EBOOK

IT Perspectives: Implementation of DevOps Platform

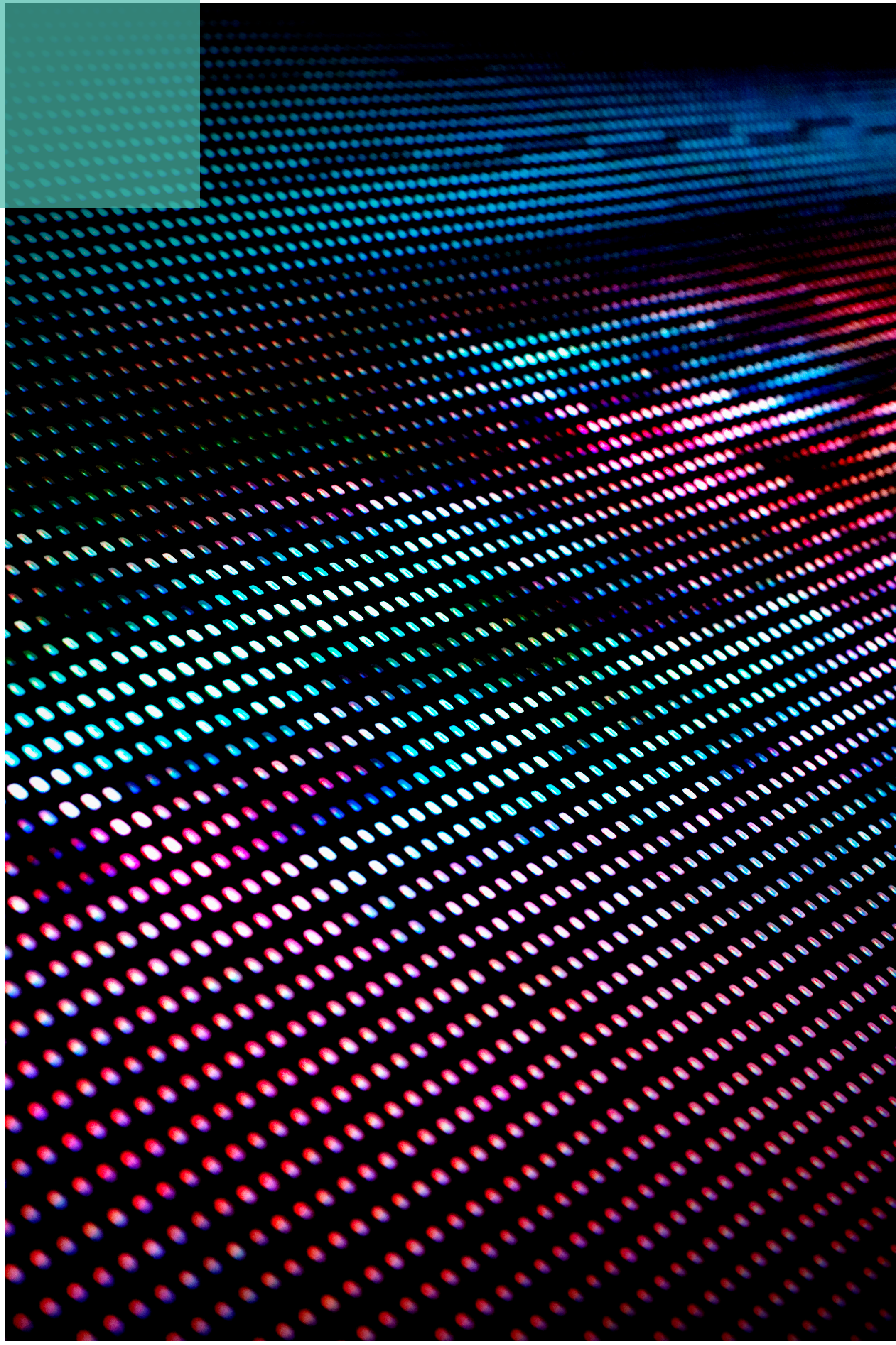
NetApp

Table of Contents

Introduction

With a desire to automate the rapidly delivery of production software changes without disruption, NetApp implemented a DevOps platform called “CloudOne” with the necessary automation and standardization to allow for frequent small production changes to business applications. The implementation allows software developers to push and pull source code, choose applications stacks, run CI/CD workflows, and rapidly produce business results.

This eBook covers how we implemented the necessary automation and standardization to allow for frequent small production changes to our business applications. We use automation as a way to improve efficiency, eliminate human error, and reduce the number of staff and man-hours needed to delivery environments.



