# An Efficient Software Delivery Framework for 5G Environment

Vijay Raghavendran
Distinguished Engineer, CX
Cisco Systems

# Contents

## Abstract

**Today's world industry is moving from a traditional hardware-centric approach to a microservices-based, software-driven, cloud-native environment. This biggest disruption is seen in 5G Service Provider (SP) architecture. The software-centric approach promotes more automation and service versatility. 5G use cases that target enterprise and industry vertical markets leverages the power of Software Defined Network by using a service-based architecture. This means an echo-system of vendors from the Application layer all the way to Physical Transport layers, opening up their systems to bring innovative services.**

**In this heterogenous vendors environment, the SP needs to increase speed of operation efficiency, enhance the user experience, and expand ongoing revenue growth. Hence, comparing to legacy software development, we need a model that can provide faster quality delivery of vendor software in a continuous incremental manner, ensuring best ROI. Based on our experience with multiple 5G Architecture, Cisco® proposes a model merging the key tenants of DevOps, CICD, and Agile Framework called Continuous Delivery and Automation Framework (CDAF).**

**Using this framework, the SP can:**

- **Automate the Vendor Software Release Lifecycle Management by extending to Vendor's CICD environment**
- **Build a thin controller platform with specific set of open-source CICD toolset for targeted function**
- **Offer a unified single user experience (UX) for end-to-end software delivery management**
- **Augment with the repetitive structured Robotic Process Automation to a more unstructured and human logic AI/ML based**
- **Optimize management of Cloud Infrastructure Resources required to support high number of Software packages**
- **Automate Test Management Test Cases selections using AI/ML**
- **Create a vendor partners echo system, to bring developers closer to the customer pre-production testbed**
- **Leverage the AI/ML based Release Certification to increase the software velocity without compromising the carrier grade requirement**

**This document provides a Case Study 5G SP Customer with a successful deployment of this framework, along with lessons learned.**

## Introduction

Whether you are an incumbent SP or a new entrant looking to provide innovative services using 5G-based architecture, you need to align your vision of overall company digital transformation.

A good starting point is a vision around automated continuous delivery of the vendor's certified quality software, providing a stable and reliable software catalog for consumption in production.

Identify key tangible outcomes mapping to the above vision. Some of these are:

- Automated Software Artifact Registration
- Automated Business Process Automation

- Automated Build Environment
- Automated Deployment
- Automated Testing and Certification
- Automated Staging for production move

To measure the success of these outcomes, identify some KPIs. To hold the vendors accountable, make these KPIs available to echo-system partner vendors in real time (through a public secure access), for transparency and proactive actions.

- Quality
  - Failures per stage
  - Failures to defect ratio
  - Failures/defects per supplier
  - Pass/Fail ratios
  - Defect escape rate (how many defects make it to production)
- Efficiency
  - Idle time (Manifests at rest)
  - Deployment time
  - Ticket/SR Volume and Queue
- Time to Market
  - Progression rates (through testing stages)
  - Deployment frequency
  - Other
- Availability
  - Error rates and budget
  - MTTD/MTTR

As part of execution of the above vision, it is imperative to visualize a production flow, as shown in Figure 1.

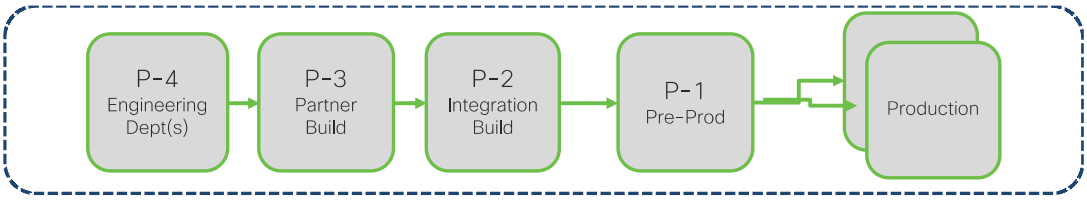


**Figure 1.**
CNF/VNF or Software Flow

Based on the SP business model, P-3, P-2, and P-1 can be managed by a third-party neutral testing company or themselves. Over time, with a high degree of automation implemented, it will make sense for SP to own as much responsibility. Initially, a compromise can be achieved by pushing P-3 and P-2 responsibility to vendors. However, due to heterogenous vendor solutions, it remains the SP's responsibility to certify the overall solution.

Expect a scenario in which the extension between SP environment and multiple vendors will expand very quickly. This means SP should have ability, system, and an operation framework to expand to multiple partners,

as shown in Figure 2.

