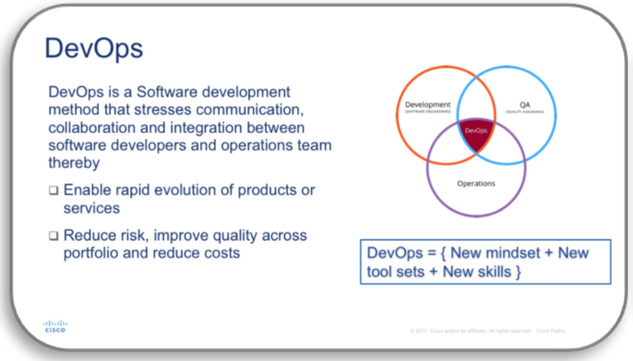
NSO “Continuous Integration”, “Continuous Delivery”

Srilakshmi Kanda, Test Engineer

NSO Developer Days 2017

Overview

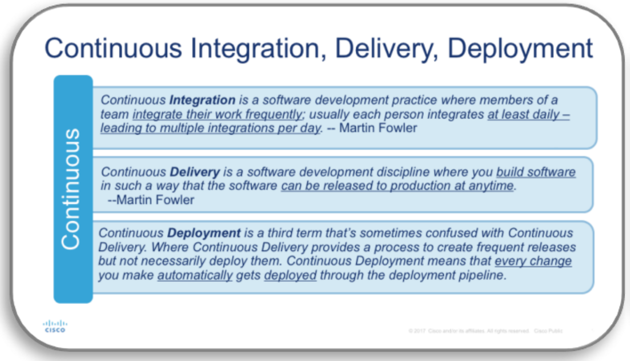


DevOps

DevOps is a Software development method that stresses communication, collaboration and integration between software developers and operations team thereby

- Enable rapid evolution of products or services
- Reduce risk, improve quality across portfolio and reduce costs

DevOps = { New mindset + New tool sets + New skills }



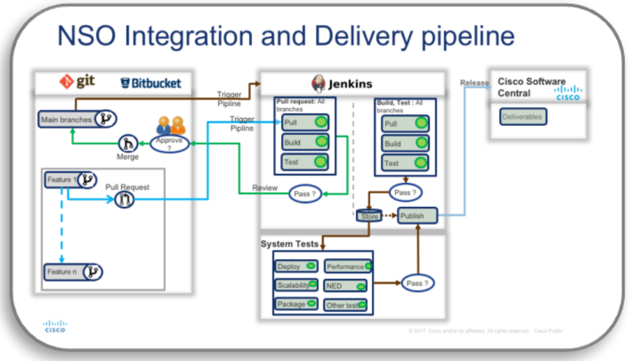
Continuous Integration, Delivery, Deployment

Continuous

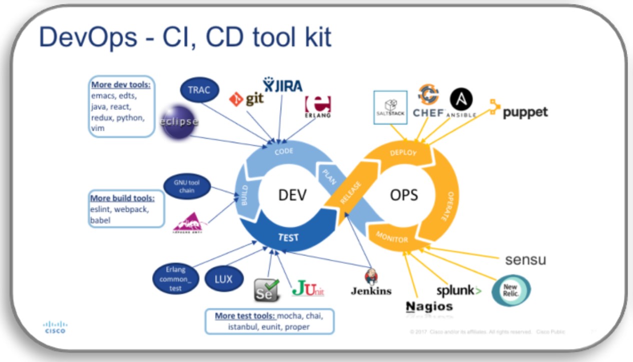
*Continuous **Integration** is a software development practice where members of a team <u>integrate their work frequently</u>; usually each person integrates <u>at least daily – leading to multiple integrations per day</u>.* -- Martin Fowler

*Continuous **Delivery** is a software development discipline where you <u>build software</u> in such a way that the software <u>can be released to production at anytime</u>.* --Martin Fowler

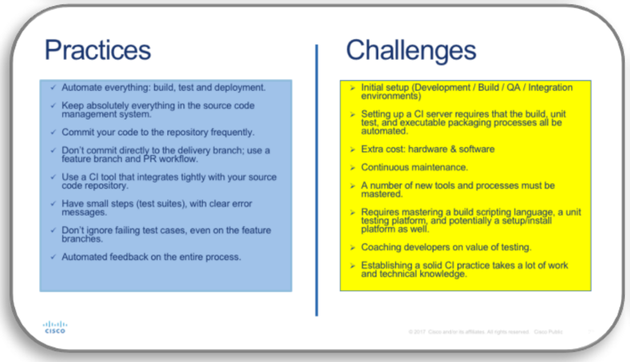
*Continuous **Deployment** is a third term that's sometimes confused with Continuous Delivery. Where Continuous Delivery provides a process to create frequent releases but not necessarily deploy them. Continuous Deployment means that <u>every change</u> you make <u>automatically</u> gets <u>deployed</u> through the deployment pipeline.*



NSO Integration and Delivery pipeline



DevOps - CI, CD tool kit



Practices

- ✓ Automate everything: build, test and deployment.
- ✓ Keep absolutely everything in the source code management system.
- ✓ Commit your code to the repository frequently.
- ✓ Don't commit directly to the delivery branch; use a feature branch and PR workflow.
- ✓ Use a CI tool that integrates tightly with your source code repository.
- ✓ Have small steps (test suites), with clear error messages.
- ✓ Don't ignore failing test cases, even on the feature branches.
- ✓ Automated feedback on the entire process.

