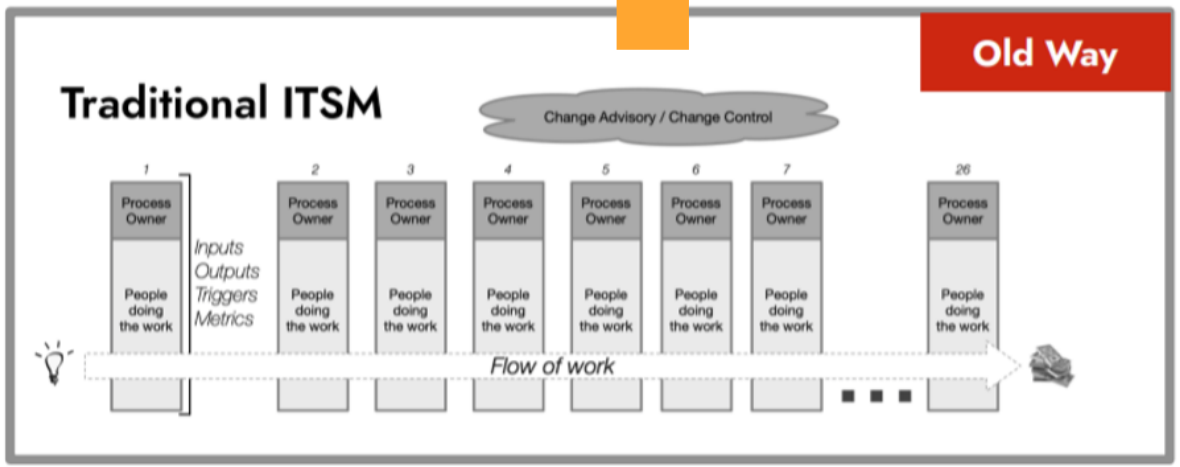
Good practices for Incident Management

One of the key responsibilities of SRE is to manage incidents of the production system(s) that they are responsible for. Within an incident, SREs contribute to debugging the system, choosing the right immediate mitigation, and organizing the incident response if it requires broader coordination

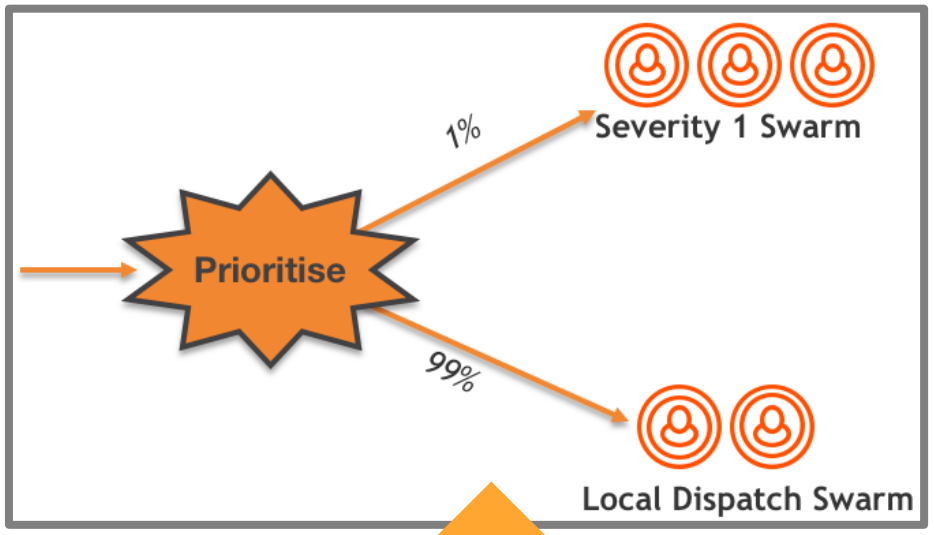


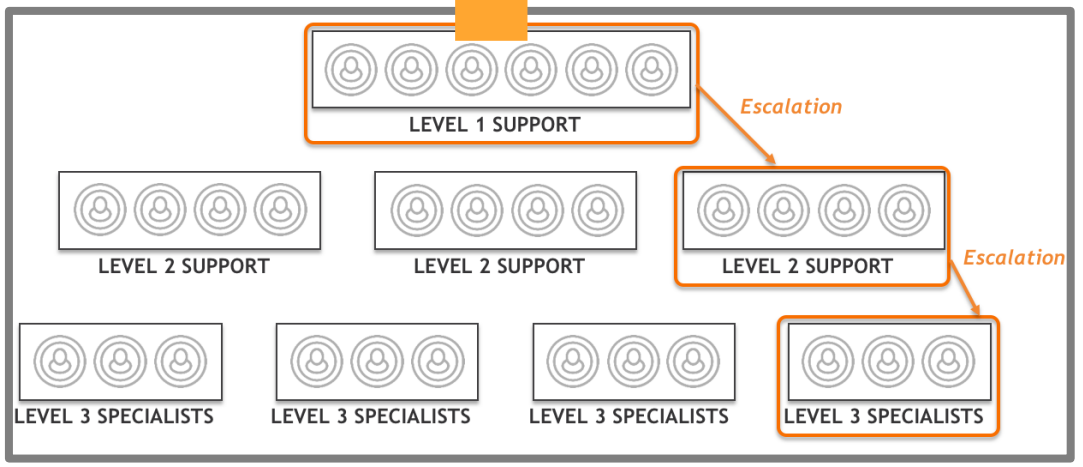
- Defect prevention
- Strategies for deploy/roll back/ roll forward (Feature Flags, Blue-Green, Canary)
- Auto remediation
- Reverting system