Implementation of a DevOps Platform Inside NetApp

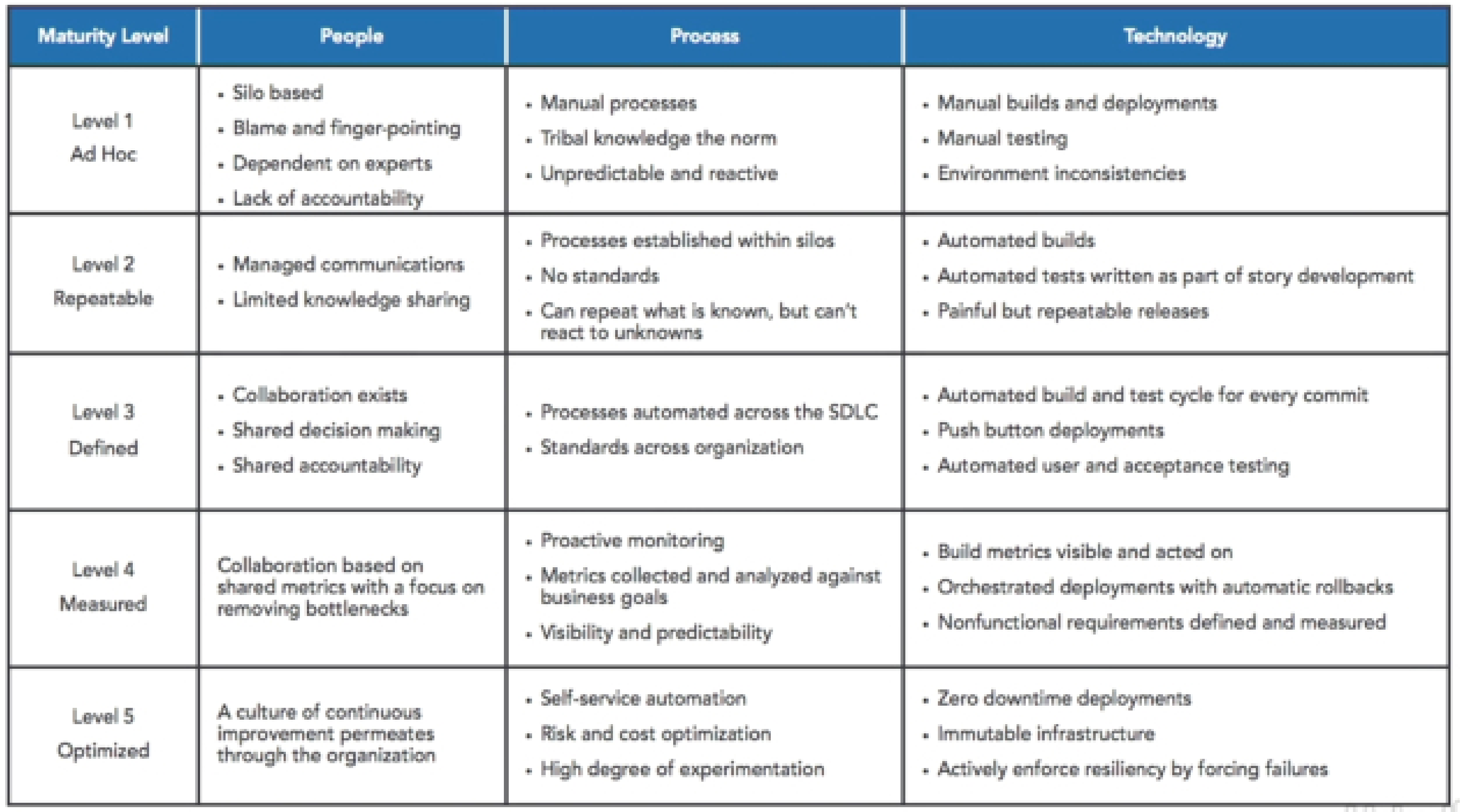
MICHAEL MORRIS
SR. DIRECTOR, IT INFRASTRUCTURE

In our ebook, “Vision for a DevOps Platform,” we detailed NetApp IT’s path to a software-defined, cloud-based solution with end-to-end DevOps workflow automation. We call this CloudOne. As the senior director for technology, strategy, and innovation within the infrastructure team, my role is to build and run the technology platform to allow our software developers to use DevOps.

CloudOne includes three major environments: developer tools, platform software, and Infrastructure-as-a-Service. The DevOps workflows are automated across all three environments. This allows software developers to push and pull source code, choose application stacks, run CI/CD workflows, and rapidly produce business results. Ultimately, developers are expected to create cloud-aware applications on CloudOne, which uses microservices architectures, running in containers, in infrastructure-agnostic designs so they are cloud-portable between public cloud providers and private cloud. Inside CloudOne, we can route application workloads to whatever cloud we want to use.

DevOps Maturity Model

We find the DevOps Maturity model posted by Sumardi Soemartopo closely aligns to our experience. Level 1 is ad hoc—meaning no DevOps other than saying the word, to a Level 5 similar to what Netflix and Facebook do with constant updates and no noticeable downtime. With CloudOne, NetApp IT meets a Level-3 classification, i.e. we have real, automated DevOps which can create, deploy and support real applications. And yet, our capabilities, experiences, and speeds are still immature. Reaching Level 5 will require a multi-year journey to improve CloudOne features, capabilities and human skillsets.



Maturity Level	People	Process	Technology
Level 1 Ad Hoc	• Silo based • Blame and finger-pointing • Dependent on experts • Lack of accountability	• Manual processes • Tribal knowledge the norm • Unpredictable and reactive	• Manual builds and deployments • Manual testing • Environment inconsistencies
Level 2 Repeatable	• Managed communications • Limited knowledge sharing	• Processes established within silos • No standards • Can repeat what is known, but can't react to unknowns	• Automated builds • Automated tests written as part of story development • Painful but repeatable releases
Level 3 Defined	• Collaboration exists • Shared decision making • Shared accountability	• Processes automated across the SDLC • Standards across organization	• Automated build and test cycle for every commit • Push button deployments • Automated user and acceptance testing
Level 4 Measured	Collaboration based on shared metrics with a focus on removing bottlenecks	• Proactive monitoring • Metrics collected and analyzed against business goals • Visibility and predictability	• Build metrics visible and acted on • Orchestrated deployments with automatic rollbacks • Nonfunctional requirements defined and measured
Level 5 Optimized	A culture of continuous improvement permeates through the organization	• Self-service automation • Risk and cost optimization • High degree of experimentation	• Zero downtime deployments • Immutable infrastructure • Actively enforce resiliency by forcing failures

DevOps Implementation Lessons

We learned three valuable lessons when addressing DevOps implementation challenges.

1) **Getting the base CloudOne environment built and ready.** It was a year-long effort from initial vision to first release. We realized early on that it takes time because of the complexity to get DevOps workflows automated across developer tools, platform software, and IaaS.

2) **Making sure we didn't build a DevOps ghost town our app development teams would not use.** Concurrently with the base CloudOne build, we educated the app development teams on cloud-aware apps and DevOps. By taking the time to provide an education focus, we were able to establish a more trusted working relationship between the infrastructure and business apps teams to create a pipeline of new and existing apps to build or migrate to CloudOne.