Challenges

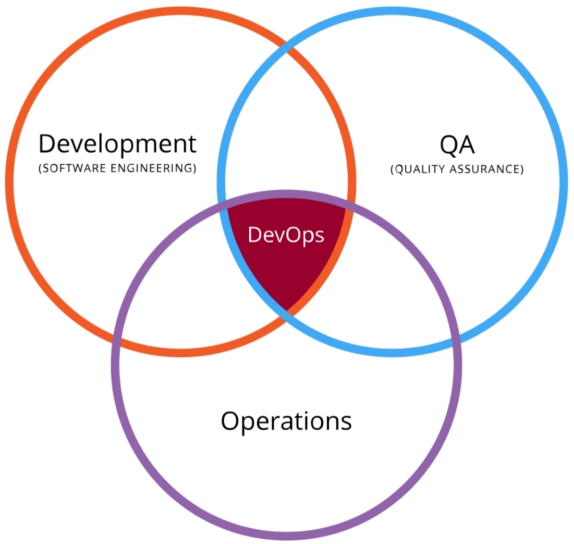
- Initial setup (Development / Build / QA / Integration environments)
- Setting up a CI server requires that the build, unit test, and executable packaging processes all be automated.
- Extra cost: hardware & software
- Continuous maintenance.
- A number of new tools and processes must be mastered.
- Requires mastering a build scripting language, a unit testing platform, and potentially a setup/install platform as well.
- Coaching developers on value of testing.
- Establishing a solid CI practice takes a lot of work and technical knowledge.

Q&A

DevOps

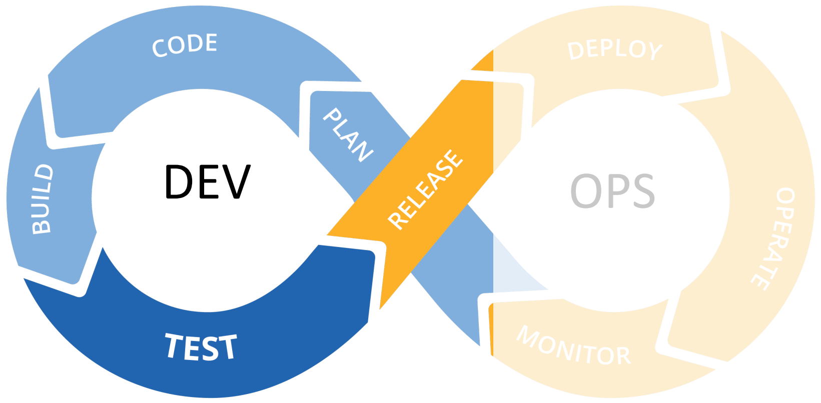
DevOps is a Software development method that stresses communication, collaboration and integration between software developers and operations team thereby

- ❑ Enable rapid evolution of products or services
- ❑ Reduce risk, improve quality across portfolio and reduce costs



DevOps = { New mindset + New tool sets + New skills }

DevOps Life Cycle



Continuous Feedback

- Plan and Measure
  - Continuous Business Planning
- Develop and Test
  - Collaborative Development and Continuous Integration / Testing
- Release and Deploy
  - Continuous Release and Deployment
- Monitor and Optimize
  - Continuous Monitoring

Continuous Integration, Delivery, Deployment

Continuous

*Continuous **Integration** is a software development practice where members of a team integrate their work frequently; usually each person integrates at least daily – leading to multiple integrations per day.* -- Martin Fowler

*Continuous **Delivery** is a software development discipline where you build software in such a way that the software can be released to production at anytime.*
--Martin Fowler

*Continuous **Deployment** is a third term that's sometimes confused with Continuous Delivery. Where Continuous Delivery provides a process to create frequent releases but not necessarily deploy them. Continuous Deployment means that every change you make automatically gets deployed through the deployment pipeline.*

Why CI and CD ?

- To encourage a culture of incremental development
- To ensure our system is working all the time. To ensure that the build is always in a “green” state.
- To improve the visibility of the current state of the build. (failed, successful, etc.)
- To establish greater confidence in software product from the development team.
- To reduce risks.
- To reduce repetitive manual process.
- To receive Regular feedback
- To Reduce integration pain
- To enable concurrent development
- To Increase automation

Prerequisites for CI

Version Control: Checking all the project scripts into a central repository (code, test, configuration)

