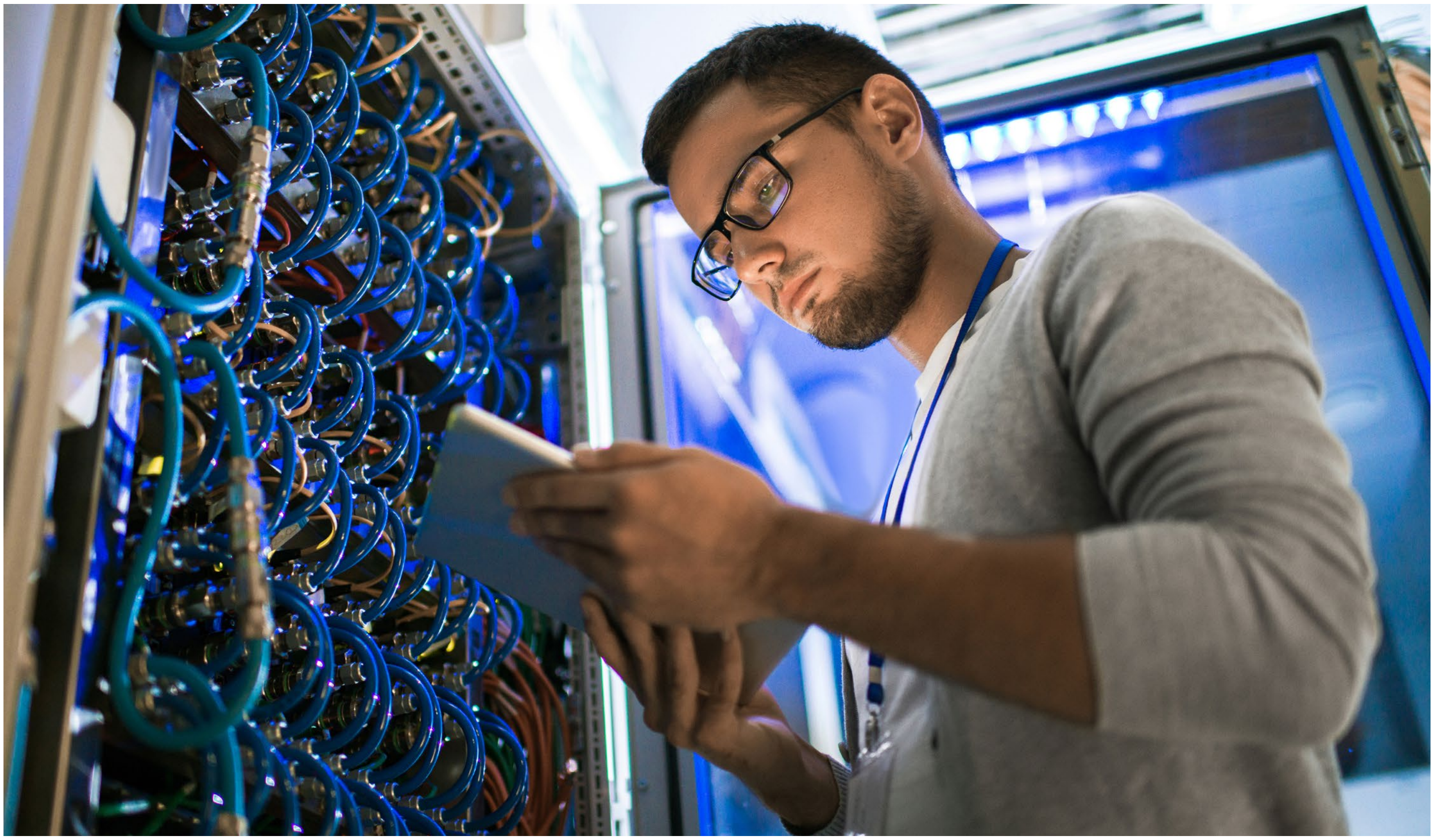
White Paper



Engineering DevOps For Networks

(NetDevOps)

Introduction

Networks are no longer composed of black boxes with strictly fixed functionality. With technologies such as 5G and SD-WAN introducing the ability to deploy an almost endless array of features, the SDN market is projected to grow at compound annual growth rate (CAGR) of more than **almost 24%** during forecast period 2019-2023. Network Equipment Manufacturers (NEMs) and Network Service Providers (NSPs) business strategies are leaping to leverage these Software-Defined Networking (SDN) market trends. The complexity of deploying 5G and SD-WAN, in turn, is driving NEMs and NSPs to collaborate, enabling them to achieve their goals and objectives in a more synergistic manner.

Successful strategies to improve business outcomes for Network Equipment Manufacturers (NEMs) and Network Service Providers (NSPs) require engineering three business processes:

01 **Rapid delivery of innovative new feature releases** generates top-line revenue and improves business competitiveness.

02 **Operational efficiencies** improve resource utilization and save money for the business bottom line.

03 **Improvements to product and services quality** improve end-customer satisfaction and improve business growth.

" . . . .

To fully realize the benefits of DevOps for networks requires end-to-end engineering of the value stream, referred to as NetDevOps.

