Getting started with cloud native

A primer for enterprises modernizing their legacy IT infrastructure

RTInsights
Accelerate Your Business with Real-time Decisions

chronosphere

Executive summary

A reliable cloud native environment is essential for the survival of enterprises today. Moving to a modern microservices- and container-based architecture promises speed, efficiency, availability, and the ability to innovate faster — key advantages enterprises need to compete in a world where a new generation of born-in-the-cloud companies are luring away customers hungry for new features, fast transactions, and always-on service. Add in economic uncertainty, and the competitive stakes for enterprises soar: A simple search delay on an online retailer's site could lose a loyal customer, and coveted revenue, to a more innovative and reliable competitor.

Encroaching competition from nimble organizations. Uncertain global economy. Savvy, demanding customers. It's more important than ever to transition to a modern, cloud native technology stack and best practices that can deliver:

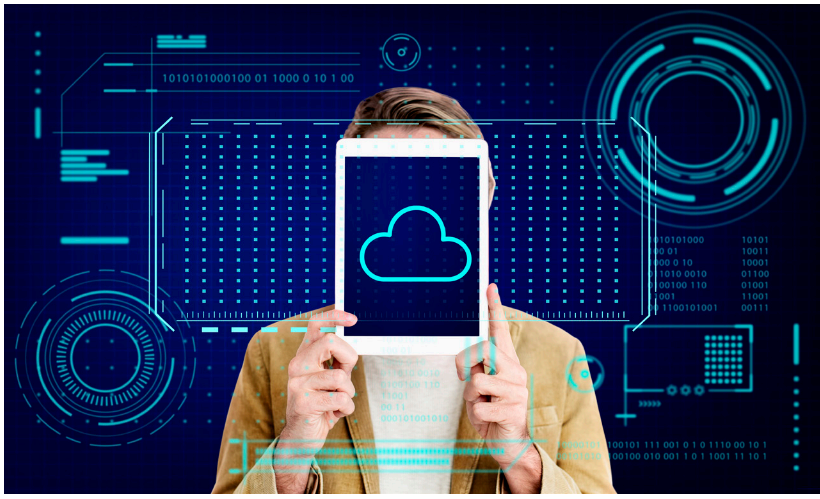
- **A highly available and more reliable service.** Cloud native best practices enable you to build a more resilient product and service.
- **More flexibility and interoperability.** Cloud native environments are not only more portable, but they also provide the ability to scale up and down dynamically and on demand.
- **Speed and more efficiency.** Engineers can iterate faster to handle increased customer expectations.

Buyer beware — cloud native is challenging: The benefits of adopting cloud native technologies are impossible to ignore — for proof, look no further than Gartner's prediction that 90% of companies will be cloud native by 2027. But there are also challenges that come with the shift from a traditional to a modern environment: If the transition to cloud native lacks proper planning and tools, enterprises risk unprecedented data volume, increased costs, downtime, reduced engineering productivity, and, yes, customer dissatisfaction.

Impressive adoption rates also tell us something else. There are many tech decision-makers still in the learning curve about cloud native basics. No matter where you are in your cloud native journey — just getting started or ready to expand adoption beyond a single business unit — this ebook will answer essential questions ranging from "What are the elements and benefits of cloud native?" to "How am I going to monitor this modern environment?"

There are four key areas to a "getting started with cloud native" primer, so let's dive into them.

1. What is cloud native?



Cloud native is the architecture of choice for building and deploying modern applications. In contrast to monolithic application development, cloud native applications or services are loosely coupled with explicitly described dependencies. As a result:

- Applications and processes run in software containers as isolated units.
- Independent services and resources are managed by central orchestration processes to improve resource usage and reduce maintenance costs.
- Businesses get a highly dynamic system that is composed of independent processes that work together to provide business value.

Fundamentally, a cloud native architecture makes use of microservices and containers that leverage public or private cloud platforms as the preferred deployment infrastructure.