- Check in regularly to mainline
- Managing your development workspace

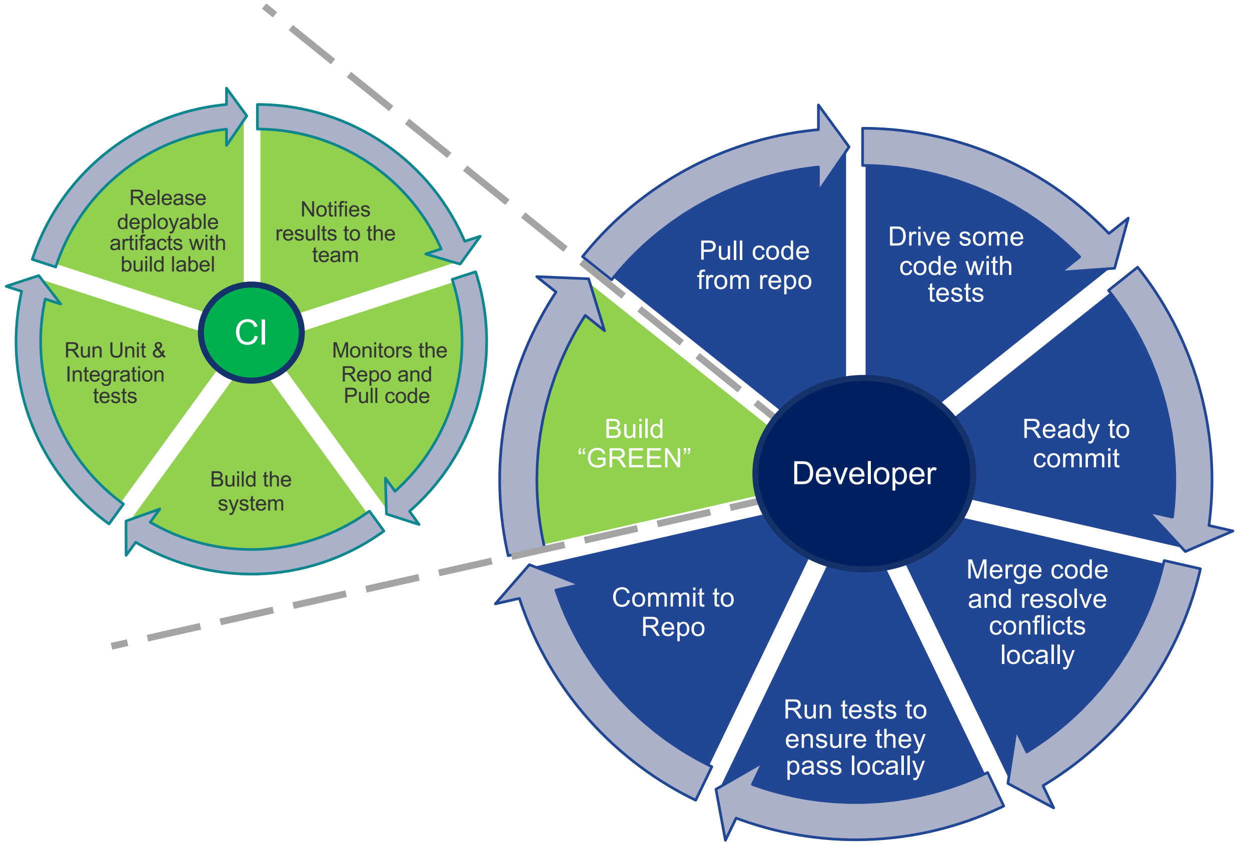
An automated build: Automating the compilation, testing and delivering processes. (Create a comprehensive test suite.)

- Keep the build and test process short
- Standardizing automation.

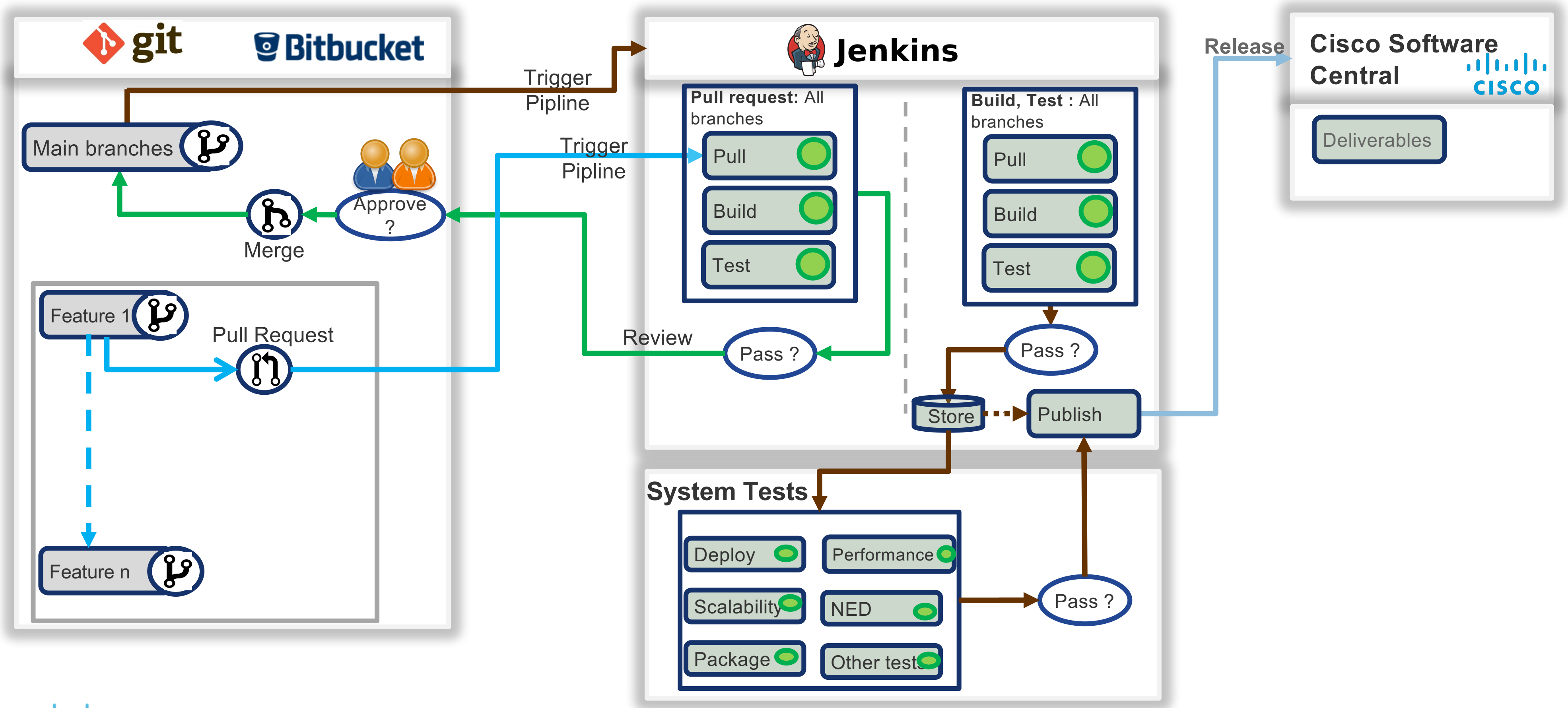
Agreement of the team: CI is a practice rather than a tool and thus requires the team's input.