. . . . "

To engineer rapid delivery of new feature releases, while increasing operational efficiencies and improving product and services quality, is challenging. To that end, many NEMS and NSPs are strategically transforming their value streams with DevOps engineering practices. Yet they run into roadblocks when trying to adopt DevOps because the value stream for network capabilities starts with feature development within a NEM (The DEV side of DevOps) and completes with the deployment to the live network managed by a NSP (The Ops side of DevOps). To fully realize the benefits of DevOps for networks requires end-to-end engineering of the value stream, referred to as NetDevOps.

According to Marc Hornbeek, in his book "Engineering DevOps from Chaos to Continuous Improvement…and Beyond," "NetDevOps extends capabilities of DevOps into the network. As networks begin to embrace more software-defined networks (SDN) and virtualized network functions (VNFs) for delivery of networks as services (NaaS)."[1] Figure 1: DevOps Value Streams for Networks illustrates the idea and goals for NetDevOps.

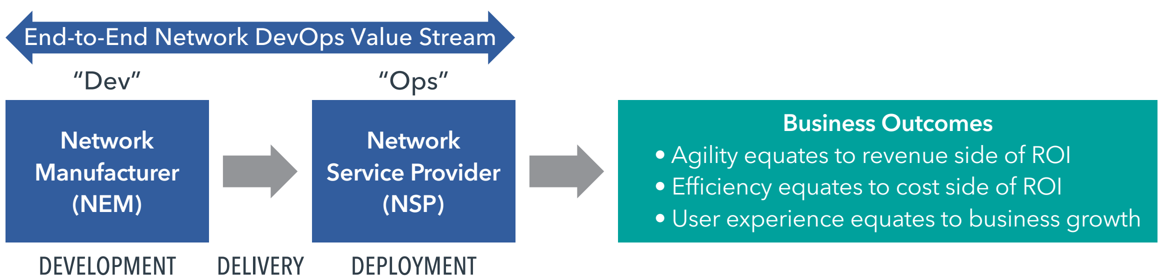


Figure 1. DevOps Value Stream for Networks

As the Figure 1 indicates, the traditional Continuous Integration/Continuous Delivery (CI/CD) model is not sufficient to describe NetDevOps pipelines. Instead a Continuous Development/Continuous Delivery/Continuous Deployment (CD/CD/CD) or CD3 is a more appropriate pipeline model for NetDevOps. This paper explains the challenges and solutions for engineering end-to-end DevOps value streams for networks that encompass both the NEMs and NSPs so they may accomplish business outcomes: rapid delivery of innovative feature releases, operational efficiencies, and improved quality. In particular, the paper highlights the following critical requirement for NetDevOps.

Part A: Engineering DevOps for Networks Concepts

DevOps for networks share many of the same challenges as DevOps for enterprise software systems. Both must address common issues with leadership, culture, design, integration, testing, monitoring, infrastructure, security, and delivery/deployment—referred to as the Nine Pillars of DevOps.[1]

Similarly, DevOps for networks can leverage many of the same solutions and concepts as DevOps for enterprise software systems. Like DevOps, NetDevOps solutions involve:

1. change management of all software and configuration artifacts that are necessary to build, deploy and operate the network,
2. removing bottlenecks by implementing Continuous Development (CD) / Continuous Delivery (CD) and Deployment (CD) toolchains,
3. orchestrating infrastructures such as test and deployment environments that, as much as possible replicate the live production environments,
4. automation of processes and tests in a manner that "shift-left" activities as much as possible, and
5. automated checks and balances that ensure the new faster processes do not sacrifice security or quality.

"In its simplest terms, NetDevOps can be defined as an intersection of Networking and DevOps. It is open communication, done through automation and using Infrastructure as Code (IaC)."[2] "Network Automation is the result of the NetDevOps approach. As DevOps places a large emphasis on automating processes, NetDevOps looks to share this goal."[3] "The sheer volume and speed of software updates for large networks is not feasible using manual methods. Automation tools for live network software updates are vital."[4]

The NEM's portion of the end-to-end stream for networks ends with the delivery of new capabilities to the NSPs. The NSP's portion of the value stream ends with deployments of new capabilities to live networks. The delivery and deployment capabilities are separated across the NEM/NSP chain of processes.

In other words, NEMs facilitate CD/CD as Continuous Development/Continuous Delivery and NSPs facilitate CD/CD as Continuous Delivery/Continuous Deployment. Engineering DevOps for networks is really a CD/CD/CD process, rather than a CI/CD process. Since Replace Paragraphs with: "Continuous Delivery is shared between NEMs and the NSP, the total end-to-end value stream can be referred to as CD/CD/CD or CD3. This is illustrated in Figure 2:

