Whitepaper

Optimizing Continuous Delivery of Software using Containers

Accelerate business productivity and agility by using containers to optimize continuous delivery

Introduction

In the past, implementing a continuous delivery (CD) workflow was a mark of innovation and efficiency for businesses - The DevOps revolution had been embraced, and digital transformation was at the forefront.

But times have changed. Continuous delivery is no longer cutting-edge. It's a must-have for businesses seeking to stay competitive by keeping applications up-to-date, and quickly delivering enhancements to users.

How can companies stay ahead of the pack in an age when continuous delivery is the norm, not the exception? The answer is to optimize your continuous delivery pipeline by making it as efficient as possible.

That's because not all continuous delivery chains are created equal. Businesses that streamline CD pipelines using open technologies like containers (which help DevOps teams to double-down on the efficiencies introduced by a CD workflow) are the businesses that will remain most competitive.

This whitepaper explains how and why to optimize your CD pipeline using containers, on-demand provisioning, open standards, and other next-generation approaches to software delivery.

Summary

- Continuous delivery is no longer cutting-edge. It's a must-have for businesses seeking to stay competitive
- Businesses that streamline continuous delivery of software using open technologies like containers will remain most competitive.
- Adopting CD offers several key business advantages such as organizational agility, reduction of staff costs, customer satisfaction, improved software quality, recruitment of talent.
- Often, businesses make mistakes that undercut the value of continuous delivery—or cause them to fall short of achieving the full benefits of a CD pipeline.
- Containers are the key to building an agile, optimized delivery chain.

Why Your Business Needs Continuous Delivery

There is no question that companies today need continuous delivery. A CD pipeline offers several key business advantages, which include:

- **Organizational agility.** When software delivery is continuous, your organization has the maximum ability to react to new demands in the marketplace, the availability of new technologies, new business goals, and so on. In other words, CD ensures agility, an essential quality for businesses seeking to thrive in a rapidly changing world.
- **Reduction of staff costs.** A CD pipeline allows your developers, QA team, ITOps admins and everyone else who plays a role in software delivery to work in parallel. It also enables collaboration, which leads to faster resolution of problems. In both of these ways, a well-designed CD pipeline leads to better use of your team's time and lower staffing costs.
- **Customer satisfaction.** Customers no longer expect to wait months or years to receive feature enhancements or other updates in software. In order to keep your customers satisfied, you need to deliver application updates continuously. In this respect, continuous delivery is the only way to guard your customer base against competitors working to lure those clients away with faster software update cycles.
- **Software quality.** By enabling continuous testing of code changes, and ensuring that software changes are thoroughly vetted before they are released into production, CD pipelines play a crucial role in maximizing software quality. Equally significant—They mitigate the risk of releasing a buggy application change into production, when the cost (in terms of time, money and reputation) of rolling it back is much greater than when problems are caught early in the delivery chain.
- **Recruitment.** CD is important to attract the best talent to your company. Top developers and admins expect to use CD pipelines and the agile frameworks that come with them. If you're not doing CD, you'll face recruitment challenges, which will ultimately harm your business by making it difficult to build the best team.

These reasons make CD pipelines an essential part of business operations today. CD is no longer a nice-to-have, or something that matters only to the technical team. It's equally crucial for running a competitive, efficient, high-achieving business.

What Not to Do: Continuous Delivery Antipatterns

CD pipelines are no longer a new idea, and most businesses already have them in place, but that does not mean they are making the most of them. In many cases, businesses make mistakes that undercut the value of continuous delivery—or cause them to fall short of achieving the full benefits of a CD pipeline.

The biggest mistakes include:

- **Equating continuous integration with continuous delivery.** Continuous integration (CI) servers, which help to automate the process of testing and staging code, are one important part of a CD pipeline. On its own, however, CI falls far short of constituting CD. A CD pipeline also needs to include solutions for automating code management, security checks, infrastructure provisioning, application deployment and release cycle feedback.
- **Adopting CD platforms that lock your business in.** One of the primary goals of a CD pipeline, and DevOps as a whole, is enabling agility, including the ability to switch between frameworks, hosting environments (and so on) easily. If you adopt a CD pipeline that is built using proprietary tools, rather than ones based on open source software and open standards, you can't scale and adapt as your business and development needs change. The benefit of your CD pipeline is therefore undercut.