Microservices provide the loosely coupled application architecture, which enables deployment in highly distributed patterns. Additionally, microservices support a growing ecosystem of solutions that can complement or extend a cloud platform.

Containers are important because developing, deploying, and maintaining applications requires a lot of ongoing work. Containers offer a way for processes and applications to be bundled and run. They are portable and easy to scale. They can be used throughout an application's lifecycle, from development to test to production. They also allow large applications to be broken into smaller components and presented to other applications as microservices.

Kubernetes (also called K8s because of the number of letters between "K" and "s") is the most popular open source platform that is used to orchestrate containers. Once engineers configure their desired infrastructure state, Kubernetes then uses automation to sync said state to its platform. Organizations can run Kubernetes with containers on bare metal, virtual machines, public cloud, private cloud, and hybrid cloud.

The cloud native and DevOps connection

Cloud native is the intersection of two kinds of changes. One is a software and technical architecture around microservices and containers, and the other is an organizational change known as DevOps. DevOps is a practice that breaks down the silos between development teams and central IT Operations teams where the engineers who write the software are also responsible for operating it. This is critical in a cloud native era, as distributed systems are so complex the operations must be run by the teams who built them.

With cloud native and DevOps, small teams work on discrete projects, which can easily be rolled up into the composite app. They can work faster without all of the hassles of operating as part of a larger team. Amazon Executive Chairman Jeff Bezos felt that this small team approach was such a benefit he popularized the concept of the two-pizza team, which is the number of people that can be fed by two pizzas. As the theory goes, the smaller the team, the better the collaboration between members. And such collaboration is critical because software releases are done at a much faster pace than ever before.

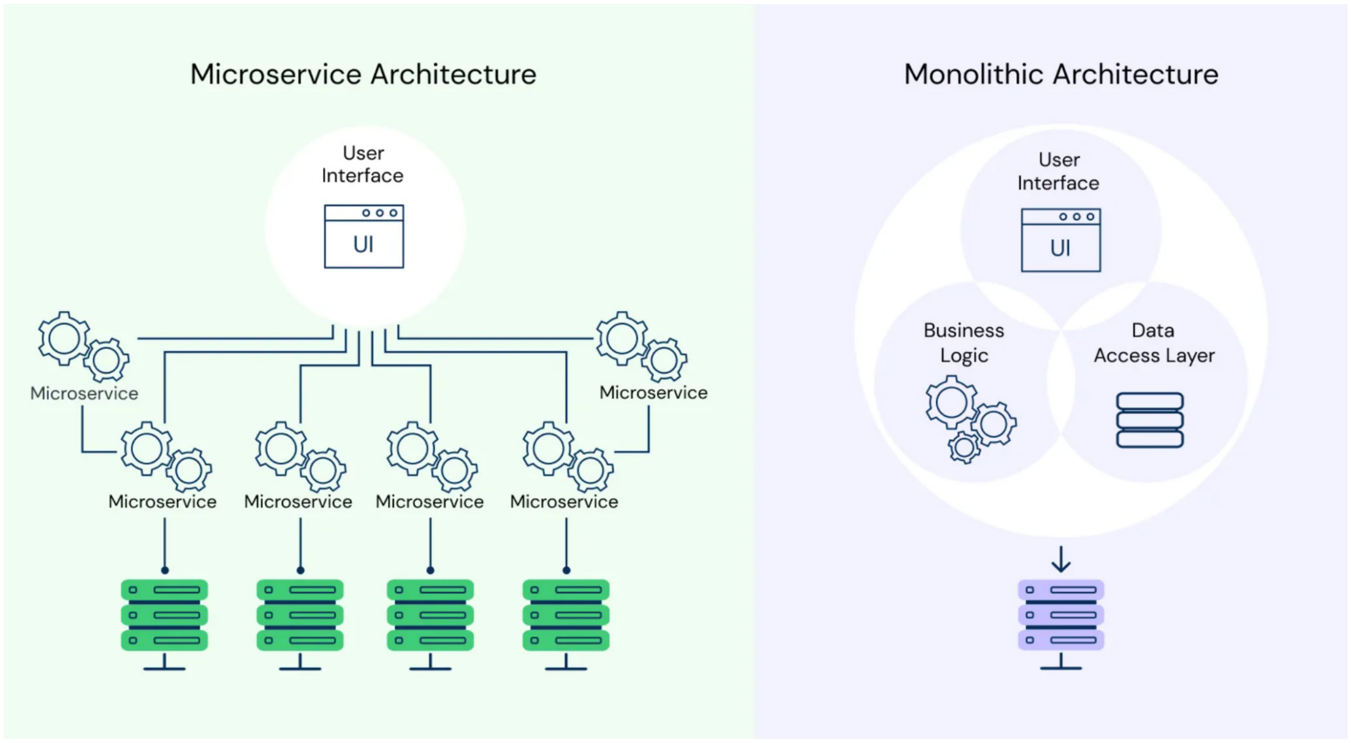
Together, cloud native and DevOps allow organizations to rapidly create and frequently update applications to meet ever-changing business opportunities. They help cater to stakeholders and a user base that expects (and demands) apps to be high availability, responsive, and incorporate the newest technologies as they emerge.

