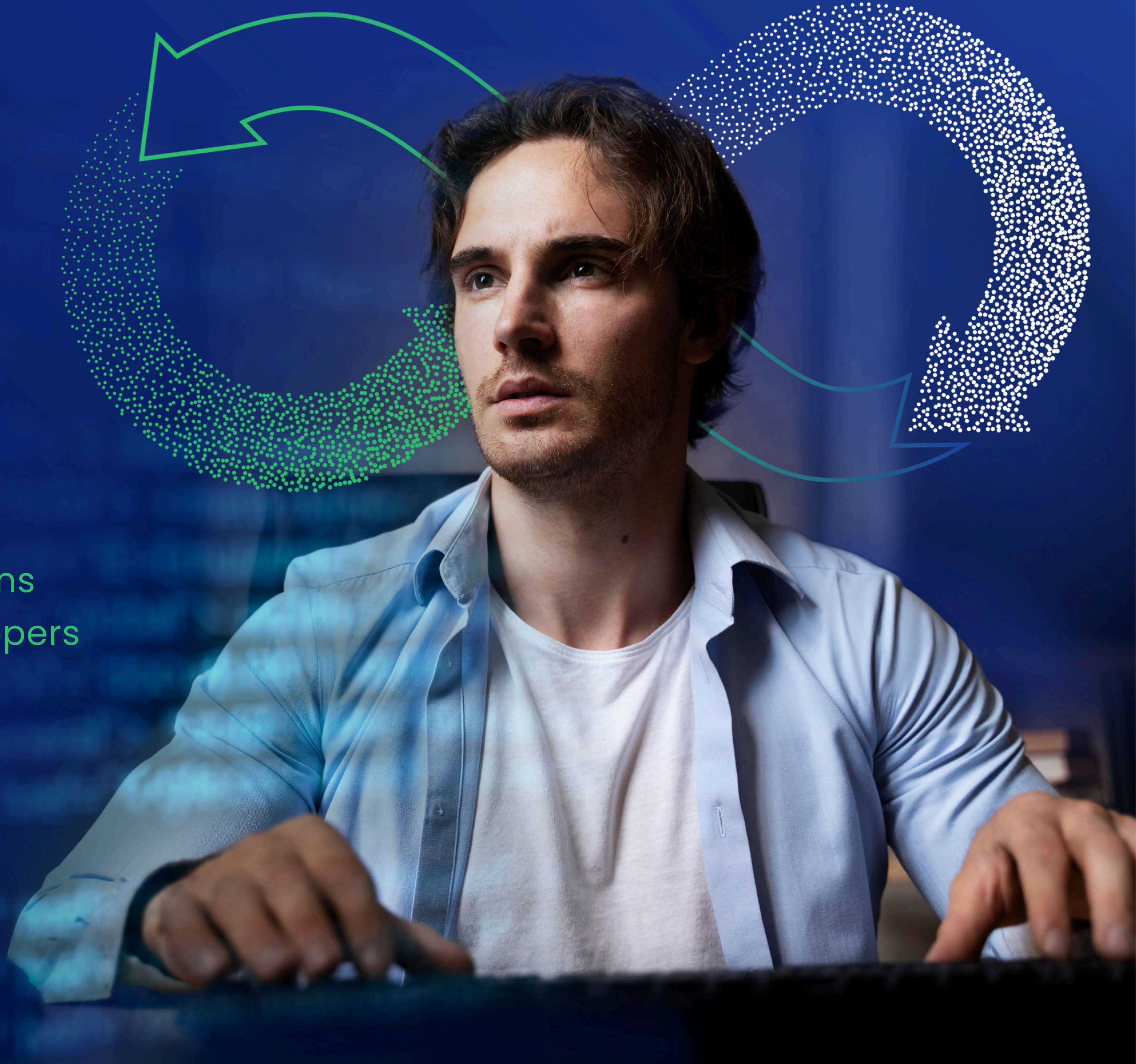
chronosphere

5 steps to improve developer efficiency in a cloud native world

How to align DevOps and cloud native operations with observability best practices so your developers drive even greater business value



chronosphere

You've taken the leap. Now make the most of it

Virtually everyone now agrees that cloud native is the architecture of choice for building and deploying modern applications, with 91% of enterprises saying they'll increase spending on technology supporting cloud native app development over the next 12-18 months.[1]

The power of cloud native environments

The way modern apps are built and work is possible because of a few critical technologies:

Microservices provide a loosely coupled application architecture, which enables deployment of applications in distributed modules.

