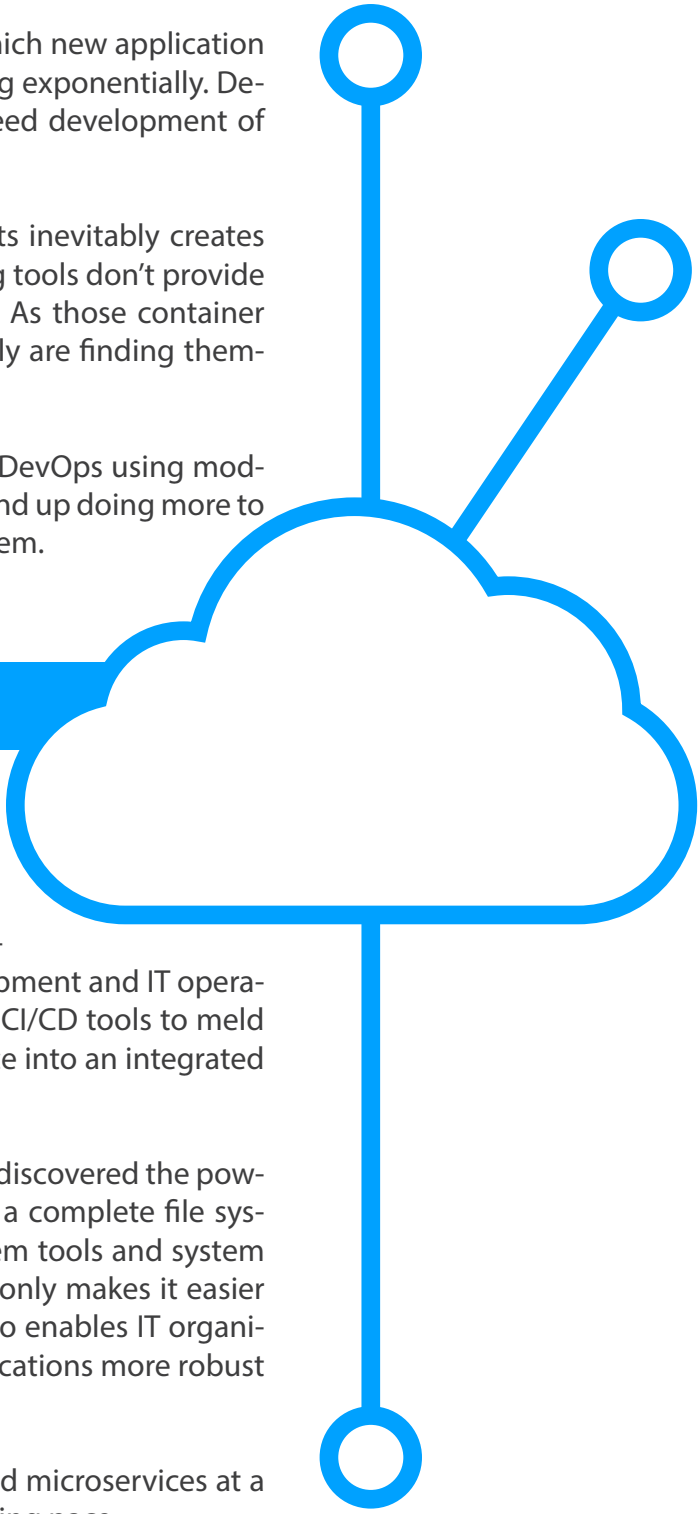
Microservices Sprawl: How Not to Be Overun

Table of Contents

Thanks to the rise of containers and microservices, the rate at which new application projects are moving into production environments is accelerating exponentially. Developers have been employing containers for some time to speed development of applications.

But the arrival of these applications to production environments inevitably creates new challenges for IT operations teams, since most IT monitoring tools don't provide visibility into the containers that make up those microservices. As those container applications move into production, IT operations teams suddenly are finding themselves potentially flying blind.

In fact, unless IT operations finds new approaches to managing DevOps using modern IT monitoring solutions, containers and microservices may wind up doing more to exacerbate longstanding DevOps issues than they do to solve them.