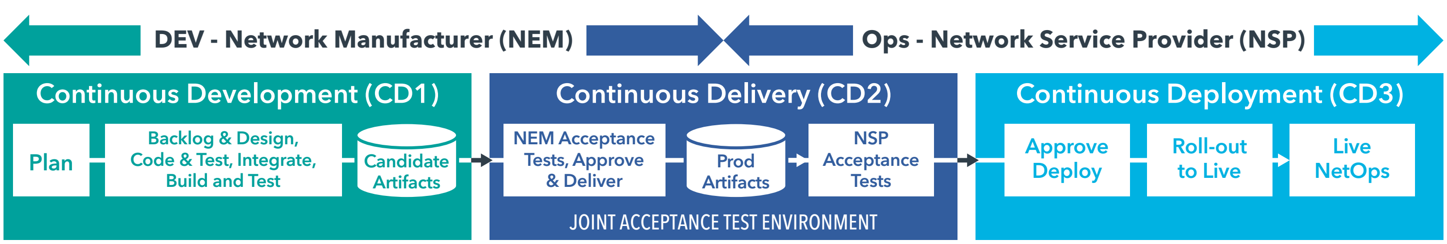


Figure 2: DevOps CD/CD/CD Value Stream for Networks

"

Since Continuous Delivery is shared between NEMS and the NSP, the total end-to-end value stream can be referred to as CD/CD/CD or CD3.

"

Need for Engineering DevOps for Networks

As networks begin to embrace more software-defined networks (SDN) and virtualized network functions (VNFs) for delivery of networks as services (NaaS), the ability to enable and manage network changes must become a core competency. This core competency requires deep understanding of DevOps engineering practices that enable automated deployments of network elements and network topologies from development, pre-deployment testing, and into production.

NetDevOps methods provide the required best practices to realize network elements as software, images and configuration files over ephemeral cloud infrastructures or real physical hardware, or a combination of cloud and physical resources.

Many network components operate on Whitebox hardware or somewhere in a data center on the network.

> "The importance of infrastructure as code and cloudification of hybrid test environments such as lab-as-a-service and test-as-a-service platforms become very effective in realizing NetDevOps pipelines and productive continuous test routines."

> "The hybrid networking requirement makes what is an already complex, heterogeneous environment even more complex to test."

Enabling automated network services in this new networking paradigm still requires the use of legacy physical systems to cooperate with next-generation virtual network elements, many of which will operate on Whitebox hardware or somewhere in a data center on the network. The hybrid networking requirement makes what is an already very complex, heterogeneous environment even more complex to test. In these cases, infrastructure as code and the cloudification of hybrid test environments such as lab-as-a-service and test-as-a-service platforms become highly effective in delivering NetDevOps pipelines and productive continuous test routines.

Figure 3: DevOps Value Stream Blueprint for Networks illustrates the major system structures required to engineer a DevOps Value Stream for networks and their general relationships with each other.

