# 2020 Trends to Watch: Cloud-Native Development

## Kubernetes impacts every sphere of cloud-native computing

Publication Date: 17 Dec 2019 | Product code: INT003-000416

Michael Azoff

# Summary

## Catalyst

As large enterprises grapple with digital transformation, there is confusion around what defines cloud-native computing and how to achieve it. There is even more confusion around Kubernetes and how to create a complete Kubernetes technology stack, as the ecosystem shows every sign of getting larger year on year, with new "essential" projects going live. This report provides Omdia's definition of cloud-native computing and navigates through the latest expansion of the Kubernetes ecosystem, and what to look out for in 2020, including the impact of Kubernetes on DevOps. It also explains the relevance of cloud-native computing to artificial intelligence (AI), and how cloud-native computing and edge computing overlap. Finally, Omdia provides a review of which programming languages are trending and gaining developer mindshare.

## Omdia view

Omdia's definition of cloud-native computing is prescriptive: the first foundation is to use agile methodologies in development, building on this with DevOps adoption across IT and, ideally, in the organization as well, and using microservices software architecture, with deployment on the cloud (wherever it is, on-premises or public). Omdia's view is that the DevOps component is essential as it creates the continuous delivery pipelines that makes deploying microservices in containers possible at scale.

In contrast, the latest definition from the Cloud Native Computing Foundation (CNCF), cloud-native definition v1.0, has switched to being largely descriptive of the benefits:

"Cloud-native technologies empower organizations to:

- **Scale:** Build and run scalable applications in modern, dynamic environments. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.
- **Create:** These techniques enable loosely coupled systems that are resilient, manageable, and observable.
- **Enable:** Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.
- **Use:** Run on public, private, and hybrid clouds."

The trouble with just describing benefits is that it leaves novices clueless as to how to achieve those benefits. We hope reading both Omdia's and the CNCF's definitions will prove helpful – as mentioned, agile and DevOps are important foundations for cloud-native computing and to assume the world is now agile and DevOps (as opposed to still trying) is optimistic for organizations not in the high-tech business.

Serverless computing also fits in with the above definitions. The Knative open source project, currently managed by Google, continues to carry high hopes for bringing portability to serverless computing. This is important for large enterprises that do not wish to be locked into one public cloud provider, and instead be able to move serverless applications across different providers: AWS

Lambda, Google Cloud Functions, and Microsoft Azure Functions. Google says that by the end of 2020 these features will be available in Knative.

The Kubernetes technology stack continues to grow because, as more microservices-based applications reach production, new challenges emerge. As cloud native solves old problems, it also introduces new ones. Software engineers and architects must decide when is the right time to build a cloud-native application and when to stay with the traditional monolith (which may range from an advanced service-oriented architecture to a simple monolith). There are overheads in the supporting infrastructure for microservices; for example, it would be overkill to supply a simple application with a service mesh.

As enterprises build more AI applications, they will want to connect this with their cloud-native computing. The advantage of Kubernetes to AI is that it decouples the application from infrastructure, and this abstraction lets data scientists and machine-learning (ML) experts focus on the application while Kubernetes takes care of how it is deployed and its needs for execution. However, ML applications are more complex than traditional software applications; for example, there is a complex connection between the algorithms and the different data sets (train, test, validation, and production) over the application lifecycle, and the need to monitor performance and prevent model drift. This is where open source projects such as Kubeflow fill the ML lifecycle gap on Kubernetes.

Finally, on the general programming language front we see the phenomenal rise in popularity of Kotlin, which targets the JVM, JavaScript, or native code (using LLVM). First created in 2011 by JetBrains, it doubled usage year on year until Google adopted it in 2017 as a fully supported official Android language and assured its continued rise. Java has also been modernized for the cloud-native world with much-needed acceleration through open source projects GraalVM and Quarkus.

## Key messages

- Kubernetes is the central force driving the cloud-native ecosystem.
- Edge computing grows as an extension of cloud-native computing.
- Kotlin language popularity rises, and Java is not dead, again.

# Recommendations

## Recommendations for enterprises

Most organizations are at some point on their digital transformation journey and cloud-native computing will play a part in that. To get the most out of the cloud, applications should be cloud native. However, most large enterprises have a legacy application backlog, so the question for them is how to modernize and integrate the new with the old. We hear that greenfield projects are typically being built from the ground up using cloud-native computing. Omdia's recommendations are therefore to use this experience as a launchpad for legacy modernization.