The New Pace of DevOps

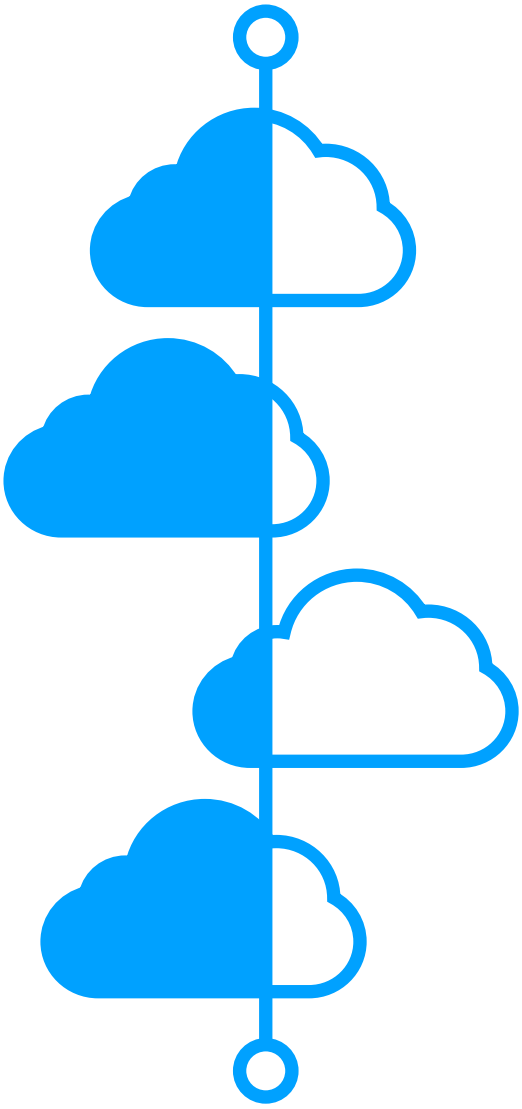
While the pace of DevOps adoption varies by organization, there is no doubt that agile development methodologies have been embraced widely at some level. The most advanced IT organizations have embraced continuous integration and continuous delivery (CI/CD) to achieve a substantial competitive advantage over rivals. Rather than approaching application development and IT operations as orthogonal processes, modern IT organizations employ CI/CD tools to meld application development, testing, refreshes and quality assurance into an integrated framework

Most of those modern IT organizations also have simultaneously discovered the power of containers such as Docker. A Docker container consists of a complete file system that contains everything, including the code, runtime, system tools and system libraries, which an application needs to run. That capability not only makes it easier to package application code to make it more portable, but it also enables IT organizations to adopt a microservices approach to IT that makes applications more robust and flexible.

Alas, there is a tendency for developers to employ containers and microservices at a rate with which many IT operations teams have a hard time keeping pace.

Microservices Usage

A recent survey of more than 500 IT professionals conducted by Docker Inc. found that more than half of respondents are running at least one container application in a production environment. But 90 percent also reported employing Docker containers in application development. On average, these organizations are seeing a factor 13 increase in the rate at which applications are being released in their organizations. A full 85 percent said that after adopting Docker containers, they are seeing improvements in their overall approach to IT operations.



500 IT Professionals Surveyed

More than half of respondents are running at least one container application in a production environment.

90 percent also reported employing Docker containers in application development.

On average, these organizations are seeing a factor 13 increase in the rate at which applications are being released in their organizations.

85 percent said that after adopting Docker containers, they are seeing improvements in their overall approach to IT operations.

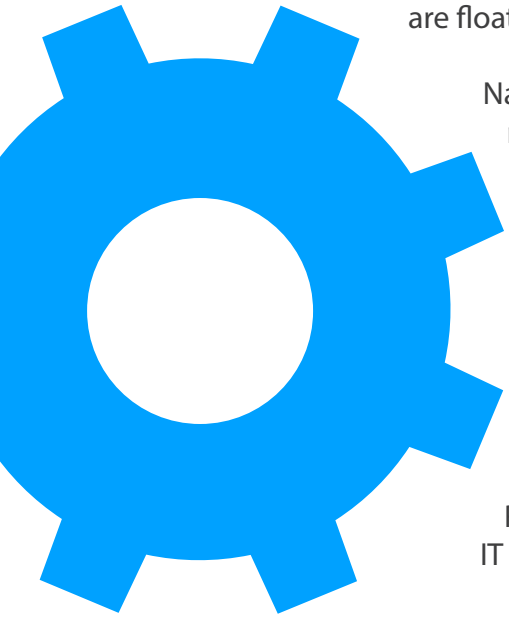
Microservices enabled by containers are gaining popularity because they enable developers to isolate functions easily. That, in turn, makes it easier to build new applications and update old ones. At a time when developers are under more pressure than ever to deliver more robust applications faster, microservices enabled by containers provide developers with crucial new capabilities.

