The Six Fundamental Pillars of Managing Microservice Applications

Contents

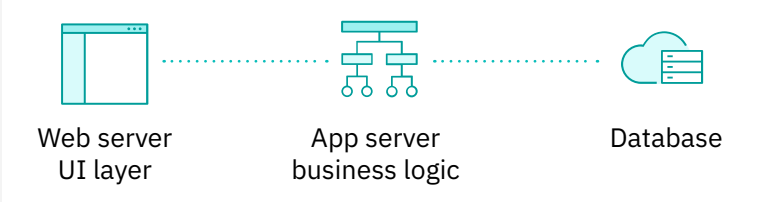
01
It’s a brave new world of applications

The modern application stack has evolved. From cloud native to containers, micros-ervices and Kubernetes orchestration, a new set of infrastructure technologies has emerged to help organizations digitally transform and move their applications faster.

In these modern applications, there’s one constant—constant change!

To effectively manage modern cloud-native microservice applications, DevOps teams need real-time visibility, automated observability and a way for all application stakeholders to get the necessary data in a way that makes sense to them at the exact time they need it.

Monolithic app



Microservices Architecture

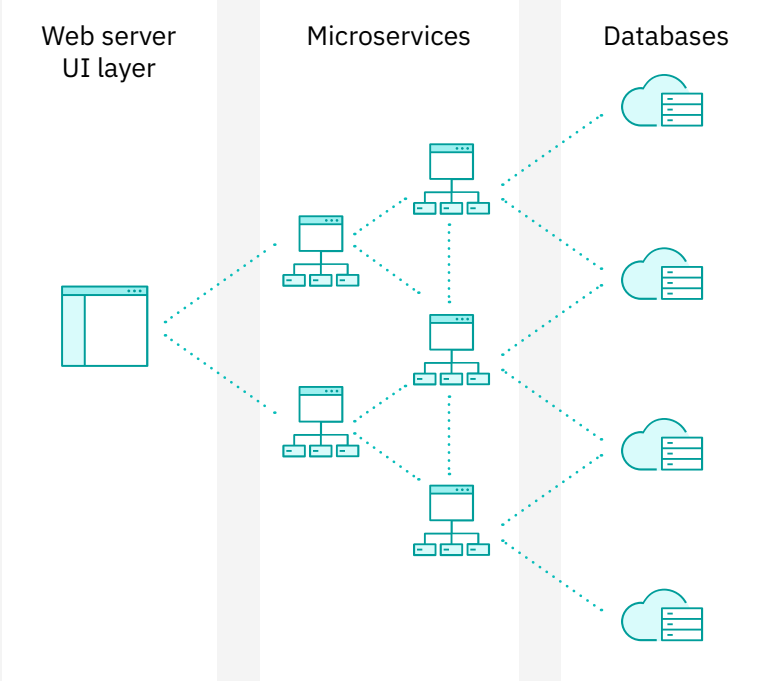


Figure 1. Not only are there more layers to a cloud-native microservice application, but each layer is more complex.

02

The difficulty of modern applications

As organizations adjust their technology stack to facilitate faster development and deployment of their applications, they face new requirements for basic performance monitoring and application observability and visibility.

The challenges of managing the performance of modern applications are consistent across programming languages, cloud providers and even different technology choices. What makes these applications difficult to deal with is the vast distributed nature of the applications coupled with the constant change bombarding both the Dev and Ops teams.

New application stack requires new monitoring requirements

The complete set of requirements to effectively manage these highly dynamic, massively distributed application environments is specific and built around the idea of all stakeholders getting the information they need when they need it.

With an eye toward what it takes to effectively monitor the performance of cloud-native applications, we at Instana, an IBM Company, present the six pillars of managing modern microservice applications. We detail the six critical concepts of monitoring and management that are critical to any solution's ability to provide continuous value to teams responsible for operating these applications.

With a not-so-subtle focus on automation and AI-assisted technology, here are the pillars of effective application performance monitoring (APM) for modern applications. Before we break down each pillar, it's important to note that if any single pillar is missing, the other five collapse. Thus, if a monitoring or observability solution doesn't have all six pillars, it runs the risk of having management gaps that could lead to application issues and many wasted bridge calls trying to fix them.

03

The six pillars of cloud-native microservices APM

1. Automatic, continuous discovery and mapping

There's an old saying in computing: "garbage in, garbage out." If we're to apply an AI approach to performance management, then the core model and data set must be up to date and impeccable. Meanwhile, DevOps needs up-to-date visibility for an accurate picture of its application's structure and dependencies. With zero configuration, our agent continuously and automatically discovers components, architecture and dependencies of the application's full technical stack and the request patterns or map of the distributed services.