The monolithic architecture had its time and place

We just discussed how a microservices architecture is a structured manner for deploying a collection of distributed yet interdependent services in an organization. They are game-changing compared to some past application development methodologies, allowing development teams to work independently and at a cloud native scale.

In comparison, with a monolithic architecture, all elements of an application are tightly integrated. A simple change to one, say, the need to support a new front end, requires making that change and then recompiling the entire application. There are typically three advantages to this architecture:

- **Simple to develop:** many development tools support monolithic application creation.
- **Simple to deploy**: deploy a single file or directory to your runtime.
- **Simple to scale**: scaling the application is easily done by running multiple copies behind some sort of load balancer.



Monolithic architecture disadvantages

The monolithic model is more traditional and certainly has some pros, but it will slow down enterprises needing to scale and compete in a world where the name of the game is fast, reliable, innovative application development.

- **Scalability:** Individual components aren't easily scalable.
- **Flexibility:** A monolith is constrained by the technologies already used in the system and is often not portable to new environments (i.e., across clouds).
- **Reliability:** An application's availability can be impacted by module errors.
- **Deployment:** The entire monolith needs to be redeployed when there are changes.
- **Development speed:** Development is more complex and slower when a large, monolithic application is involved.

2. Benefits — and challenges — of a cloud native architecture

Applications built using a cloud native architecture offer a faster time to market, higher scalable, efficient development, and improved reliability. Let's look at the advantages in greater detail.

- **Faster time to market:** A cloud native approach to developing applications speeds development times. The component nature of cloud native apps allows development to be distributed to multiple teams. And the work of these teams can be done independently. Each service owner can work on their component of the app simultaneously. One group is not dependent on another group finishing its aspect of the app before they can start on their own.

  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 must be created for each new application.

  Change one thing in the code for a monolithic structure, and it affects everything across the board. Microservices are independently deployed and don't affect other services.

- **Efficiency:** As noted, a cloud native approach lets smaller development teams work in parallel on a larger application. The idea is that a smaller team spends less time managing timetables, in meetings, and keeping people up to date and more time doing what needs to be done.