- Feedback Mechanism
- It is a Cultural Movement.

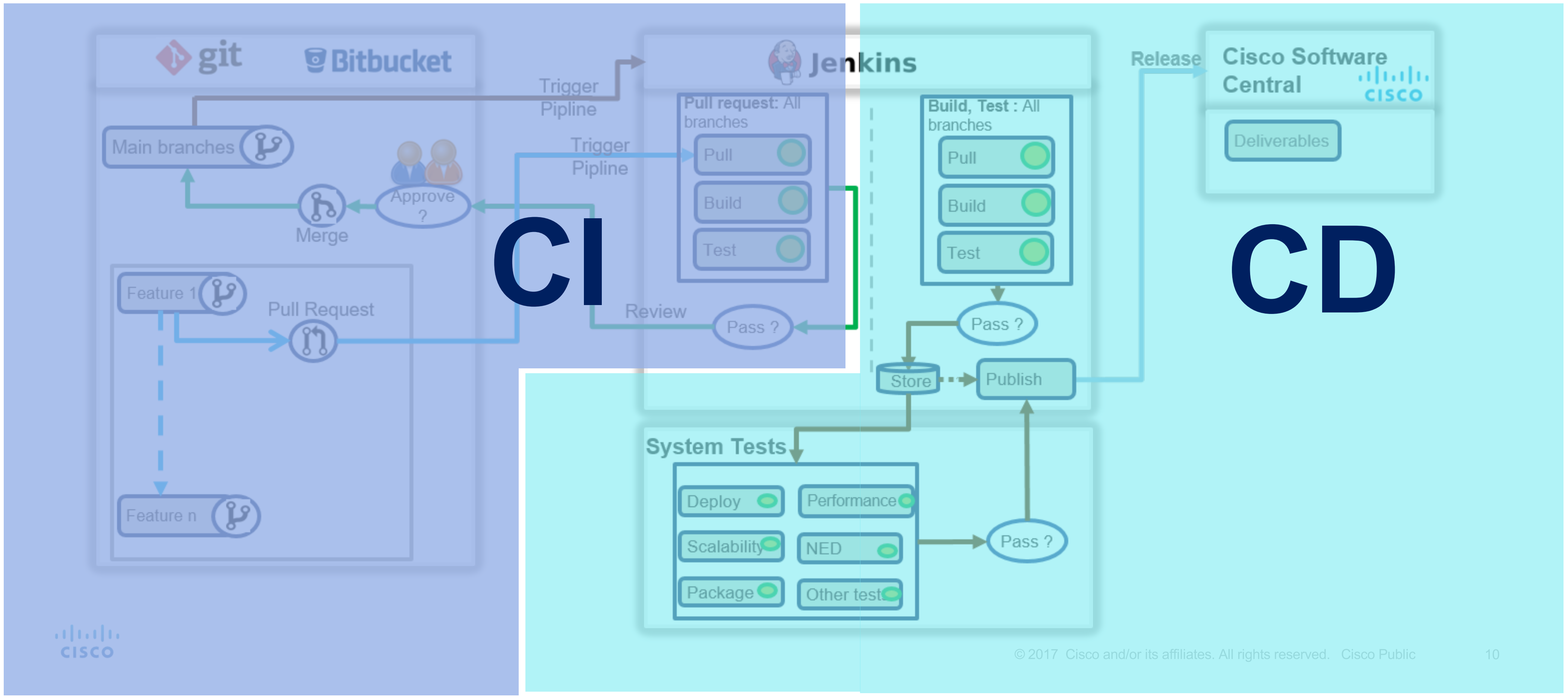
Execution Cycle of CI
(Developer and CI server steps)


CI
Release deployable artifacts with build label
Notifies results to the team
Monitors the Repo and Pull code
Build the system
Run Unit & Integration tests
Developer
Pull code from repo
Drive some code with tests
Ready to commit
Merge code and resolve conflicts locally
Run tests to ensure they pass locally
Commit to Repo
Build "GREEN"

