Part 1: Why Does Cloud Native Development Matter?

Understanding the elements of a cloud-native development approach

Many aspects of our socio-economic lives are built around applications. That's why businesses and organizations must seek the best way to design, build, and use those applications to stay competitive. Most businesses would agree that cloud-native application development is the embodiment of the most effective approach to building, running, and improving applications.

This is still a theoretical belief for far too many businesses. Many have yet to embark on a cloud-native development approach that can optimize application performance and cloud value. These businesses understand there are challenges to realizing the business benefits of cloud-native app development, which we will tackle a little later.

The bigger challenge is that stakeholders rarely get a clear picture of why cloud-native development matters to the way they run their business. This blog will break down the reasons it matters starting with what it is, what it takes to make it happen, and how CTOs can better explain it to stakeholders to get buy-in that overcomes its initial challenges.

Old and New Cloud-Native Approach

Businesses generally agree that applications should be built to run in a cloud environment. But the cloud-native app development approach isn't clear to everyone. Most legacy apps are the product of dev teams using the traditional waterfall method with a phased release approach that lasts months.

It's true that traditional, monolithic app development is easier to develop, manage, and deploy. These apps become harder to manage and update as they grow because of interdependencies that impact each other. This leads to difficult app updates and failures.

Cloud-native app development takes a modular and flexible agile architecture approach that delivers:

- On-demand horizontal scaling
- Support across multiple environments
- Application portability
- Continuous development and delivery for shorter cycle times

The Cloud Native Computing Foundation (CNCF) defines cloud-native as a means to empower organizations to build and run scalable applications in public, private and hybrid clouds. Continuous integration/continuous deployment (CI/CD) pipelines are the primary workflow used in cloud-native app development along with:

- Immutable infrastructure
- Serverless
- Service meshes
- DevOps process

- Containers
- Microservices
- Declarative application programming interfaces (APIs)

Cloud-native development is all about building and updating apps quickly in a way that takes advantage of the distributed computing model. This makes them more scalable, elastic, resilient, and flexible. To understand why that matters in business terms requires breaking it down to a more business-oriented understanding of the old way of creating applications and the cloud-native approach.

The Key Elements of Cloud-Native Development

For an application to be considered 'cloud native', its architecture and development process will feature all the elements that we've laid out above. But to understand how cloud-native works, we need a basic definition of each of these elements.

Containers

Cloud-native application architecture is based on dynamically orchestrated containers. Containers hold the services that run the application like load balancing and networking. This makes the application portable so it can run in any cloud environment without integration challenges. As an organization's containers grow, Kubernetes Orchestration enables efficient management.

Microservices

Microservices architecture further splits the application into independent parts for running each application process as a service. These services communicate via application programming interfaces (APIs) or messaging where services:

- Perform a single function and operate independently
- Can be designed, tested, deployed, or replaced without affecting other services
- Enable dev teams to scale and make corrections quickly which reduces technology and business risk

As we split more applications into more microservices, service meshes make it easier to handle the challenges of running a service such as monitoring, networking, and security.

Serverless

Serverless is where the cloud provider runs the server and manages machine resource allocation rather than the Ops team. This makes it easier for Dev teams to focus on code rather than the run environment, which:

- Improves development pipeline efficiency and security
- Reduces costs by lowering the need for specialists' expertise in keeping the app running on-prem
- Further reduces server costs based on what is used as opposed to on-prem server/hardware Capex

Continuous Integration / Continuous Delivery (CI/CD)

The continuous integration and continuous delivery (CI/CD) pipeline are the centerpiece of delivering higher quality apps with faster and more stable updates via cloud-native app development. These pipelines use cloud-native tools and services like Jenkins to automatically test and deploy code modifications to the production environment.

Continuous development processes enable the delivery of code changes to a testing or production environment after the build stage. The continuous flow of new updates and features means businesses can improve services for banking customers, healthcare application service providers and eCommerce customers as just three examples. Cloud-native tools like Jenkins automate the DevOps workflow by starting testing when developers modify or add code to the app.

The DevOps culture that underpins these processes and works environment allows development and operations teams to work together towards the goal of high quality, availability, and security of applications delivered and updated quickly. The agile method drives the fast and easy changes and updates by splitting the development process into time windows with a continuous feedback loop. The result is faster, more effective, and innovative AppDev where dev and op silos are eliminated to get the app from design to minimal viable product (MVP)as fast as possible.

In part two of this blog, we'll look at how cloud-native development benefits businesses in terms of applications used for products, services, operations, and customer experience.