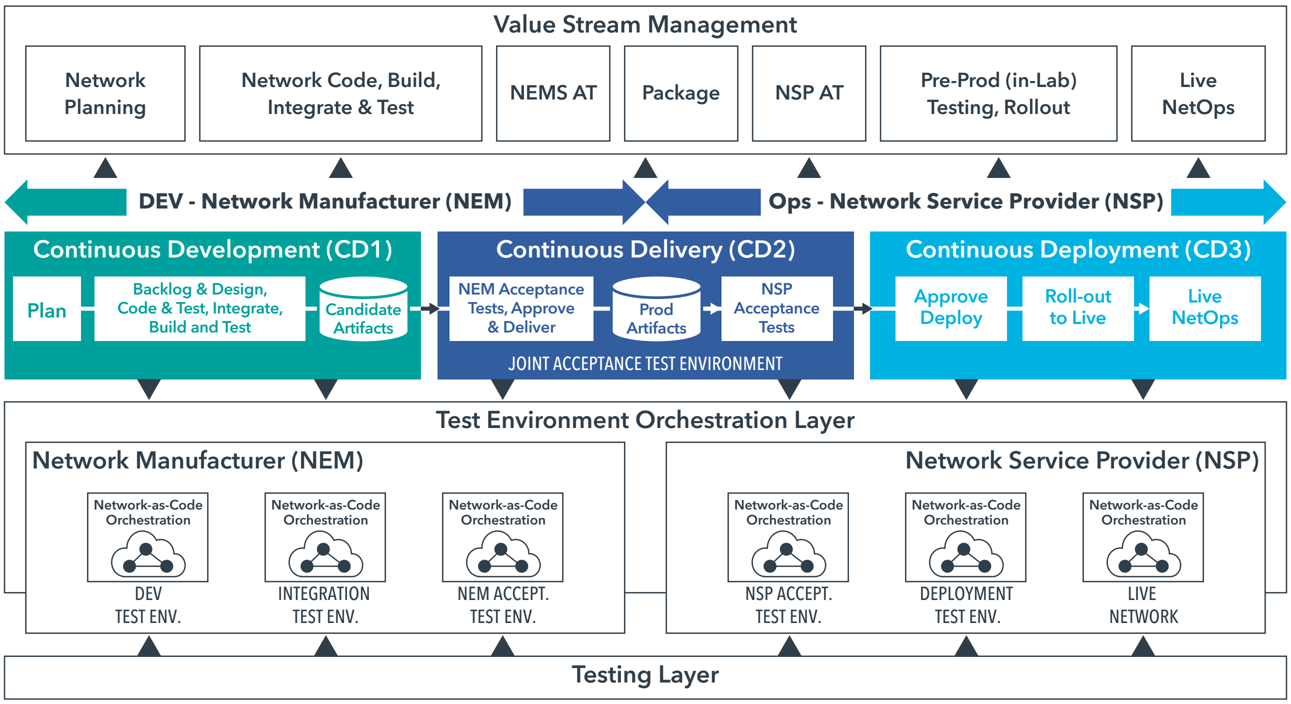


Figure 3: DevOps Value Stream Blueprint for Networks

The **top Value Stream Management** layer is responsible for observation, analysis, and overall orchestration of the major value stream stages from network planning, through to building, packaging, testing and deployment to the live network

The **CD/CD/CD pipeline** layer is responsible for execution of processes for planning, coding, integration, testing, packaging, acceptance, and rollout of releases.

The **Test Environment Orchestration** layer (often referred to "**Lab-as-a-Service** layer for in-house testing applications) is responsible for orchestrating the deployment of test environments and ultimately deployments to the live network. With DevOps the live environment is also considered a "test environment" in the context of this paper. This concept is justified because best DevOps engineering practices involve release rollouts with a "test-as-you-deploy" dark launch strategies such as Blue-Green deployments, Feature-Flag rollouts, and Canary rollouts.[1] "In a DevOps[network] environment, implemented according best practices, the laboratory is becoming an on-line operation that is reliable, usable, remotely accessible and shareable by many types of remote users that expect 24/7/365 availability."[5]

The **Testing** layer (sometimes referred to Test-as-a-Service) includes the actual testing tools and tests required to be available and executed on-demand as they are needed to support the various requirements of each stage of the CD/CD/CD pipeline.

Challenges for NEMs and NSPs

Both NEMs and NSPs share some common challenges, when it comes to the three dimensions of DevOps: People Process and Technology.[1] Listed below are some of the most prevalent ones.

1. **Legacy silos**–Many NEMS and NSPs are large, established firms that have evolved over the years yet maintained organization structures with different functions such as separate development and QA departments. This is a disadvantage compared to being integrated into small, nimble multi-functional organization units that are favored by modern modular DevOps processes. The breakup and restructure of these organizations is beyond the scope of this paper.

2. **Big monolithic software releases**–Many network systems have developed over the years as tightly coupled software monoliths that, because of the performance benefits of tight integration of the various system layers of network system architectures, are hard to dissect into smaller-management micro-service architectures favored by DevOps. This is beyond the scope of this white paper.

3. **Testing is often the biggest bottleneck**–All three of the business goals are affected by bottlenecks in the value stream. Due to the complexity of network topology variations, testing is especially challenging for network systems. Test environment orchestration and testing is typically the largest bottleneck for DevOps value streams for networks. This challenge is the centerpiece of this paper.

Challenges Specific to NEMs

NEMs are in a difficult position when it comes to certifying their products for network deployment as they typically do not have access to a realistic environment which mimics the intended deployment network. This prevents NEMs from testing against realistic deployment topologies. To alleviate this problem, NEMs will typically emulate the parts of the network where they are handing off.

Another troublesome area of testing for NEMs is the integration with third-party devices. Testing the almost infinite combinations of devices, software, and configurations that need to be tested is a cost prohibitive exercise.

Finally, a lot of the testing is performed manually, especially when it comes to new functionality. This, combined with the fact that NEM releases typically include a large amount of changes and that these changes are built in as a monolith, contributes to exceedingly long test and release cycles.

Challenges Specific to NSPs

NSPs have some specific challenges of their own. Testing of NEM products is largely a manual process, which leads to very long lead times for deployment. NSPs must contend with testing different configurations and combinations of NEM equipment and software releases, not to mention interoperability across a whole slew of networking equipment configurations and topologies. They also need to deal with a fair amount of back and forth with the NEMs when they find system configuration issues that are specific to the target deployment environment under test.

NSP labs are typically testing against fixed network environments. It's not always possible for NSPs to replicate a realistic customer deployment environment and setting up the required environment often requires a lot of manual intervention.