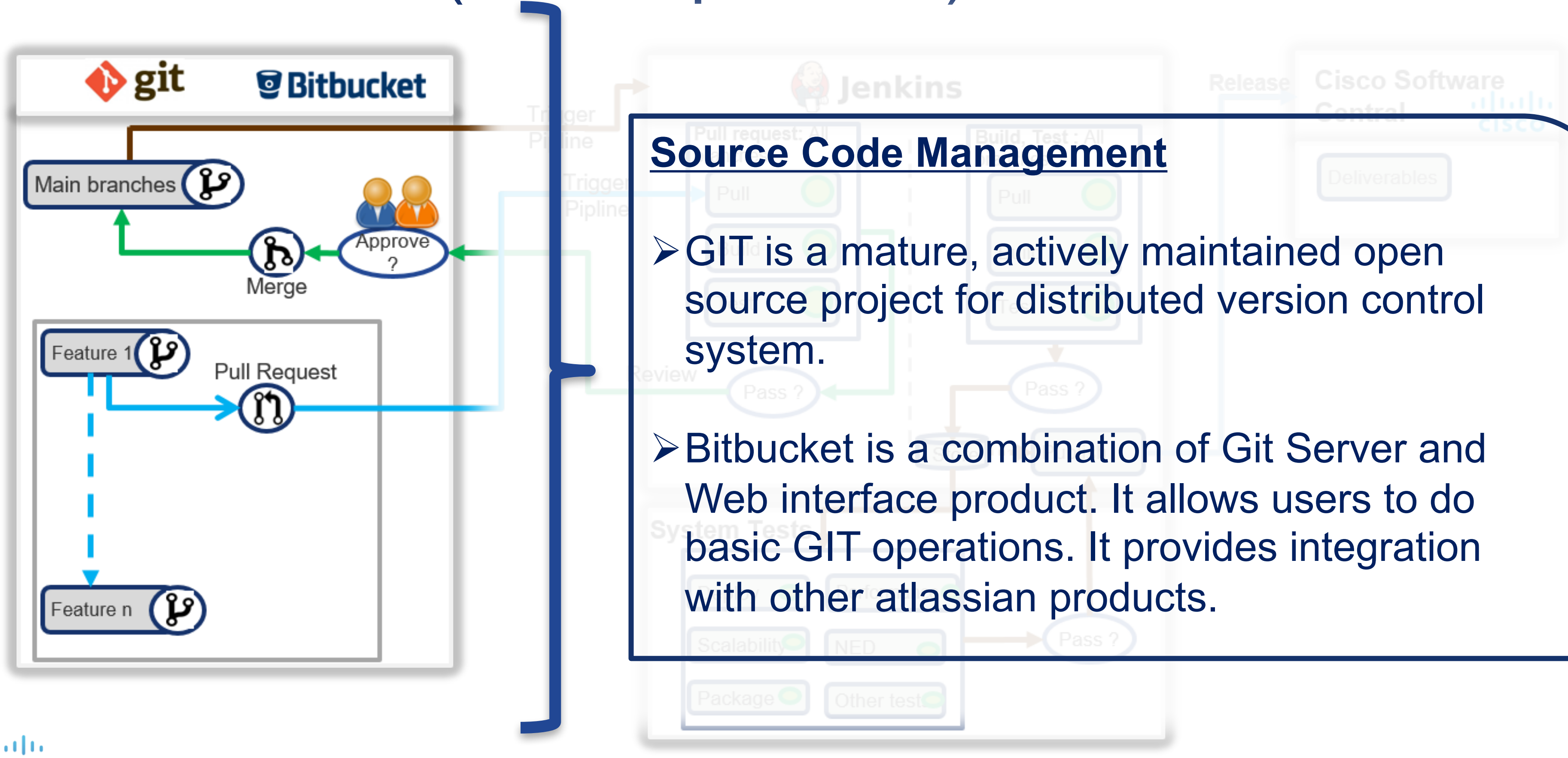
NSO Integration and Delivery pipeline



NSO Integration and Delivery pipeline



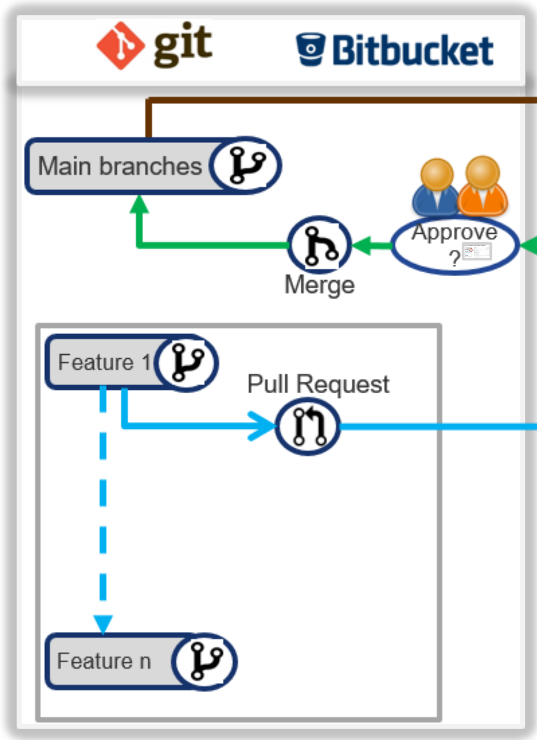
NSO Integration and Delivery pipeline - (Developer flow)



Source Code Management

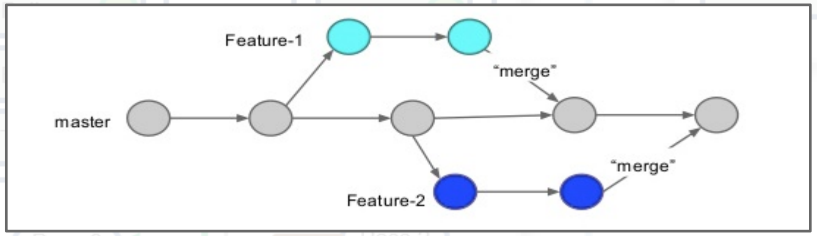
- GIT is a mature, actively maintained open source project for distributed version control system.
- Bitbucket is a combination of Git Server and Web interface product. It allows users to do basic GIT operations. It provides integration with other atlassian products.