It is surprising how many organizations want to move to microservices, containers, and Kubernetes without having the foundation pieces in place: agile, DevOps, and continuous delivery. Therefore,

Omdia's strongest recommendation is to "walk before you can run." The whole ecosystem of cloud-native computing is evolving so rapidly that there is merit in waiting for the technology to stabilize and mature and to adopt at a point where mistakes have been left behind and there has not been investment in earlier incompatible incarnations. Just this year the CNCF graduated projects in numbers; last year only Kubernetes was graduated. However, we do recommend that enterprises keep a watchful eye and have some early stage involvement in order to build up their expertise at a DIY level. However, this does not mean waiting on the sidelines to be active with cloud-native computing. Rather, Omdia recommends choosing a container platform that is built to be opinionated and therefore easy to use, removing many admin chores and simplifying the journey to deploying cloud-native applications.

## Recommendations for vendors

A key message in this report for vendors active in the cloud-native computing space is that their products need to be Kubernetes native or they will suffer in the market to competitors that are. This spans every application area from ML applications, through DevOps tooling and container platforms, to edge computing.

Another message is that we still find many large enterprises struggling with DevOps and this inhibits their ability to become cloud native. The solution is tools that are simpler to use and easier paths to becoming cloud native, such as using container platforms. The projects showcased at CNCF/KubeCon meetings are at the bleeding edge and enterprises need a far gentler path to achieve the benefits of cloud-native computing that they see being gained by the internet giants.

# Kubernetes is the central force driving the cloud-native ecosystem

## The Kubernetes ecosystem is too large and needs rationalization

Each user conference of the CNCF is larger than the previous one. The recent US meeting in San Diego, November 18–21, 2019, had 12,000 attendees, and the number of new members surpassed 500. The number of graduated projects has also increased, from one a year ago (Kubernetes) to eight currently (add Prometheus, Envoy, CoreDNS, containerd, Fluentd, Jaeger, and Vitess). Expect to see more added in 2020 as incubating projects (there are currently 15) graduate. Graduate status carries an "enterprise-ready maturity" label (for CNCF's definition see: https://github.com/cncf/toc/blob/master/process/graduation_criteria.adoc).

As CNCF reminds everyone, it is a vendor-neutral organization designed to nurture the cloud-native movement and its purpose is not to define a technology stack. It is not a kingmaker, and within its membership there are competing projects. This is fine and plays a useful role, but it does pose a problem for the rest of the world as it tries to make sense of this movement and navigate the projects. CNCF does have a landscape map of the projects but it has reached beyond being helpful: the graphic is no longer legible without zooming in. Part of the problem is that the field is continually

progressing, and useful, maybe essential, pieces are being added to the ecosystem year on year. On top of that, rationalization is in progress: CNCF listed 34 acquisitions among its members since May 2016.

The CNCF steering committee has now said that it is moving away from a first-come, first-served model to assessing projects that fill a gap in the landscape.