The six pillars of cloud-native microservices APM:

Automatic, continuous discovery and mapping

Real-time full stack application data model

Precise high-fidelity visibility

AI-powered incident monitoring, problem resolution and troubleshooting

Cloud, container and microservice native

Integration into development, deployment, and continuous integration and continuous delivery (CI/CD) pipeline tool set

2. Precise high-fidelity visibility
Enabling AI requires data precision and accuracy. After automatically discovering the components and structure of the application, IBM Observability by Instana collects the industry's most accurate monitoring data. Metric data is streamed at one-second granularity, and every request through the application is captured in a trace. This data is the source for our AI training and the basis for giving the user deep visibility into microservice applications.

IBM Observability by Instana collects all the needed data automatically and in real time:

- *Time-series metrics for the full stack*—including infrastructure, cloud, container, process, middleware and application metrics
- *Changes*—all changes, events and errors for all monitored components
- *Distributed traces*—full distributed traces, mapping every request and transaction

3. Cloud, container and microservice native
Modern applications continuously change, are deployed across hybrid clouds and take advantage of containers and orchestration such as Kubernetes and D2iQ, all in the name of high-speed delivery. Meanwhile, the microservice architecture fosters wide diversity—engineers can choose the programming language, middleware and databases best suited to code their microservice. Scale, complexity and constant change are the new realities. Older tools, designed before the advent of cloud-native technologies, can be more onerous to configure and maintain, which can lead to potential visibility gaps

IBM Observability by Instana is designed to operate in the modern world. With zero configuration, it naturally aligns with the infrastructure, clouds, containers, orchestrators, middleware and languages to automatically keep up with and visualize your microservice application.

Even in constantly changing environments such as Kubernetes, IBM Observability by Instana automatically discovers the full stack of your application and manages its performance.

4. Real-time full stack application data model

The core technology powering Instana is the internal data model, the dynamic graph.

- The graph is a model of your discovered application with all physical and logical components, its technology components, dependencies and configuration.
- The graph understands logical components such as traces, applications, services, clusters and tablespaces.
- The graph updates its model in real time, whenever any change occurs to the application, through continuous discovery by the agents.
- The graph maintains the dependency models needed to enable Instana's AI-powered, precise troubleshooting, prediction and problem resolution capabilities.
- The graph is flexible—allowing IBM Observability by Instana to dynamically describe architectures of any complexity, layering, virtualization and so on—and, thus, is robust for future architecture trends.

Without a comprehensive model that includes the architectural makeup of an application, any intelligent analysis can be limited to simple data correlation, leading to missed relationships.

Without a comprehensive model that includes the architectural makeup of an application, any intelligent analysis can be limited to simple data correlation, leading to missed relationships.

5. AI-assisted monitoring and troubleshooting

It bears repeating that modern applications—cloud-native, microservices, container-based, orchestrated and multicloud—are extremely complex and dynamic. The only constant among these applications is that they are constantly changing. In this type of system, AI must be brought to bear to discover and understand the components, dependencies and overall health of the system.

For AI to effectively work, there must be a robust underlying model that includes:

- Automatic discovery of all system components and all interdependencies between those components
- Configuration data for each system component
- Performance data for each component
- An understanding of how components are used together to produce desired outcomes

Next is real-time detection—and understanding of any events—new components, updated components and eliminated components.

The model and real-time event detection drives the AI-assisted discovery and mapping, deployment, monitoring and troubleshooting needed for production application monitoring.

The best way to break the data into pieces that can be understood by both machines and humans is through the "Google golden signals"—transaction rate, error rate, latency and saturation. These derived metrics together represent the overall health of the application system.

The final piece of monitoring a complex system is to eliminate the noise. With millions of metrics and events, how do you keep the human operators from becoming overwhelmed? The answer is simpler than you might think—focus on service impacts only. This method uses the most constrained resource—human operator time—to provide the most positive impact, maximizing the application user experience.

Even with a smaller focus on service-level incidents, though, it's practically impossible for any single stakeholder—whether Dev or Ops—to understand how pieces work together and where any individual change might impact overall service levels. That's where the final piece of AI is applied: to take the incident and events, correlate them through the dependency model and identify the triggering event. Pinpointing the time, event, system and other specifics gives the monitoring users the exact information they need to find and fix the problem before major service impacts occur.

