NTT DATA Services

WHITE PAPER | CLOUD

The Evolution and Future of Containers in Agile IT Infrastructure

FEBRUARY 2021

NTT DATA

Table of contents

Why NTT DATA?

We help organizations develop a container strategy and implement the tools and frameworks necessary to achieve it, teaching their teams how to use containers to improve agility. Empowering teams helps them deploy applications in a consistent way, scaling DevOps and cloud for greater business innovation. Developers benefit from an expedited process that allows them to consistently deliver containerized applications with built-in quality while infrastructure teams benefit from a self-service model that assures standardization across builds. And the CFO's office benefits from greater cost controls and system optimization that saves weeks of development time.

Our certified AWS consultants work with clients to help them start from a secure foundation, integrate the latest cloud-native and third-party services, and learn how to use the platform to achieve their vision of the agile enterprise.

NTT DATA offers a breadth and depth of experience building and deploying a variety of container technologies and enabling services on AWS. With years of hands-on work in our clients' environments, we've had a front-row seat to the evolution of container technologies. Put our expertise and longevity to work for you.

Introduction

Although containers generate a great deal of buzz, deployment in production has grown slowly. Only recently has it begun to accelerate as enterprises begin to experience container benefits first-hand, from their ability to facilitate faster application delivery, scalability and portability to their flexibility in policy-setting for managing resources, security and more.

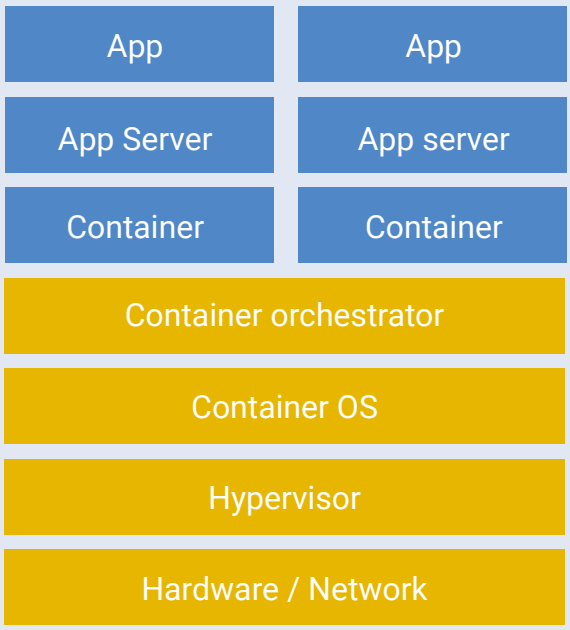
Containers play a foundational role in creating agile IT infrastructure, helping IT leaders achieve business objectives with greater flexibility and optimizing resources that enable quick pivots to meet emerging market and customer needs. Having worked with containers since 2013, NTT DATA has in-depth experience helping organizations meet the needs of both development and IT operations teams through container technology. As container technologies like Docker, Kubernetes and Red Hat OpenShift expand, they enable service delivery processes to be consolidated, drastically reducing coordination, and growing IT and developer productivity.

This paper offers a quick look at the evolution of containers and details how NTT DATA has helped companies place containers in a strategic role that optimizes IT resources and increases output. In the process, container technology gives technology teams an agile IT infrastructure that will move the organization into the future through digital transformation that addresses internal and external customer needs, market pressures and overarching business goals.

The evolution of containers: From image to ecosystem

It starts with a predefined image, instantiated to create an isolated process(es) with native OS controls (for example, Linux and Windows). Each container instance shares the machine's kernel with other containers but runs as if it's the only system on the machine. A container image is a set of logical layers of software and all its dependencies surrounded by a manifest that provides the runtime, necessary instructions and controls.

The container image is then downloaded to the host one time and can be used to spin up multiple instances on demand, allowing near instant container instantiation and scaling. The process of creating an image provides an immutable deployment package that can be used across environments, increasing the quality and consistency of the software development process.



Development embraces Docker

While the roots of container-style process automation date back to the 1970s, it wasn't until 2013 when Docker emerged with its solution that containerization grew to become mainstream. In fact, within a month of its first test release, Docker saw over 10,000 developers sign up to trial the solution. NTT DATA was among the first users of this platform. In 2013, when Docker 1.0 was released, we used the new container tools to build a solution for Auto.com where we helped the company achieve push-button infrastructure with the help of automation, cloud and Docker containers. With one Docker container for each application tier, the Chef automation tool to provision Docker containers in development and production, and the open-source Vagrant for development isolation, we were able to achieve production on a laptop.

