While cloud-native development is proliferating, the developer experience for using Kubernetes remains complex. This IDC Technology Spotlight examines the issue of cloud-native application development and looks at the role of DigitalOcean in the market.

Understanding Cloud-Native Application Development and the Developer Experience

May 2019

Written by: Arnal Dayaratna, Research Director, Software Development

Introduction

As the most widely used orchestration framework for deploying and managing containerized applications, Kubernetes — the orchestration framework that was open sourced by Google in 2014 — owes part of its meteoric adoption to the contemporary importance of distributed computing. Moreover, Kubernetes adoption has been further spurred by the increased use of cloud computing as well as containers and microservices.

Kubernetes provides tools that enable developers to deploy distributed applications that leverage services delivered by APIs. Contemporary applications typically use a multitude of services that are delivered via APIs. They require an orchestration framework such as Kubernetes that can manage application scaling in conjunction with the APIs that run on multiple servers and that collaborate by means of web-based protocols. In a software development landscape where services and data stores are increasingly decentralized and distributed, Kubernetes takes responsibility for scheduling containers on a cluster and managing container-based workloads to ensure that they run appropriately.

Containers constitute the foundational infrastructure for applications that run on Kubernetes, which automates the deployment, management, scaling, and availability of container-based applications. The ability of Kubernetes to automate the deployment and operational management of container-based applications is similar to the way in which cloud computing platforms automate the scaling, elasticity, and high availability of infrastructure-as-a-service and platform-as-a-service infrastructures. Because Kubernetes brings the automation that is typically associated with cloud computing to application development, software development that leverages Kubernetes, containers, and microservices architectures is most aptly characterized as "cloud-native development." This term refers to a modality of software development that is optimized for the cloud because of its ability to build applications that leverage the distributed services, data stores, and infrastructures that are characteristic of cloud computing.

Cloud-native applications leverage the capabilities of microservices and use containers and container orchestration frameworks. These applications deliver the benefits of microservices-based development along with the application portability enabled by containers and the automation specific to container orchestration frameworks such as Kubernetes. By using a combination of microservices, containers, and container orchestration frameworks, cloud-native applications empower developers to rapidly deploy new features and functionality while preserving both high availability and reliability.

AT A GLANCE

KEY TAKEAWAY
Managed Kubernetes offerings that simplify the developer experience have the potential to democratize Kubernetes development by allowing more developers to use Kubernetes.
Drivers for Cloud-Native Development

Business Drivers
The primary business driver for cloud-native development is the need to ship applications more quickly without compromising quality or security. Organizations are tasked with the challenge of continuously reinventing themselves to keep pace with digitally driven innovation from competitors. Cloud-native development positions organizations to safely release new features and functionality for digital solutions that scale to accommodate the fluctuating needs of end users while demonstrating high availability and performance. Moreover, cloud-native development helps organizations deliver superior customer experiences characterized by shorter release cycles and applications that are stable, reliable, and highly available.

Today’s business environment is characterized by the increased availability of data, the rapid digitization of business processes, and the prospect of disruption by digitally savvy competitors. Greater data availability is related to leadership interest in using data to drive decision making. Organizations have an increased awareness of the value of internal and external data sources for guiding decision making. This awareness has deepened the practice of data-driven decision making versus relying on intuition or an affective interpretation of a business situation or landscape.

Meanwhile, the rapid digitization of business processes in almost every industry vertical is driving organizations to digitize their own processes and operational protocols. Organizations are introducing software to every business process or operational workstream, with applications and data-driven methodologies for use cases such as quantifying the number of customers entering a brick-and-mortar store, or segmenting visitors to an ecommerce site. The rapidity and intensity of digitization have organizations embarking on full-fledged transformation initiatives aimed at digitizing business operations and processes.

Another business driver for cloud-native development involves the prospect of disruption by a digitally savvy competitor. The availability and affordability of technology have amplified the possibility that a company will be disrupted by competitors using digital solutions to obtain a competitive advantage. To counter the threat, organizations need to aggressively innovate and thoroughly understand the evolution of their business landscape and customer base.

Technical Drivers
These drivers include the necessity of deploying applications to hybrid cloud infrastructures featuring a combination of non-cloud environments, on-premises private clouds, off-premises private clouds, and public clouds. As organizations experiment with multiple types of infrastructures, the ability to transport an application from one environment to another is of paramount importance.