NSO Integration and Delivery pipeline - (Developer flow)



Development Workflow

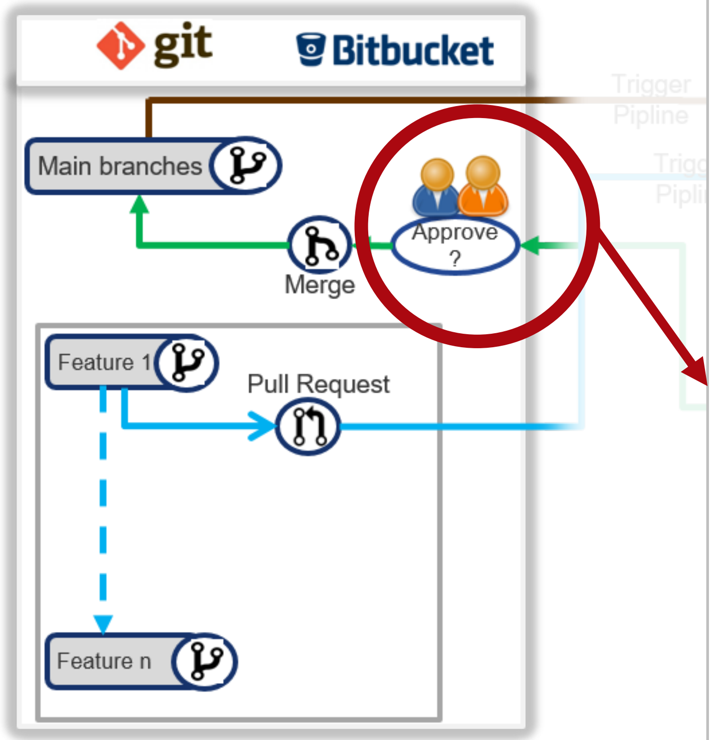
- One main branch "trunk" with a transient number of feature branches.

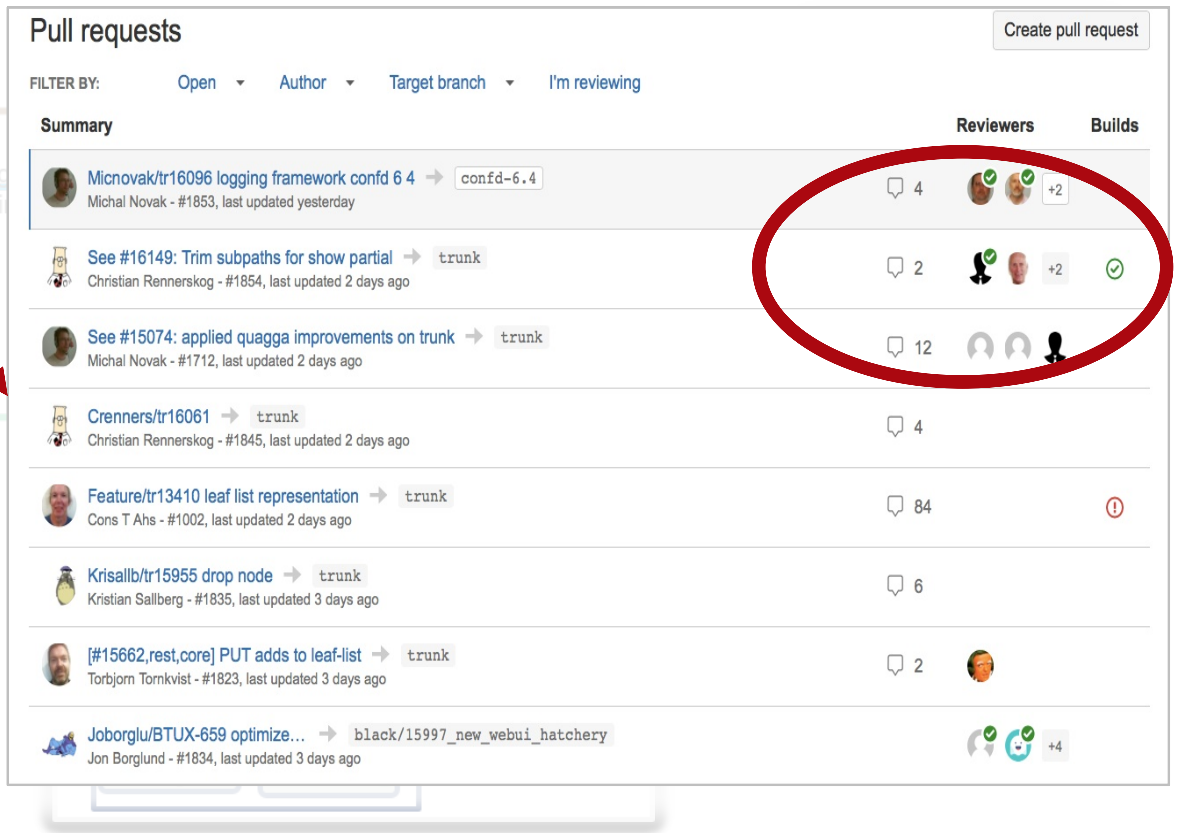


Practices

- Create a feature branch off of the main branch.
- Commit changes to the feature branch.
- Create a "pull request" (PR) targeting the main branch.
- Merge PR after it passes CI tests and team review.
- Delete feature branch after integration is complete.

