Keeping always-on services always on

How organizations can build a DevOps culture that supports always-on services

KRISHNA SAI
Head of Engineering, IT Solutions

The nature of always-on services requires continuous response from agile and DevOps teams. These teams need to think beyond reacting to a single incident and align the team structure, values, and tools to ensure that operational excellence becomes a core competency.

Today's users expect modern services to be always-on and always available. Downtime can be detrimental, causing damage to reputation and bottom line, with the average costs of downtime as high as $9,000 per minute.

In a cloud-native world, however, incidents are as much a fact of life as bugs in code. Incidents that result in downtime range from hardware and network failures to misconfiguration, resource exhaustion, data inconsistencies, and software bugs.

Always-on services require teams to think beyond reacting to a single incident and align the team structure, values, and tools to ensure that operational excellence becomes a core competency. It entails adopting the practice of you build it, you run it (YBIYRI), where the ownership of building, testing, deploying and operating a service rests with the development team. This concept puts DevOps theory into practice, and reinforces the cycle of continuous deployment, feedback, and maintenance or incident response that teams need to keep always-on services, always on.

Challenges of always-on services

Since it was first discussed 14 years ago, YBIYRI is still challenging modern development teams to live up to its promise of speeding up time-to-resolution and scaling operational best practices. Unfortunately, many teams still frame up their skills, schedules, and processes as a reaction to an incident, instead of a foundation for long-term success.

Teams often move to a YBIYRI culture without adequate preparation and the first major incident often ends up being a wake-up call. However, the reaction is often triggered by the sentiment, "we can't let incidents happen again". In an attempt to accomplish this, safety gates, checkpoints, and other procedural overhead are introduced. Also, change review boards and weekly release reviews are made part of the team rituals. Every change is scrutinized carefully in an attempt to prevent outages. While this

RELATED MATERIAL

Get started for free

Learn more →

RELATED MATERIAL

carefully in an attempt to prevent outages. While this often results in reduced incidents, it can slow development velocity and product momentum. This can become a competitive weakness as more nimble competitors can move much faster.

Learn about the benefits of DevOps

See article →

Team best practices for always-on services

Operational readiness

One of the critical shifts for YBIYRI teams is to include operational readiness as part of the sprint planning and execution cycles. Operational readiness may include:

- During development, building appropriate, high-quality alerts in the code that minimize mean time to detect (MTTD) and mean time to isolate (MTTI)
- Building monitors -- including synthetic monitors when appropriate -- to ensure dependent services operate as expected
- Allocate time to build required dashboards and train all team members to use them
- Ensure on-call team members don't have other development commitments during a sprint
- Plan "war games" for the service to ensure rollbacks operate as expected
- Plan for bandwidth in sprints to close actions from previous incident reviews
- Address security (upgrades/patches/rolling credentials) and operational issues as part of sprint cycles

These all require product owners to understand the service level objectives (SLO) and prioritize them appropriately, along with business commitments related to feature development and functionality.

Embrace incident values

Embracing incident values at the team level can create a strong foundation for a team's YBIYRI journey. Incident values guide your team in incident response. These values ensure there is a strong foundation for a sustainable culture around building and operating an always-on service. Incident values are designed to:

- Guide autonomous decision-making by people and teams in incidents and postmortems
- Build a consistent team culture that includes how to identify, manage, and learn from incidents
- Align teams as to what attitude they should bring to each part of incident identification, resolution, and reflection

An Incident Values playbook provides an excellent guide to help identify team values during incident response and to create a plan to live out those values consistently. It can help if your team struggles with customer centricity, team cohesiveness, shared understanding, service levels, or service mandates on your Health Monitor.