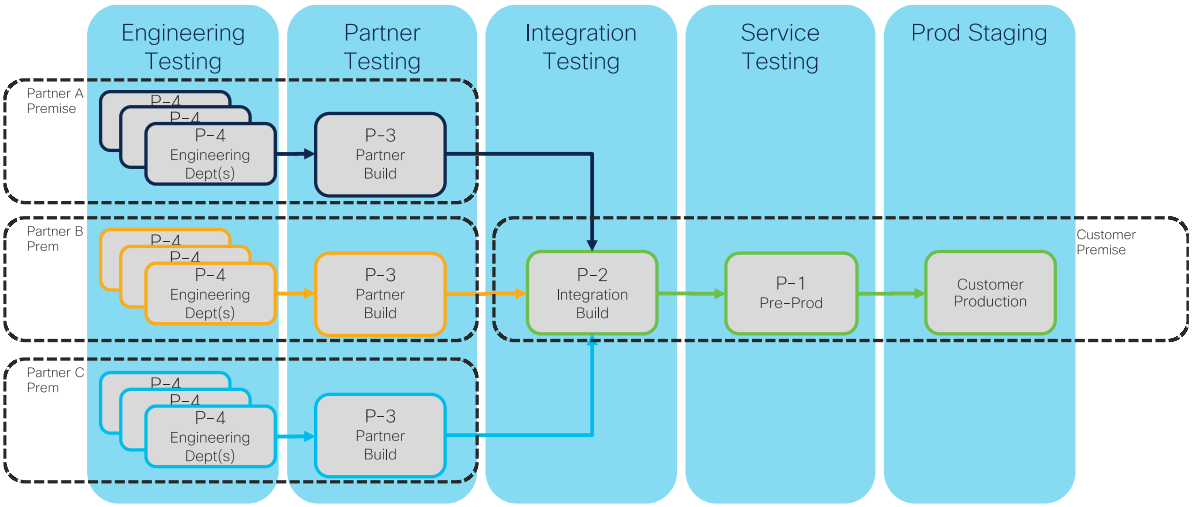


**Figure 2.**
Scale to Partners

In an agile world with multiple suppliers, the ability to regulate on a per-software feature basis becomes unmanageable and time consuming. This method is too detailed for successful operations. Our proposed solution is to organize software on its journey to production—to focus on integration and interoperability more than feature regression testing. During certification stages, software artefacts are tested in logical groups and then aggregated into larger packages for collective deployment to production. Those packages are called Manifests (see Figure 3). Manifests include configurations of devices with different formats, such as JSON, XML, or YAML, or even Ansible playbook. These are critical for DevOps or NetDevOps operations.

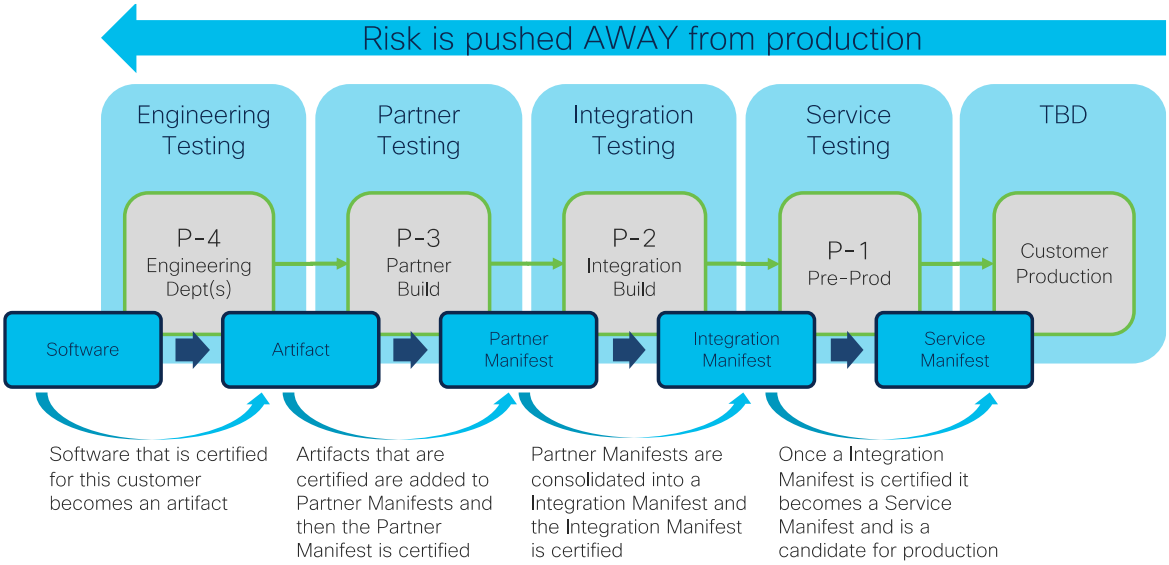


**Figure 3.**
Manifest

We highly recommend a journey map to move gradually from Semi to Full to Self-Driving DevOps Automation. There will be adoption challenges if you try to implement Full or Self-Driving DevOps automation in one go. We recommend a more modular, functional, and phased approach. This sets the stage for our proposed framework, Continuous Deployment and Automation Framework (CDAF).

CDAF enables a journey of an increased level of automation, from simple discrete workflow automations to a more complex RPA or AI/ML-based automation. This is possible due to a modular and discrete functional-based approach, to bring People, Processes, and Technology together.

This framework's achievable goal is to reduce the complexity of software deployment by modularizing and integrating various components through automation, to provide the best customer experience. See Figure 4 for details.

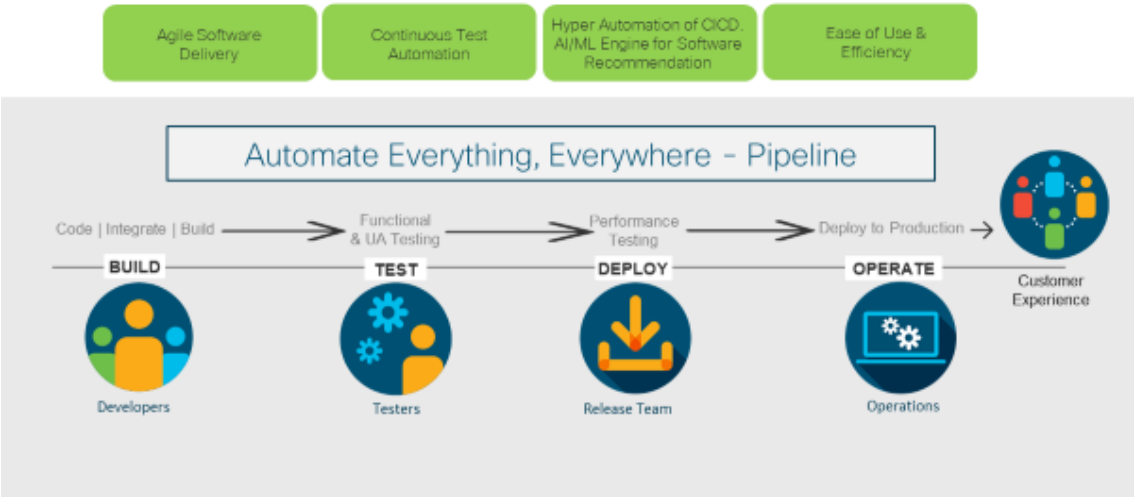


**Figure 4.**
Automate Everything

As part of the agile software delivery methodology, this framework broadly covers the following phases, as depicted in Figure 5.