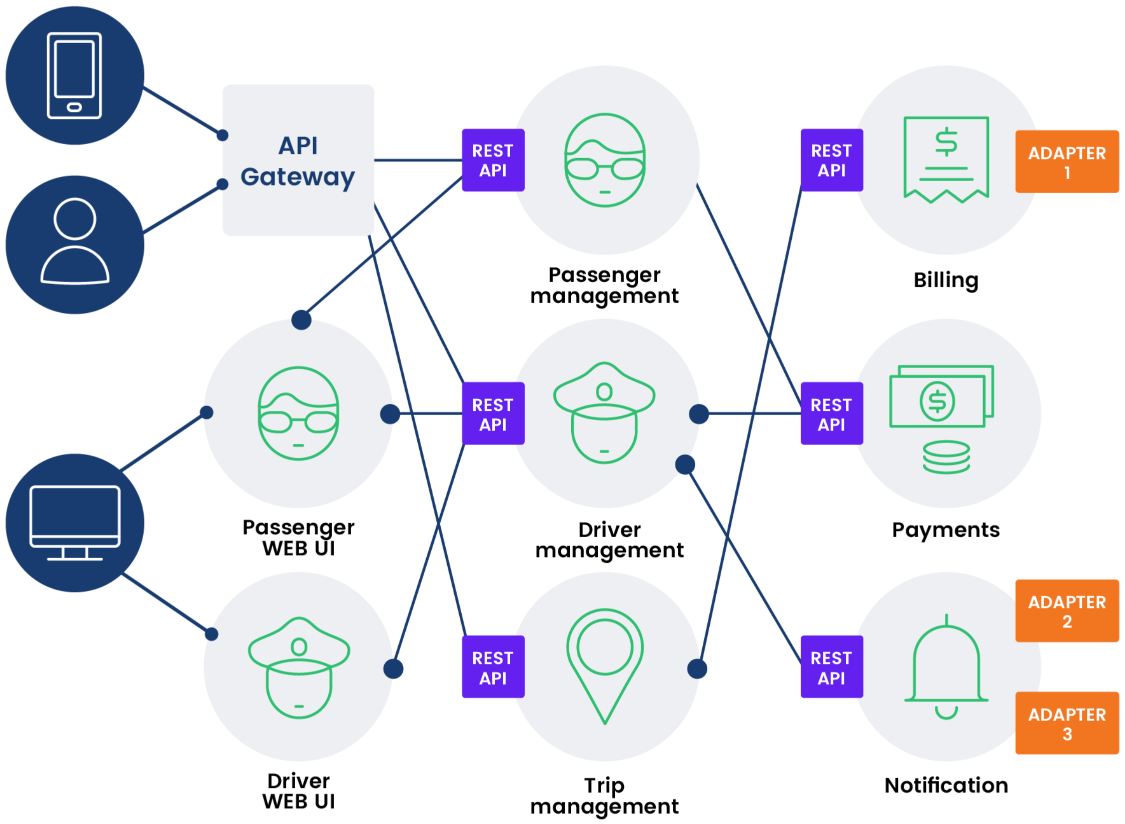
  In such a work environment, these small teams access common company resources. That allows each team to benefit from cultural knowledge acquired over time throughout the organization. And naturally, the teams can work together, benefiting from each other's best practices.

- **Scalability and agility:** In a cloud native environment, an 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 adjustment can be to accommodate long-term trends or short-term changes. For instance, a retailer having a seasonal sale can increase the capacity of their shopping cart and search services to accommodate the surge in orders. Similarly, a financial institution seeing an increase in fraudulent activity may scale up machine learning fraud detection services.

If you run everything through one monolithic application, it's hard to manage the massive scale of services and respond to changing market conditions as an application grows.

- **Reliability and resiliency:** Because cloud native systems are based on loosely coupled, interchangeable components, they are less vulnerable to a larger set of failures compared to the classical monolithic application. If one microservice fails, it rarely causes an application-wide outage, although it could degrade performance or functionality. 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. In short, in cloud native environments, the "blast radius" is much smaller when a component fails. When something fails, instead of impacting the entire application, a smaller set of services or functions may be impacted.



CAPTION: A real world example of a microservices architecture in action – Uber's architecture circa 2018.

Cloud native also comes with challenges

Competitive benefits notwithstanding, cloud native adoption comes with its own set of challenges. None are insurmountable thanks to modern tooling, but understanding what you're getting into with microservices and containers will set you up for success on your cloud native journey.

Complexity can impede engineer productivity

The inherent design of microservices leads to significant complexity. Imagine a microservices architecture featuring thousands of interdependent services — it becomes much more difficult and time-consuming to isolate issues. Even visualizing these services and their connections is challenging, let alone wrapping your head around it. When microservices are so independent of each other, it's not always easy to manage compatibility and other effects of different versions and workloads.