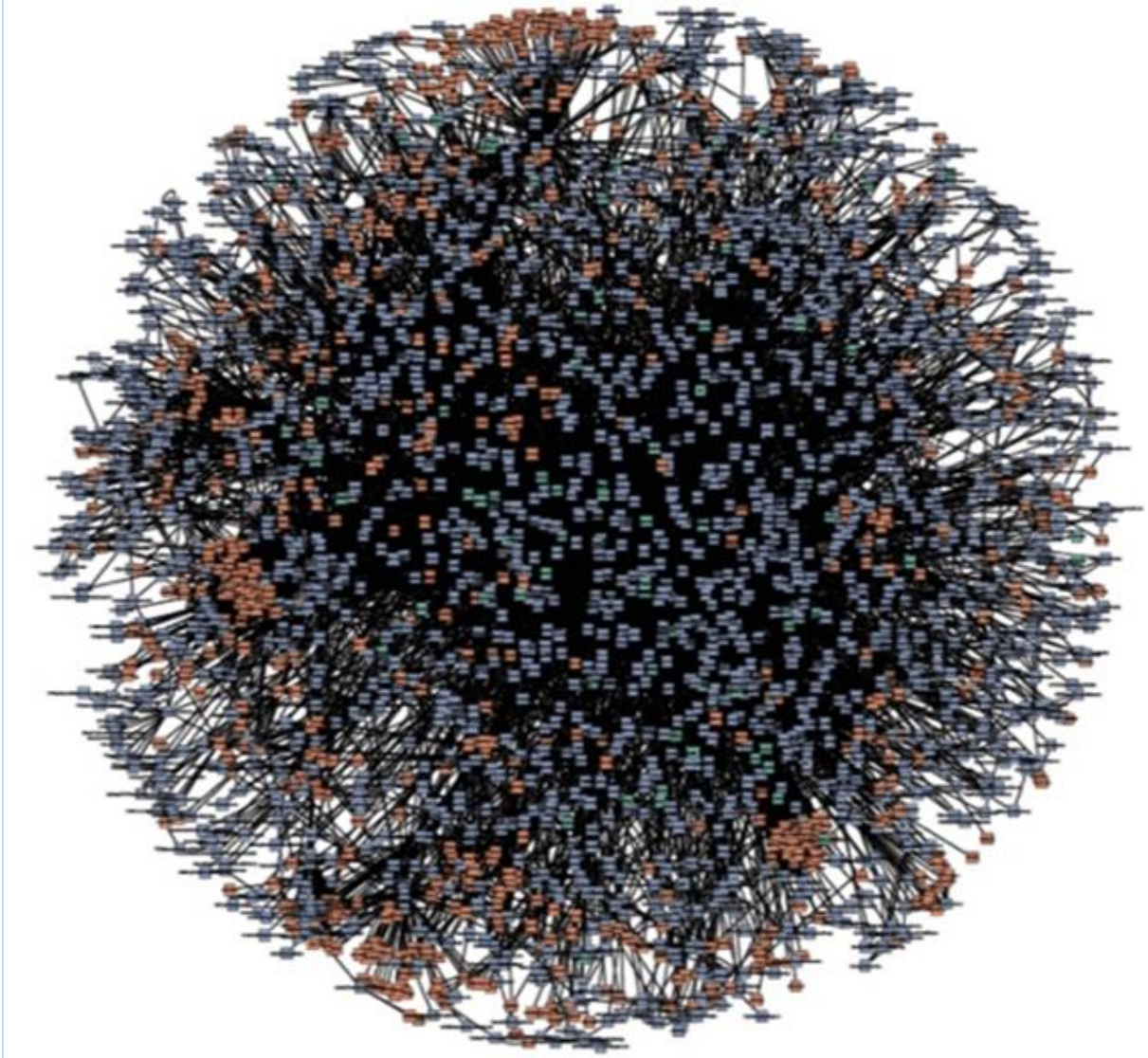
Here are a few examples of meetings and CNCF projects that attracted Omdia's attention at the San Diego event:

- **Certified Kubernetes Conformance.** CNCF runs this certification to ensure that every vendor's version of Kubernetes supports the required APIs and will be interoperable with applications built on other certified versions.
- **CloudEvents.** To create a consistent and portable event publishing system, CloudEvents is a specification for describing event data in a standard way, making it easy to declare and deliver events across services and platforms. Currently in early stage development, its working group has received much industry interest, and the end goal is to offer the specification to the CNCF.
- **DeployHub.** One of the reasons for moving away from monoliths to microservices is to avoid the tight coupling of objects in a monolith – a typical diagram illustrating this coupling looks like a hairball. But a diagram of the connections between microservices in a large system such as at Amazon looks just as complicated (see the "Death Star" diagram in Figure 1). DeployHub (a software configuration management solution offering microservices cataloging, versioning, and deployment information) resolves this complexity by creating a cloud-native version of the CMDB but automating the configuration information (metadata) of which microservices connect with which other services. It is a solution to a problem that large cloud-native applications will face. (DeployHub is not a CNCF project but it does have a related open source project called Ortelius.io, which it hopes will become an "incubating" CNCF project.)
- **Network Service Mesh (NSM).** Quoting Ed Warnicke (Cisco), Frederick Kautz (doc.ai), and Nikolay Nikolaev (VMware): "This is a community-driven CNCF Sandbox project that aims to simplify connectivity between workloads, independent of where they are running. It extends an IP reachability domain to workloads running in multiple clusters, legacy environments, on-premises, or in a public cloud, communicating with the protocols they are currently using. NSM works at a workload granularity and brings the service mesh from HTTP down to IP. Applications and application service meshes (e.g., Istio) run unaltered on top, leaving the hybrid/multicloud IP connectivity to NSM."
- **Open Policy Agent.** An open source policy engine and an incubating CNCF project, it provides a declarative domain-specific language for authoring policy across a cloud-native application deployment, from development, to integration, to container clusters in production.
- **Vitess.** Managing data is one of the more difficult areas in cloud-native applications. Vitess offers a MySQL database solution that can scale easily and, for migration purposes, be moved across any Kubernetes environment. First launched in 2010, this CNCF project graduated in 2019, and has some notable users; for example, all YouTube database traffic since 2011.