1. Scope Management (SM)
2. Development Management (DM)
3. Build Management(BM)
4. Test Management (TM)
5. Governance Management (GM)
6. Release Management (RM)
7. Deployment Management

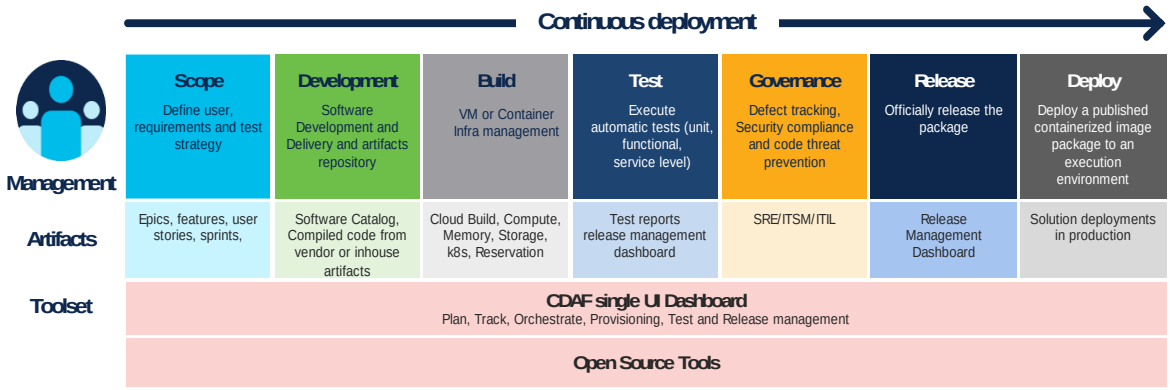


**Figure 5.**
CDAF

The framework proposes leveraging various DevOps Open Source CICD Tools. It is tools agnostic and recommends using tools with the best API available and published for integration. In many of our SP implementations, we were able to leverage existing DevOps CICD tools such as JIRA, CA Rally, Jenkins, GitHub, and JFrog Artifactory.

We suggest building a CDAF Dashboard or a controller with a full stack and a microservices architecture. This controller is pivotal for automation success as it is equivalent to a brain controlling various modular component to provide a unified UX. This UX is a very important point to convince multi vendors to adopt using a single framework and platform. Allowing each vendor to bring their own CICD tools and processes will result in lack of accountability and poor outcome.

## Details of Framework

### Scope Management

Scope Management brings various organizations together to implement the vision of CxO, starting with Program Management Office (PMO) and Architecture Management Office (AMO). This is a very important starting point of the CDAF.

CDAF proposes implementing the Scaled Agile Framework (SAFe) architecture to aid collaboration between different teams, as shown in Figure 6.

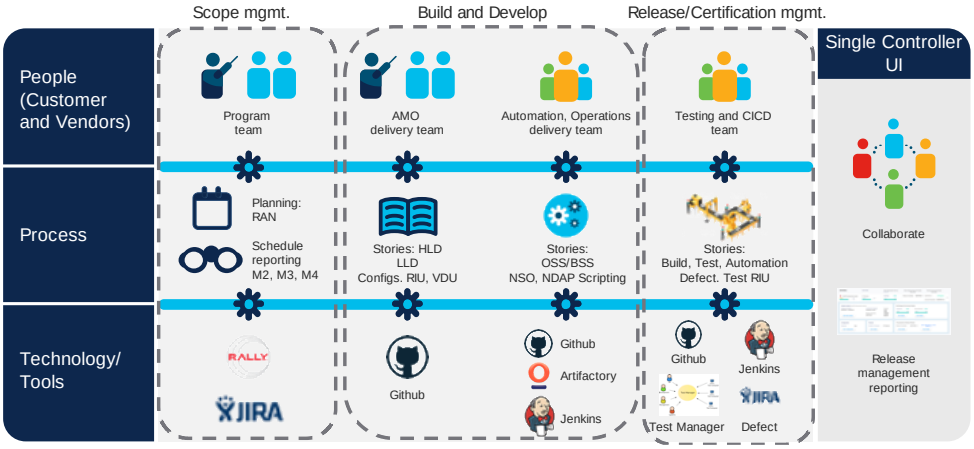


**Figure 6.**
SAFe – Bridging People, Process, and Technology

It is also important to identify the stakeholders involved in the process. Approval process of various workflows should be captured from the interview with different departments, as shown in Figure 7. This is crucial as we go through the journey of RPA or AI/ML-based approval.