The infrastructure layer is not any simpler. Kubernetes is notoriously challenging to operate, in part because the ephemeral nature of containers means some may only live for a few seconds or minutes. There are many moving parts in a container orchestration system that all must be configured and maintained correctly.

All told, cloud native complexity places a new burden on engineers who are responsible for performance and reliability.

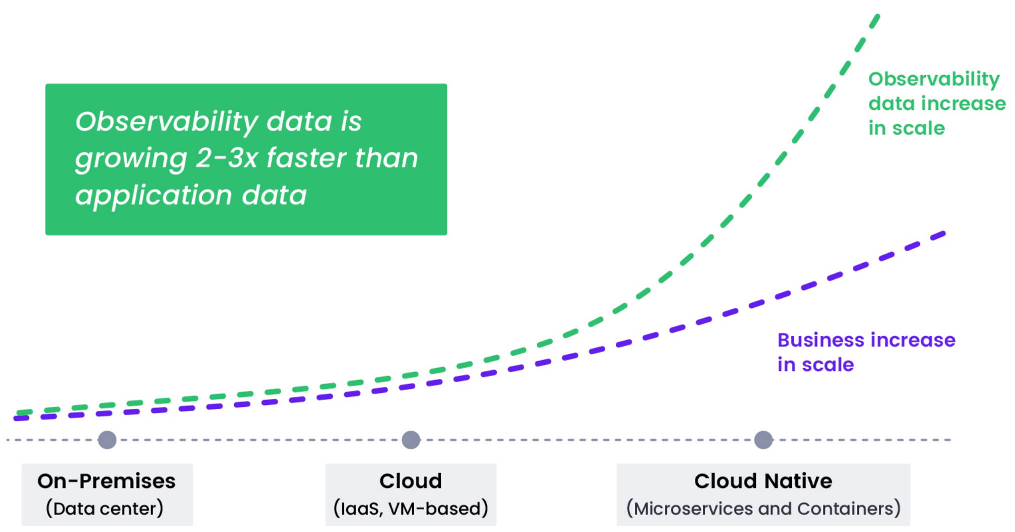
Unprecedented observability data volume

With cloud native agility comes an explosion of observability data (e.g., metrics, logs, traces, events) that can slow down teams while they're trying to solve customer-facing problems. Cloud native environments — especially as you start scaling them — emit massive amounts of observability data — somewhere between 10x and 100x more than traditional VM-based environments. Each container emits the same volume of telemetry data as a VM, and scaling containers into the thousands and collecting more and more complex data (e.g., higher data cardinality) results in data volume becoming unmanageable.

ESG estimates 69% of companies are concerned with the rate of their observability data growth.

This data alone can be challenging to manage, but the problem is made worse by developers adding more labels to their metrics, causing the number of elements in a set to spike. The unprecedented growth in observability data makes dashboards and queries run slow or not at all and makes engineers spend an inordinate amount of time just looking for the right data to fix a problem.

Cloud native impact on observability data volumes



Unpredictable and rapidly increasing observability costs

A key consideration for enterprises embarking on a cloud native journey: More observability data volume does not necessarily mean better customer experiences. Unchecked data growth does, however, lead to increased costs.

The explosive growth in data volume and the need for engineers to collect an ever-increasing breadth of data has broken the economics and value of existing infrastructure and application monitoring and tools. Costs can unexpectedly spike from a single developer rolling out new code. And ultimately, observability data costs can *exceed* the cost of the underlying infrastructure.

There are also other softer costs to consider with a transition to a modern architecture as well. Your traditional APM (application performance monitoring) solution wasn't designed to handle ever-increasing data volumes and massive cardinality, which can impact system uptime. When your systems are down, business stops, customers jump ship, and revenue is lost.