3) **Keeping key engineers focused.** My new favorite analogy of CloudOne is to think of it like a train which rolls down a never-ending track, taking on new cars (or features) and passengers (applications) with no end in sight. CloudOne is a service which will go on for years, so the engineers who have participated in the initial build need to stay engaged throughout. We are creating a CloudOne operating model using agile practices and frequent releases to maintain engagement.

Role of NetApp Technology

As Customer-1, our goal is to use NetApp products, services, and reference designs early and as much as possible. We also spend a significant amount of time speaking to the NetApp product teams, giving our feedback on products, sharing good and bad implementation results, and influencing new features and roadmaps. In this way, we contribute to improving NetApp's customer experience and satisfaction.

CloudOne uses NetApp ONTAP Ansible playbooks for infrastructure automation and the NetApp Trident plug-in for volume creation in OpenShift. For our private cloud we are using these NetApp technologies:

- Hyperconverged Infrastructure (HCI) to provide compute, memory and (local) storage for containerized application components
- All Flash FAS (AFF) for the stateful (NFS, iSCSI) storage platform for databases and high-performance workloads
- StorageGRID as the repository for heavy object storage usage given cloud-aware apps are inherently stateless and should use a stateless storage mechanism.

These cloud-friendly technologies are tangential to our existing install-base products and are particularly important to our cloud journey. Ultimately, all DevOps tools, software, and applications created by DevOps workflows need infrastructure. Taking an infrastructure product, simplifying it, and making it software-defined has allowed us to plug into DevOps environments like CloudOne. That's the future of IT—inserting good products into existing DevOps environments through APIs and automation. •

Day 1 Onboarding Applications

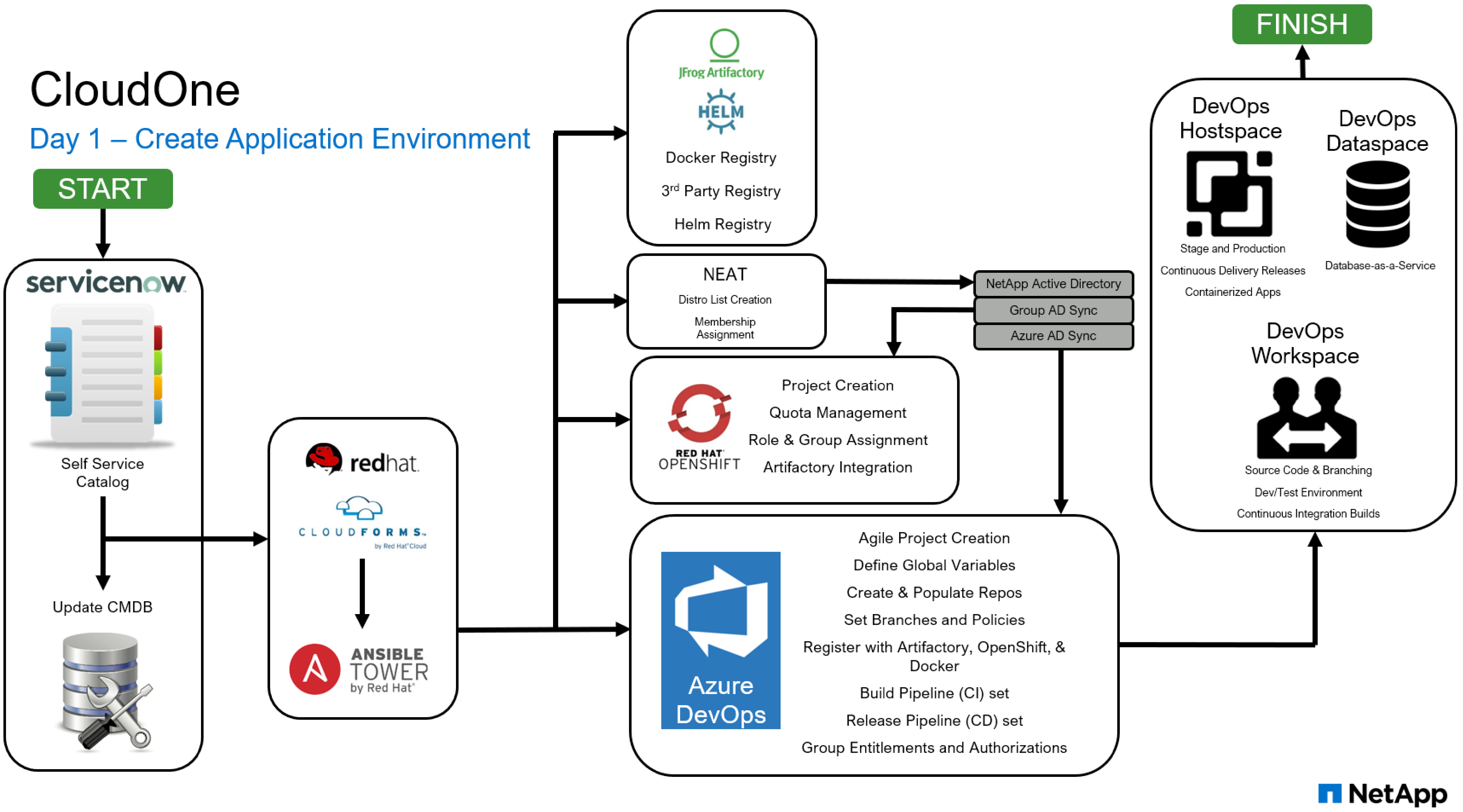
MOHAN RAJ
IT SR. MANAGER, CLOUDONE DEVELOPER EXPERIENCE

Over the past 18 months I have been part of an exciting NetApp IT initiative to build a DevOps platform that provides the cloud services, automation, and CI/CD release models that our application development teams need to build cloud native applications. We call the platform CloudOne, as it provides one consistent developer experience, irrespective of the destination being private or public cloud.