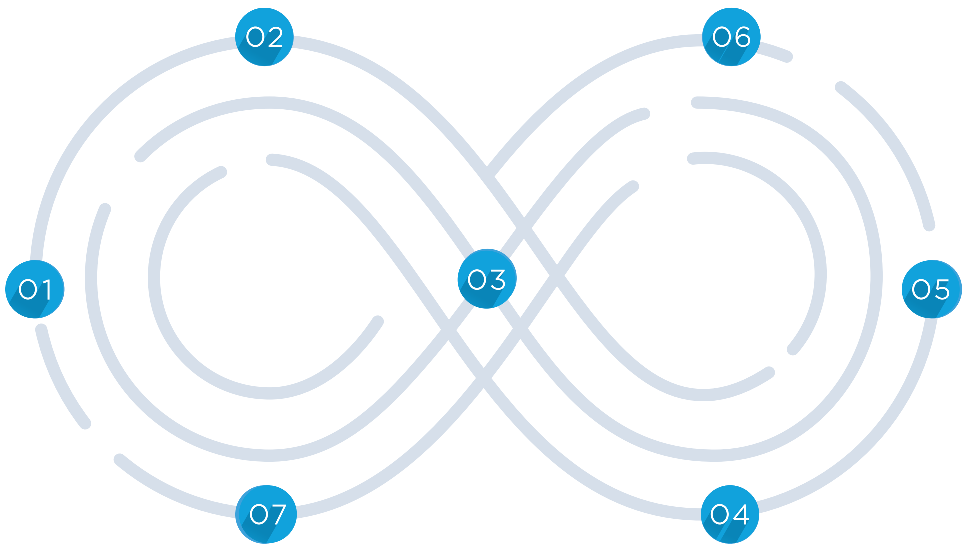
These factors all contribute to exceedingly long test cycles for network system acceptance and deployments. In the following sections, we will describe how NEMs and NSPs can overcome the challenges outlined above. We will also discuss how to develop an effective blueprint for a collaborative ecosystem that allows both the NEMs and NSPs to achieve their business goals of agility, efficiency, and quality in a more synergistic manner

.

Part B: Practical Applications of Engineering DevOps for Networks

While DevOps for networks, NEMs and NSPs have specific challenges, the same overall engineering approach to creating the Seven-Step DevOps Transformation Blueprint illustrated in Figure 4 and described in the "Engineering DevOps" solution applies.[1]

As indicated in Figure 4, it is recommended to start any DevOps transformation by getting senior leaders that oversee the product or service to agree on the strategic Vision and sponsor key players to drive the transformation.



01	Visioning	**Strategic, Sponsors, Partners** Leadership sponsors quality metrics team to lead actions.
02	Alignment	**Leadership, Team, Applications** Quality metrics team assigns risk priorities—people, process and technologies.
03	Assessment	**Discovery, 9 Pillars of Maturity, Deep-Dive, Value Stream Map** Select ACTIONABLE metrics that best match highest risk priorities—pillars and structural.
04	Solution	**Future State, Roadmap, Epics, Realignment** Select a value stream management platform (VSMP) to perform a backbone for metrics tools and analytics.
05	Realize	**Projects, Stories, Tasks, POC, Validation, Training, Deployment, Governance** Select tools and coding projects that fit the selected metrics.
06	Operationalize	**Monitor SLI / SLO / SLA, SRE Controls, Retrospectives** Put in place highest priority metrics with a model application.
07	Expansion	**Continuous Flow, Enterprise Adoption, Continuous Feedback, Continuous Improvement** Operationalize, measure, iterate.

Figure 4: Seven-Step—DevOps Transformation Blueprint

- Next, the leadership team should Align (agree to) a model application to be a "lighthouse" for the organization to showcase its progress.
- An Assessment is conducted on the model application to discover its current state relative to best practices for DevOps, to refine the goals and draw a current-state value stream map.
- During the Solution stage, a future state value stream map, an implementation roadmap, and Agile plans are defined and agreed with leadership.
- During the Realize phase, the project is executed. This may involve trials of different tools as part of evaluating solution alternatives.
- After implementation, the Operationalize phase includes monitoring the performance of the solution and making adjustments to plans according to feedback from operational experience.
- Finally, during the Expansion phase, the solution is adopted or modified as needed to suit other applications.

As indicated previously, testing is often the biggest bottleneck that needs to be addressed for DevOps for networks. This paper focuses on this key bottleneck, which is critical for networks.

Key Components of the DevOps Solution for Networks

Testing of networking solutions typically requires a lab where equipment can be staged and connected to create the required environment to test against. Most of today's lab operations are supported using manual processes. The very act of setting up and tearing down a test bed is hands-on, intensely time-consuming, and requires significant time before and after testing. Introducing these manual lab processes in DevOps-based processes is counterproductive and cripples the DevOps initiative.

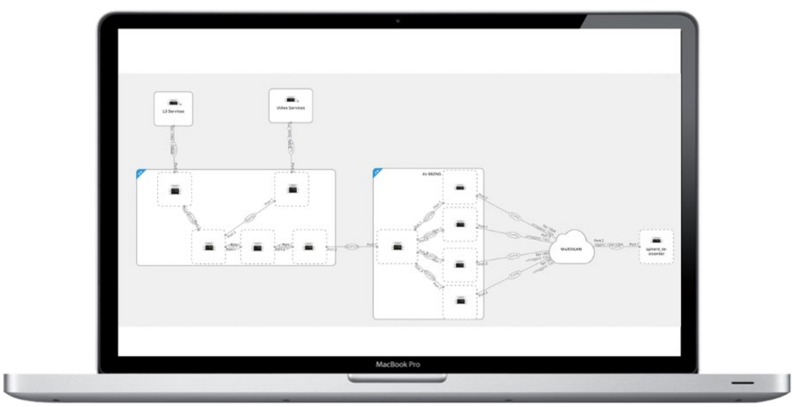


Figure 5: Topology of Lab-as-a-Service

> "Cloudification of the lab involves the virtualization of the entire lab infrastructure and services into a cloud environment which makes it accessible to anyone, anywhere, at any time."