This, in turn, helped Auto.com test code in a production-like environment without extensive hardware and resource investment, as well as minimize bugs that occurred due to a mismatch in local, quality assurance (QA) and production environments. With the introduction of Docker containers, Auto.com was able to speed its iteration times, grow continuity, experiment and innovate more while reacting with greater agility to changing market needs. We were proud to present this solution alongside Auto.com at the inaugural DockerCon[1], sharing how organizations can use Docker to grow DevOps agility.

Growing DevOps productivity

The following year, our client Pristine, which supplies software-as-a-service solutions to healthcare providers, showcased its Docker solution on stage at AWS re:Invent 2014. We helped Pristine address its rapid growth by building and applying a templatized infrastructure model to scale its application and ensure compliance with HIPAA. Using Docker, Pristine simplified the workflow from development to production and streamlined its blue/green deployment process. The result: the ability to rapidly deploy new infrastructure and ensure a high degree of uptime for mission-critical and health-critical situations. In the process, Pristine achieved "10X more DevOps work in 1/10th of the time," according to Pristine Cofounder and CTO Patrick Kolencherry in his AWS re:Invent 2014 presentation.[2]

Container revolution kicks in

Following an initial focus on containers in preproduction environments to automate release pipelines for greater deployment consistency, organizations stopped experimenting and started deploying at scale.

For example, in 2015 we worked with Rent-A-Center (RAC) to roll out an ecommerce platform to support its online shopper workflow. In the process, we created the first ever production containerized SAP Hybris environment. Working closely with the RAC team, we built an AWS-based microservices architecture based on Docker containers deployed on Amazon Elastic Container Service (ECS). The Dockerized SAP Hybris environment offered low-touch maintenance, easy setup and automation. In addition, we implemented autoscaling on top of Amazon ECS that allowed each service to scale up and down to independently address changes in demand. The result was a new ecommerce website that was implemented rapidly. According to the company, it took fewer than six months to get the ecommerce site up and running.[3] And, once it did, RAC was able to adeptly handle Black Friday's 42% increase in traffic. The infrastructure easily scaled to support 9 million hits, maintaining the same response time to page loading.

This same year, we helped Fugro, a multinational enterprise that collects and provides a highly specialized interpretation of geological data at land and sea, launch a new internet of things (IoT) service built on Docker containers and AWS. Providing a high degree of uptime, the remote service ensured the security of collected data while enabling portability so the environment could be quickly replicated in new global regions on demand. For Fugro we used Docker containers to create microservices that met its need for constant availability and reliability of service despite ships at sea with limited connectivity. And Docker became a central component in the company's continuous delivery cycle, illustrating how Docker could be used to create redundancy and high uptime. Fugro presented the solution at the 2016 DockerCon.[4]

Greenfield applications move to containers

While it wasn't until 2019 that 95% of new applications would use containers, in 2016 we were already working with organizations like payment solutions provider Verifone to develop greenfield applications on container technologies.[6] To help the company take advantage of a new market opportunity, we combined AWS with Docker to create a new cloud-based, streamlined infrastructure from the ground up. It was designed to address high availability, portability across multiple environments, a high degree of automation to increase agility and immutable infrastructure-driven security.

Heritage of the container

In the early 1950s, Malcolm McLean developed the first intermodal shipping container, sailing the maiden voyage of a retrofitted WWII tanker stacked with truck containers from Newark, New Jersey, to Houston, Texas. The concept quickly gained popularity because containers greatly reduced the time and expense involved in loading and unloading a ship, creating a 36x cost savings. And, importantly, got the ship more quickly back to sea for, as McLean notes, "a ship earns money only when she's at sea."[5] Similarly, container technology greatly reduces manual overhead, saving human and financial resources. And, just as a ship earns money at sea, organizations benefit when they deliver solutions faster that satisfy partner and customer needs and desires.