My team automated the processes to onboard applications onto the platform as well as the continuous integration, continuous deployment (CI/CD) process to rapidly move application changes to production. The onboarding process starts with when a development team goes to our Self-Service Catalog (on ServiceNow) to identify the type of application being onboarded, and making selections around type of application stack and environment sizing requirements of their application. This starts an automated workflow that updates the CMDB—our single source of truth—so that we don't lose track of any assets that get created in the onboarding process.

CloudOne

Day 1 – Create Application Environment



Above: *NetApp IT's CloudOne DevOps platform Day 1 process.*

Using APIs, ServiceNow then hands off to Red Hat CloudForms, which invokes Ansible automation. Ansible Tower to perform four key routines.

1. Create the repository for all binaries using jFrog Artifactory, the place where we store all of the binaries get created during the application CI/CD process. I. It is during this step that the Docker registry, as well as third party registry and Helm registry, are established for the application. Helm is used to manage Kubernetes applications.

2. Create email distributions lists (through Active Directory) to do role-based authorizations for OpenShift and Azure DevOps. Our lists are created using an internal tool called NEAT, NetApp Enterprise Action Tool. It is important that we have clearly defined roles embedded into the process so that changes are not made in production without proper approval. It is our embedded checks-and-balances based on preassigned responsibilities and accountabilities.

3. Configure the Red Hat OpenShift environment which creates and manages the containers. It manages the quotas, assigns roles and groups, and integrates Artifactory.

4. Configure Azure DevOps Project to create the developer environment including everything from code repositories to code branching to the CI/CD process itself that gets created for the application

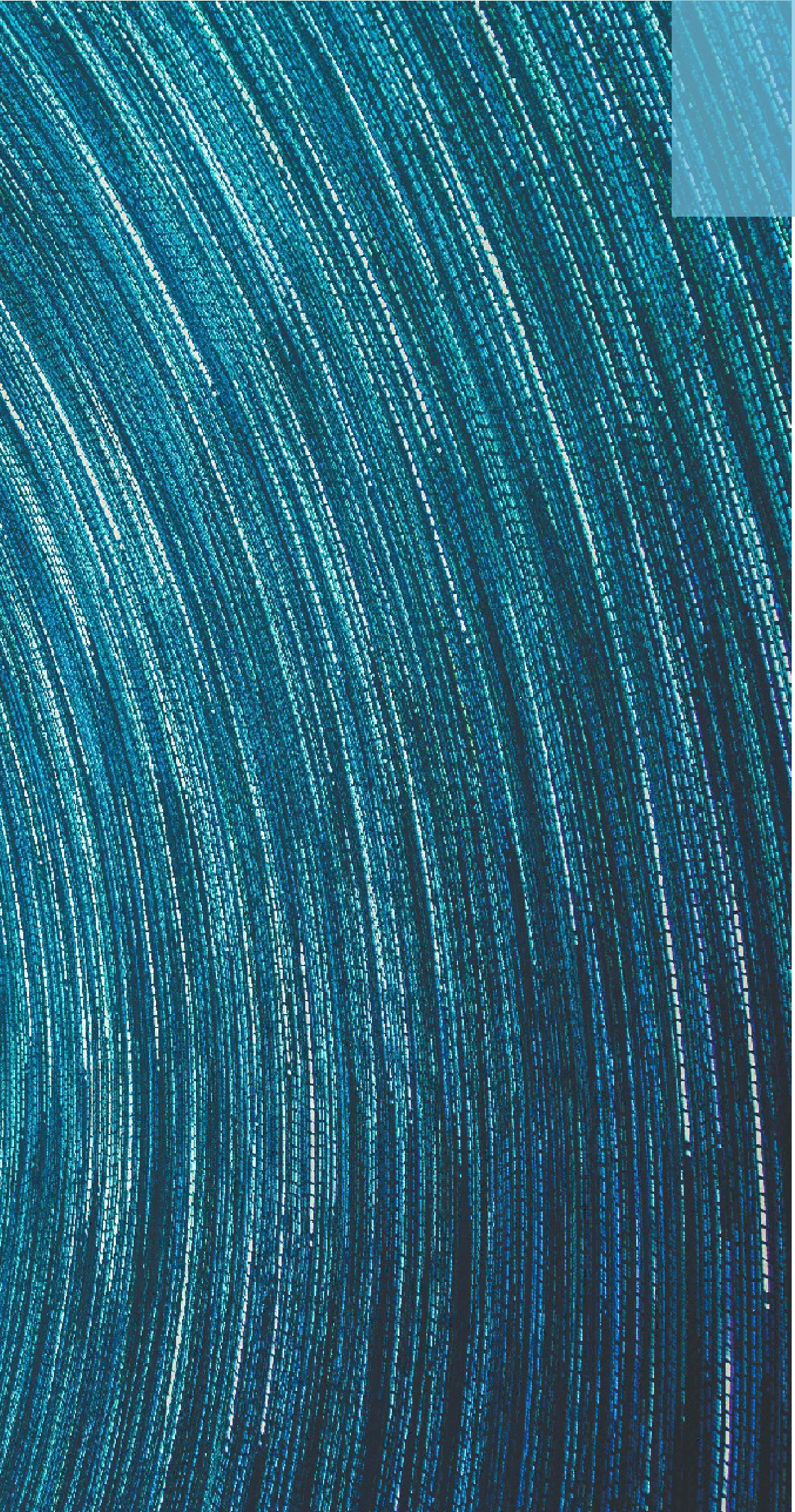
It takes about 30 minutes to onboard an application through this automated process that creates the DevOps Workspace and the DevOps Hostspace. If needed, we provide the DevOps Dataspace which provides database as a service.

Workspace: Where developers do their work before an application goes live. It is the non-production environment. For example, developers can create code branches, develop the code, create pull requests and trigger continuous integration builds. It's where they can test out new features, fix bugs, or try out new capabilities using our standards-based technology product catalog.

Hostspace: Where the stage and production application run, and where the continuous delivery releases will ultimately culminate. Our goal is to take the work that the developers do in the workspace, and via our CI/CD process, transition the changes to stage and production hostspace.