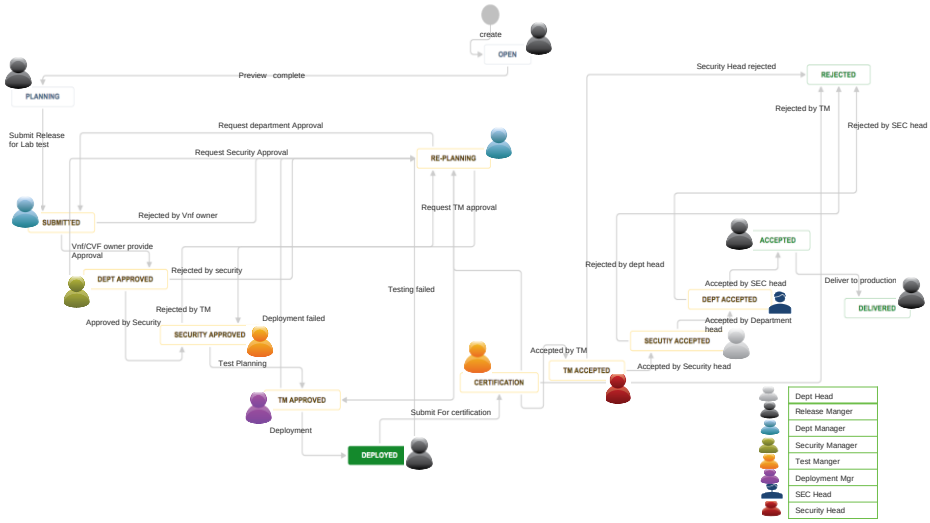


**Figure 7.**
End-to-End Business Process Workflow

A simple SAFe approach is as follows:

1. Use an inclusive team approach. Open up the access to the common tool for all vendor points of contact to be part of one extended team.
2. Take the existing functional groups working on the project as the Teams under SAFe.
3. The portfolio is captured as Initiative with numerous Epic or Features, as shown in Figure 8.
4. Epic will be broken down into stories or smaller tasks.
5. We recommend creating a simple and standard hierarchy based on SAFe and not dependent on tools like Rally or JIRA. This structure helps in mapping the user story, not only inside the PMO or AMO or Automation groups, but also across different groups. This is critical to map the Software Delivery flow.
6. Prepare the Agile Delivery Framework jointly with your vendors to capture Deliverables and Roles and Responsibilities, as shown in Figure 9.