Building off this foundation, Verifone launched two projects with microservices architectures that allowed it to increase developer agility, in turn shortening the time to market of new applications associated with these projects. The company also decreased costs, driven by reduced wait times for IT when standing up environments, and gained a competitive advantage through a much faster time to market of its greenfield application. Containers were a key technology in the launch of a new business line in record time that also resulted in a new, dedicated DevOps team.

Container ecosystem emerges to further fuel expansion

Expedited by the introduction of container orchestrators like Kubernetes, Marathon-Mesos and Docker Swarm, containerization saw another period of rapid growth. Southwest Airlines approached NTT DATA in 2017 about migrating and replatforming several of its legacy on-premises applications to AWS. We used KOPS to deploy Docker Swarm instances on AWS. Using the open-source tool with Amazon Simple Storage Service (S3) and Docker, we automated the cluster creation and versioning processes.

In addition to automation, we configured KOPS for security, pairing it with CIS benchmark images for the host OS that deploy within each Docker container. Building CIS configuration guidelines into the solution helped the airline proactively safeguard against security threats. And, by implementing Kubernetes clusters in development, QA and production environments, the airline was able to effectively and efficiently deploy services, replatforming and migrating its applications as services on AWS.

This same year, we began work with G6 Hospitality. The company had implemented a modernization effort known at the organization as IT 2.0.[8] Designed to remove its dependency on legacy systems that were self-limiting, IT 2.0 embraced cloud-native technologies like containers to grow the company's flexibility and agility.

One of the first applications addressed was HotelKey PMS, G6's property management system. We migrated the application to an OpenShift container-based microservices architecture on AWS. Using repeatable patterns, we modernized several other applications, such as the company's reservation system, for which we built a microservices-based architecture using Amazon Elastic Kubernetes Service (EKS). In addition to giving developers greater flexibility and the ability to innovate, the modernized system delivers direct business benefits. For example, within two months of deploying the new reservation platform, the company decreased Online Travel Agency (OTA) reservation synchronization errors by more than 50%, in turn freeing nearly 40 hours/week in manual corrections by specialized call center agents.

> "We had a legacy system that had been used for a long time that was slow, difficult to maintain and couldn't handle the volume of data we process using the cloud. Today, we use a cloud-based system that operates with iPads at the front desks of our hotels. With everything in the cloud, any of our nearly 1,400 general managers can buy or replace an iPad, download our app and get back to work right away, serving our guests and making time-sensitive decisions regarding hotel operations."
>
> — Jessie Burgess, CIO, G6 Hospitality

Kubernetes migration factory

An enterprise media group sought help with its IT modernization plan, in which it had determined an AWS cloud migration was the ideal path. This firm had over 200 applications that were assessed and flagged for application replatforming. For these apps, we worked with the firm and its operations, security and business units to develop five patterns to move the applications to a modern, containerized infrastructure.

The first pattern developed was for applications with an architecture running Java/Tomcat in containers on top of a Kubernetes platform. We then developed a "Kubernetes pipeline" to deploy Kubernetes on AWS clusters as self-service on top of AWS, allowing for apps that may or may not use session state as well as apps with and without the need for shared storage. The point-and-click Kubernetes factory sped the company's time to market, allowing developers to launch Kubernetes clusters on demand. Once initial challenges were worked out and a final workflow established, the firm was able to launch two to three new services each week on the new system.

Moving to production: The future of containers

Development teams have quickly embraced container technology for use in development environments. Yet, when it comes to deploying containers in production, the challenge for many organizations has been where to start. With thousands of applications in the technology portfolio, knowing which to prioritize for refactoring for containerization and which are better off being retired, replaced or simply lifted and shifted to the cloud can be overwhelming. To help with this process we've created a decision-making guide, "Strategies for Large-Scale Cloud Migration," but asking if the application or service maps to any of the following goals can also help guide prioritization efforts.[7]

Modernize legacy applications

Container technologies can help organizations gain independence from traditional environments with new ways of packaging and architecting in which developers can modernize legacy applications using the best-fit language. Because containers isolate the application and its dependencies, there's no impact to other containers or the operating system. This allows developers to focus on the application components and not on other externalities — making them more productive, and applications faster to deploy, more portable and resilient. Last, developers can quickly and easily update application components, making quick changes to keep their applications modern and meeting ever-changing customer needs.