Source: Damon Edwards



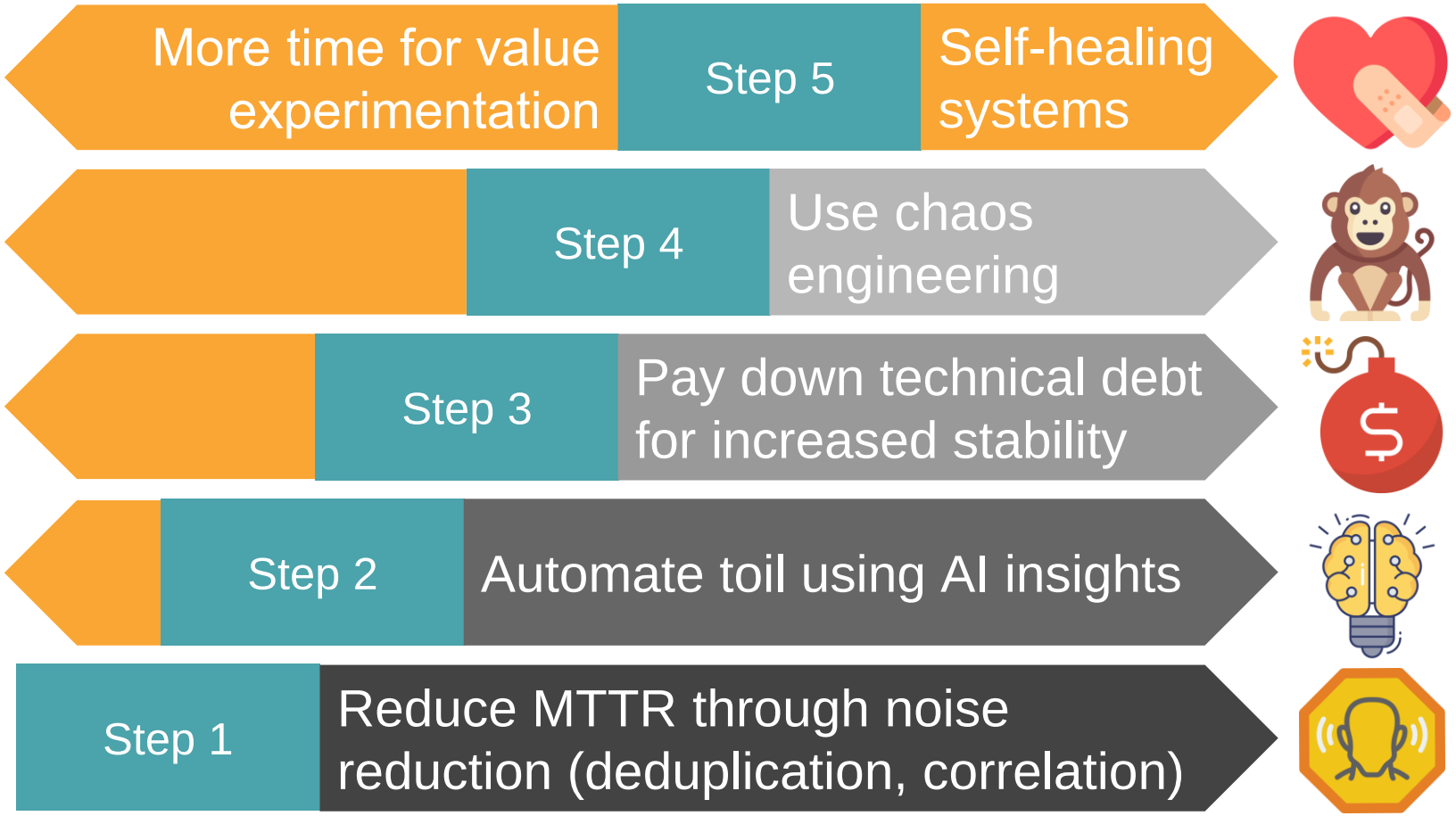


Source: Jon Stevens-Hall

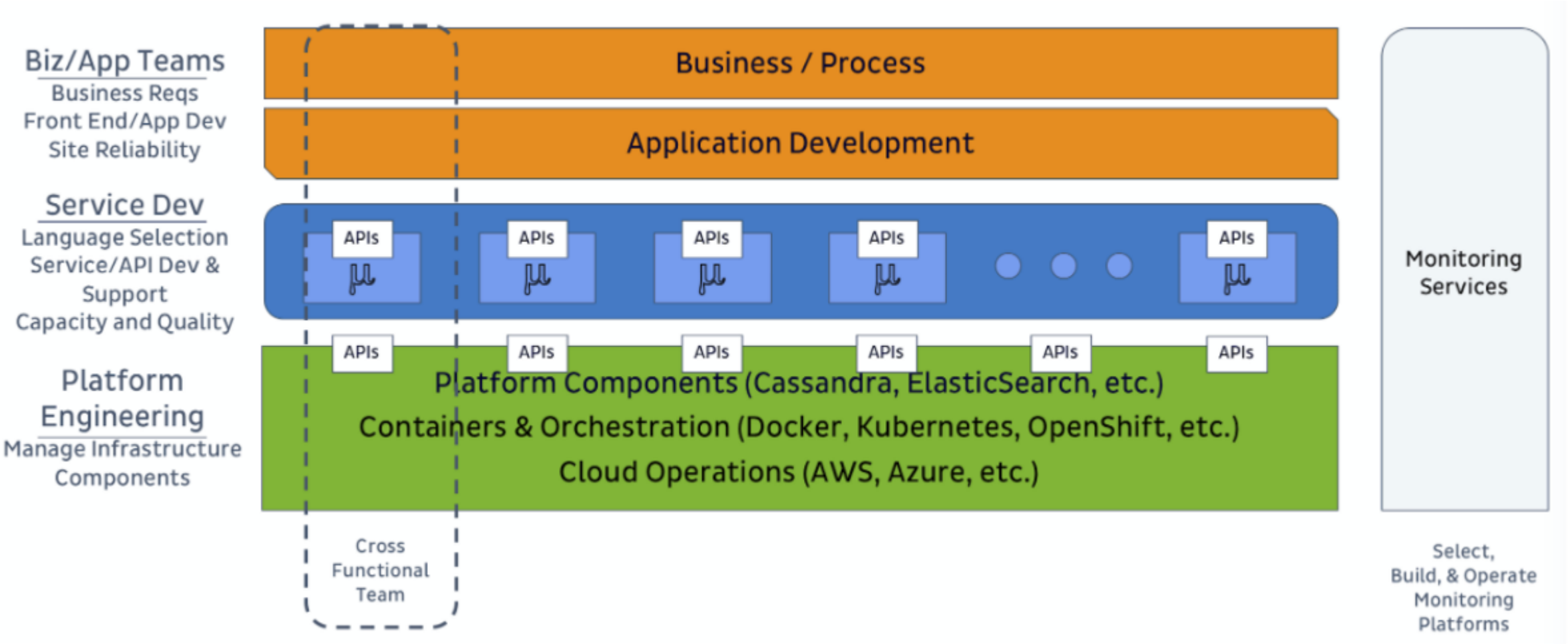
Let’s ask Mark

What’s your view on intelligent swarming?

IM, Observability, AIOps and the SRE



Platform SRE



Platform SRE ushers in self-service

- The Platform provides “self-service” provision of infrastructure, functionalities, configurations and environments that can be consumed by development teams , third parties e.g. distributed teams and partners
- Embedded governance, controls and standards are built-in
- End-to-end deployment automation, infrastructure playbooks of a service or application
- Abstraction of infrastructure specific implementations for multi/hybrid cloud through runbooks and playbooks
- In-Source code, products built by platform teams can be extended or enhanced by SRE/Dev/Ops or any other

Chaos Engineering

The discipline of experimenting on a distributed system in order to build confidence in the system's ability to withstand turbulent conditions.