Dataspace: Representing Database as a Service (DBaaS), this provides the productivity, performance, and data security needed for databases.

Upon completion of this onboarding processes, both the workspace and hostspace(s) are created so that the CI/CD process can automatically work. Once it's done, then the development team can use the CloudOne DevOps platform to rapidly move application changes to production. This is the continuous integration, continuous deployment (CI/CD) process of our platform. •



Day 2 Rapid and Frequent CI/CD

MOHAN RAJ
IT SR. MANAGER, CLOUDONE DEVELOPER EXPERIENCE

When introducing our CloudOne platform inside NetApp, we wanted to meet certain minimum viable product (MVP) features including the capability to allow rapid, frequent production change through Continuous Integration, Continuous Deployment (CI/CD). We wanted our developers focused on writing code and rapidly releasing application changes, and not be concerned about other IT services like servers, storage, networks, and more.

With our CloudOne platform, we provide automation for frequent small production application changes. Automation helps to reduce the risk associated with large-scale production deployments while adhering to DevOps principles like moving fast and making small changes quickly. With traditional IT there are change control processes that often involve large changes, with some requiring the application to be taken offline. All of this takes time and introduces potential problems for something going wrong. It is not fast.

My favorite feature of the CloudOne platform is how the development (DEV), stage (STG), and production (PRD) environments are handled at the application layer. In our traditional data center model, the DEV-STG-PRD environments are dedicated and sit idle until used. With CloudOne, these environments dynamically spin up and down as needed at the application layer; the infrastructure and platform reside in the operational production (PRD) environment.

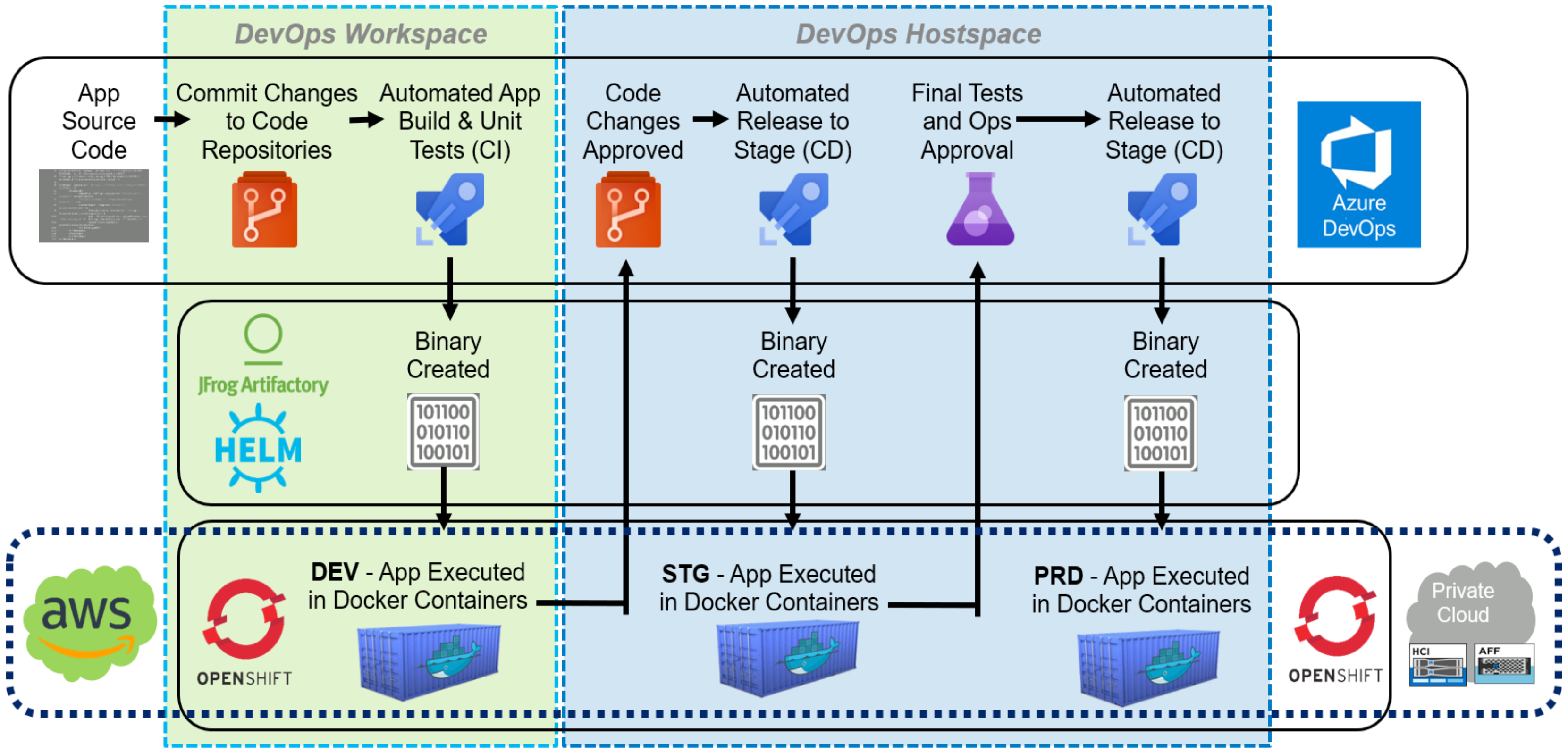
CloudOne application onboarding

Our CloudOne MVP introduction happened in November 2018. The MVP goal was to onboard simple apps onto the CloudOne platform to prove out the technology, test the process, and confirm support roles. During application onboarding, we create what we call a DevOps Workspace and DevOps Host Space, and if needed, a DevOps Dataspace.

Putting CI/CD processes to work

Applications are onboarded once. During this onboarding, both the workspace and host space are created so that the CI/CD process can automatically work. Once it's done, then the development team can use CloudOne to rapidly move application changes to production. This is the continuous integration, continuous deployment (CI/CD) process of CloudOne.