Implement microservices and advanced cloud architectures

Taking modernization one step further, organizations use container technologies to break what would be a complex application into modules, each of which runs within its own container. Referred to as a microservices approach, applications built this way are easier to change and update because they can be developed concurrently, with each team's unique preferences. This cloud-native approach can be applied to both legacy applications and greenfield opportunities.

Increase agility and further DevSecOps

Containers help further DevSecOps efforts by effectively shifting security left to development. Security becomes a key consideration in the build effort, rather than an issue to be managed by operations at deployment. As immutable infrastructure, containers aren't patched like other systems, but simply replaced, reducing the opportunity for security bugs or human error. These dynamics work together to bolster DevSecOps efforts. With automation, containers offer the opportunity to grow consistency, standardizing across builds for greater quality that allows cost controls and system optimization to be built into the system. Automation extends across the application lifecycle to include fully automated continuous innovation and continuous delivery (CI/CD) pipelines, automated testing and more, all of which can contribute to greater team efficiency and productivity.

Creating an agile IT infrastructure with containers

Containers can play a foundational role in creating an agile IT infrastructure. With other technologies such as cloud, configuration management and CI/CD, containers help remove hours of mundane, tactical work and firefighting through automation. This enables teams to increase the time spent on strategic work that brings value to customers. When it comes to driving agility, enhancing IT productivity and ultimately delivering business outcomes, containers benefit the business with:

- **Scalability.** Container technology encourages scalability because new containers can easily be spun up, quickly increasing capacity. Conversely, solutions like Amazon ECS automate the process of draining a container instance in preparation for maintenance or cluster scale down.
- **Team efficiency**. A container image includes all the code and dependencies needed, so containers can easily and consistently run regardless of where they're deployed. This grows team efficiency because less time is spent designing for and/or diagnosing issues in different environments and more time can be spent delivering new features and functionality to customers.
- **Portability.** With self-contained dependencies, container instances can be easily deployed to different clouds, operating systems and even hardware platforms. This translates into create once, deploy anywhere ease of portability for enterprise applications.
- **Application lifecycle management support.** Container technology supports agile and DevOps development, test and production through immutability, enhanced efficiency and more. Yet, containers also streamline CI/CD efforts because they more readily glide through build, test and deployment with the same container image that eliminates restrictions on technologies used in the process and reduces the need for manual intervention.
- **Optimized resource utilization.** The architectural approach of containerization allows containers to deliver greater resource optimization (CPU/memory) for large-scale workloads. This efficiency is attained because the application utilizes compute more effectively running as a container with a shared OS.
- **Immutable infrastructure.** Containers can be a key component of an immutable infrastructure, an approach that relies on treating application or service components as replaceable. When a change needs to be made, a new instance is easily created, and the old instance removed and replaced with the new. The ability to easily remove and replace old instances with new immutable ones decreases risk because it removes the opportunity for change-related security concerns — from configuration drift to simple human errors.
- **Support for use cases like microservices.** Containers are an ideal way to deploy microservices given how easy they are to spin up, their isolated environment and the ability to facilitate application lifecycle management. By implementing microservices in their own containers, enterprises can further streamline development and deployment, speeding digita transformation efforts.

Technology considerations and best practices

To attain these benefits, organizations should follow industry best practices for container security, operational excellence and system standardization. Based on our extensive container experience over the years, we recommend the following:

- **Start with a strong foundation.** Before diving into a containerization project, it's important that an organization first have a strong cloud foundation. Create an AWS account architecture (with accounts for billing, production, non-production, etc.), network connectivity and more. Before an application is ever deployed, deploy AWS services that establish the foundation for workloads. We recommend AWS Control Tower, AWS Organizations, AWS Identity and Access Management (IAM), AWS Single Sign-On (SSO), AWS CloudTrail, AWS Security Hub, AWS Config and Amazon GuardDuty. These services all work together to form a security baseline. While some think these AWS security services are already enabled when starting an AWS account, it's important to note they are not.