Use of containers and microservices, however, is not limited only to new applications. Savvy developers also are dropping legacy applications into containers, making it easier to treat them as a large microservice. Then, over time, developers will use additional containers to isolate various functions of that application to create a true microservice.

In some organizations developers are taking over the entire lifecycle management of containerized applications constructed using microservices. Rather than handing off microservices to a separate IT operations team, developers are being held accountable for maintaining the code they create. That specific approach to DevOps is not yet embraced by every organization; many enterprise IT organizations, for a variety of reasons, will continue to have dedicated IT operations teams for years to come. But at the current rate of DevOps adoption, support for those applications constructed using microservices will be changing.

Container Metrics

Docker containers generate metrics; they just don't do it in a way most IT organizations expect. Not only are there commands for collecting Docker container metric data, there also are metrics that are specific to Docker containers, such as CPU throttling. In addition, as Docker container technology evolves, different versions are floating about, each with its own type of metrics associated with it.



Naturally, an IT team can opt to check all these Docker metrics manually. But an IT monitoring tool would can provide the information in much less time. In fact, one of the primary benefits of an IT monitoring tool is that it enables IT organizations to spend more time fixing problems than looking for their source.

Of course, Docker is not the only container technology an IT organization might want to monitor. Linux systems, SAP in-memory databases and the Cloud Foundry platform-as-a-service (PaaS) environment all make use of their own container technologies. Understanding that, organizations should choose an IT monitoring tool that is extensible to support other container technologies.

Regardless of the container technology employed, however, there are significant IT operations challenges ahead for IT organizations embracing microservices.

Issues with Containers

Memory technologies are at the center of the container universe. Container cluster managers look at the total memory available on the host against the amount of memory being requested by containers to decide on which host a new container should be launched. If a cluster manager is unable to find a host with sufficient resources for the container, the deployment will not take place.

Memory technologies are at the center of the container universe.

Memory issues are only the beginning. Each container usually runs a single process with its own virtual network and associated storage resources, and any number of short- and long-term containers can share one or more of the same underlying physical resources. For example, Docker images might be several hundreds of megabytes in size because they have their own operating environment attached. Cleaning up after those images have come and gone requires constant attention to ensure disk resources are available for other container applications. Because of these issues, container resource-sharing requires proactive enforcement of resource usage limits.

Other potential sources of contention include the internal networking inside the cluster and the network virtualization software used to connect containers running on different clusters. Usually there are two different types of networking technologies deployed inside the cluster and out. The handoffs between those two networking technologies must be monitored closely to ensure optimal performance in environments running multiple container clusters. That can be even more challenging when connecting container clusters running on-premises with ones running in a public cloud and accessed across a wide area network.

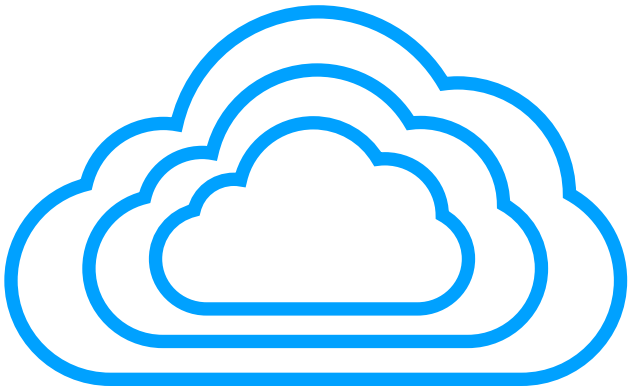
More challenging still, each of these services most likely will be created using multiple types of programming languages, in addition to their own unique syntax. Managing all the application programming interfaces (APIs) that the containers need to invoke can be its own full-time job. Add in load balancers and messaging layers between the services, and things can become complex, all promises of microservices simplicity notwithstanding.

Naturally, these potential container contention issues mean IT organizations need more visibility than ever into what is occurring in their environments, where containers can number thousands. In an ideal world, an IT monitoring platform should be able to alert IT organizations to issues. The robustness of the IT monitoring tool is, of course, of paramount consideration—a poorly implemented IT monitoring solution easily could wind up generating thousands of alerts because of the potential for system resource conflict.