The skills gap is challenging to fill

Traditional IT and app development skills may not necessarily translate to cloud native. Instead, enterprises getting started on a digital transformation initiative need to start thinking ASAP about acquiring the latest skills, such as engineers with knowledge of open source software (OSS), especially Prometheus monitoring systems. In fact, 24% of respondents in a recent Enterprise Strategy Group (ESG) report on the mainstreaming of cloud native apps and

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.

methodologies found that one of the biggest challenges with cloud native apps was a lack of resources and skills.

Still, finding talent is half the battle. Compounding the skills gap issue is the fact that many engineers working with cloud native tools experience frustration and burnout due to the drain on their time troubleshooting issues — and inadequate (legacy) tools are typically to blame. Those that burn out contribute to the skills challenge because they must be replaced.

How are you going to monitor it all?

You're more than likely already leveraging APM and infrastructure monitoring tools, and by now, also realizing these tools aren't going to cut it in a modern environment. Time to start breaking down the most important questions: How are you going to monitor it all?

3. Your existing APM and infrastructure monitoring tools aren't equipped for cloud native

Enterprises monitoring early cloud native workloads only need access to simple performance and availability data. In this scenario, the siloed nature of these platforms isn't an obstacle to keeping applications or infrastructure running and healthy. So, traditional APM and infrastructure monitoring tools do the job.

But as organizations begin their cloud native initiatives and make use of DevOps principles to speed application development, more is needed. APM and infrastructure monitoring tools simply cannot provide the scalability, reliability, and shared data insights needed to rapidly deliver cloud native applications at scale.

Legacy tool shortcomings

Here are some key ways legacy monitoring tools fail to meet cloud native challenges laid out in the previous section. All of these shortcomings will cause acute pain as your cloud native environment grows, and should be factors that are considered when devising your modernization plan:

- **Can't navigate microservices.** Legacy tools are unable to navigate and highlight all of the interdependencies of a microservices environment, making it nearly impossible to detect and remediate issues in a timely manner.

- **Lack of control.** APM and infrastructure monitoring solutions lack data controls and visibility into observability data usage across teams and individuals. Simple code changes or new deployments can result in surprise overages.
- **Vendor lock-in.** Proprietary solutions make it nearly impossible to switch tools, leaving you powerless when prices go up.

And though these may seem like engineering-centric challenges, they end up having a big impact on overall business health:

- **Costs increase.** Because the pricing models for these tools is aligned to data ingestion, users, or hosts, and there are no mechanisms to control data growth, it's easy for costs to spiral out of control.
- **Teams end up flying blind.** Rapidly rising costs force teams to restrict custom metrics and cardinality tags, negatively impacting metrics stack behavior visualization and causing teams to "fly blind."
- **Developer productivity plummets.** Engineers are spending long nights and weekends troubleshooting. Burnout sets in. The skills gap worsens.
- **Downtime and data loss.** Service-level agreements (SLAs) and service-level objectives (SLOs) aren't being met. Small changes lead to data loss.

These shortcomings have consequences due to the way modern businesses operate. Customer experience and application responsiveness are critical differentiators. Anything that impacts either of these things can drive away customers, infuriate internal workers, or alienate partners. Today, rather than waiting for problems — including performance degradation, disruption, and downtime — to happen, businesses need to be ahead of the issues. They need to anticipate problems in the making and take corrective actions before they impact the application or the user.

4. The need for cloud native observability

Enterprises going from monolithic infrastructures to cloud native environments will fail without a modern approach to observability. It's a fact. Traditional monitoring alone will not do. But while, as mentioned above, the challenges cloud native adoption brings are real — they are not insurmountable by any means.

Arming teams with modern observability that is purpose-built for cloud native environments will allow them to quickly detect and remediate issues across the environment. Your applications will work as expected. Customers will be happy. Revenue will be protected.

In a recent Enterprise Observability Survey 71% of engineers say their business can't innovate effectively without good observability.

Here's what to look for in a cloud native observability solution.

Control data … and costs

Your cloud native observability solution should help you understand how much data you have and where it's coming from, as well as make it simpler to quickly find the data you need to solve issues and achieve business results.