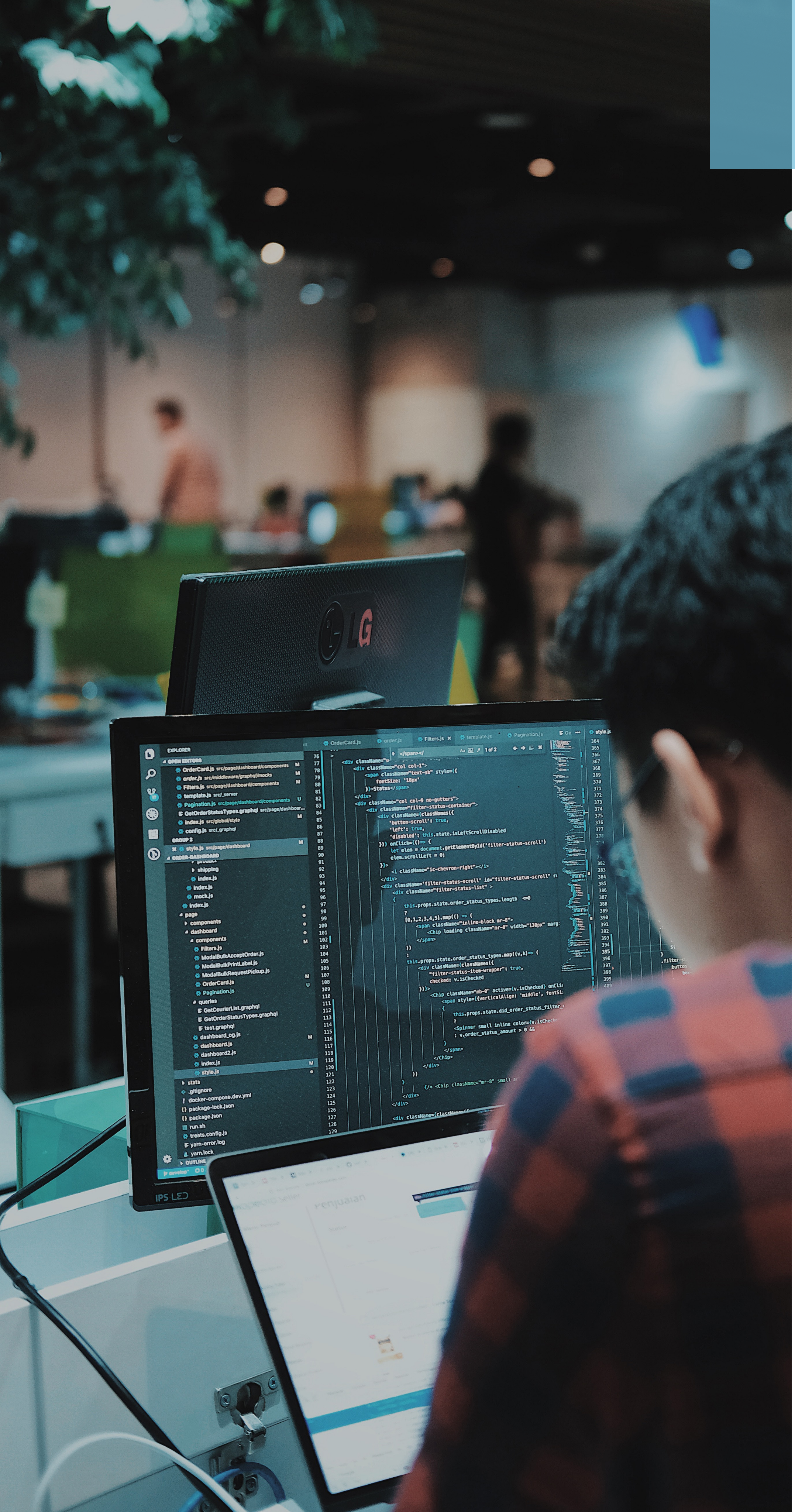
There are three levels of CloudOne for application CI/CD. Our developers do


DevOps Workspace
DevOps Hostspace
App Source Code
Commit Changes to Code Repositories
Automated App Build & Unit Tests (CI)
Code Changes Approved
Automated Release to Stage (CD)
Final Tests and Ops Approval
Automated Release to Stage (CD)
Azure DevOps
JFrog Artifactory
HELM
Binary Created
Binary Created
Binary Created
aws
OPENSHIFT
DEV - App Executed in Docker Containers
STG - App Executed in Docker Containers
PRD - App Executed in Docker Containers
OPENSHIFT
Private Cloud
HCI
AFF

most of their work in the first level using primarily Microsoft Azure DevOps. This service provides a full application lifecycle management ecosystem including dashboards, analytics, wiki's, and project and Kanban boards. It is where the developer's code repository resides for the applications and where they work on source code.

When an application developer submits a code change to the code repository, it kicks off an automated application build. The build combines the application stack, along with the code changes, to create a binary that is deployed within the workspace (non-production). Within about three to four minutes—that's how long the automation takes—the developer can see their change to the application running in a container and can determine if the change or bug fix worked. If it worked, the Continuous Delivery (CD) process takes the binary that is in the workspace and copies it into the stage environment in the host space. The code change can then be approved for release to production.

Based on role-based authorization in OpenShift and Azure DevOps, the developer, development lead, UAT



technician, and even operations, can do the final testing and approvals of small changes. Those involved with the change must be satisfied. Once the final approval is made, then the binary image is moved to production via automation. With OpenShift we can migrate each container one at a time to the new binary image so that the application never goes down. The whole process to implement a change—from the developer working on the code all the way to production—can be less than 15 minutes assuming no delay in approvals.

By adhering to an end-to-end, automated DevOps workflows in CloudOne, our developers can spend their time writing code and releasing changes, and not have to deal with infrastructure or platform services. The automation provides the ability for frequent, small production application changes to reduce production deployment risk.