In fact, as IT environments become more dense in the age of converged and hyperconverged infrastructure (HCI) configured with lots of Flash memory, there is a corresponding rise in management challenges created by the applications trying to share that IT infrastructure. Containers are an especially efficient means for driving up IT utilization rates. But with all that increased usage, IT organizations need to be more concerned with bottlenecks in their systems, which can spring up at any moment.

The Rise of Microservices at Scale

As with things in life, there can be too much of a good thing when it comes to microservices based on containers. Microservices and the containers they are composed of tend to be ephemeral in nature; most containers are stateless and not tied to specific database, which makes it easier for developers to scale their applications dynamically. The life span of a stateless container is relatively short because when developers want to add functionality, they simply replace entire sets of containers. There are stateful instances of container applications that make use of a database running inside a container. But most container applications are of the stateless variety.

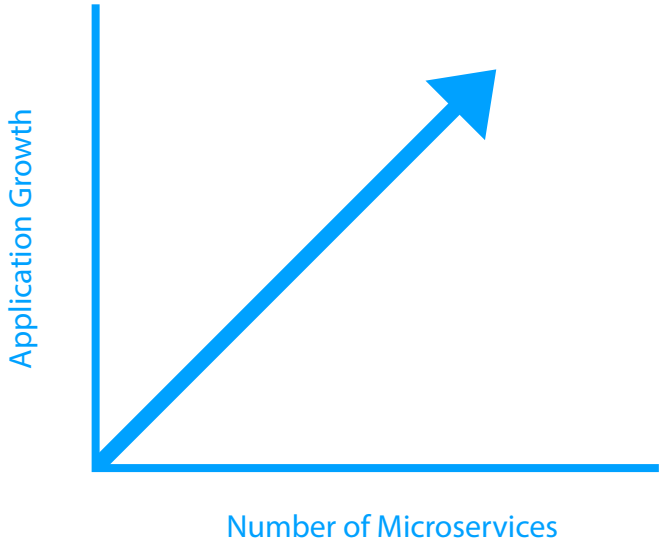


Most containers are stateless and not tied to specific database, which makes it easier for developers to scale their applications dynamically.

Regardless of the type, however, microservices tend to multiply once employed. Application developers find it much easier to add new functionality by creating yet another microservice that allows them to update and manage those functions in isolation from other microservices. Before too long, IT organizations find themselves trying to manage hundreds of thousands of microservices comprising an untold number of containers that might be replaced at any time. But the laws of physics still apply even to containers, and all those microservices are contending for access to the same limited set of physical IT infrastructure resources.

The Coming IT Complexity Paradox Created by Microservices Sprawl

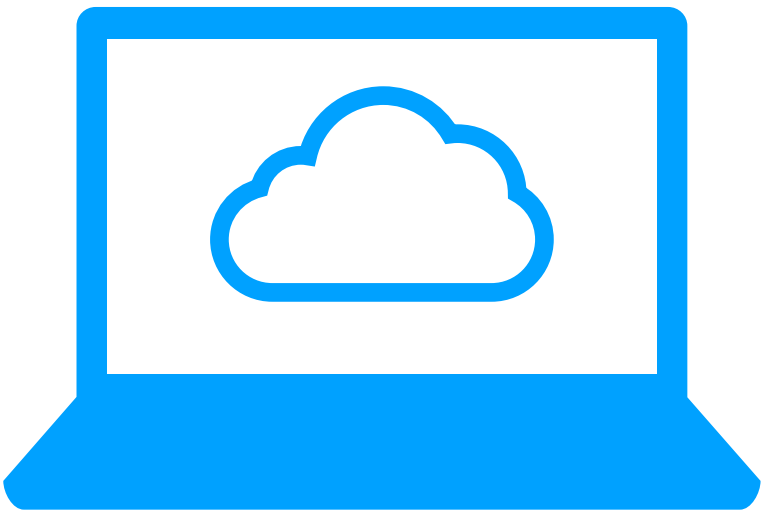
In theory, microservices are supposed to make managing IT easier. Various functions of an application are isolated in specific sets of containers that can be updated more easily in isolation. But as the application grows over time, the number of microservices increases. Each of those microservices can consist of any number of containers that most IT operations teams don't see coming or going.



IT operations teams, therefore, can't determine which of the containers running are invoking what resource. Before long, the entire IT environment becomes unbalanced. Most of the microservices may be hitting the same servers and storage systems while the rest of the IT environment sits idle. As a result, application performance starts to degrade and, before long, developers and IT operations managers are pointing fingers at each other.