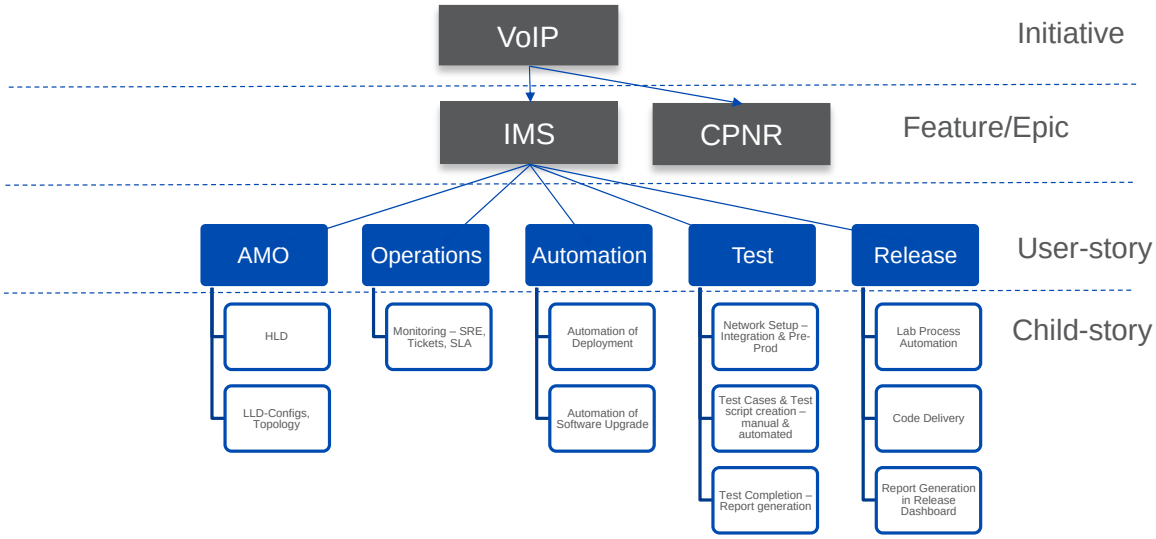


**Figure 8.**
SAFe Approach for CDAF

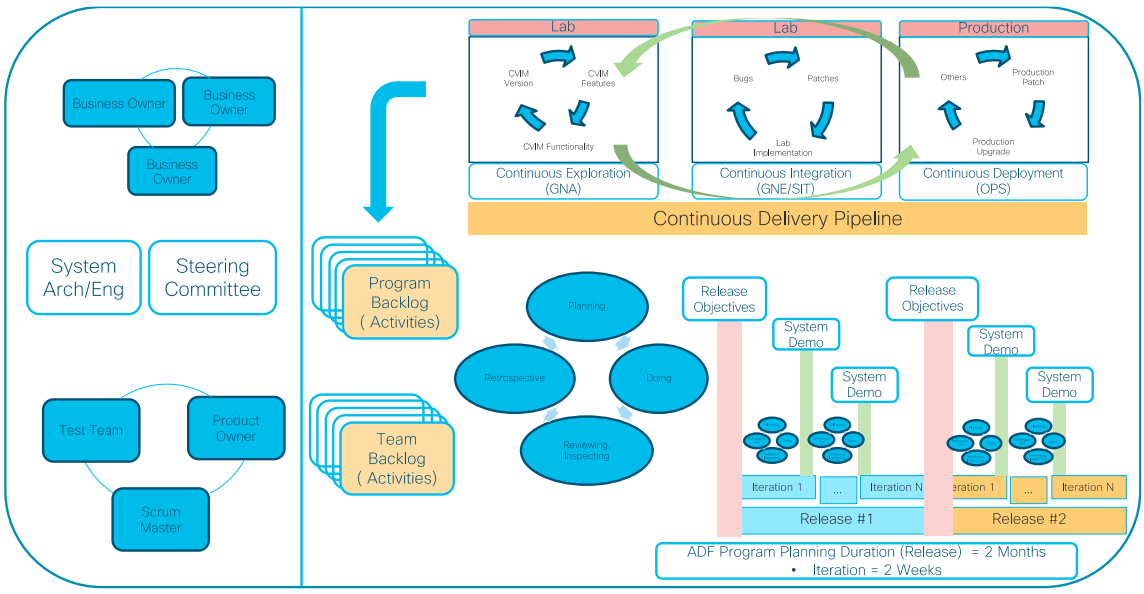


**Figure 9.**
Agile Delivery Framework

## Development Management

Whether custom software is being developed by customer partner, or software packages are provided by vendors, Development Management provides the ability to continuously deliver software to the customer's premises. We recommend starting with a software catalog approach, which should include the following key components:

- Self-vendor Registration
- Automated Software Delivery from vendors
- View of releases by specific vendor
- All application release details and hardware requirement specifications
- Identification of the current software release in the lab, as well as the last certified software
- Identification of software behavior deployed in production
- View of release-specific KPIs, such as defects, test results

Figure 10 is an example of the above components through a single UI. The software catalog should be able to provide every vendor's software manifest, along with KPIs. To attain this, a secure capability Automated Software Delivery can be a push or pull model. Usually, a pull model from the SP environment can be implemented easily.

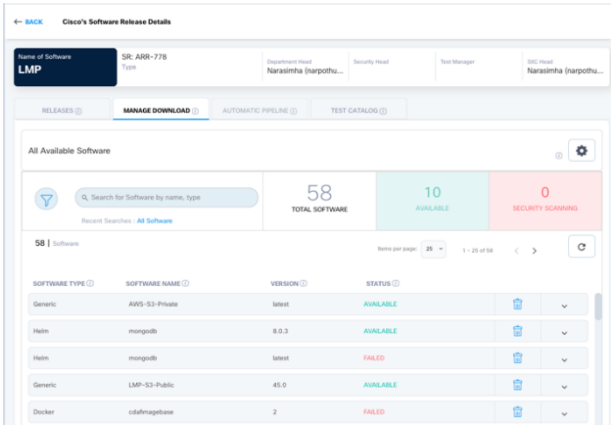


**Figure 10.**
Software Catalog

## Build Management

This is a critical phase of the framework. Today's 5G architecture may be built on OpenStack VMs services or K8s Container services. When a vendor provides their software, as part of the manifest, they have to provide resources required and configuration parameters. These will be leveraged as part of the automated pipeline to build the Development or Integration or Staging Environment.

The following steps provide an example of how to implement a Build Management:

1. Type of Infrastructure: OpenStack VMs or K8s Containers
2. Develop a Resource Request feature for manual or auto discovery through Helm parsing.
3. Create a Resource Management Dashboard to track your Build Environment resource usage, as shown in Figure 11.
4. Orchestration integration: For OpenStack, use NSO or Ansible. For K8s, you can choose Jenkins along with Helm, or even a simple microservice calling the APIs of K8s wrapper service.

5. Reservation System: Using data from (2) and (3), you can create an approval process.
6. Successful approval from (5) will result in a deployment pipeline using (4), as shown in Figure 12.

The build resource can be moved from Development to Staging, and eventually to Production.