At Atlassian, we embrace the following incident values at the team level:

	Atlassian Value	Stage & Incident Value	Rationale
	Build with heart and balance	**Detect** Atlassian knows before our customers do	A balanced service includes effective monitoring and alerting to detect incidents before our customers do. The best monitoring alerts us to problems before they become incidents.
	Play as a team	**Respond** Escalate, escalate, escalate	We don't mind being woken up for an incident, even if we aren't needed. But we do mind if we aren't woken up when we should've been. We may not always have the answers, so "don't hesitate to escalate."
	Don't !@#$ the customer	**Recover** Stuff happens, clean it up quickly	Our customers don't care why a service goes down, only that we restore it as quickly as possible. Never hesitate to resolve an incident quickly so we can minimize impact to our customers.
	Open company, no bullshit	**Learn** Always blameless	Incidents are part of running always-on services. We improve services by holding teams accountable, not by blaming.
	Be the change you seek	**Improve** Never have the same incident twice	Identify the root cause so we can prevent the incident from occurring again. Commit to delivering specific changes by specific dates.

Tools for an always-on enterprise

In addition to strong practices and culture, companies running always-on services need the right tools. Teams with mature DevOps practices use tools to facilitate agile project planning and sprints, CI/CD, automation, and advanced monitoring and alerting capabilities.

A modern incident management tool like Opsgenie ensures you receive important alerts delivered to your preferred notification channel(s) with the lowest latencies. It also includes the ability to group alerts to filter numerous alerts, especially when several alerts are generated from a single error or failure. An alert management tool must seamlessly integrate with your team's tools (e.g., log management, crash

reporting) so that it naturally fits into your team's development and operational rhythm.

Each team is different in terms of workflows, policies, and stakeholders. The alert management tool must be able to customize on-call schedules and routing rules to handle alerts based on their source and payload. Often the alerts may warrant an escalation to an incident. The tool should manage an incident without distractions by automatically creating an incident manager. This allows you to manage the incident like a war room with all the information handy, with integrations to communication and collaboration tools. Finally, the tool must provide advanced reporting and analytics to gain insight into areas of success and identify opportunities for improvement. It should reveal the sources of alerts, the team's performance in responding, and how on-call workloads are distributed.

In conclusion...

The modern consumer's desire for always-on services has become less of a want and more of a need. Many companies adopt a YBIYRI culture to develop the agility required to satisfy these demands. The challenge is that many companies aren't equipped with the appropriate tools and necessary team structures/practices to sustain this velocity.

If you are planning to shift to a YBIYRI DevOps culture for your team, here are some steps to take:

- Prepare your team to own all phases of development and operation of the application or service
- Ensure alignment with product owners so that SLOs are prioritized in sprint planning
- Embrace a set of incident values that guide the behavior of your team in response to an incident
- Empower your team with a modern alert and incident management tool like Opsgenie, which is reliable, fast, and flexible

Download our free incident management handbook and get started with Opsgenie for free.

KRISHNA SAI

Krishna Sai is the Head of Engineering, IT Solutions at Atlassian. He has over two decades of engineering/technology leadership in several startups and companies including Atlassian, Groupon, and Polycom. He lives in Bengaluru, India and is passionate about building products that impact how teams collaborate.

SHARE THIS ARTICLE

NEXT TOPIC

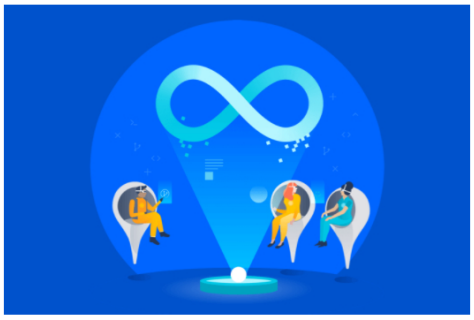
DevOps Frameworks →

Recommended reading

Bookmark these resources to learn about types of DevOps teams, or for ongoing updates about DevOps at Atlassian.

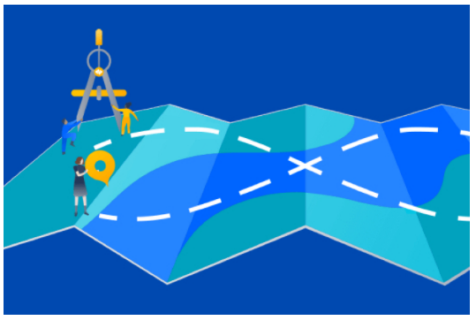
DevOps community

Learn more →



Simulation workshop

Learn more →



Get started for free

Learn more →

Sign up for our DevOps newsletter

Email address

Sign up

Copyright © 2021 Atlassian