CNCF also has an end-user community (which uses cloud-native technology but does not sell any cloud-native services externally), and this currently stands at 128 companies. Benefits of membership include sharing best practices, voicing concerns directly to CNCF, attending meetings, recruiting talent, and staying educated. There are also CNCF user groups set up for telecoms, financial services, and research.

**Figure 1: Structure of microservices at Amazon**



Source: Divante (see https://divante.com/blog/10-companies-that-implemented-the-microservice-architecture-and-paved-the-way-for-others)

# The rise of Kubernetes-based container platforms

Given the difficulty of navigating the cloud-native ecosystem, especially the one around Kubernetes, there is a high demand for easy-to-administer development platforms that deliver applications in Kubernetes-managed containers. This is the space of Red Hat OpenShift, Pivotal Platform, Cloud

Foundry, Rancher, Canonical, and more. However, two new players have entered this segment of the market: HPE and Mirantis.

First, HPE launched the HPE Container Platform at the San Diego event. Kubernetes based, it is designed to run on any bare metal or virtualized infrastructure from the edge and to the cloud. What is different about HPE's product compared with rivals is its built-in tooling for managing stateful workloads, built on the acquisitions of BlueData (November 2018) and MapR (August 2019). BlueData provides containerization for ML and data analytics workloads, with automated provisioning and management across large-scale distributed environments. MapR adds an enterprise-grade, data fabric and persistent container storage for ML, data analytics, IoT, and other use cases. The platform also leverages BlueData's open source KubeDirector project for complex stateful workloads.

Mirantis is in the process of acquiring the Docker Enterprise platform from Docker; this complements its Kubernetes-as-a-service offering, which will be merged with the Docker platform. The rise of Kubernetes has weakened Docker and illustrates how essential it is today to be a part of the Kubernetes ecosystem, and Mirantis allows that integration.

The acquisition of Pivotal by VMware and the introduction of Kubernetes into the Pivotal Platform is another indication of how essential it is today to be Kubernetes native.

In December 2017, Canonical and Rancher partnered to offer a joint Kubernetes Cloud Native Platform, combining Canonical's distribution of Kubernetes (which is always no more than two weeks behind the latest open source version) and Rancher. Canonical also offers a lightweight Kubernetes, called MicroK8s, which is a fully conformant (i.e., certified) Kubernetes version. MicroK8s is positioned for offline development, prototyping, and testing, as well as being ideal for IoT, edge, and appliance use cases. For multicloud container clusters, Canonical offers Charmed Kubernetes, making use of Charm operations automation, and suitable for bare metal or virtualized environments.

Omdia expects to see the battle for market share of enterprise-grade ease-of-use Kubernetes platforms to heighten in 2020, as large enterprises build more cloud-native applications and struggle to navigate the DIY world of CNCF-style Kubernetes projects.

# The Kubernetes-friendly ML lifecycle

Dominating the Kubernetes-native ML lifecycle tools is Kubeflow. Kubeflow is open source and currently project managed by Google, which donated it to the community. The project describes it as a multi-architecture, multitenancy, multicloud framework for running ML pipelines. The aim is to decouple data scientists and ML experts from cloud-native infrastructure concerns, letting them focus on AI applications. Accepting again the premise that Kubernetes has won, and cloud-native applications are going to run on Kubernetes-native infrastructure, in 2020 there will be a scramble for all ML lifecycle tool vendors to add Kubeflow compatibility if they have not already.

# DevOps and the role of Tekton

The practical side of DevOps is continuous integration (CI) and continuous delivery (CD) capability. The original tooling in this space has largely addressed moving monolithic applications into production along CD pipelines, but the advent of cloud-native computing has made CI/CD an

essential foundation for deploying containers. The next step for the CI/CD tool vendors is to make their tools Kubernetes native to ease the deployment of workloads and containers into Kubernetes environments, especially at scale, with possibly thousands of containers. This challenge is being addressed at the Continuous Delivery Foundation (CDF), an offshoot of the Linux Foundation.