Properties of a Chaos Experiment

- Define steady state
- Formulate hypothesis
- Outline methodology
- Identify blast radius
- Observability is key
- Readily abortable

Getting started with Chaos Engineering

From a technical point of view, they are easy to set up and do not have to be sophisticated in terms of implementation

- Get relevant people in a room who are responsible for a system or set of systems
- Shut off a component that the system should be robust enough to tolerate without it
- Record the learning outcomes and bring the component back online

Apply It: Build a Game Day event
What / Who / When / Where / How

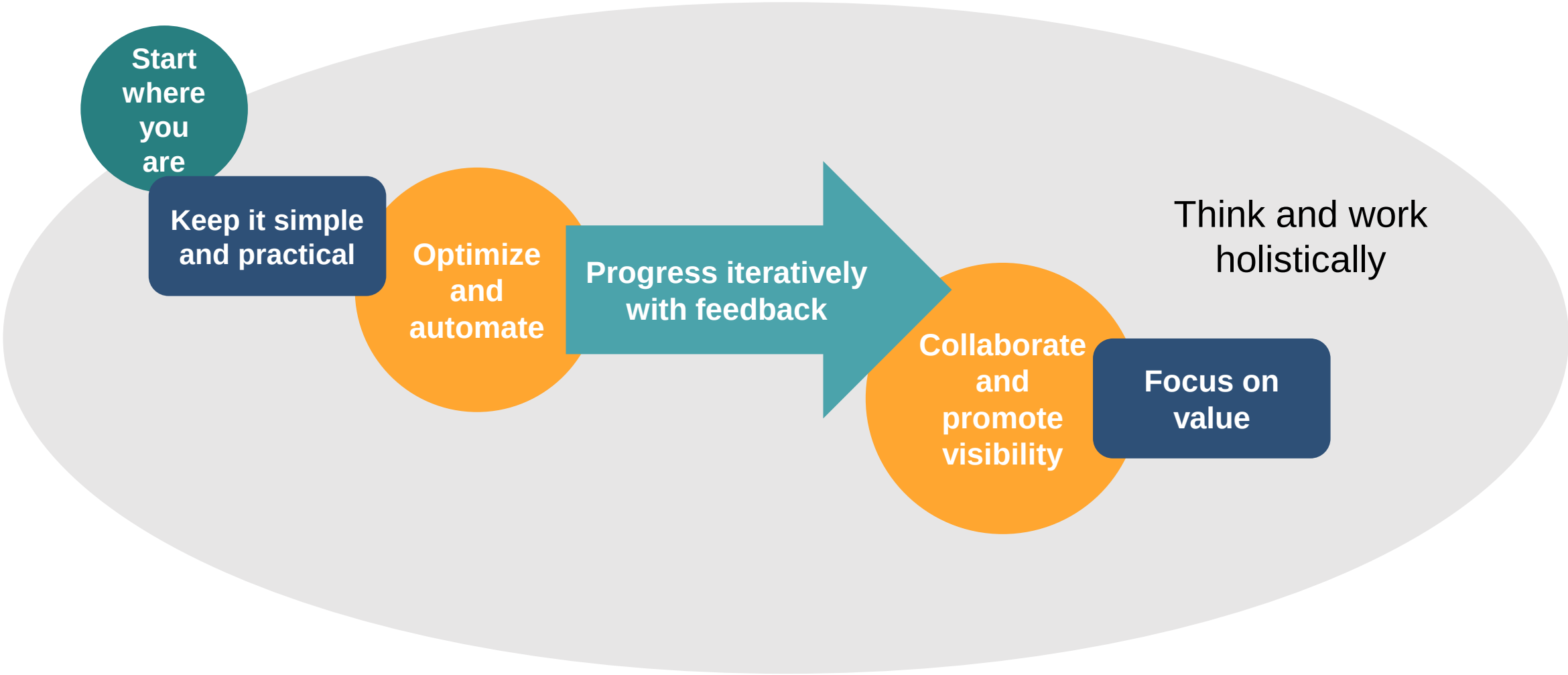
Note: You don't need to run your GameDay in production! Insights can come from conducting experiments first in a staging or test environment

Implementation challenges

And how to overcome them

You don't have enough cross-team usage or buy-in	Your difficult and dense process is slowing down incident response	Postmortems are underutilized and don't encompass in-depth learnings	You wait for incidents to happen	You stop at incident management without SLOs
Loop in sales, support and customer success	Customize lightweight, lean checklists	Automate: turn unstructured postmortems into a taxonomy with metadata & data	Practice chaos engineering	Automate: visualize your metrics

Where to start



DevOps Institute

ADVANCING THE HUMANS OF DEVOPS

THANK YOU!