6. Integration into development and CI/CD pipelines

As application teams start taking advantage of new technology and architecture choices, releases become more frequent, are deployed by more individuals and impact more users. Modern APM solutions must recognize this shift in the application delivery process and integrate with the latest set of mission-critical tools—development and CI/CD pipeline management solutions.

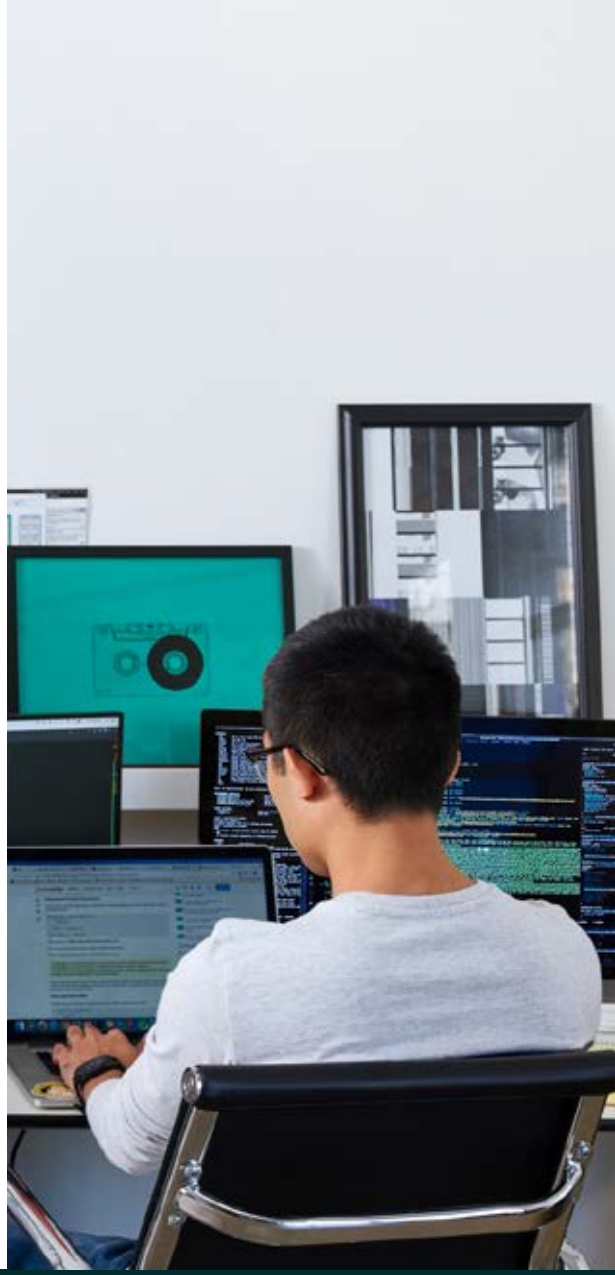
Of course, effectively managing release performance in an integrated manner requires a few capabilities that we've previously discussed, namely, one-second metric granularity and three-second alerting. After a software release, the faster the Ops team can know whether or not the release has issues, the better.

Beyond the fidelity and granularity of the metric set, it's important for the APM solution to understand—programmatically—when releases occur, what their impact is, and whether or not the releases achieved the appropriate performance and success levels. There are three critical capabilities you should be looking for:

- Real-time update and change detection
- Release identification and tracking
- Immediate feedback on new component performance

Only with all three capabilities can you hope to have an effective APM tool that relates to modern software practices such as continuous delivery and agile development cycles.

The bonus feature that you want to look for is integration with a release management solution, such as Jenkins, which allows you to not only auto-tag releases but also understand where potential hotspots are.



04

About Instana, an IBM Company

Instana, an IBM Company, provides an **Enterprise Observability Platform** with **automated application performance monitoring** capabilities to businesses operating complex, modern, cloud-native applications no matter where they reside—on premises or in public and private clouds, including mobile devices or IBM Z® mainframe computers.

Control modern hybrid applications with Instana's AI-powered discovery of deep contextual dependencies inside hybrid applications. Instana also provides visibility into development pipelines to help enable closed-loop DevOps automation.

These capabilities provide actionable feedback needed for customers as they optimize application performance, enable innovation and mitigate risk, helping DevOps increase efficiency and add value to software delivery pipelines while meeting their service-level and business-level objectives.

Learn more →

IBM Corporation
New Orchard Road
Armonk, NY 10504

Produced in the United States of America
April 2021