Containers allow large applications to be broken into smaller components and presented to other applications as microservices. They are portable and easy to scale, and can be used throughout an application's lifecycle, from development to test to production.

Kubernetes (K8s) is the most popular open source platform to orchestrate containers, and K8s can run with containers on bare metal, virtual machines (VMs), in public clouds, private clouds, and hybrid clouds.

The state of cloud native application development

61%

We build and deploy cloud native applications based on a microservices architecture

36%

We build and deploy cloud native applications using a traditional (e.g., multi-tier) approach

4%

We plan to build and deploy cloud native applications within 12 months

[1] *Enterprise Strategy Group. "The Mainstreaming of Cloud-native Apps and Methodologies," 2023.*

chronosphere

The cloud native and DevOps connection

Cloud native intersects with another kind of change that directly impacts your developers. While cloud native is a software and technical architectural shift, DevOps is an organizational transformation happening in businesses that is critical to getting the most out of cloud native.

DevOps is a practice that breaks down the silos between development teams and central IT operations teams so that the engineers who write the software are also responsible for operating it. This is imperative in a cloud native era, as distributed systems are so complex that operating them must be done by the teams who know them best.

DevOps is immensely popular. As of 2023, 84% of developers worked in DevOps-related environments.[2] Moreover, the wholesale move of businesses to a DevOps model has caused time to restore systems outages to continuously decline.

When you combine cloud native with DevOps, small teams work on discrete projects, which can easily be rolled up into composite apps. Teams can work faster without all of the hassles of operating as part of a larger team. As the theory goes, the smaller the team, the more efficiently members can collaborate. And such collaboration is important because software is released at a much faster pace than ever before.

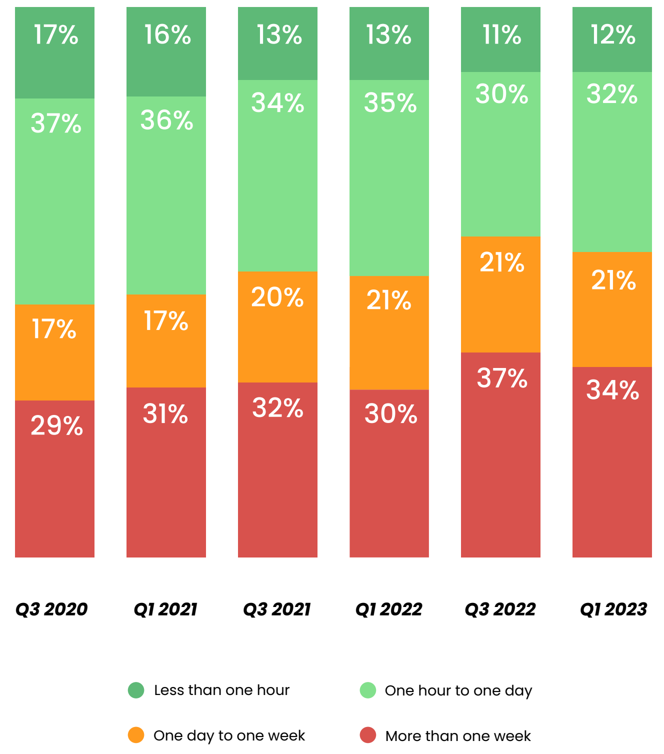
Together, cloud native and DevOps allow your organization to rapidly create and frequently update applications to seize ever-changing business opportunities. They help cater to your stakeholders and users who expect — or demand — the apps they use to be high availability, responsive, and incorporate the newest technologies.

This ebook outlines five ways you can combine cloud native and DevOps with observability best practices to boost efficiency for your developers as change continues.