Traditional APM and infrastructure monitoring tools lack the ability to efficiently manage the exponential growth of observability data and the technical and organizational complexities of cloud native environments. APM and infrastructure monitoring tools require you to collect, store, and pay for all your data regardless of the value you get from it.

With a central control plane, you optimize costs as data grows without any surprise budget overruns. You get persistent system reliability and improve your user experience. You enjoy platform-generated recommendations that optimize performance. You also don't waste so much valuable engineering resources on troubleshooting but rather reduce noise and redirect your engineers to solve problems faster. As a capability of a modern cloud native observability solution, a central control plane allows you to refine, transform, and manage your observability data based on need, context, and utility. That way, you can analyze and understand the value of your observability data faster, including what's useful and what's waste.

Avoid vendor lock-in with open source compatibility

Proprietary formats not only make it difficult for engineers to learn how to use systems, but they add customization complexity. Modern cloud native observability solutions natively integrate with open source standards such as Prometheus and OpenTelemetry, which eliminates switching costs. In times of economic uncertainty, and when tech talent is scarce, you'll want to invest in a solution that is open to possibilities.

Ensure availability and reliability

When your observability platform is down - even short, intermittent outages or performance degradations - your team is flying blind with no visibility into your services. The good news is that you don't have to live with unpredictable and unreliable observability.

Your modernization plan should include working with an observability vendor that offers a 99.9% uptime SLA, which can be confirmed by finding out what the actual delivered uptime has been for the past 12 months. Also, dig in a little to understand how vendors define and monitor SLAs and at what point it notifies customers of problems. A best-in-class solution will proactively monitor its own systems for downtime and count any period greater than a few minutes of system inaccessibility as downtime, prompting immediate customer notification.

Predict and resolve customer-facing problems faster

A cloud native observability solution can improve engineering metrics such as mean time to remediate (MTTR) and mean time to detect (MTTD) as well as time to deploy. But that's not all. It can also provide real-time insights that help improve business key performance indicators (KPIs) such as payment failures, orders submitted/processed, or application latency that can hurt the customer experience.

Improving the reliability of your observability tool has been shown by Forrester Research to reduce severe incidents by 75% annually.

Promote strong developer productivity from the jump

Something enterprises should try to avoid in transitioning to a modern environment: today's engineering on-call shifts are stressful because people can't find the right data, run queries quickly, or remediate issues fast. Most APM tools were introduced more than a decade ago when most engineering teams were organized in a top-down hierarchical fashion. In a DevOps world, developers own responsibility for the operations of their applications. The best way to support a modern environment that's been organized with small, interdependent engineering teams is with an observability solution that supports workflows aligned with how your distributed, interdependent engineering teams are now operating.

Your observability vendor should be a partner in your cloud native success

Technical expertise isn't a nice to have; it's a must-have for successful businesses. Vendor support experts help teams meet service-level agreements. Therefore, your observability vendor should offer customer support experts that are always available to help you navigate your cloud native journey — at no additional charge.

How Chronosphere helps cloud native initiatives succeed

Many enterprises getting started in cloud native are teaming with Chronosphere, which brings real-world expertise and an observability platform made for cloud native environments. Chronosphere's co-founders cut their observability teeth while at Uber and went on to create the only purpose-built, SaaS monitoring solution for cloud native environments.

Chronosphere gets to the heart of solving the top challenges enterprises will face in their transition to a modern, cloud native environment:

- **Data growth and cost:** As observability data grows, costs skyrocket and become a significant budget item, often second only to infrastructure costs. To control costs, many organizations are forced to make trade-offs that sacrifice visibility, reduce business insights, and increase risk.
- **Developer productivity and customer satisfaction:** The massive amount of data leaves engineers drowning in a flood of noise when troubleshooting, ultimately leading to longer incidents that impact customers.