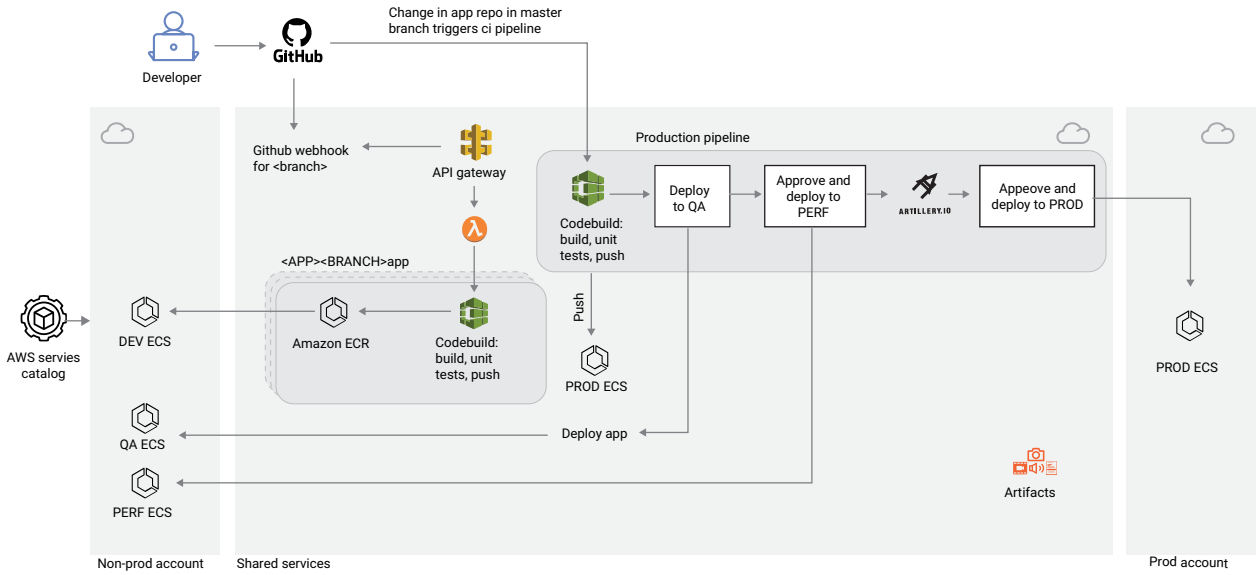


Figure 1: Deploy Containers for AWS code pipeline

Deploy Containers for AWS

To help even more enterprises on their container journey, NTT DATA developed a reference architecture called Deploy Containers for AWS. Based on automation and deployment best practices from hundreds of client implementations, it cuts eight to 10 weeks from the containerization journey by giving teams a sound foundation from which to start. The containerization process creates dozens of questions, many of which have long-term implications. While teams new to the process may find themselves frozen as they stop at each crossroad to ensure they make the best design choice, Deploy Containers for AWS helps speed the deployment of containerized applications. It shrinks the time developers spend on infrastructure setup and configuration while enabling efficient developer workflows with short feedback loops when developing application code. This, in turn, enables enterprises to begin containerizing in as little as a week.

- **Automate secure account creation**. The number of accounts needed will quickly blossom. As a result, it's very important to find a way to create AWS accounts with a consistent security baseline. And, once done, it's just as important to automate it so accounts can be consistently created, meeting a secure standard.
- **Achieve availability.** Replicating an environment across several availability zones grows resiliency. When coupled with multiple virtual private clouds for each environment and multiple AWS accounts to separate non-production, production and PCI workloads, organizations can create a resilient infrastructure and application that results in consistent availability.
- **Standardize the stack**. While there are a variety of tools to use, one set isn't necessarily better than the other. Standardization must take precedence over using specific tools for specific use cases. Find the tools that will work best in most situations and standardize. The time savings will come in handy when inevitable production issues occur later; teams will be able to scale learning and troubleshoot faster, which leads to greater improvement over time.
- **Secure the DevOps foundation**. To ensure proper code promotion through the desired environments, it's important to have a secure DevOps foundation. With it, organizations can implement infrastructure as code (IaC) and pipelines to deploy, manage and update each of the environments (for example, development, QA and production). Cloud-native serverless tools reduce management while simultaneously introducing best practices. Most importantly, a DevOps approach allows teams to deploy applications in a consistent way, scaling for greater business innovation. By leveraging AWS cloud-native tools, organizations can rapidly deploy containers and expand and customize their use with time, giving the business greater agility in response to ever-changing market demands.
- **Self-service portal for launching new environments and services**. Another way to extend standardization is through a Service Catalog, where developers can input their information and begin deploying containerized applications within minutes. A Service Catalog allows developers, regardless of their knowledge level, to deploy standardized, security approved infrastructure within minutes. A self-service portal also eliminates lengthy legacy ticket processes while scaling to many developers.
- **Build a CI/CD pipeline.** To build consistency across multiple applications, an application-agnostic pipeline/process is critical. Be sure to include QA, staging and production environments, so developers don't have to rebuild it for each application. Once built, the CI/CD pipeline should act as the interface between the application layer and the base platform, benefiting the system with greater standardization, yet deploying each application individually.
- **Performance testing.** As shown in Figure 2, performance testing should be conducted as part of the application pipeline. By building performance testing into the application pipeline, developers can help ensure the scalability of code promoted to production.