CDF is currently home to four CI/CD open source projects: Jenkins, Jenkins X, Spinnaker, and Tekton. Tekton is gaining interest with its focus on being Kubernetes native (the project was donated by Google). Its aim is to provide Kubernetes-style resources for declaring CI/CD pipelines. The project is still in an early stage of development but has contributors such as Google, IBM, and Red Hat on board and we expect to hear more about Tekton in 2020.

# Edge computing grows as an extension of cloud-native computing

## Choosing regular or lightweight containers for the edge, or both

The landscape in edge computing is also taking shape; Linux Foundation (LF) Edge is acting as an umbrella organization for many projects, such as EdgeX. The anticipation of general 5G rollout is spurring industry players to position themselves on the edge. However, it is clear that it will not be the extreme edge and the cloud; rather, there will be a spectrum of edge gateways and midway points where aggregation and computing will take place. Computing at the edge will filter and process live “big data” streams and forward to a next level for further compute and aggregation. The results may go back to the extreme edge, or to a next higher-level gateway, or maybe to the cloud. However, there are many use cases where going direct to the cloud causes too many latencies. We also see hardware chip vendors, whether CPU or accelerators, increasingly addressing edge computing, especially for AI workloads in inference mode.

The players in the market are beginning to stake out their positions. Omdia spoke with Red Hat at the San Diego event, and it explained that its strategy in edge computing was to develop this capability on OpenShift for cloud-native applications, which will deploy standard containers to the edge. LF Edge is host to several open source projects that target the edge. Project Edge Virtualization Engine (EVE) deploys a hypervisor to bare metal hardware that will run orchestration services and lightweight containers. LF Edge is also host to EdgeX Foundry, which is building an IoT/edge open framework for interoperable plug-and-play microservices-based components.