[2] *Continuous Delivery Foundation, "State of Continuous Delivery Report," 2023.*

Time to restore service is shrinking

Software delivery performance - Time to restore service



Q1 2023 | The State of Continuous Delivery - The Evolution of Software Delivery Performance

chronosphere

Cloud native and DevOps together offer tremendous benefits

Applications built using a cloud native architecture with a DevOps organizational structure offer tremendous benefits to your organization, from faster time to market and better scalability to more efficient development and improved reliability.

Faster time to market

The component nature of cloud native apps allows software development tasks to be distributed to multiple for developers teams, and the work of these teams can be done independently. Each service owner can work on a particular component of the app simultaneously. Any issues in testing or deployment can be dealt with by developers immediately. Additionally, cloud native apps allow components to be reused. So, rather than creating a new front-end for every new app or a new buy capability, existing ones can be used on a new app. Reusing various elements greatly reduces the total amount of code that your developers must create for each new application.

Easy automation

Automation is one of the most significant advantages for developers, which uses practices such as continuous integration and continuous delivery (CI/CD) to automate development, delivery, testing, and deployment. Automating processes in a cloud native environment reduces human error, cuts cost, and speeds time to market.

Efficiency

A cloud native/developer-centric approach lets smaller development teams work in parallel on larger applications. The idea is that a smaller team spends less time managing timetables, attending status meetings, and keeping people up to date. They can devote more time to doing what needs to be done. In such a work environment, these small teams also access common company resources. That allows each team to benefit from cultural knowledge acquired over time throughout the organization. And naturally, the teams can work together, benefitting from each other's best practices.

Scalability and agility

In a cloud native/DevOps environment, your organization can readily scale different functional areas of an application as needed. Specifically, running elements of a cloud native application on public clouds builds the capability to dynamically adjust compute, storage, and other resources to match usage. The developers can monitor and make the adjustments easily—either up or down to accommodate both long-term trends or short-term changes.

Reliability and resiliency

Because cloud native systems are based on loosely coupled, interchangeable components, they are less vulnerable to large-scale failures than classical monolithic application. If one microservice fails, it rarely causes an application-wide outage, although it could degrade performance or functionality until fixed. Similarly, containers are designed to be ephemeral, and the failure of one node will have little to no impact on the operations of the cluster. When something fails, instead of impacting the entire application, a smaller set of services can be impacted. And because of the nature of the DevOps culture, your team that built the service is situated to speedily fix the issue.

Observability challenges that slow down developer efficiency

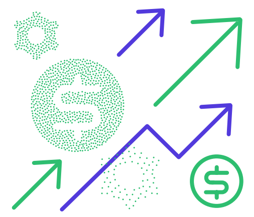
Yet cloud native environments can slow down your developers, especially if you depend upon traditional application performance monitoring (APM) and infrastructure monitoring tools to keep them up and high performing. As the complexity of cloud native, distributed systems grows, the ability to monitor, understand, and optimize these systems becomes increasingly critical. Traditional observability platforms, while providing a significant level of insight, often come with their unique challenges.

Complexity – Most APM and infrastructure monitoring tools were introduced more than a decade ago, when engineering teams were typically organized into a top-down hierarchy. Your developers built dashboards, reports, and other artifacts from the bottom up, and from their own worldviews, in isolation from the rest of the organization. This allowed developers to execute fairly quickly, but it caused problems when trying to scale. Indeed, application complexity is by far the greatest challenge to cloud native environments (59%).[3]

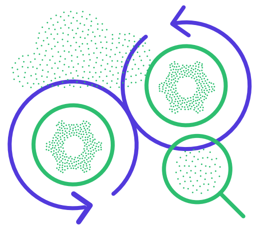
Unprecedented observability data volume growth – Sixty-nine percent of enterprises agree that their observability data is growing at a "concerning" rate, with 20% of respondents identifying this growth as a top-three priority to address.[4] Such growth slows down troubleshooting when using traditional APM and infrastructure monitoring solutions. Your developers can't find the right data, run queries quickly, or remediate issues as fast as they need to for exceptional customer experience. Developers can spend long nights and weekends trying to find and solve the root cause of problems. Burnout sets in, and in worst case scenarios, you face high turnover of your most valued developers.