- **Investing too much time maintaining the CD pipeline.** A CD pipeline should help you complete tasks—not add more tasks. If your CD pipeline requires a significant amount of time to set up and maintain, it begins to defeat its own purpose. This is why CD pipelines that are delivered as a service (and therefore require minimal setup or management on the part of users) are advantageous.

Optimizing Your CD Pipeline with Container Technology

How can businesses not only avoid the mistakes identified above, but also make the very most of their CD pipelines? The following is a set of guidelines to adhere to as you work towards optimizing the continuous delivery chain you already have in place, or building a new one from scratch.

You'll notice that containers play a central role in helping to construct the CD pipeline outlined below. That's because containers are the key to building an agile, optimized delivery chain. Unlike old-generation technologies, like virtual machines, containers introduce crucial efficiencies to the way you manage software delivery.

- **Be holistic.** Your CD pipeline should automate not just development and testing (the tasks on which traditional CD tools have focused), but everything from software design to post-release production and management. Your CD tool should encompass all of these processes, providing a single pane of glass for viewing and managing absolutely all parts of the software delivery process. Containers help here by providing consistent software environments for all states of the pipeline, which in turn will minimize variables and maximize visibility.

- **Be flexible.** The CD pipeline you need today may not suit your needs tomorrow. For this reason, it is essential to design your CD pipeline in a flexible way that can be adapted to meet changing needs. Containers provide flexibility because they can host apps written using any framework or language and are highly portable between different environments.

- **Think in terms of roles.** In most cases, not everyone involved in your CD workflow needs to have access to all CD tools and processes. Unnecessary access is a security risk and a source of needless complexity. That's why it's important to design a CD pipeline that enables role-based access control and features, with different roles assigned to different groups—such as developers, the QA team, the security team and the ITOps group. Containers enable fine-tuned access control because you can allow or deny access on a per-container basis.

- **Build provisioning into the pipeline.** Infrastructure provisioning is not a strict requirement for a CD pipeline. But including on-demand provisioning within your pipeline and integrating provisioning workflows with the rest of your application delivery will help you to make the most of CD, and avoid costly delays that could result from a slow or inconsistent provisioning process. Containers can help you achieve this because they make applications infrastructure-agnostic and portable, thereby reducing the amount of infrastructure provisioning required.

- **Use open tools.** Building your CD pipeline using open source tools and open standards guarantees portability and interoperability, and prevents the risk of vendor lock-in. In other words, it maximizes your agility. Using containers helps you stay open because Docker and other container technologies are built using open source code and open standards—a characteristic that makes containers different from virtual machine platforms, most of which are proprietary.

- **Adopt the latest innovations, including containers.** The infrastructure that provides the backbone of your CD pipeline should consist of the most innovative and agile technologies, such as Docker containers. In addition to being an open source technology, Docker containers help to streamline software delivery by providing a consistent environment for testing, staging and deployment, and by creating a framework for creating agile, microservices-based applications.

- **Be consistent and secure.** Security should be built into every stage of your CD pipeline. Containers help to mitigate security risks by allowing you to deploy applications in a consistent and immutable way, thereby reducing the number of variables and opportunities for error that could lead to intrusion.

- **Consume your pipeline as a service.** A CD pipeline built using components that are delivered as a service is easier—and therefore less costly—for your staff to set up. It also requires minimal maintenance. The end result is a pipeline with the lowest total cost of ownership, and one that can scale up or down easily in size as your needs fluctuate.