I realize it is difficult to fully understand how a development platform works without a demonstration. Click on this video clip to see how easy it is to fix a bug in a demo application we created called Tiny URL Tool. By automating our release process end-to-end, we can rapidly make a small change to this app without downtime, risk, or duplicated supporting infrastructure. •

Building and Scaling our Private Cloud with HCI, AFF, and StorageGRID

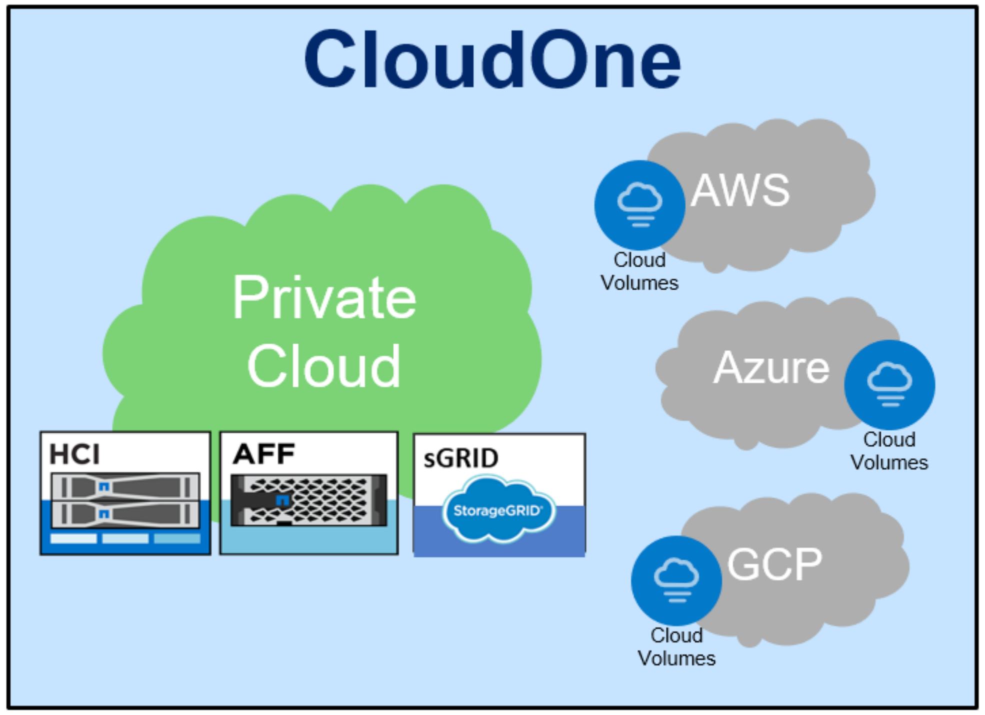
GOPI SIRINENI
SR. MANAGER, CLOUD AND COMPUTE SERVICES

One of the main priorities for our IT shop is to deliver rapid, frequent business software capabilities through automation and cloud technologies. In a NetApp IT eBook called "Vision for a DevOps Platform," we shared the vision of our CloudOne platform as a software-controlled and orchestrated development platform to build and run cloud-aware applications using DevOps and CI/CD delivery models.

The goal of CloudOne is to track and manage cloud costs with visibility across our multi-cloud enterprise. We focus on taking advantage of the elastic, burstable capability of the public cloud with its variable cost model, or the fixed cost structure of our private cloud. Either option requires dial tone-like infrastructure that is scalable and supports microservices and containers.

For our private cloud, we chose three NetApp technologies that met our requirements and are controlled and orchestrated through software.

- NetApp Hyperconverged Infrastructure (HCI)



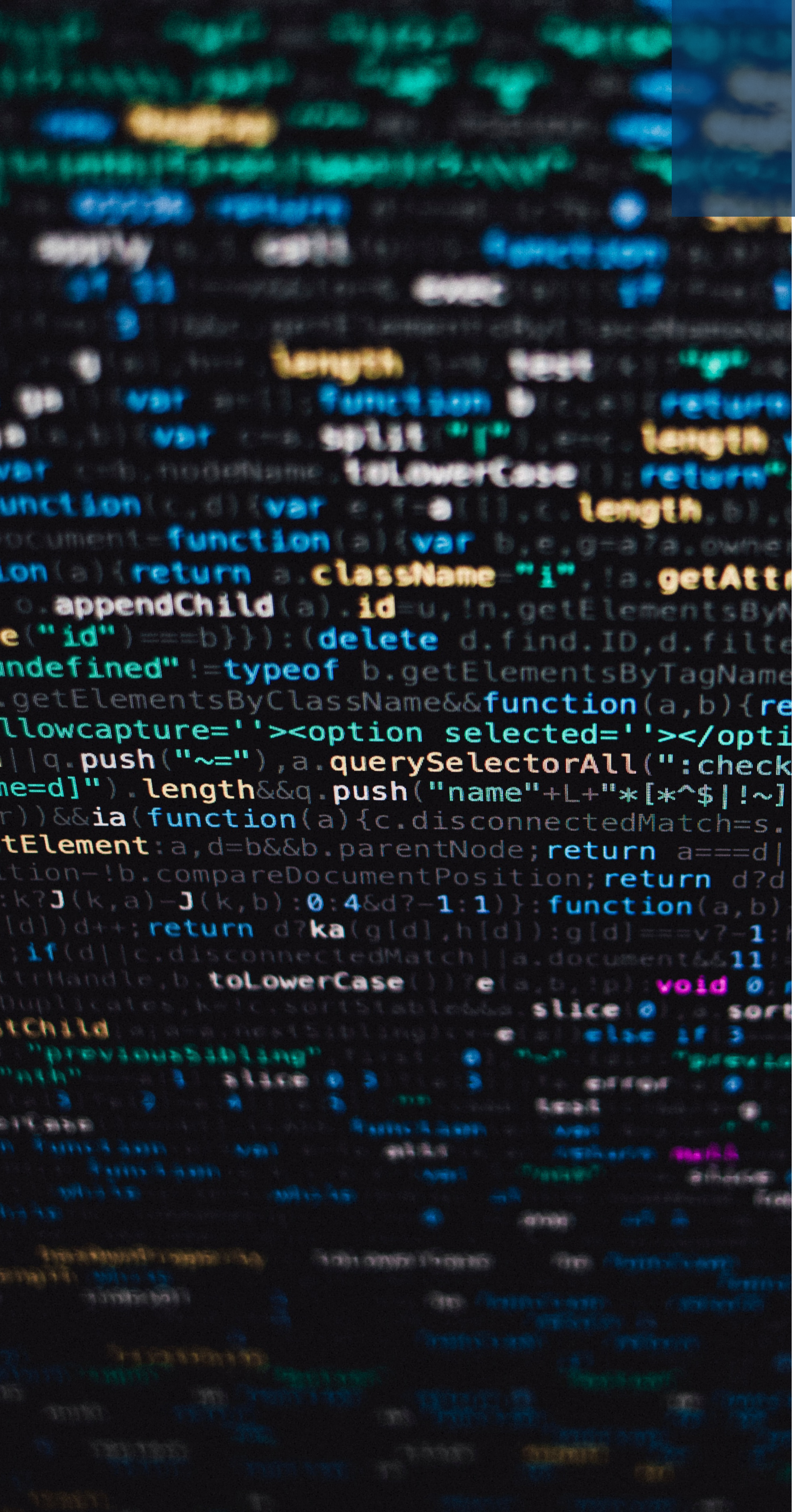
CloudOne: NetApp IT's software controlled and orchestrated development platform to build and run cloud-aware applications using DevOps and CI/CD delivery models