Chronosphere provides deep insights into every layer of your stack — from infrastructure to applications to the business. 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. It delivers capabilities that benefit central observability teams and makes the lives of engineering teams easier by streamlining workflows, accelerating remediation, and improving both engineer efficiency and quality of life. In real-world customer environments, Chronosphere is reducing data volumes by 98% and giving organizations 99.99% availability.

What makes Chronosphere unique?

Chronosphere enables you to reduce observability data volume and cost to improve performance and deliver more business value. Here are a few key capabilities:

- **Control plane:** Chronosphere gives customers the ability to understand cost and value of observability data in real time. After understanding the cost and value of data, Chronosphere enables customers to take action without touching source code or redeploying. We allow customers to aggregate or downsample data, remove high cardinality labels, or drop non-valuable data. The result is reduced cost and improved performance without alert or query impact. Lastly, using the Chronosphere control plane, customers can understand usage by team, then assign guardrails to delegate responsibility to dev teams.
- **Reliability and performance at scale:** Chronosphere offers the most reliable observability platform with 99.99% historical uptime, automatically measured per customer. The underlying technology behind the Chronosphere platform is proven to handle more than two billion data points per second and 20 billion active time series with millisecond latency. In cloud native environments, there are often slow dashboards and alerts due to the sheer volume of data. Query Accelerator

automatically detects slow dashboards and queries and optimizes them behind the scenes to improve performance without customers ever having to touch them.

- **Outcome-driven workflows:** Unlike every other product that is optimized for telemetry input types, Chronosphere is optimized for outcomes: fastest detection, troubleshooting, and remediation. To do this, we surface relevant data and dependencies to your developers in curated and contextualized views. This allows customers to dramatically improve their troubleshooting process, regardless of engineering expertise, in order to achieve a 75% reduction in MTTR.
- **Open source compatibility:** Chronosphere natively integrates with open source standards such as Prometheus and OpenTelemetry. That allows organizations to preserve their monitoring and observability investments.
- **World-class customer support:** Chronosphere customer support experts are always available to help you navigate your cloud native journey. No additional charge. You get the success services you need today and for tomorrow, such as legacy tool migration, historical data on-boarding, and alert setting. Also, for many enterprises, open source software, such as Prometheus, PromQL, and OpenTelemetry, will be new. Chronosphere offers training and enablement that not only teaches the core features of the product but also includes a curriculum that helps engineers build their OSS knowledge and skills.

Some real-world examples of Chronosphere's Control Plane in action include:

- **Robinhood** - 80% reduction in data and improved query latency by 8x and MTTD by 4X.
- **Snap** - 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.

The bottom line

A Total Economic Impact™(TEI) study conducted by Forrester Consulting on behalf of Chronosphere quantified the productivity improvements and cost savings of its observability solution.

The study noted that customers can improve “reliability and reduced incidents to give time back to key resource groups such as developers, engineers, and help-desk workers. In addition to time spent in triage during downtime from incidents, customers avoid permanent revenue loss that occurred during critical downtime events.”

All of that translates into hard savings. The study found that customers using Chronosphere’s cloud native observability platform returned $7.9 million in benefits over three years, with $4.9 million in cost savings, for a 165% return on investment. **The average payback period was less than six months.**

To learn more about how Chronosphere can help on your journey to cloud native and modernization, get in touch.

About RTInsights

RTInsights is an independent, expert-driven web resource for senior business and IT enterprise professionals in vertical industries. We help our readers understand how they can transform their businesses to higher-value outcomes and new business models with AI, real-time analytics, and IoT. We provide clarity and direction amid the often-confusing array of approaches and vendor solutions. We provide our partners with a unique combination of services and deep domain expertise to improve their product marketing, lead generation, and thought leadership activity.

About Chronosphere

Chronosphere is the only observability platform that puts you back in control by taming rampant data growth and cloud native complexity, delivering increased business confidence. Chronosphere is backed by venture capital investors Greylock, Lux Capital, General Atlantic, Addition, and Founders Fund. The team is distributed, with major hubs in New York, Seattle, and Vilnius.

Learn more and watch a demo at chronosphere.io