And now over to Mark

aws marketplace

Mark Kriaf

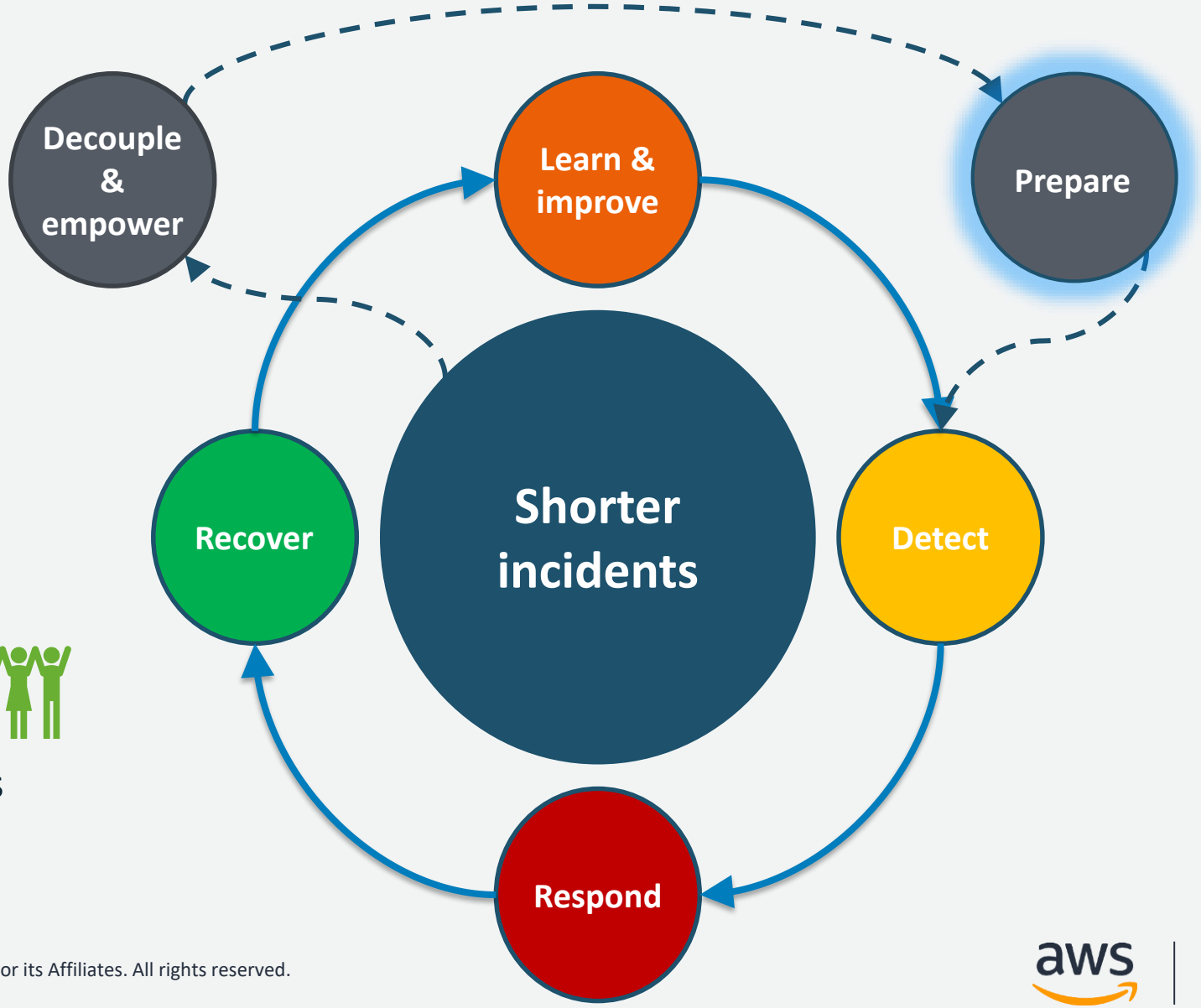
Partner Solutions Architect at AWS

markkriaf

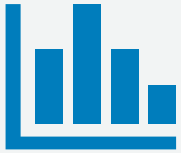
Agenda

- The uptime flywheel - prepare
- Prepare for downtime
- AWS Services for incident management
- The uptime flywheel – learn
- Chaos engineering
- AWS Fault Injection Simulator
- Summary and Marketplace next steps

The uptime flywheel @ AWS



“2-pizza” teams



Weekly OpsMetrics meeting

aws | aws marketplace

Prepare for downtime

1. Detect
    - What could go wrong?
    - How would I know?
2. Respond and recover
    - Who needs to be engaged?
    - What do they need to do to diagnose?
        - Procedures and scripts
        - Where do we collaborate?
3. Learn and improve
    - How did we respond?
    - What actions will we take?

aws | aws marketplace

AWS Systems Manager

Centralize operational data from multiple AWS services and automate tasks across your AWS resources

Benefits

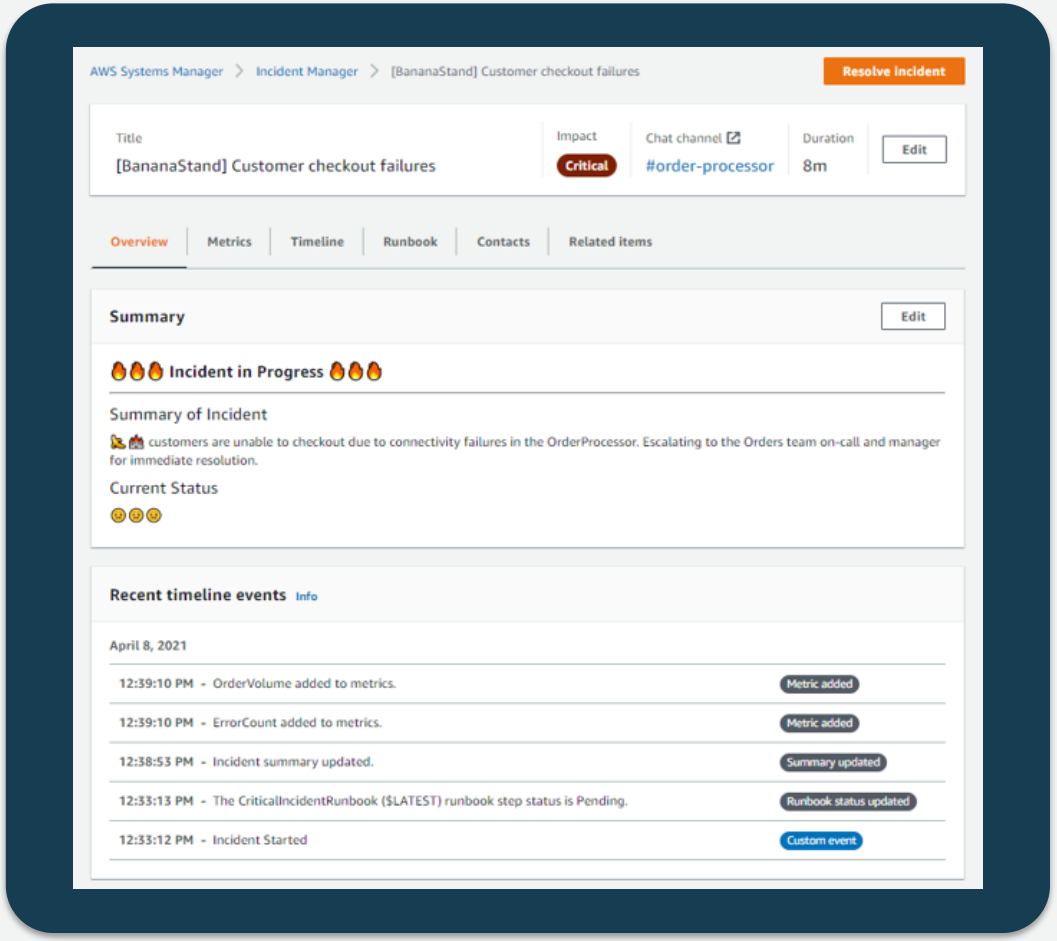
- Simplify resources and application management
- Easy to operate and securely manage multi-cloud infrastructures at scale
- Resolve critical application availability and performance
- Prepare for and manage incidents efficiently with automated response

aws | aws marketplace

Incident Manager

Resolve application issues faster with automated response plans

- Specify a response plan to critical application alarms, including who to engage, what runbook to follow, and where to collaborate
- Notify the right people immediately with SMS, voice, and escalations (additional partner integrations coming soon)
- Single console to track incidents from detection to mitigation and post-incident analysis, including timeline, runbooks, metrics, etc.
- Collaborate in Slack via AWS Chatbot to resolve incidents
- Identify post-incident action items, such as improving alarms or automating runbooks steps, using Amazon's post-incident analysis template and track them in OpsCenter



aws | aws marketplace

Overview | Metrics | Timeline | Runbook | Engagements | Related items | Properties