The solution: Cloudification of the lab. Cloudification of the lab involves the virtualization of the entire lab infrastructure and services into a cloud environment (private or public) which makes the testing accessible to anyone, anywhere, at any time. This enables users to create the required topologies from any remote location, and to build and deploy their test bed in the lab so they can run their test. With a complete end-to-end Lab-as-a-Service (LaaS) solution, users can build a test bed and request a test bed or a test environment to be built. Figure 5 illustrates a typical network topology, showing a LaaS system for an end-to-end network configuration used for testing.

With a DevOps pipeline that is 100% automated, the lab environment is optimized to enable continuous testing and delivery. This capability is not limited to just physical lab environments. Lab environments that use public or private cloud services need to be included. That's because many manual processes still exist, including deploying VMs, containers, and CNFs; connecting these virtual environments together; and deploying services. With this capability, automated testing can be performed in the required environment. Once the testing is completed, the environment is released and the equipment can be allocated to support other requests. Velocity™, offered by Spirent, is a leading solution provider in this space.

> "Today, NEM and NSP market leaders are moving towards the inclusion of LaaS in a DevOps pipeline driven by critical business objectives. Those that are not going down this path take the risk of falling behind."

A second component critical to the success of DevOps deployments is automation. To achieve speed and agility, it's imperative to automate all the testing processes and configure them to run automatically when the deployment of the test environment is completed. This requires a specialized and mature automation testing framework through which one can quickly script, test, and deploy automated tests that operate with a high degree of reliability.

Value Stream Delivery Model

Now we're armed with the knowledge of the fundamental components required to support a DevOps environment for the testing of networking products and solutions. Next, let's have a look at how we can put these pieces together to create a blueprint for a DevOps environment that can help NSPs solve their challenges by:

- accelerating the pace of product/service delivery,
- maximizing quality,
- minimizing risk, and
- achieving significant cost savings with high ROI.

To begin, let's consider the big picture. The first major area of inefficiency, when looking at delivery of products and services to a network, is the fact that NEMs and NSPs typically act as two separate and independent silos. Bridging this chasm between the NEMs and NSPs is strategically important for the future of the networking industry. For example, NFV provides opportunities to reduce OpEx and improve the customer experience, but at the same time it introduces additional layers of complexity at an operational level. NEM products are no longer black boxes with fixed functionality. VNF allows them to deploy an almost endless array of features which is best suited to their business needs. Leading NSPs are looking at a more collaborative model that allows both the NEMs and NSPs to achieve their goals and objectives in a more synergistic manner.

This can be accomplished by establishing and end-to-end DevOps Value Stream in which both NEMs and NSPs are active contributors. As shown in the diagram below, we can see a value stream that allows the NEMs and NSPs to contribute directly to the end-product–the network services to be deployed.

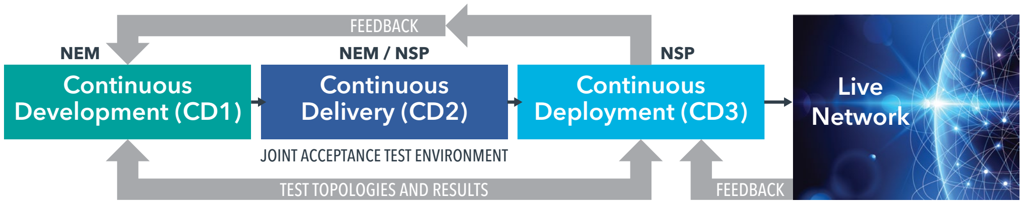


Figure 6: DevOps Value Stream Feedback

In Figure 6, we introduce the concept of end-to-end value stream feedback for network deployments. Feedback is the critical ingredient that enables continuous improvement. Here, we illustrate two types of feedback. First, network feedback is intended to provide key information to the NSPs about how network services are performing and how they can be improved. For example, using the feedback to derive more realistic wireless call models and usage to provide a more streamlined user experience at lower cost. The second type of feedback is Product feedback which is intended for NEMs to provide key information about how the NEM products and services are performing and how they can be improved. For instance, being able to determine how additional features impact resource utilization in a live environment.

As stated earlier, an important challenge is the size of the software deliveries. Steps must be taken to reduce the size of these software deliveries as smaller deliveries translate into higher quality, shorter cycle time, more timely feedback, and ultimately greater profit [Reinertsen, 2009].[6] This may sound counter-intuitive but there are plenty of examples today that make this point in a resounding fashion. Companies such as Amazon (AWS), Facebook, and Google have mastered this and understand the implications clearly. Reducing the sizes of the software deliveries will accelerate the delivery into the network and place more focus on smaller deliverables will enable higher quality levels. These translate directly to greater profit.