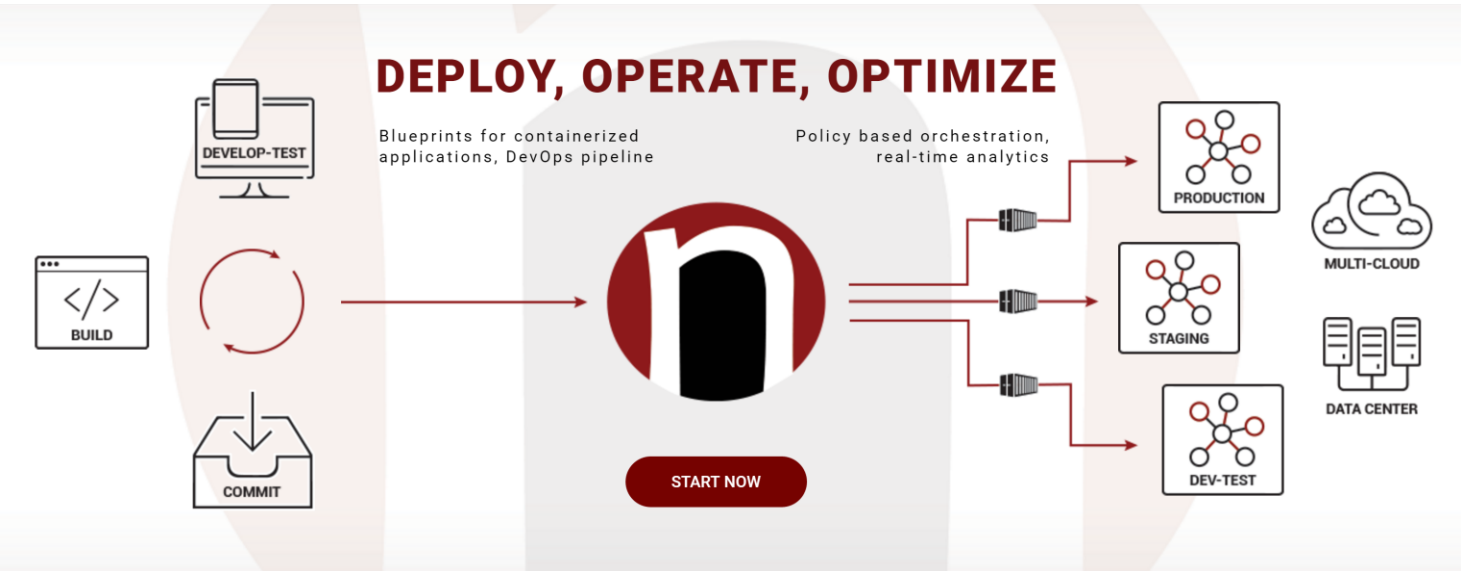
- **Think in terms of enterprise-readiness.** Your CD pipeline should be scalable and agile enough to support large-scale enterprise software deployments—not just internal workflows or experimentation carried out by your IT team. For this reason, a CD pipeline that is truly ready for enterprise production use should be constantly on your mind as you design and build your CD pipeline. Containers help in this regard by maximizing scalability. A single server can host thousands of containers, whereas only a few dozen virtual machines can be supported on a single host.

- **Use the resources you already have in place.** A cost-efficient CD pipeline should work with the networking, storage and other systems that you already use. It should not require you to migrate wholesale to new platforms. CD pipelines that are flexible and designed for agility will ensure that you can build your CD pipeline with the resources you already own (and that your IT team already knows how to use) rather than starting from scratch. A CD pipeline built using containers provides this flexibility because containers can host apps developed using any kind of technology or framework. They can also run on both Linux and Windows servers. You do not need to make major changes to your toolset or environment to use containers.

A CD pipeline designed and created according to the principles above will not just make the lives of your IT staff easier, it will also maximize return on investment in software delivery by keeping IT processes lean, efficient and agile by guaranteeing better and faster software updates to customers. And it will also help you stand out from competitors who fail to optimize software delivery.

About Nirmata

Nirmata is a complete solution for deploying, operating, and optimizing containerized applications on any cloud.



Securely connect your physical or virtual hosts to Nirmata and create fully automated, continuous delivery pipelines for your applications. Nirmata is delivered as an always on, highly scalable, and secure cloud service, so your applications are always monitored and automatically managed to meet desired service levels.

You can learn more about Nirmata *here*, or signup for *a free 15-day trial*. No credit card required!

Check out our *blogs* & watch our *demo videos*!

email: info@nirmata.com
web: www.nirmata.com