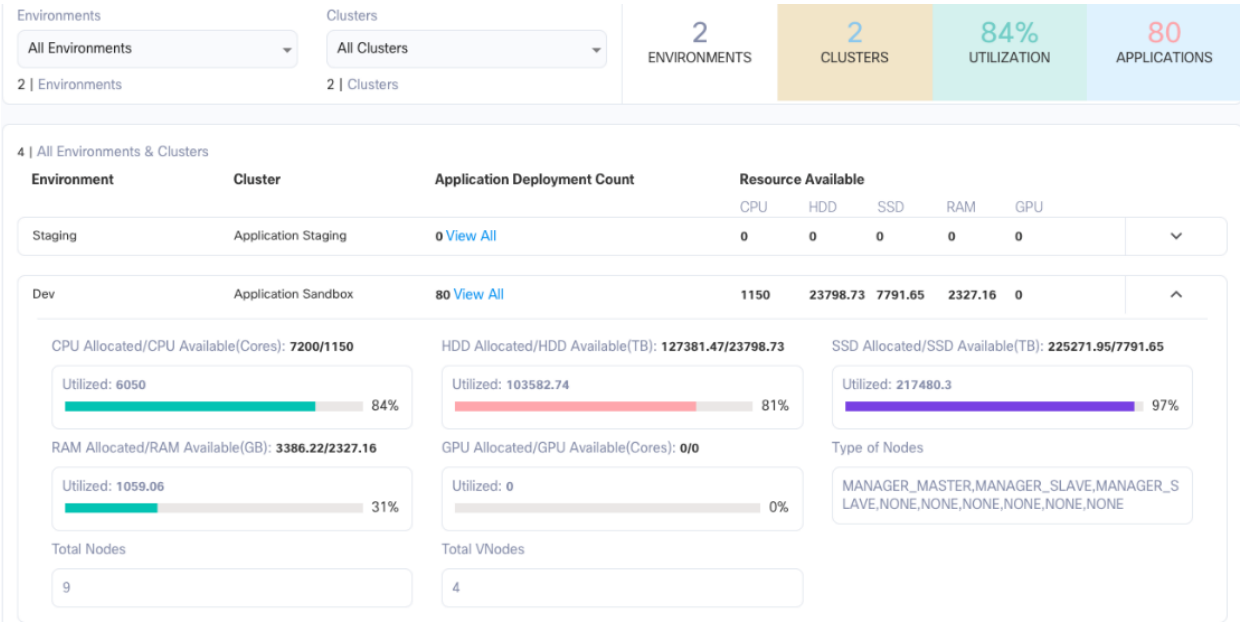


**Figure 11.**
Resource Management Dashboard

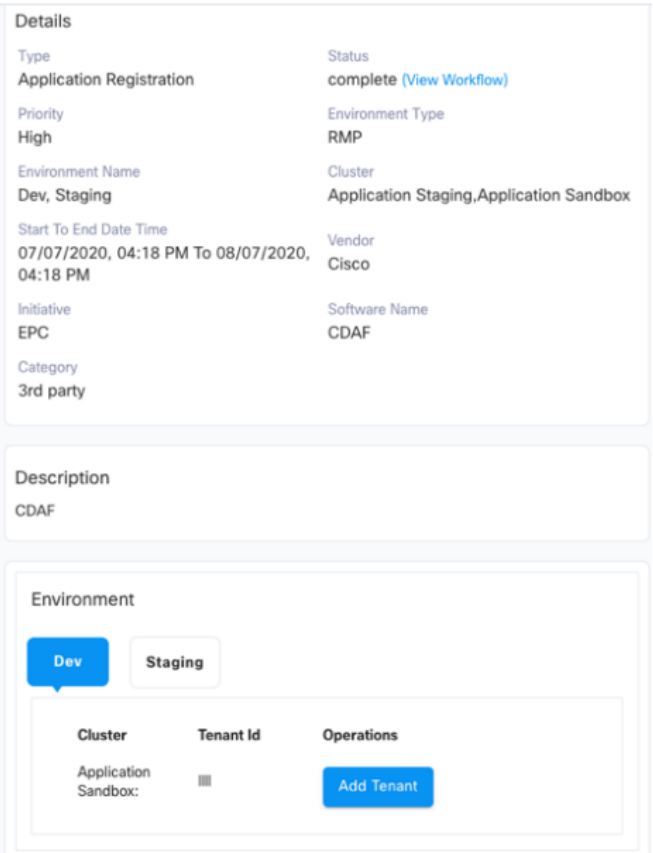


**Figure 12.**
Successful Build and a Deployment

## Test Management

Test Management is necessary for CDAF and CI Scope because it is required for Certification. Using a Test Manager (TM) with API capability can help make this process less tedious. We recommend the following steps to assist in providing efficient, accurate, targeted testing:

1. Map the initiative from scope management from Initiative to Feature/Epic to a corresponding project creation in the TM.
2. Maintain a Master Test Catalog per vendor in TM. Vendor testers write the Test Case (TC) in the TM tool. Corresponding test scripts are maintained in GitHub. Make sure the TM has the appropriate tag schema to identify types of TCs.
3. Start with a basic selection mechanism in the common Dashboard to map the Scope to TCs using tags. Over time, leverage the AI/ML search mechanism to identify the correct test cases, eliminating tags entered manually. This helps the user select the optimal test case for respective project scope testing.
4. Create automated pipeline to trigger the Test Project after a successful Build. This process should be scheduled and queued from a common UI outside the TM. Schedule Automated versus Manual TCs at different time periods. Run automated TCs at night. Schedule serial execution or parallel execution of TCs.
5. Test results should be maintained in a single DB outside the TM. This will be a single source of truth for Scope Management, Test Management, Release Management, and overall Governance of the entire test process. See Figure 13 for an example of depicting all data in one snapshot.

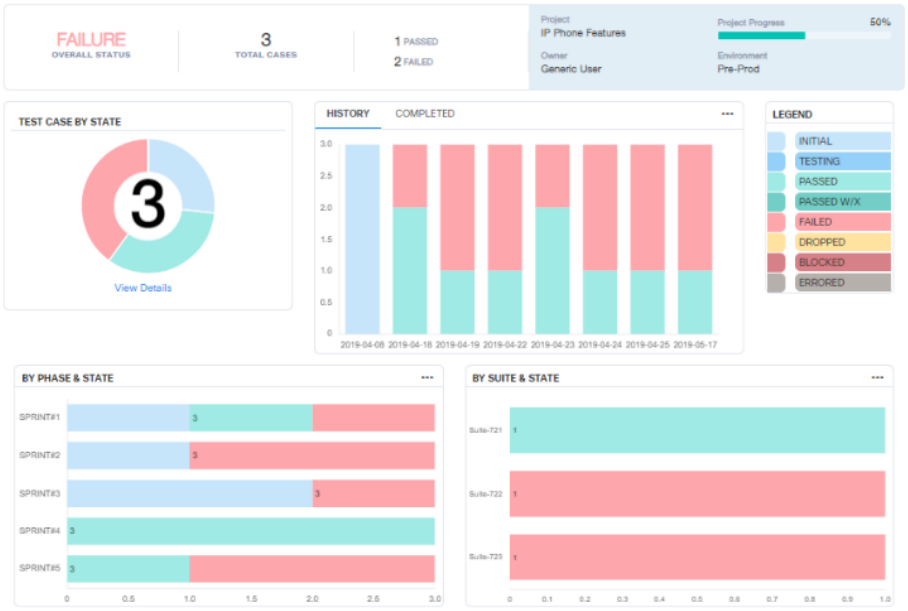


**Figure 13.**
Example of Test Management Dashboard

## Governance

Governance covers various guardrails around process, standard conformance, and security. Various KPIs to measure a vendor's performance (such as defects) also should be captured. This is not restricted to proprietary offerings of vendors. It should be integrated with SP-specific security processes or equivalent processes, if any.