IT monitoring tools that are optimized for containers prevent that from happening by providing the visibility required to identify dependencies to help IT operations teams optimize I/O across the entire data center environment. Specifically, a modern IT operations platform should be able to detect automatically the endpoints that make up a microservices architecture. Developers and IT operations teams should be able to monitor key performance indicators (KPIs) such as calls made to endpoints per minute, average response times and errors per minute from the time an endpoint was created to its deployment in a production environment. Also, they should have access to snapshots that include detailed diagnostics to drill down and isolate the root cause of any performance issues affecting any microservice.

Another major challenge is containers' ability to run almost anywhere. While today they are deployed most frequently on virtual machines hosted in cloud services, soon we may see more containers on both virtual machines and physical servers running on-premises or in a platform-as-a-service environment spanning a hybrid cloud computing environment. An IT monitoring platform should be able to track containers in any environment regardless of where they were created or deployed.



Microservices based on those containers can scale up and down rapidly, which makes it challenging to identify what specific microservices are being invoked by multiple applications and track their performance. An IT monitoring tool must be able to identify and track logically and over an extended amount of time which microservices are attached to which applications.

What's more, microservices often need to scale to support thousands of simultaneous requests, which raises the specter of system bottlenecks as multiple microservices attempt to request the same data. An IT operation tool should have a thread contention analyzer that helps identify, for example, the block time, blocking object and the blocking line of code. Armed with that information, developers and IT operations teams can minimize the time required to isolate and resolve application performance issues.

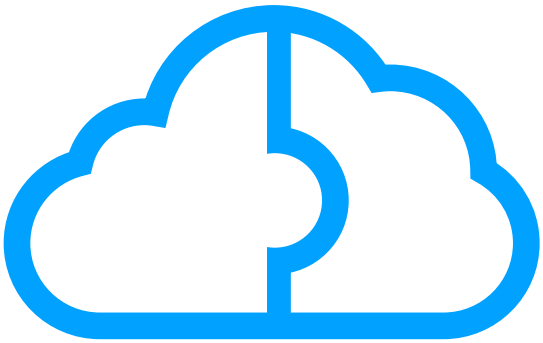
There also are multiple container orchestration platforms such as Kubernetes, Mesos and the Docker orchestration services provided by Docker Inc. that must be considered. It may not be uncommon for IT organizations to deploy multiple types of container orchestration platforms over time. The container monitoring tool they pick should be compatible with all the major container orchestration platforms.

Finally, the analytics provided by an IT operations platform is critical. Containers may look deceptively stable in terms of total IT infrastructure consumption, but container environments are some of the most dynamic ever seen in a data center. Some container services run for a long time, while others run for a few minutes. Over time, IT operations teams must be able to discern patterns in container behavior and consumption to prioritize access to systems resources. In addition, IT organizations eventually will create duplicate microservices. Discovering those duplicate services early in the development cycle will go a long way to reducing any technical debt an IT organization might incur as the microservices environment expands to include multiple application development teams. There may be good reason to duplicate a service, but IT operations teams should be made aware of that duplication before that new service is deployed.

Given the number of containers involved, most IT organizations will rely on some level of automation to address many of these issues. But all the automation in the world is not useful unless it's informed by accurate analytics.

Containers may look deceptively stable in terms of total IT infrastructure consumption, but container environments are some of the most dynamic ever seen in a data center.

Need for More Holistic Approach to IT Monitoring

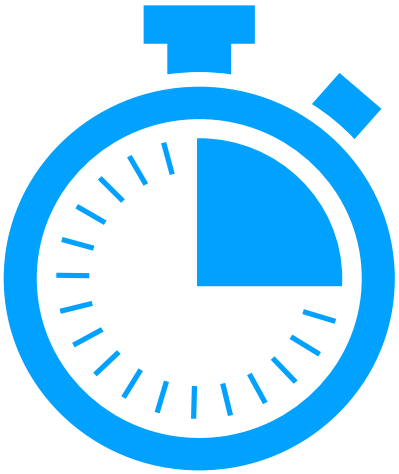


Microservices are only a piece of the IT puzzle. Legacy applications are not going away anytime soon; IT organizations have been crafting services around them for decades. There may be a lot of enthusiasm for microservices these days, but from an operational perspective IT organizations need a common framework for monitoring microservices alongside all the existing "macro" services they have been delivering for years.