A critical aspect of the DevOps Value Stream is test automation. As with any DevOps initiative, automation plays a vital role in its implementation. We earlier described the automation layer responsible for moving along the various parts of a DevOps value stream. We will now introduce two additional automation layers: environment automation and test automation. As noted above, automating the test environment is essential in order to accurately capture and recreate the environment under which a test must be executed. It is not possible to run automated tests without an accurate representation of the test environment. It is also not possible to run tests in a DevOps environment without the ability to automate the creation of the required test environment. This capability enables NEMs and NSPs to validate environment definitions and allows them to build a library of automated tests that can be shared and executed arbitrarily along the value stream. This also enables NEMs to conduct automated tests along with their software deliveries so these automated tests can be used downstream by the NSPs to validate the delivery.

"Having the ability to automate the test environment and create a library of automated test cases shared by both NEMs and NSPs, it now becomes possible for NSPs to share their test environments with NEMS."

The last piece of the DevOps Value Stream is about creating shared test environments. Once NSPs have the ability to automate the test environment and create a common library of automated test cases for both NEMs and NSPs, it now becomes possible for NSPs to share their test environments with NEMs. Doing this solves two problems: first, the NEMs are able to test in an environment that correctly represents the target network environment. Second, the NEMs are able to validate the test automation they deliver to NSPs. The NSPs can automate the validation of products delivered by the NEMs and further automate the testing of their end of the value stream. This is only possible if the shared test environment is automated to accurately build the necessary test environment. Enabling NSPs with automation minimizes the rework cycle caused by bug discovery in the NSP part of the value stream, thereby reducing time and cost for both the NEMs and NSPs.

Scaling the Value Stream Delivery Model

Figure 7, Network Roll-out Strategy illustrates that implementation can be done incrementally for each value stream, when multiple NEMs are involved in the development and delivery of new products. Service providers can use a common testing ecosystem for large scale testing and use a specific discrete environment to test individual value streams.

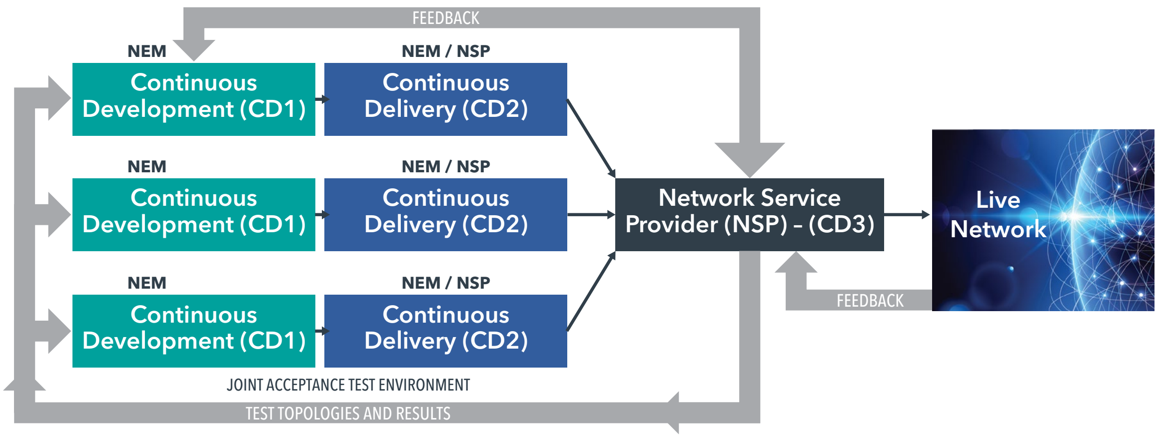


Figure 7: Network Roll-out Strategy

In this scaled view of the deployment model, NSPs must consider the dependencies between each value stream. This may require sequencing of each value stream delivery to ensure the network is not adversely affected because of incompatibility between the value streams. The network deployment approach is also a critical factor. As mentioned previously, NSPs must consider how changes are rolled out to enable the detection and recovery from failures introduced as a result of a change from any of the value streams.

The Test Environment Orchestration Layer (see Figure 3) helps supports this scalability by enabling the sharing, scheduling, and preparation of test environments or test environment components across the multiple value streams. Manual processes or fixed environments are not going to scale to the ever-increasing complexity of deployments which NSPs face today.

"

With the DevOps Value Stream for Networks in place, the NSPs have an effective blueprint for creating a collaborative ecosystem which allows both the NEMs and NSPs to achieve their goals and objectives in a more synergistic manner.

"

With the DevOps Value Stream for Networks in place, the NSPs have an effective blueprint for creating a collaborative ecosystem that allows both the NEMs and NSPs to achieve their goals and objectives in a more synergistic manner. With this blueprint in place, NSPs can accelerate the delivery of network services from typically months today to just days, thereby improving top-line revenue and business competitiveness. Reducing the size of the software deliveries into the NSPs leads to higher quality, shorter cycle time, more timely feedback, and ultimately greater profit.

The ability to dynamically create test environments allows NSPs to maximize the utilization of their resources, eliminate stranded resources, and increase quality by proving expanding test coverage in a richer environment. These operational efficiencies significantly improve resource utilizations and contribute directly to the business bottom line.

With the acceleration of service delivery and service quality, this will contribute to improved end-customer satisfaction and business growth.