A Defect Management system extending from SP to vendors consists of:

- **Defect Discovery:** During various test stages (Integration, Solution and Pre-prod), issues will be found and should be reported on ticketing tools. The capability through common UX should bypass user logging into the tools.

- **Defect Resolution:** CDAF recommends vendors to track defects until resolution. The Vendor Defect system should be integrated with the tool; a single Defect tool can be used for faster resolution across all vendors.
- **Defect Management for Solution Release:** SP should be able to define the open defect criteria for vendors to be compliant for any solution release.

### Release Management

Release Management is chiefly concerned with how changes flow through pre-production environments. The goal is a successful release with deployment of changes into the production environment, with a minimum of disruption. Release Management involves:

- Configuration Management
- Release and Deployment Management
- Designing
- Planning
- Rollout Planning
- Testing Communication

Figure 14 provides an example of RM implementation of the CDAF dashboard:

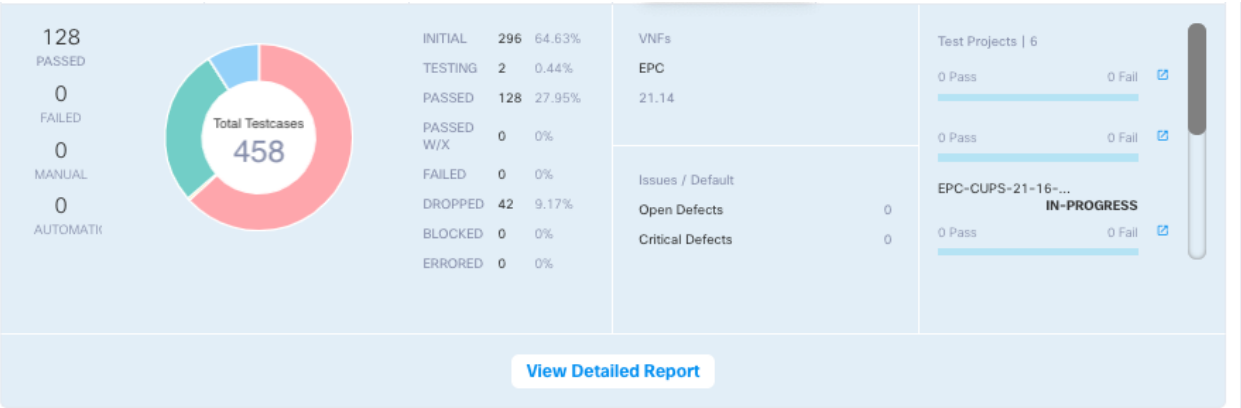


**Figure 14.**

Example of a Release Management Dashboard

## AI/ML in CDAF

AI/ML engines can be developed to solve multiple complex problems and to improve accuracy and productivity. This section highlights a few examples of Use Case implementations to illustrate the power of AI/ML. The Framework, combined with a central DB with all information, makes the AI/ML engine very powerful.

At a high level, the software development lifecycle can be broken down to three stages (see Figure 15). You can start with few sample Use Cases to improve productivity, agility, and accuracy in these areas.

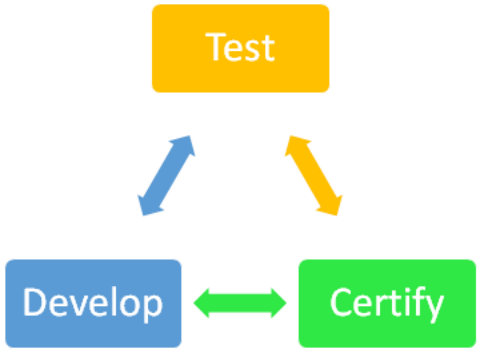


**Figure 15.**
**Software Development Lifecycle**

- Test Stage:
  - Using NLP for Test case de-duplication helps.
  - Test case ranking provides the ability to move from supervised to unsupervised learning, based on user behaviors.
  - Test case search allows new users to narrow down the Test Cases based on expert recommendation.
  - Test project simplified “onboarding” simplifies Test project creation (as the name suggests).
- Certify:
  - ML-based Software Recommendation uses performance attributes from the production network to recommend the feasibility of a new tested software.
  - ML-based Software Certification automates the release certification process workflow by minimizing human approvals.
- Develop:
  - AI/ML-based Knowledge Base captures the true multi-vendor lab performance and troubleshooting data for operation support.
  - ML-based Chatbot assists various functional groups in resolving issues.

## Lessons Learned

A strong executive sponsorship is required to implement this framework. Some key lessons learned are:

1. Waterfall approach of a legacy SP to an Agile approach is a huge transformation effort. Multiple organizations should sign up, and an executive mandate should be pushed.
2. Start with a modularized approach. For example, Test Management can be conducted without interdependency of Scope Management. After successful transformation effort of (1), integration with Scope Management can happen. Eventually, you can simplify the end-to-end flow.
3. Involve users in an iterative approach. For a successful adoption, it is imperative that users provide their problem statements along with their alternate approach. The CDAF value proposition is compelling to replace their existing approach.
4. Vendors have CICD tools preference. Choose the best open source (Enterprise License) and adopted tools, and provide the ability for vendors to bring some of their own scripts for the tools. Tool examples include Jenkins and Ansible.
5. Set the Defect types and priority levels with a vendor consultation. Everyone should have a clear understanding of an issue versus a defect; not all issues are defects.

6. If you want to put the onus of testing on vendors, it is fine during the initial stage. However, you should take responsibility sooner, as a heterogenous environment of a 5G solution is difficult for any vendor to recreate in their environment.
7. Convince all vendors to adopt a single TM or a single platform and framework. This will help in having one single DB. A single DB helps in automating and building a closed loop self-driving approach.
8. Keep the BPMN flow minimal in Release Management or Build management. Manual approval can delay process approval. With RPA or AI/ML, you can eliminate Manual approval processes.