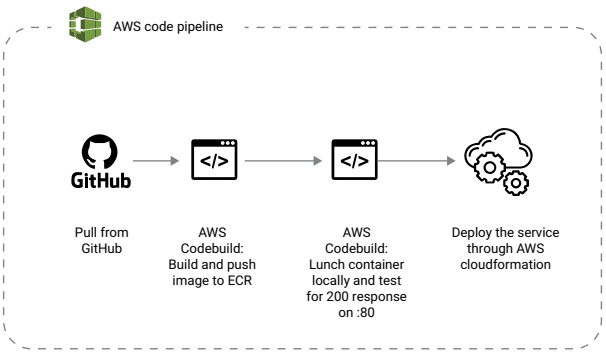


Figure 2: Performance testing should be part of your automated pipeline.

- **Secrets management.** Container security and secrets management are of the utmost importance. Enforce security best practices, like automated secrets management, without creating friction by giving container services access to application secrets only as needed. Once the containers run on AWS, they can assume an IAM role to obtain temporary credentials.
- **Rely on reference architectures and quick starts.** With best practices built in, reference architectures and quick starts can help get organizations most of the way to their goal. For example, clients find that our Deploy Containers for AWS reference architecture provides about 80% of what they need to get started while also providing a solid best practice architecture. Quick starts may have certain limitations, but they provide a good starting point from which to tailor to unique needs, reaching the end goal much faster.

Sources

1. DockerCon 2014. “Lessons from Using Docker to Improve Web Developer Productivity” June 9, 2014. https://www.slideshare.net/Flux7Labs/docker-con-lessons-from-using-docker-to-improve-web-developer-productivity-36261623

2. AWS. “AWS re:Invent 2014 | Pristine Helps Medical Professionals Deliver Higher Quality Care Using AWS.” December 2, 2014. https://www.youtube.com/watch?v=ktKzSuD9InM&feature=youtu.be&t=2m45s

3. AWS. “Rent-A-Center Case Study.” https://aws.amazon.com/solutions/case-studies/rent-a-center/

4. DockerCon 2016. “Overseeing Ship’s Surveys & Surveyors Globally Using IoT & Docker – Use Case Track.” July 11, 2016. https://www.youtube.com/watch?v=S7B-pEDoSeE&feature=youtu.be

5. Ebeling, C.E. Winter. “Evolution of a Box.” Invention and Technology Vol. 23. Winter 2019.

6. Emily Mell. “The evolution of containers: Docker, Kubernetes and the future.” TechTarget. April 2, 2020. https://searchitoperations.techtarget.com/feature/Dive-into-the-decades-long-history-of-container-technology

7. AWS. “G6 Hospitality Reduces Code Deployment Time by Over 80 Percent.” https://aws.amazon.com/partners/success/g6-hospitality-flux7/

8. Flux7, an NTT DATA Company. “Strategies for Large-Scale Cloud Migration.” https://us.nttdata.com/en/-/media/assets/white-paper/496685-large-scale-cloud-migration-whitepaper.pdf

Visit **nttdataservices.com** to learn more.

NTT DATA Services, a global digital business and IT services leader, is the largest business unit outside Japan of NTT DATA Corporation and part of NTT Group. With our consultative approach, we leverage deep industry expertise and leading-edge technologies powered by AI, automation and cloud to create practical and scalable solutions that contribute to society and help clients worldwide accelerate their digital journeys.

NTT DATA
Trusted Global Innovator