Cloud-Native Applications—The Journey Continues

Cloud native has been gaining good momentum through established alliances, foundations, and conferences. So, it is not a buzzword any longer. The main reason to develop cloud-native applications is to exploit cloud features, such as up/down autoscaling and autoprovisioning.

Looking at what is out there in terms of what the term cloud native really means, we find heterogeneity. As expected, there is no one single definition. The reason is likely because the field is evolving and organizations or companies try to define it in a way that fits their strategy or portfolio. But what we all can agree on is that cloud native applications will take some form of Distributed Architecture. Here are a couple of example definitions in the market. The Linux Foundation, which seems to be active in this area, launched the Cloud Native Computing Foundation (CNCF) in December 2015. CNCF’s mission is to “create and drive the adoption of a new computing paradigm that is optimized for modern distributed systems environments capable of scaling to tens of thousands of self-healing multi-tenant nodes”. Further, its main push is containers, Microservices, and all the orchestration around them. This is more of an architectural view of Cloud Native. Pivotal defines cloud native as a framework or an umbrella term that includes Composable Architectures, Automated Processes, Collaborative Culture, and Structured platform; meaning, Pivotal views it as more than architecture. Regardless, the one that has the most comprehensive and easy to use orchestration framework will likely garner much market share the faster.

We have had cloud-native applications for some time now: the likes of Netflix, Twitter, Facebook, and Microsoft Office365. We have been seeing that cloud providers such as Microsoft, Amazon, and Google have realized there will be a great demand for cloud-native applications. For that reason, they are providing in their cloud offerings ingredients such as containers, orchestration tools, and frameworks to design microservices, to build cloud native applications. For example, in Azure, Microsoft is supporting containers, has Azure service fabric, and Azure app services. The Azure service fabric is a distributed architecture that allows building microservices.

One question I have been asked is whether developing cloud native applications will resolve the cloud interoperability issue and allow or ease portability. Clearly the answer is “not really”. If you are going to develop cloud-native application on, for example, Microsoft Azure, it will rely on all what Azure supports in terms of containers’ orchestration. So, moving the applications to Amazon Web
Services (AWS) will clearly not work or be at least be a major pain to modify before it can run on AWS. Another question I usually hear is, “Will it be possible to convert legacy applications to become cloud-native?” The answer is, definitely yes. It all depends on how much time, effort and money you are willing to spend.

Every company should be able to benefit from cloud-native applications. So if you are developing new applications that will be deployed in the cloud, you should seriously think about developing them as cloud native, especially if you are happy with the cloud provider you are working with. But also remember the portability point I made earlier.

Cloud native architectures will bring their own challenges. I do not believe we have a full understanding of the complexities or challenges cloud-native will bring to architectures for networking and/or storage, as well as performance, availability, and reliability.

This special issue of the magazine is on Cloud Native Applications and it is led by Dennis Gannon from the School of Informatics and Computing at Indiana University, Roger Barga from Amazon Web Services, and Neel Sundaresan from Microsoft Corporation. I urge readers to read their introduction as it provides a very informative and detailed overview of Cloud-Native. This is definitely a must read.

The columns in this issue cover cloud-native as well as other topics. The Cloud Technologies column looks at cloud migration for cloud-native applications. Cloud and the law looks at the use of Blockchain for security and privacy for data in the cloud. Cloud Economics looks at the economics of Fog Computing; Joe Weinman refers to it as Fogonomics. Finally, the StandardsNow column went through all the columns that we covered in the last three years and did a comprehensive overview.

With the above in mind, I define a cloud native application as an application that has been built from the ground up as a distributed architecture that leverages every aspect of the cloud for better performance, reliability, scalability, and every other *ility you can think of. All other approaches, such as finding a way to make a legacy application benefit from cloud features, are varieties of cloud native. For example, when transforming a legacy application into microservices that run in containers, it becomes sort-of cloud native as it may not include all the tight connections in its software architecture to exploit the cloud.