## Conclusion

CDAF provides capabilities beyond traditional CICD to meet the requirements of an SP carrier grade requirements. The impact on KPI improvements is huge, specifically on Time-to-markets, Software quality, Agility, and Transparency index. We have measured KPI improvements over 5X times, as explained in the following Case Study.

### Case Study

A greenfield advanced 4G/5G Service Provider wanted to implement an Agile CICD Framework. The CTO's expectations were:

- Improved Code Quality
- Accelerated multivendor co-development
- Faster innovation

The expectation was that the business outcome from CICD automation would result in cost savings in product testing certification without compromising quality.

A vision structured around the CTO's expectations resulted in the development of a platform based on CDAF, as shown in Figure 16.

The initial focus was on their Mobility Backend Cloud Infrastructure based on Cisco® OpenStack (CVIM). Now, the SP has built a parallel infrastructure for a Cloud Native Application, based on Kubernetes as an Operating or Orchestration System. The expectation is that vendors will provide the Containerized Network Function (CNF).

Some core attributes of the CICD platform developed are:

- Microservices-based architecture with open-source tools (GitHub, Artifactory, Jenkins, CA Rally, JIRA, and Ansible)—which are easy to deploy and integrate. This allowed co-development of the platform, with the customer building their own Use Cases and integrating through microservices.
- Agile software lifecycle management for co-development between multiple vendors. Due to the customer mandate, all vendors were forced to use the platform. An automatic code pull from the vendor's multiple type repository (Docker, AWS-S3, GitHub, and public accessible Artifactory) was implemented.
- AI/ML-based software release management, certifying and deploying multiple package versions. Primarily, AI/ML was used with NLP to enforce approval of certification results. Additional AI/ML use cases on test cases improvement are enhanced constantly, while being deployed and adopted by multiple vendors.
- Agile defect tracking and resolution from direct access to the customer's on-premises resources to BU development.

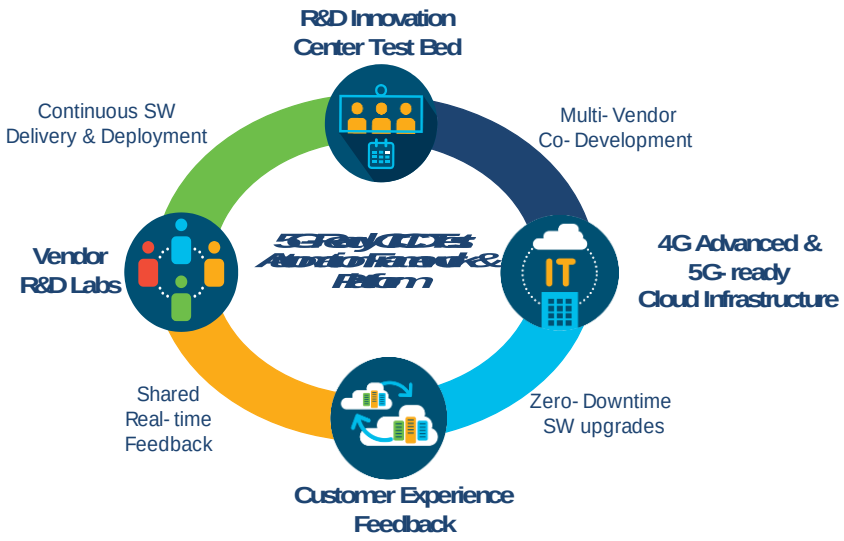


**Figure 16.**
Case Study of a 4G/5G Implementation

In an 18-month timeframe of implementing CDAF, following are some KPIs we captured:

1. **High adoption by vendors:** Managed 70+ vendors and 1900 users securely.
2. **High number of software packages:** More than 300 vendor packages from different types.
3. **Number of releases:** 720 releases addressed and 300 certified for deployment.
4. **Compliance:** More than 75% release-certified software deployed in production environment.
5. **Test Cases:** More than 45,000 Test Cases developed by multiple vendors.
6. **AI/ML Duplicate/Quality Elimination:** More than 60% of TCs deprioritized from databases.
7. **Issues/Defects:** More than 1,500 vendor Issues/Defects identified and resolved.

## References


[1] 2020 IEEE 3rd 5G World Forum(5GWF) , Vijay Raghavendran, Sudipta Debnath - CDAF - an Efficient 5G Service Delivery Framework for Agile/DevOps Environment

[2] SAFe Provided by SCALED AGILE – scaledagileframework.com

[3] Kubernetes, Cloud Native, and the Future of the Software – Kubernetes.io Blog

[4] Developer Tools on AWS

[5] Microservices on spring.io

[6] Devops.com, “5 Metrics you should know to understand your Engineering Efficiency,” offered by CircleCI, January 3, 2020. *(references)*

[7] Derek E. Weeks, 31 Reference Architectures for DevOps and Continuous Delivery, April 22, 2015

[8] Vijay Raghavendran and Noam Ben Gal, “The CDAF customer Proposal document” unpublished.

[9] Joseph Callen, Aaron Weitekamp, Pep Mauri “Reference Architectures 2017 Application CI/CD on OpenShift Container Platform with Jenkins,” Red Hat



**Americas Headquarters**
Cisco Systems, Inc.
San Jose, CA

**Asia Pacific Headquarters**
Cisco Systems (USA) Pte. Ltd.
Singapore

**Europe Headquarters**
Cisco Systems International BV Amsterdam,
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **https://www.cisco.com/go/offices**.





Printed in USA C11-743921-00 06/20