NSO Integration and Delivery pipeline - (Developer flow)

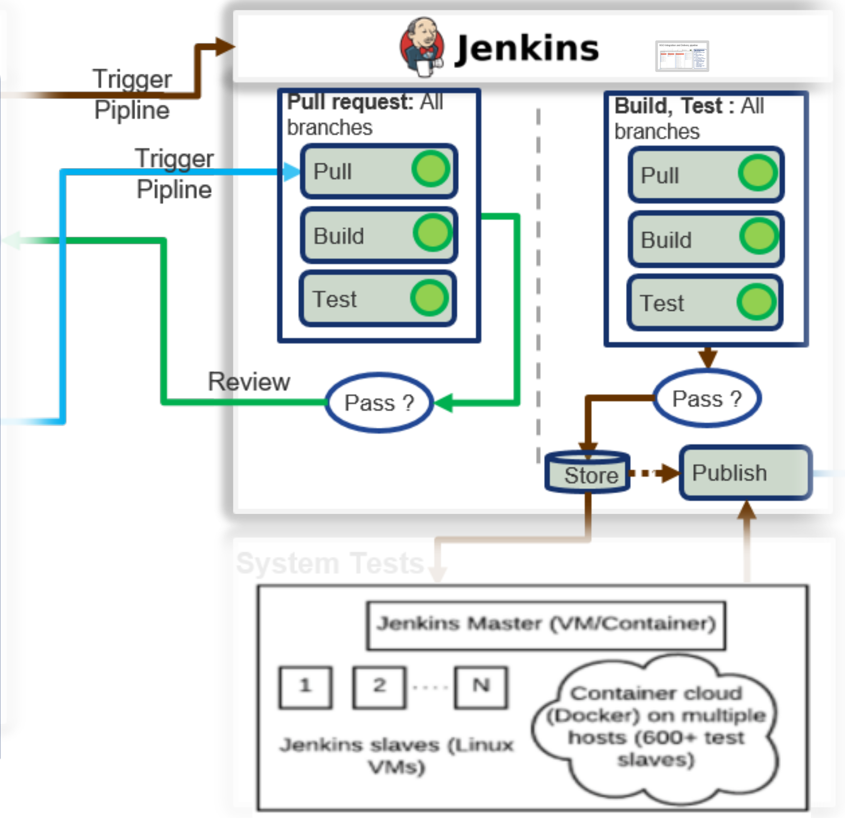




Summary		Reviewers	Builds
Micnovak/tr16096 logging framework confd 6 4 → confd-6.4 Michal Novak - #1853, last updated yesterday	4	+2	
See #16149: Trim subpaths for show partial → trunk Christian Rennerskog - #1854, last updated 2 days ago	2	+2	✓
See #15074: applied quagga improvements on trunk → trunk Michal Novak - #1712, last updated 2 days ago	12		
Crenners/tr16061 → trunk Christian Rennerskog - #1845, last updated 2 days ago	4		
Feature/tr13410 leaf list representation → trunk Cons T Ahs - #1002, last updated 2 days ago	84		!
Krisallb/tr15955 drop node → trunk Kristian Sallberg - #1835, last updated 3 days ago	6		
[#15662,rest,core] PUT adds to leaf-list → trunk Torbjorn Tornkvist - #1823, last updated 3 days ago	2		
Joborglu/BTUX-659 optimize... → black/15997_new_webui_hatchery Jon Borglund - #1834, last updated 3 days ago		+4	

NSO Integration and Delivery pipeline (CI)

- Jenkins is an open source automation server. With Jenkins, organizations can accelerate the software development process through automation.
- Jenkins manages and controls development lifecycle processes of all kinds, including build, document, test, package, stage, deployment, static analysis and many more.