runs container-based technologies like Kubernetes and Docker and gives the containers exactly what they need: a little compute, a little memory and a little storage. It operates at a fixed cost and provides the proper compute resources and the ability to incorporate an API-controlled technology for storage.

- NetApp StorageGRID as the repository for object storage use. It is perfect for unstructured data that sprawls, may be retained forever, is rarely updated, and is meant to be accessed by multiple application services across geographical boundaries. Cloud-aware applications are inherently stateless, so they should take advantage of a stateless storage mechanism.
- NetApp All Flash FAS (AFF) acts as a stateful (NFS, iSCSI) storage platform for our databases and high performance workloads.

NetApp also leans on tools like RedHat Ansible to automate software provisioning, configuration management, and application deployment. To automate their storage configuration tasks, the team relies on NetApp Ansible modules for ONTAP, Element OS and E-Series. For example, by using Ansible ONTAP modules they have reduced Day 0 build times from a 2-day manual process to an automated, 10-minute process (read more here).

Ultimately, what we are accomplishing with CloudOne for DevOps is to rapidly deliver application changes at the speed of business change; NetApp technologies make this possible. Our CloudOne architecture includes not only the foundation of our private cloud moving forward, but the opportunity to use the same automated orchestration to deploy resources or workloads into AWS, Azure, Google Cloud or any other public cloud provider. We want to automate services—regardless of the cloud provider—so our developers can focus on writing code and releasing application changes seamlessly using our CloudOne DevOps platform. •



Provisioning Results: DevOps vs. Traditional Environment

MOHAN RAJ
IT SR. MANAGER, CLOUDONE DEVELOPER EXPERIENCE

I have worked in IT for over 25 years and am intrigued by how the world is being changed by software. It feels like every company today is defined by their ability to rapidly meet customer's needs via software. This software driven world demands rapid application development. Regardless if it is an IoT solution, a mobile banking app or a customer support app, DevOps software developers and operations staff are now being asked to work collaboratively on projects from start to finish. Traditional silos and governance methods are being displaced by efforts to use microservices architectures running in containers using cloud for rapid software creation and change. This is certainly true inside NetApp.

Over the past two years I have been part of an exciting NetApp IT initiative to build our DevOps platform that provides the cloud services, automation, and CI/CD release models that our application development teams need to build cloud native applications. We call the platform CloudOne, as it provides one consistent developer experience, irrespective of the destination being private or public cloud. My team automated the processes to onboard applications onto our CloudOne platform as well as the continuous integration, continuous deployment (CI/CD) process to rapidly move application changes to production.

As a result of our automation efforts, we have significantly reduced the time to provision environments in CloudOne compared to the traditional approach (results shown on the right).

Initially we only looked at automation to improve the efficiency and provisioning of hostspaces in less than 30 minutes. We take the work that the developers do in the worksapce, and via our automated CI/CD process, transition the changes to production in the hostspace.

Automation eliminates human error and reduces the number of staff and manhours needed to deliver environments. The time saved is now spent on delivering new capabilities within our platform or on any number of other value-added activities going on in IT. Automating delivery is critical to remaining relevant and competitive in today's IT climate of rapid software delivery. •

Provisioning Comparison	Traditional Approach	DevOps Environment
# of tickets needed to provision new application environment	7 tickets to create an environment including: • Demand initiation • Domain name • VM creation request • Firewall changes • Load balancer changes • SSL certification request • Middleware installation	1 self-service request* Hostspace using a self-service portal **Customized domain name requires additional tickets*
Typical Time needed to complete request	1-2 weeks for normal request	2 hours
# of environment provisioned in past 6 months	40 environments	72 environments
Code Build Automation (CI)	Manual build or using exist build automation	Provided as default; part of environment provisioning automation
Code Deployment (CD) Automation	Manual deployment or update takes 1-2 weeks of additional effort	Provided as default; part of environment provisioning automation
Environment Scaling	1-2 weeks of manual effort	Software-driven elastic scaling
Security Scan	Ad hoc	Provided as default; part of environment provisioning automation
Access Provisioning	Submit manual tickets to added/modify access	Provided as default; part of environment provisioning automation
Storage Provisioning	Volumes mounted part of VM provisioning	Provisioned dynamically during deployment

Authors

Michael Morris

Mohan Raj

Gopi Sirineni

For information on NetApp DevOps solutions, visit **NetApp.com/DevOps**