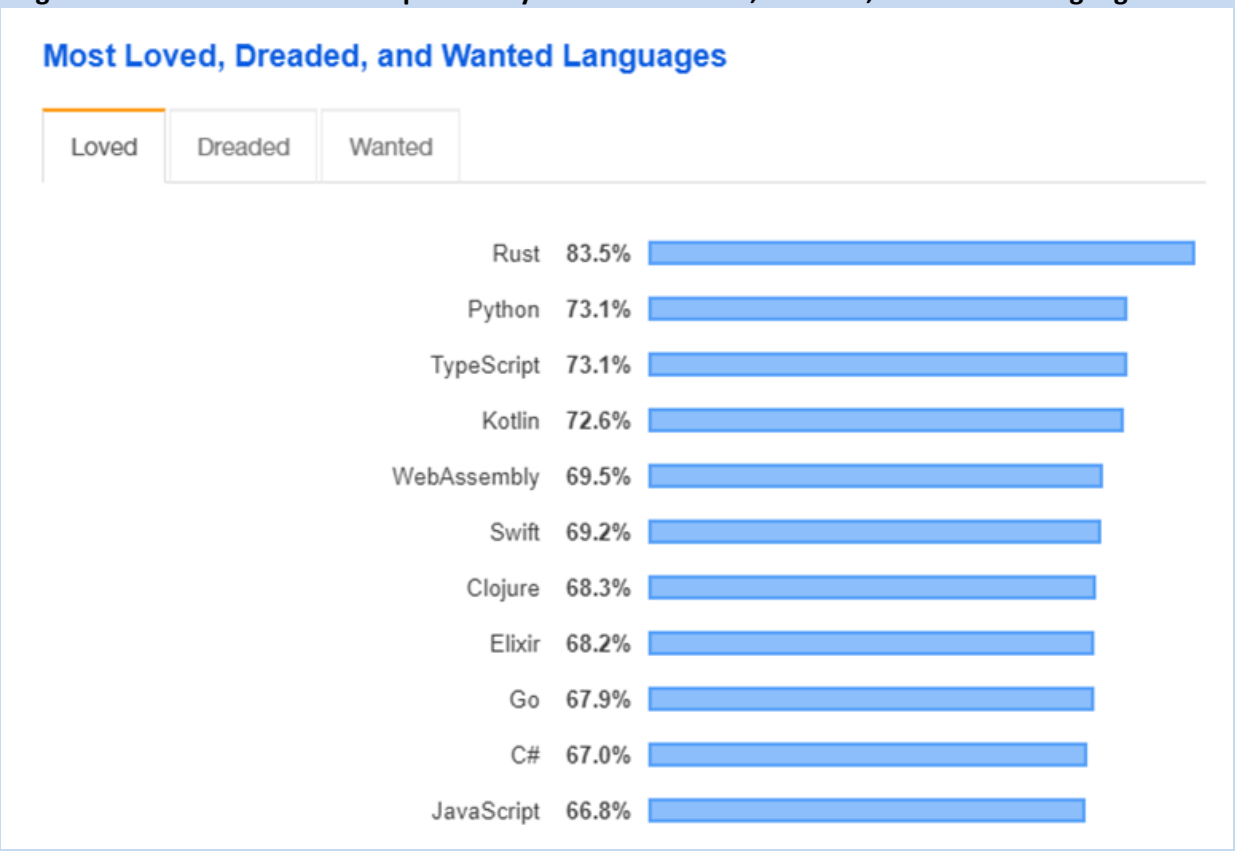
 

# Kotlin language popularity rises, and Java is not dead, again

## Kotlin is fourth most loved language

The developer website, Stack Overflow, runs an annual developer survey, which includes the question on "most loved/dreaded/wanted" programming language. Most loved is defined as "percentage of developers who are developing with the language or technology and have expressed interest in continuing to develop with it." Kotlin was rated second in the 2018 survey and fourth in the 2019 survey – see Figure 2 (Rust was first in both surveys). Developers tend to desire new green grass, especially if it is in demand and the salaries are commensurate. Another language popularity index is the TIOBE index (www.tiobe.com/tiobe-index/), which currently places Kotlin at number 35, and has seen it rise fast over the last few years.

**Figure 2: Stack Overflow Developer Survey 2019: Most loved, dreaded, and wanted languages**




Source: Stack Overflow

The question you ask, no doubt, is why bring it to your attention? Kotlin is a JVM language, so it benefits from the Java libraries and will natively integrate with Java code – this is a big plus in the enterprise world, which has a huge amount of legacy Java. Having been chosen by Google in 2017 as a first-class Android language, it is proving popular for building Android apps, taking less code than Android Java to achieve the same end. It is also safer – Kotlin was designed from the ground up to deal with a cause of 70% app crashes in Java: NullPointerException (we won't go into how it deals

with nulls here but suffice to say it avoids null-related crashes.) The language has evolved beyond the JVM – the originator, JetBrains, has released Kotlin JS (which compiles Kotlin to JavaScript), so can be used for user interface development. It has also released Kotlin/Native, which compiles directly to machine code using LLVM without needing a virtual machine.

Kotlin is one of a new generation of languages, making company with Clojure, Go, Rust, and Swift, it is designed to do more with less code. Omdia believes 2020 will see Kotlin rise further in popularity, and although Kotlin now has wider appeal than just for the Java community, we think it is still worth note for Java shops.

## Java has been accelerated for cloud-native computing

Red Hat announced the release of Quarkus 1.0 at the end of November 2019. Quarkus accelerates the startup time of Java applications – essential for cloud-native applications that may spin up and down a microservice over milliseconds and cannot endure traditional load times. It is Kubernetes native and works with Java standards, frameworks, and libraries, including Eclipse MicroProfile, Apache Kafka, RESTEasy (JAX-RS), Hibernate ORM (JPA), CDI, Spring, Infinispan, and Camel.

Related to Quarkus is the open source project from Oracle, GraalVM, a Java VM and JDK based on HotSpot OpenJDK. The first production-ready version of GraalVM 19.0 was released in May 2019. It implements ahead-of-time compilation for speeding up execution and, together with Quarkus, shows how the Java community is bringing Java into the cloud-native age.

Therefore, Java is not dead for the time being and, as we always say, it is more than just a programming language – it is a programming platform. With so many enterprises invested in Java, it is good to see the community bringing about its evolution.

# Appendix

## Methodology

Omdia has talked with technology vendors, consumers, and independent third parties to understand the state of the software development market. Together with its decades of experience in the field researching and consulting, Omdia can identify the key patterns emerging.

## Further reading

*2020 Trends to Watch: Cloud Computing*, INT003-000402 (October 2019)

## Author

Michael Azoff, Distinguished Analyst, IT Infrastructure

michael.azoff@omdia.com

## Citation Policy

Request external citation and usage of Omdia research and data via citations@omdia.com.

# Omdia Consulting

We hope that this analysis will help you make informed and imaginative business decisions. If you have further requirements, Omdia's consulting team may be able to help you. For more information about Omdia's consulting capabilities, please contact us directly at consulting@omdia.com.

# Copyright notice and disclaimer

## CONTACT US

omdia.com

askananalyst@omdia.com