[3] *GigaOm. "A GigaOm Survey: Driving Better Outcomes with APM and Observability," 2022.*

[4] *Enterprise Strategy Group. "Observability and Demystifying AIOps," 2023.*



Unpredictable and rapidly increasing observability costs – The pricing models for traditional APM and infrastructure monitoring tools mean that costs increase as data ingestion volumes or your numbers of users or hosts grow. Moreover, such tools typically don't include any mechanisms to control data growth. It's therefore easy for observability expenses to spiral out of control. Often, this drives developers to attempt to control costs by restricting custom metrics and minimizing cardinality tags. Although doing this can stabilize costs, it negatively impacts their ability to visualize the behaviors of their environment, causing them to "fly blind" at critical moments.



A widening skills gap and challenges recruiting talent with the right skills – It's a tough market out there for finding qualified developers. Unfortunately, traditional APM and infrastructure monitoring tools are designed in such a way that impedes easy analysis of—and thus understanding of—observability data, so only the most advanced users can diagnose issues. This puts you at a disadvantage the more you depend on your cloud native applications. According to the DevOps Institute, for almost one-third (31%) of developers, the speed of change, innovation, and pressure to transform drive the biggest developer skill challenges businesses currently face.[5]

[5] *DevOps Institute. "Upskilling IT 2023 Report," 2023.*

The most in-demand developer skills[6] are as follows:

51%

Process and framework skills
(familiarity with key process frameworks such as DevOps, Agile, SRE, ITIL®)

46%

Technical skills
(cloud platforms, serverless, microservices, APIs, programming languages, etc.)

44%

Human skills
(collaboration, communication, social skills)

41%

Leadership skills
(coaching, guiding, decision-making, etc.)

41%

Digital skills
(digital fluency, understanding digital systems)

40%

Automation skills
(replacing of manual processes, tasks or events with automation)

37%

Cognitive skills
(analytical capabilities, quantitative and statistical knowledge)

35%

Business skills
(business acumen, financial management, marketing, sales, etc.)

[6] *DevOps Institute. "Upskilling IT 2023 Report," 2023.*

5 steps to improve developer efficiency in a cloud native world

The following steps can help your organization improve developer efficiency to help ensure your organization is satisfying customer expectations while controlling costs.

Step 1: Get the right cloud native tools for the way you want to work

First, deploy tools that automatically present each of your teams with just the data that is relevant to them, cutting out all the other noise. The only way to do this—and to support a modern environment organized with small, interdependent engineering teams—is with a cloud native observability solution that supports workflows aligned with how your distributed teams now operate.

In a cloud native world, your developers own responsibility for the operations of their applications, rather than throwing them over the wall to an IT Ops team. As the mantra goes, "you build it, you own it."

A developer logging into a more traditional infrastructure or application monitoring tool is overwhelmed by a sea of data with very little ability to navigate. Finding the right data is nearly impossible. To make it worse, dashboards load slowly or not at all. On-call shifts are chaotic, and newer team members end up escalating issues to expert power users who are the only ones who seem to know where to look.

Consequently, observability solutions need to support workflows more aligned with how distributed, interdependent developer teams work.

Step 2: Analyze, refine, and operate observability data at scale

Next, empower your teams to analyze their observability data to understand what is useful and what is waste. Enable them to shape their data to improve its usefulness and eliminate what they don't need. And ensure the data is continuously optimized so dashboards, alerts, and queries are fast and deliver the information developers need to do their jobs. All of this should be done without touching source code and redeploying, so it doesn't slow down your developers.

But acknowledge that everyone needs to understand their usage and take ownership of it. This means performing chargeback/showback, just as infrastructure and cloud costs are allocated today in many businesses. Once you have visibility into consumption, you can take steps to make this more predictable, by giving portions of observability system capacity, with guardrails (quotas) for each team. By doing this, not only can the observability team ensure one developer team can't crowd out other teams, but they also delegate responsibility for optimization decisions to the teams themselves – always a good thing.