Another key driver of cloud-native development involves the complexity of an application and the need to add new features and functionality to it in the future. In the case of a well-defined application with limited functionality (a banking application that allows users to make deposits, withdrawals, and transfers as well as check statement balances), a monolithic application architecture deployed on a traditional virtual machine (VM) may well suffice for the next generation of the application’s life cycle. However, in cases where software updates amplify an application’s complexity, a microservices architecture can help developers make incremental changes that deliver little to no disruption to the rest of the application. A microservices architecture empowers developers to add complexity to an application while isolating the effect to one or more microservices. As an application increases in complexity, microservices also help with debugging and application life-cycle management considerations related to identifying the root causes of an error or performance degradation.
One of the defining features of cloud-native applications is the integration of automation into the infrastructure on which applications are deployed. For example, cloud orchestration frameworks take responsibility for the provisioning of container-based infrastructures as well as application deployment and the scaling and self-healing of clusters of containers. In addition, container orchestration frameworks such as Kubernetes are responsible for identity and access management, role-based access control, and governance management. By enhancing the automation of container infrastructures, cloud-native development lets developers focus on writing code instead of managing the infrastructure on which their applications are deployed. This automation of operational responsibilities related to deployment and infrastructure operations enables developers to take on high-value tasks such as development of security, user experience (UX), and data management. IDC believes that as deployment platforms proliferate, cloud-native development practices that abstract away the complexities of deployment environments and the minutiae of operationally managing them will become increasingly important. As such, cloud-native development represents the leading indicator of a trend in which developers have less responsibility for the operational management of applications and their environments.

**Microservices Architectures**

Microservices architectures structure an application as a suite of loosely coupled services that implement business capabilities. In a microservices model, the individual services that constitute an application can be independently deployed while sharing a modicum of centralized management. Moreover, the individual services may be developed in different languages and storage technologies.

Microservices application design allows the propagation of rapid changes to individual modules within an application, enabling the delivery of rapid application changes and updates.

Moreover, microservices architectures allow finely grained scaling of application components as opposed to the monolithic application’s requirement to scale the entire application. This ability to use finely grained, targeted scaling of application components not only optimizes application performance but also leads to cost savings resulting from optimized resource consumption.

**Containers**

A container is a package of software that contains everything that the software requires to run. Containers include the executable software applications as well as relevant toolchains, libraries, and settings. Because containers have everything a software application needs to run, they allow the same software application to run in different environments. For example, a container that houses a Python-based application can run on different versions of Linux or Windows.

Containers facilitate microservices application architectures by providing a standard unit of software packaging for application development. For example, the database layer of an application might run in one container while the front end runs in a different container.

The lightweight quality of containers means that more of them can be deployed on a single machine. In addition, containers are typically megabytes in size and start up in seconds, whereas VMs are gigabytes in size and start up in minutes.
**Container Orchestration**
Container orchestration frameworks automate the management of container provisioning, deployment, scaling, discovery, scheduling, and self-healing.

Kubernetes facilitates horizontal autoscaling, persistent volumes, the ability to use webhooks, and rolling updates.

Kubernetes is notable because it can deploy on any infrastructure, inclusive of a multitude of public clouds as well as on-premises infrastructures. Other examples of container orchestration frameworks are Docker Swarm and Apache Mesos.

**Benefits**

**Development Velocity**
The microservices-based quality of cloud-native applications allows development teams to work on individual microservices in parallel, enabling developers to work on application development for multiple business functions concurrently. The accelerated velocity of cloud-native development shortens time to market and increases opportunities to deliver innovation.

**Agility**
Applications are architected as microservices so that developers can rapidly change an application's components without disrupting the rest of the application. This ability to create changes to an application independently of its other components underpins the implementation of a decentralized governance model that enables developers to make changes to an application without requiring the permission of other development teams.

**Superior Customer Experience and Business Innovation**
Cloud-native development takes an agile approach to creating new applications that lets developers respond to market developments and customer demands more quickly via a continuous integration and continuous delivery mechanism.

**Infrastructure Portability**
Cloud-native applications are more easily transported across different infrastructures because their container-based infrastructure abstracts away dependencies on their external environment. Cloud-native applications can be migrated across cloud infrastructures and from one on-premises infrastructure to another or from an on-premises environment to a cloud-based environment.

**Granular Application Scalability**
Developers can customize the scaling of individual containers in contrast to unilaterally scaling the entire application. This ability to apply granular scaling control over constituent parts of an application optimizes scaling and enables more efficient resource consumption. Because containers are lightweight, they can be deployed with a greater density than VMs on a server. Moreover, the lightweight quality of containers allows developers to rapidly create and destroy them, facilitating more efficient resource consumption compared with the provisioning of VMs.
Trends

In a global survey of more than 3,000 developers, IDC found that developers are using either container-as-a-service (CaaS) deployments or in-house container deployments, have firm plans to implement container solutions, or are generally educating themselves about container solutions (see Figure 1).

Figure 1: Developer Usage of and Familiarity with Containers for Production-Grade Applications

Some key observations about Figure 1 are as follows:

» 23.5% of developers are either generally educating themselves about or evaluating CaaS solutions, while 20.0% are doing the same with respect to in-house container solutions.

» 13.1% of respondents have firm plans to implement containers within the next 12 months.

At least 20% of developers are educating themselves about or evaluating container solutions. As such, the market features a notable segment of developers who have yet to obtain firsthand experience developing, deploying, and managing container-based applications. However, the developers in this segment deem container technologies sufficiently important that they are devoting time to deepen their familiarity with them.

Importantly, adoption of public cloud–based CaaS solutions differed between non-mission-critical applications and mission-critical applications: For non-mission-critical applications, 17.1% of developers adopted CaaS solutions while only 12.0% adopted in-house container solutions. Conversely, for mission-critical applications, 17.5% of developers opted for in-house container solutions while 11.4% chose CaaS solutions. The key takeaway is that organizations tended to select...
in-house container solutions for mission-critical applications because CaaS solutions are still at a relatively early stage of their adoption life cycle and continue to experience battle-testing and intensive scrutiny with respect to their readiness for mission-critical applications and workloads.