Rather than trying to stitch together all that data from a variety of monitoring tools, a single IT monitoring platform can provide all the context an IT organization needs to respond dynamically to rapidly changing IT conditions. That context must include metrics pertaining to specific containers and the rest of IT infrastructure environment. After all, while performance attributes of a specific container might be interesting, that information is truly useful only when it can be compared to whatever else is happening within the IT environment.

Without that capability, an IT team could waste hours in war rooms trying to prove their innocence whenever a problem arises. Given the thousands of containers operating at any given time, IT teams could wind up trying to replicate a problem that might exist intermittently or for a few minutes.

Compliance Concerns



A big part of the IT management challenge is making sure IT operations can keep pace with the rate at which applications are being developed in the age of containers. Developers traditionally are not known for communicating their intentions clearly.

In addition, most developers don't have much time for compliance. Most, if not all, companies must comply with various regulations. Containers that can spin up and down rapidly can create a regulatory nightmare; as such, IT organizations must be able to document what occurred inside their IT environments and when.

Compliance officers are not likely to be impressed by any development practice that hinders visibility for the IT organization. IT monitoring tools that keep track of historical data associated with each microservice can be the first step in helping compliance officers accept and even endorse microservices architectures.

Changing the DevOps Culture

As compelling as new technology is, IT leaders know that modern technology is as much about people and processes as it is about the technology itself. Microservices and containers change almost every aspect of how IT is managed. Instead of a comparatively small number of monolithic applications, the future of IT will be defined by hundreds of thousands of microservices that developers mix and match to drive innovation.

That may make developers happy, but the rest of the IT operations team may need some time and guidance to make this adjustment. Most people are resistant to change, IT operations teams—through hard-won experience—generally are conservative when it comes to embracing new technologies. It's not that they don't appreciate their potential, but they are the ones usually held accountable should something go wrong.

A DevOps culture that makes developers more responsible for the quality of the application experience will go a long way to ensuring better code is delivered when employing containers and microservices. However, the processes necessary to achieve that goal do not happen of their own accord. In an ideal world, developers and IT operations teams work hand in glove, and for that to occur, everyone must understand their duties and responsibilities. In some cases, organizations will have to make significant investments in training to ensure everyone has the skills and expertise required to succeed.

Of course, the best way to start that cultural transformation is to provide a set of IT operations tools that make it easier for IT operations professionals to understand what's occurring inside an IT environment that has changed dramatically.

Conclusion

Containers and microservices will be crucial technologies driving next-generation digital business applications. IT leaders are under intense pressure to create more agile IT environments. Containers are a core technology enabling IT organizations to increase the number of applications and projects being worked on simultaneously. Just as importantly, those applications now can run anywhere there is enough IT infrastructure resources. The data center is just the beginning: Before too long, thousands of Internet-of-Things (IoT) applications also will be running outside the data center.

Containers are a core technology enabling IT organizations to increase the number of applications and projects being worked on simultaneously.

Developers, meanwhile, are forcing the issue by employing Docker containers—with or without permission. It's often not until that first Docker application arrives in production that IT operations teams begin to comprehend the enormity of the change at hand. In effect, deploying and managing containerized applications is an ambitious exercise in distributed computing.

However, containers and microservices offer limited visibility, which can be an issue with those processes that are the lifeblood of a digital business. If something goes awry, it's more than an application being down; entire streams of revenue are not being generated. Telling a business leader that the IT organization can't discover, let alone fix, the problem because there is no visibility into a containerized application is a career-limiting conversation that no one wants to—or ever should—have.

Worse yet, numerous contention issues may force an IT organization to abandon microservices and containers, while rival organizations use them to achieve great success. Containers and microservices may not be for the faint of heart. But they also are quickly becoming table stakes in the era of digital business competition. Organizations that fail to master them will find themselves overwhelmed by competitors employing microservices and containers to transform the end-user application experience.

A modern IT operations platform gives IT teams the confidence to deploy microservices at scale in production environments. Without visibility into those microservices, most IT organizations will be uncomfortable with any applications for which they can't identify easily the issues that might cause an application to fail or, more commonly, its performance to degrade. The quality of the IT operations experience directly impacts how fast any IT operations team will transition to containers and microservices.

Of course, that's not stopping organizations from adopting microservices. Companies are moving to microservices from monolithic applications to gain agility and reduce application development lead times. At the same time, however, the old IT adage about not being able to manage what you can't see applies. With IT operations, anything that can't be both seen and understood usually doesn't make it into the IT production environment.