Runbook Info

Cancel

Runbook
BananaStand-MajorIncident (v1)

Execution details
Execution details: BananaStand-MajorIncident

Status
Waiting

00:00:00 (UTC-7:00)

13:23:28 Triage — Waiting

Determine customer impact

- View the **Metrics** tab of the incident or navigate to your CloudWatch Dashboards to find key performance indicators (KPIs) that show the extent of customer impact.
- Use CloudWatch Synthetics and Contributor Insights to identify real-time failures in customer workflows.

Communicate customer impact

Update the following fields to accurately describe the incident:

- **Title -** The title should be quickly recognizable by the team and specific to the particular incident.
- **Summary -** The summary should contain the most important and up-to-date information to quickly onboard new responders to the incident.
- **Impact -** Select one of the following impact ratings to describe the incident:
  - 1 – Critical impact, full application failure that impacts many to all customers.
  - 2 – High impact, partial application failure with impact to many customers.
  - 3 – Medium impact, the application is providing reduced service to many customers.
  - 4 – Low impact, the application is providing reduced service to few customers.
  - 5 – No impact, customers are not currently impacted but urgent action is needed to avoid impact.

Resume

--:--:-- Diagnosis — Pending

Rollback

- Look for recent changes to the production environment that might have caused the incident. Engage the responsible team using the **Contacts** tab of the incident.
- Rollback these changes if possible.

Locate failures

- Review metrics and alarms related to your Application. Add any related metrics and alarms to the **Metrics** tab of the incident.
- Use CloudWatch ServiceLens to troubleshoot issues across multiple services.
- Investigate the possibility of ongoing incidents across your organization. Check for known incidents and issues in AWS using Personal Health Dashboard. Add related links to the **Related Items** tab of the incident.
- Avoid going too deep in diagnosing the failure and focus on how to mitigate the customer impact. Update the **Timeline** tab of the incident when a possible diagnosis is identified.

--:--:-- Diagnosis — Pending

Rollback

- Look for recent changes to the production environment that might have caused the incident. Engage the responsible team using the **Contacts** tab of the incident.
- Rollback these changes if possible.

Locate failures

- Review metrics and alarms related to your Application. Add any related metrics and alarms to the **Metrics** tab of the incident.
- Use CloudWatch ServiceLens to troubleshoot issues across multiple services.
- Investigate the possibility of ongoing incidents across your organization. Check for known incidents and issues in AWS using Personal Health Dashboard. Add related links to the **Related Items** tab of the incident.
- Avoid going too deep in diagnosing the failure and focus on how to mitigate the customer impact. Update the **Timeline** tab of the incident when a possible diagnosis is identified.

--:--:-- Mitigation — Pending

Collaborate

- Communicate any changes or important information from the previous step to the members of the associated chat channel for this incident. Ask for input on possible ways to mitigate customer impact.
- Engage additional contacts or teams using their escalation plan from the **Contacts** tab.
- If necessary, prepare an emergency change request in Change Manager.

Implement mitigation

- Consider re-routing customer traffic or throttling incoming requests to reduce customer impact.
- Look for common runbooks in Automation or run commands using Run Command.
- Update the **Timeline** tab of the incident when a possible mitigation is identified. If needed, review the mitigation with others in the associated chat channel before proceeding.

--:--:-- Recovery — Pending

Monitor customer impact

- View the **Metrics** tab of the incident to monitor for recovery of your key performance indicators (KPIs).
- Update the **Impact** field in the incident when customer impact has been reduced or resolved.