IDC considers cloud-native development to be a nascent, but rapidly growing practice that is likely to gain significant adoption for production-grade and mission-critical applications within the next 12–24 months.

**Considering DigitalOcean**

DigitalOcean is a cloud hosting service provider that provides a platform for developers at small to medium-sized enterprises to deploy, manage, and scale applications. The company’s flagship product is a virtual server called a Droplet that features solid state drives. DigitalOcean Droplets provide a range of memory and CPU configurations as well as predictable, pay-as-you-go pricing and a portfolio of tools that facilitate collaboration, backups and security, monitoring capabilities, block storage, APIs, and networking functionality. In May 2018, DigitalOcean introduced a fully managed Kubernetes service that takes responsibility for provisioning and managing Kubernetes clusters.

DigitalOcean is known for simplified developer experiences that let developers deploy a Droplet in less than a minute and a Kubernetes cluster in less than five minutes. DigitalOcean’s Managed Kubernetes offering illustrates the company’s recognition of the importance of cloud-native development to contemporary application development. The service focuses on the developer experience by prioritizing simplicity, speed, and transparency as they relate to the process of provisioning, managing, and monitoring a Kubernetes cluster.

The master node in DigitalOcean’s Managed Kubernetes service is free of charge, which means developers pay only for worker nodes. Developers can use the service to automate application deployments as well as continuously deliver application updates and enhancements, with zero downtime. Kubernetes takes responsibility for optimally scheduling application operations to maximize resource efficiency and minimize cost. In addition, DigitalOcean’s Managed Kubernetes service manages the horizontal scaling of pods to ensure the availability of adequate instances of application services to accommodate the needs of end users. It also takes responsibility for the self-healing of the cluster and supports the deployment of Kubernetes-based applications to any cloud or environment that similarly supports the framework.

The simplified user experience enabled by DigitalOcean's Managed Kubernetes service enables developers to use Kubernetes without relying on IT operations engineers to support deployment, scaling, monitoring, and application updates and releases. This innovation promises to democratize Kubernetes development to developers and subsequently accelerate its adoption by opening the door for enterprise developers to more actively leverage Kubernetes for cloud-native development use cases.

IDC believes that DigitalOcean's Managed Kubernetes offering not only underscores the company's commitment to a simplified developer experience but also demonstrates the company's ability to provide infrastructure solutions that cater to the needs of enterprises. Given that the complexity of the Kubernetes-related developer experience remains one of the key obstacles to enterprise adoption, DigitalOcean's offering represents an important breakthrough in the landscape of Kubernetes-related development tools because of its ability to simplify, enrich, and streamline the developer experience for deploying and managing Kubernetes clusters.
Challenges
DigitalOcean’s Managed Kubernetes service builds on experience understanding and serving the needs of developers. The company’s Managed Kubernetes offering illustrates DigitalOcean’s understanding of the developer experience by allowing developers to rapidly deploy, provision, and manage clusters and their associated applications. That said, DigitalOcean will need to overcome its perception as a developer-focused company to gain traction within the enterprise with respect to its Kubernetes offering. Enterprises are likely to perceive the offering as a service that caters to the needs of individual developers and start-ups. To gain market share, DigitalOcean will need to sharpen its messaging with respect to its capabilities to deliver enterprise-grade customer service and high availability.

Another challenge is the relative scarcity of cloud-native developers. Few developers can boast fluency in Kubernetes, let alone the triad constituted by the conjunction of containers, microservices, and Kubernetes. DigitalOcean’s Managed Kubernetes service specializes in simplifying Kubernetes deployment in ways that give it a competitive advantage with respect to the challenge of developer adoption of Kubernetes. The ability to simplify the developer experience should prove especially useful given the complexity of the developer experience specific to Kubernetes at present.

Conclusion
Cloud-native applications represent a particularly useful paradigm for building applications that empower developers to make smaller changes more frequently and to quickly evolve applications to incorporate the features desired by customers. The increased automation enabled by Kubernetes allows developers to focus on writing code, while Kubernetes takes responsibility for the scaling, self-healing, and provisioning and deployment of containers. Even though Kubernetes represents a powerful tool for the deployment and management of cloud-native applications, it remains heavily oriented toward IT operators at present and is less accessible to developers because of the complexity of its developer experience. Managed Kubernetes services that simplify and streamline the use of Kubernetes promise to enrich the development of cloud-native applications and transform Kubernetes from a technology used primarily by operators to a technology that is actively used by developers as well.

About the Analyst

** Arnal Dayaratna, Research Director, Software Development  

Dr. Arnal Dayaratna is Research Director, Software Development at IDC. Dr. Dayaratna focuses on software developer demographics, modalities of software development, trends in programming languages and other application development tools, and the intersection of these development environments and the many emerging technologies that are enabling and driving digital transformation. Dr. Dayaratna’s research examines how the changing nature of software development relates to broader trends in the technology landscape.**