Step 3: Provide standardized templates and best practices to developer teams

No one should have to reinvent the wheel. Give your teams a standard set of dashboards, alerts, and integrations they can start with and customize from there. Providing your teams with service-level objective (SLO) and service-level agreement (SLA) definitions will also give them a head start and have the added benefit of standardizing definitions of performance across your organization.

Leading cloud native observability platforms will provide you with out-of-the box dashboards, metrics, and integrations that you can adjust to fit your particular needs.

Step 4: Optimize for speed and performance

Choose an observability platform that rapidly loads real-time dashboards and quickly responds to queries, presented in a way that less time is spent worrying about metrics and more time is spent working on value-added activities. Optimizing speed and performance comes down to asking (and answering) three questions:

1. **How quickly are your developers notified when something is wrong?**
2. **When something does go wrong, how fast can it be triaged so that you understand the impact?**
3. **How speedily can you figure out the underlying root cause of the issue to fix the problem?**

The goal is fast remediation—and always honing these steps to optimize performance.

Financial technology company Affirm saved 14,000 engineering hours per year just by focusing on dashboard load times and querying to triage much faster.

Step 5: Ensure tooling is usable by all levels of developers, not just power users

Too many observability tools today are hyper focused on power users. They leave novice and casual users behind. The result? When an incident occurs, your on-call developers can rarely solve the problem themselves and must escalate to more experienced (and much more expensive) senior developers who should be working on more important problems rather than firefighting.

A recent Enterprise Observability Survey revealed that 63% of respondents were at least somewhat concerned about a lack of engineering skills ahead of an organization's digital transformation or application modernization.[7]

It's imperative to get rid of the hero culture and choose a cloud native observability tool where any developer can easily troubleshoot any application or service.

[7] *Chronosphere. "2023 Cloud Native Observability Report," 2023.*

chronosphere

Discover how Chronosphere can boost developer efficiency

If your company is getting started with cloud native, Chronosphere is an ideal partner with the only purpose-built, SaaS monitoring solution for cloud native environments that boosts developer efficiency. The Chronosphere platform reduces customer observability data volumes by 60%, on average, while improving key metrics such as time to detection and time to remediation.

Chronosphere delivers capabilities that benefit central observability teams and makes the lives of developer teams easier by streamlining workflows, accelerating remediation, and improving both developer efficiency and quality of life. In real-world customer environments, Chronosphere is reducing data volumes by 98% and giving organizations 99.99% availability.

At the heart of Chronosphere's approach to observability and improving developer efficiency is the Control Plane. It overcomes the challenges traditional observability platforms present and offers a transformative solution for cloud native environments with four interconnected components:

- **Operate focuses on efficiency** – It provides platform-generated optimization opportunities and mechanisms to ensure fast and effective queries. This leads to quicker problem-solving and time savings, improving developer efficiency.
- **Analyze bestows understanding** – It helps you understand your observability data's value and cost in real time.
- **Refine aspect empowers your teams to transform data based on need, context, and utility** – Dynamic shaping policies can be implemented and adjusted on the fly to reduce noise, improve data usefulness, and control costs.
- **Centralized governance assigns accountability and control to teams** – By setting quotas and providing ownership of your data to teams, it helps contain cardinality, control long-term data growth, and avoid budget overruns.

chronosphere

The Chronosphere Control Plane in action

Robinhood

An **80%** reduction in data and improved query latency by **8x** and MTTD by **4X**

Snap Inc.

Reduced data volumes by more than **50%** and cut the number of on-call pages by **90%**

Abnormal

98% reduction in data volumes and improved MTTD by **8x**

Learn more

The new world of cloud native coupled with a DevOps organizational approach promises to drive not just IT efficiencies and cost savings, but business results that feed into your strategic objectives. But your developers face a number of observability challenges in this environment that can adversely impact their efficiency.

These five best practices steps tell you how to continue transforming your developer environment from a cost center to a source of competitive advantage, and why Chronosphere is the best solution and partner to help make your transition happen successfully.

See how Chronosphere can boost your developer efficiency.