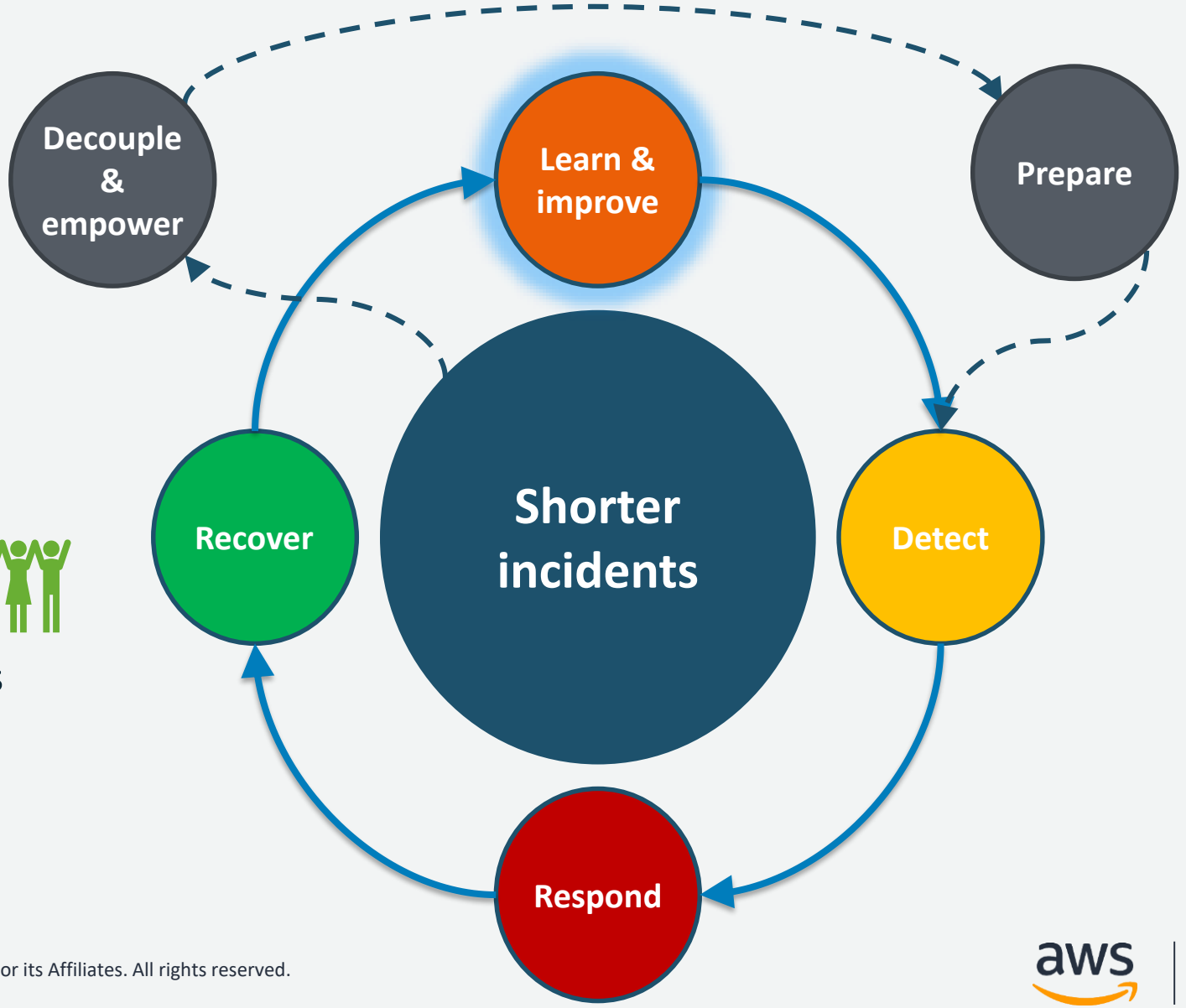
Identify action items

- Add entries in the **Timeline** tab of the incident to record key decisions and actions taken, including temporary mitigations that might have been implemented.
- Create a **Post-Incident Analysis** when the incident is closed in order to identify and track action items in OpsCenter.

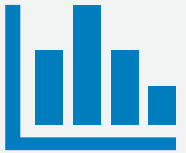
23:59:59

aws | aws marketplace

The uptime flywheel @ AWS



"2-pizza" teams



Weekly OpsMetrics meeting

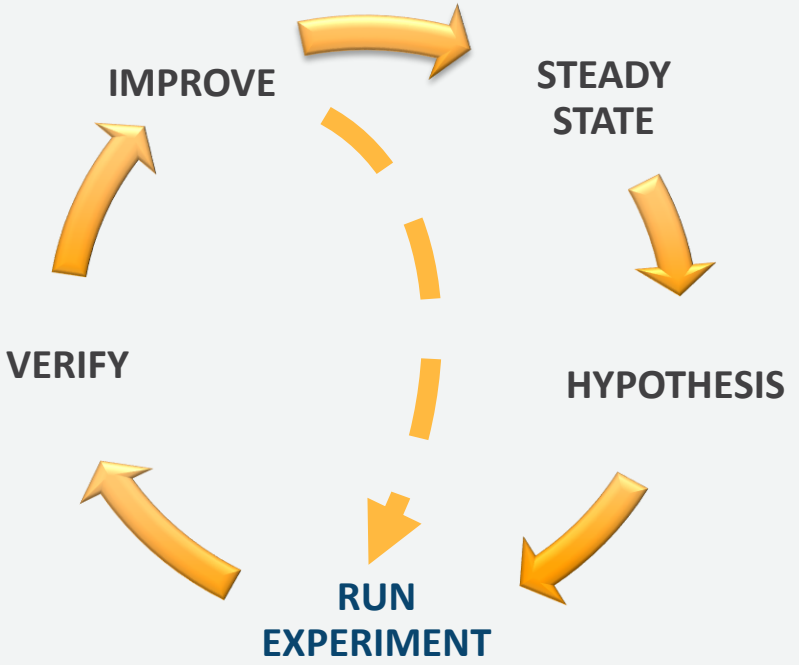
aws | aws marketplace

Chaos engineering

Experiment to ensure that the impact of failures is mitigated

Chaos experiment

- Inject events that simulate
    - Hardware failures, like servers dying
    - Software failures, like malformed responses
    - Nonfailure events, like spikes in traffic or scaling events
    - Any event capable of disrupting steady state