Summary

This paper defined a blueprint for how to engineer NetDevOps using end-to-end CD/CD/CD (CD3) value streams for networks. The paper showed how engineering practices in which NEMS positioned as Dev, and NSPs positioned as Ops, cooperating through a common Delivery Stage, can realize important business outcomes including rapid delivery of new feature releases, while improving operational efficiencies and product and services quality.

The approach can help NEMS and NSPs struggling with digital transformations that are not considering the end-to-end benefits of a CD/CD/CD (CD3) value stream approach.

The paper explained that a common bottleneck for network value streams is test environment (lab) orchestration and testing of complex network topologies. Solutions involving Network Test Orchestration (Lab-as-a-Service) tools such as Spirent Velocity™ can greatly reduce the bottlenecks through automation of testing processes and tests.

References

1. "Engineering DevOps From Chaos to Continuous Improvement" … and Beyond. Marc Hornbeek, November 2019, mybook.to/engineeringdevops

2. "NetDevOps: When DevOps Comes To Networking", Shilpi,m August 23 2019, https://opensenselabs.com/blog/articles/netdevops

3. "What is NetDevOps?", Logan Rivenes, October 2016, https://datapath.io/resources/blog/what-is-netdevops/

4. "Continuous Testing Live Network DevOps", Marc Hornbeek, 2016, https://devops.com/continuous-testing-live-network-devops/

5. From Laboratories to DevOps Factories for Continuous Testing, Marc Hornbeek, May 2016, DevOps.com, https://devops.com/laboratories-devops-factories-continuous-testing/

6. Reinertsen, D. G. (2009). The principles of Product Development Flow: Second Generation Lean Product Development. Celeritas Publishing.

About the Authors

Marc Hornbeek, a.k.a., DevOps-the-Gray esq. is **CEO and Principal Consultant** at Engineering DevOps Consulting, author of the book Engineering DevOps , **Analyst** for the Accelerated Strategy Group and **Ambassador** of The DevOps Institute. Marc is a specialist / expert at applying a deep knowledge of engineering practices of Continuous testing, Test Automation, Risk Management, and Continuous Quality Assurance methodologies to DevOps transformations. Marc applies his unique, comprehensive ***Engineering Blueprints,*** **Seven-Step DevOps Transformation Blueprint** and ***9 DevOps Pillars*** discovery and assessment tools, together with targeted workshops to create actionable and comprehensive ***DevOps transformation roadmaps and strategic plans***. Marc is an IEEE Outstanding Engineer, and 44-year Senior of IEEE member. He is a **DevOps leadership advisor**/mentor. He is the original author of the ***Continuous Delivery Architect (CDA) and DevOps Test Engineering (DTE)*** **certification courses** that are offered by the **DevOps Institute**. He is a Blogger on DevOps.com and a ***freelance writer of DevOps content*** including webinars, and white papers. His education includes engineering and executive business degrees and multiple certifications from the DevOps Institute. Email Marc for DevOps consulting, training, writing and speaking engagements at mhornbeek@engineeringdevops.com, call him at +1 805 908 5789 or use this link to book a live 20-minute DevOps chat.

Pierre Frigon is the founder of iLeadAgile and is a recognized Agile leader with experience leading product development teams in several different roles including software development, system integration, system test, and test automation. Pierre is highly experienced in leading acceleration projects to solve some of the most complex deployments and automation challenges with demonstrated speedups of up to 300x with high ROIs. Pierre has also published several articles on the practical aspects of CI/CD and DevOps implementation.

Pierre has led many global teams responsible for the development of automation frameworks, automated regression, automated lab environments and system testing of next generation telecommunication products for national carriers operating in the access, wireless backhaul and edge routing space. He is a Certified Data Center Designer,
Agile Leadership coach and co-owner of Management 3.0. Pierre can be contacted via email at Pierre.Frigon@iLeadAgile.com.

About Spirent Communications

Spirent Communications (LSE: SPT) is a global leader with deep expertise and decades of experience in testing, assurance, analytics and security, serving developers, service providers, and enterprise networks.

We help bring clarity to increasingly complex technological and business challenges.

Spirent's customers have made a promise to their customers to deliver superior performance. Spirent assures that those promises are fulfilled.

For more information, visit: www.spirent.com

About the Sponsor

Spirent Communications (www.spirent.com) provides innovative products and services to help the world communicate and collaborate faster, better and more securely. Spirent's customers operate in markets characterized by relentless traffic growth, ever-increasing security challenges and continual pressure to innovate while generating revenue and managing operating costs. Our testing and measurement capabilities are needed now more than ever to assure our customers can deliver on the promises they make to their customers to keep them safe, secure and connected.

spirent™
Promise. Assured.

Contact Us

For more information, call your Spirent sales representative or visit us on the web at www.spirent.com/ContactSpirent.

www.spirent.com

Americas 1-800-SPIRENT
+1-800-774-7368 | sales@spirent.com

Europe and the Middle East
+44 (0) 1293 767979 | emeainfo@spirent.com

Asia and the Pacific
+86-10-8518-2539 | salesasia@spirent.com

Rev A | 06/20