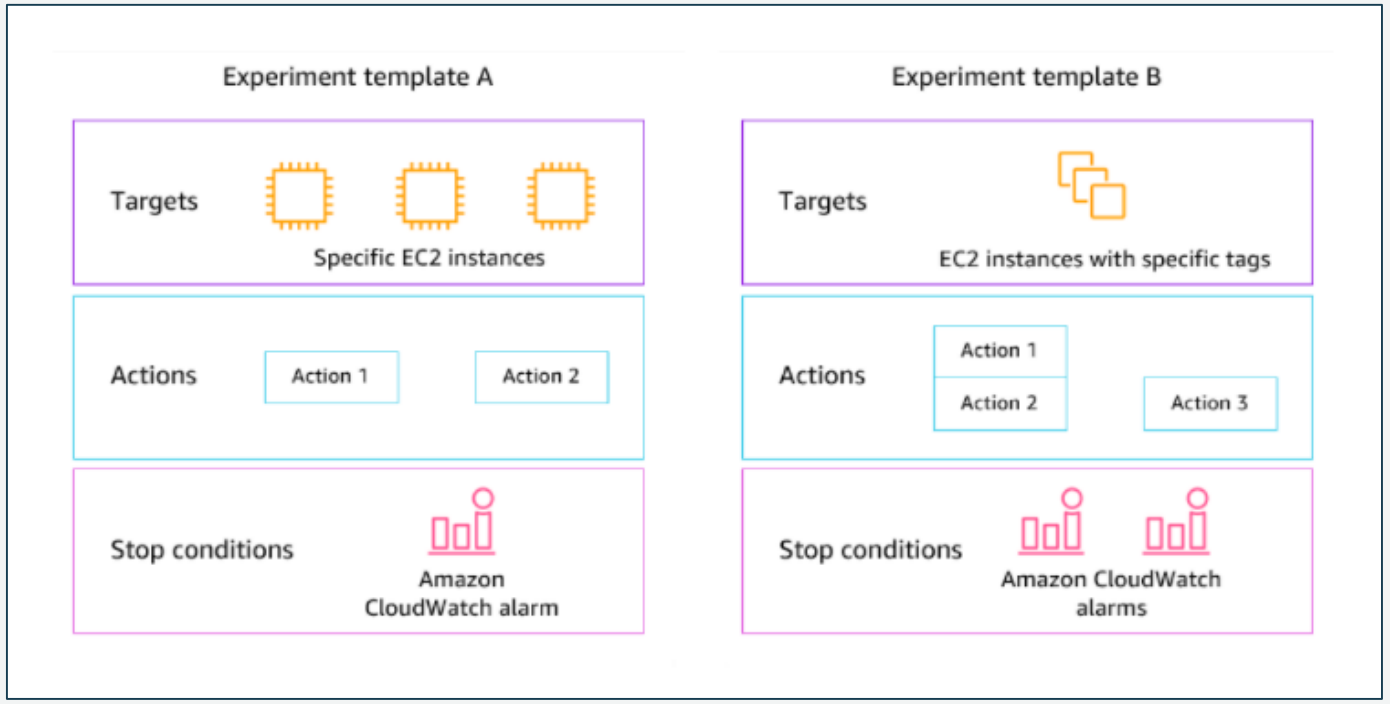


aws | aws marketplace

AWS Fault Injection Simulator

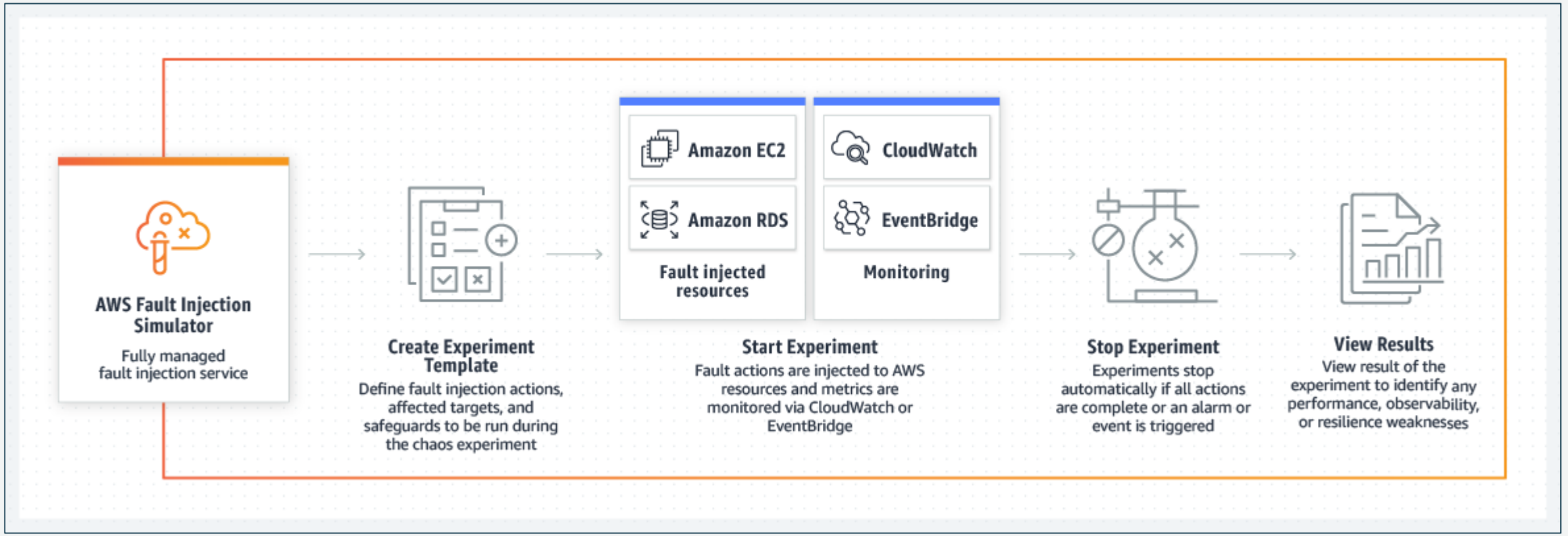
Improve resiliency and performance with controlled experiments

- A fast and easy way to get started with fault injection experiments
- Validate how your application performs on AWS
- Safeguard fault injection experiments
- Improve application performance, resiliency, and observability
- Get comprehensive insights by generating real-world failure conditions



aws | aws marketplace

AWS Fault Injection Simulator



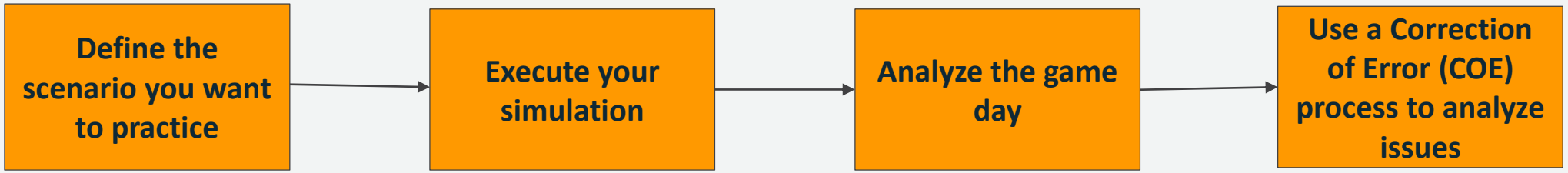
aws | aws marketplace

AWS Fault Injection Simulator use case: periodic game days

Why conduct a game day?

- Simulate a failure or event to test systems, processes, and team responses
- Should cover the areas of operations, security, reliability, performance, and cost
- Can be carried out with replicas of your production environment using AWS CloudFormation
- Should involve all personnel who normally operate a workload

Game day process

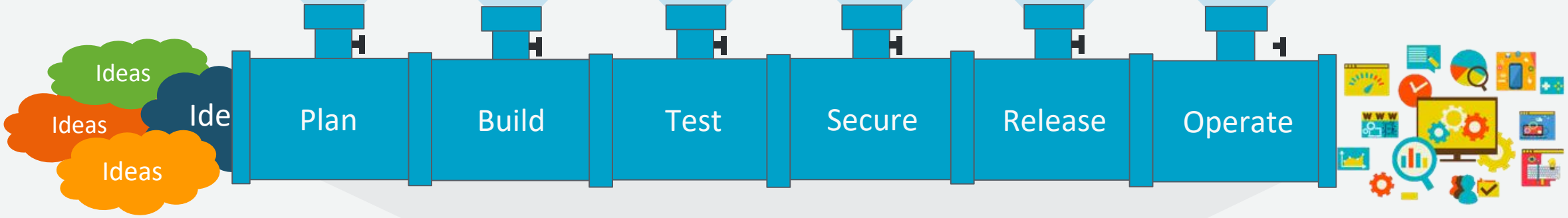


aws | aws marketplace

AWS Marketplace: Destination for third-party solutions to use with AWS

DevOps Core practices

- Collaboration & communication
- Microservices and everything-as-code
- Continuous integration
- Testing & quality management
- Continuous delivery
- Security & compliance
- Monitoring & observability
- Incident management



Sample AWS Marketplace solution providers

Gremlin · PagerDuty · moogsoft · bigpanda

JFrog · TRICENTIS · LaunchDarkly · MICRO FOCUS · splunk> · HashiCorp · CHEF · New Relic. · CloudBees.

1,600+ vendors | **8,000+** products

aws | aws marketplace

aws marketplace

8,000+ listings • **1,600+** ISVs • **24** regions • **290,000+** customers • **1.5M+** subscriptions

AWS Marketplace DevOps Workshop Series participating partner hands-on labs

GitLab, JFrog, circleci, harness, CloudBees., armory, puppet, Quali, TRICENTIS, MICRO FOCUS, sumo logic, LaunchDarkly, GENYMOTION, TREND MICRO, PagerDuty, Gremlin, New Relic., Gatling, dynatrace, APPDYNAMICS, CONTRAST SECURITY, CORESTACK

And more coming soon!

aws | aws marketplace

How can you get started?

Find

A breadth of DevOps solutions:

perspectium

dynatrace

PagerDuty

Gremlin

APPDYNAMICS part of Cisco

New Relic.

moogsoft

splunk>

sumo logic

THUNDRA

lumigo

bigpanda

Buy

Through flexible pricing options:

Free trial

Pay-as-you-go

Budget alignment

Bring Your Own License (BYOL)

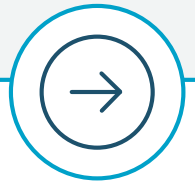
Private Offers

Billing consolidation

Enterprise Discount Program

Private Marketplace

Deploy



With multiple deployment options:

SaaS

Amazon Machine Image (AMI)

CloudFormation Template

Containers

Amazon EKS/ Amazon ECS

AI / ML models

AWS Data Exchange

AWS Control Tower

aws | aws marketplace

Next steps

- Bookmark the Workshop Series landing page, check back for new content or subscribe to email updates
- Start your lab of choice
- Move on to Module 8: DevSecOps
- Visit the AWS Marketplace website to experiment with DevOps tooling

aws | aws marketplace

Module 7 Hands-on Labs

DevOps Workshop Series

Module 7: SRE and Incident Management Hands-on Labs

Learn from expert presenters and get hands on experience with some of the best tools in DevOps.

Step 1: Watch the workshop presentation

Thanks for registering! You will receive a confirmation email with your link to the presentation live on October 13th and soon afterwards the recording and slides will be available on this page.

Step 2: Get hands on experience with AWS and some of the best tools in DevOps

Watch a quick demo to see which tool you'd like to get started with then complete a tutorial at your own pace. You'll recieve a sharable completion badge for each tool you master! Labs will be available on the date of the presentation.

PagerDuty

Application performance monitoring solution that offers real-time monitoring across on-premises and AWS environments.

Lab coming soon>

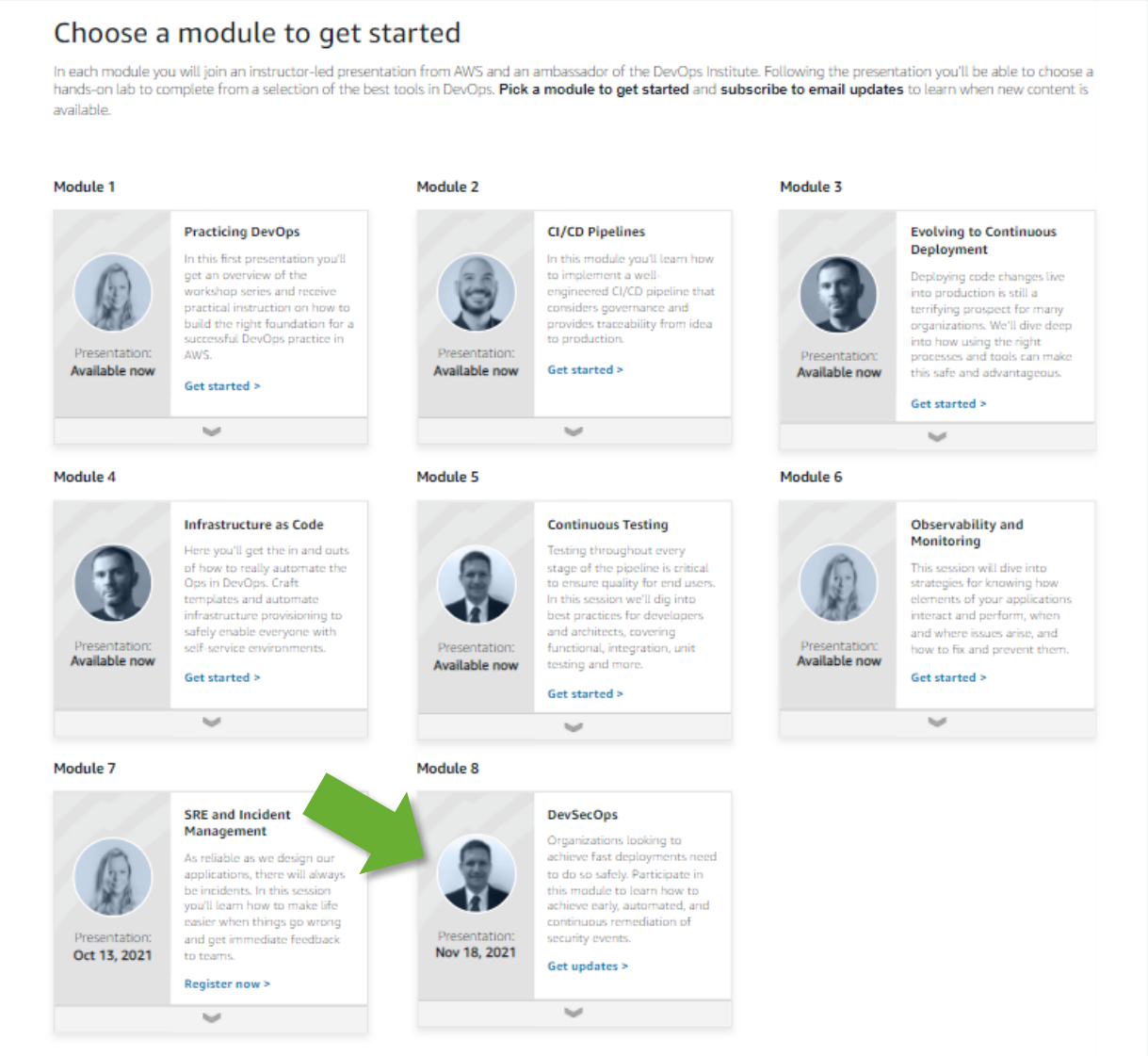
Gremlin

Tests the reliability of cloud infrastructure and applications by running experiments against hosts, containers, functions, or Kubernetes.

Start your hands-on lab >

aws | aws marketplace

Move on to Module 8: DevSecOps



https://pages.awscloud.com/awsmp-h2-dev-aws-marketplace-devops-workshop-series.html

aws | aws marketplace

THANK YOU!