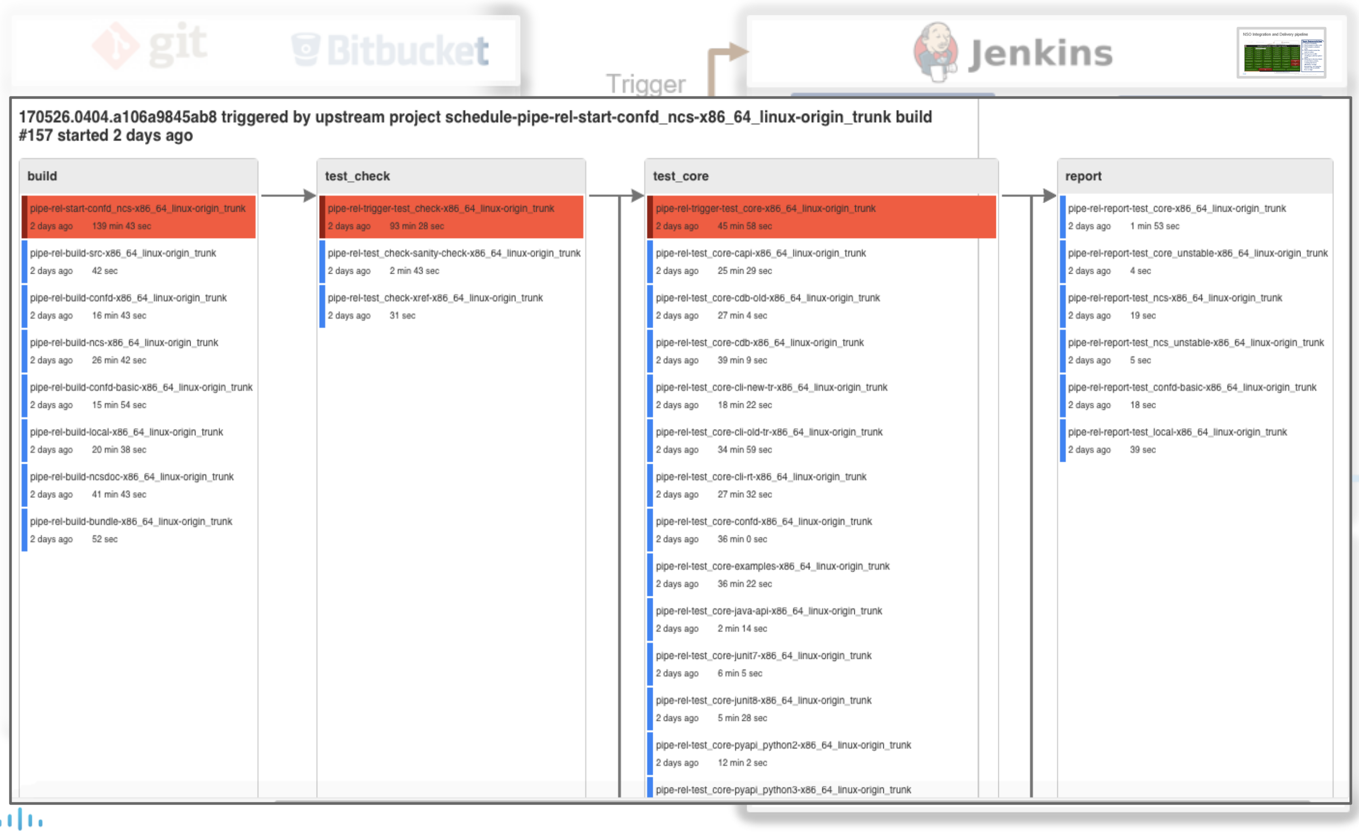
Trigger Pipline
Trigger Pipline
Review
Jenkins
Pull request: All branches
Pull
Build
Test
Pass ?
Build, Test : All branches
Pull
Build
Test
Pass ?
Store
Publish
System Tests
Jenkins Master (VM/Container)
1
2
N
Jenkins slaves (Linux VMs)
Container cloud (Docker) on multiple hosts (600+ test slaves)

Test Infrastructure

Builds and Tests

- Parallel execution of test jobs.
- Runs on Docker containers and macos servers.
- Git read-only caching server (Gitolite) used for the test machines.
- Nexus Binary Repository dependencies

NSO Integration and Delivery pipeline (CI)



170526.0404.a106a9845ab8 triggered by upstream project schedule-pipe-rel-start-confd_ncs-x86_64_linux-origin_trunk build #157 started 2 days ago

build

pipe-rel-start-confd_ncs-x86_64_linux-origin_trunk
2 days ago 139 min 43 sec
pipe-rel-build-src-x86_64_linux-origin_trunk
2 days ago 42 sec
pipe-rel-build-confd-x86_64_linux-origin_trunk
2 days ago 16 min 43 sec
pipe-rel-build-ncs-x86_64_linux-origin_trunk
2 days ago 26 min 42 sec
pipe-rel-build-confd-basic-x86_64_linux-origin_trunk
2 days ago 15 min 54 sec
pipe-rel-build-local-x86_64_linux-origin_trunk
2 days ago 20 min 38 sec
pipe-rel-build-ncsdoc-x86_64_linux-origin_trunk
2 days ago 41 min 43 sec
pipe-rel-build-bundle-x86_64_linux-origin_trunk
2 days ago 52 sec

test_check

pipe-rel-trigger-test_check-x86_64_linux-origin_trunk
2 days ago 93 min 28 sec
pipe-rel-test_check-sanity-check-x86_64_linux-origin_trunk
2 days ago 2 min 43 sec
pipe-rel-test_check-xref-x86_64_linux-origin_trunk
2 days ago 31 sec

test_core

pipe-rel-trigger-test_core-x86_64_linux-origin_trunk
2 days ago 45 min 58 sec
pipe-rel-test_core-capi-x86_64_linux-origin_trunk
2 days ago 25 min 29 sec
pipe-rel-test_core-cdb-old-x86_64_linux-origin_trunk
2 days ago 27 min 4 sec
pipe-rel-test_core-cdb-x86_64_linux-origin_trunk
2 days ago 39 min 9 sec
pipe-rel-test_core-cli-new-tr-x86_64_linux-origin_trunk
2 days ago 18 min 22 sec
pipe-rel-test_core-cli-old-tr-x86_64_linux-origin_trunk
2 days ago 34 min 59 sec
pipe-rel-test_core-cli-rt-x86_64_linux-origin_trunk
2 days ago 27 min 32 sec
pipe-rel-test_core-confd-x86_64_linux-origin_trunk
2 days ago 36 min 0 sec
pipe-rel-test_core-examples-x86_64_linux-origin_trunk
2 days ago 36 min 22 sec
pipe-rel-test_core-java-api-x86_64_linux-origin_trunk
2 days ago 2 min 14 sec
pipe-rel-test_core-junit7-x86_64_linux-origin_trunk
2 days ago 6 min 5 sec
pipe-rel-test_core-junit8-x86_64_linux-origin_trunk
2 days ago 5 min 28 sec
pipe-rel-test_core-pyapi_python2-x86_64_linux-origin_trunk
2 days ago 12 min 2 sec
pipe-rel-test_core-pyapi_python3-x86_64_linux-origin_trunk

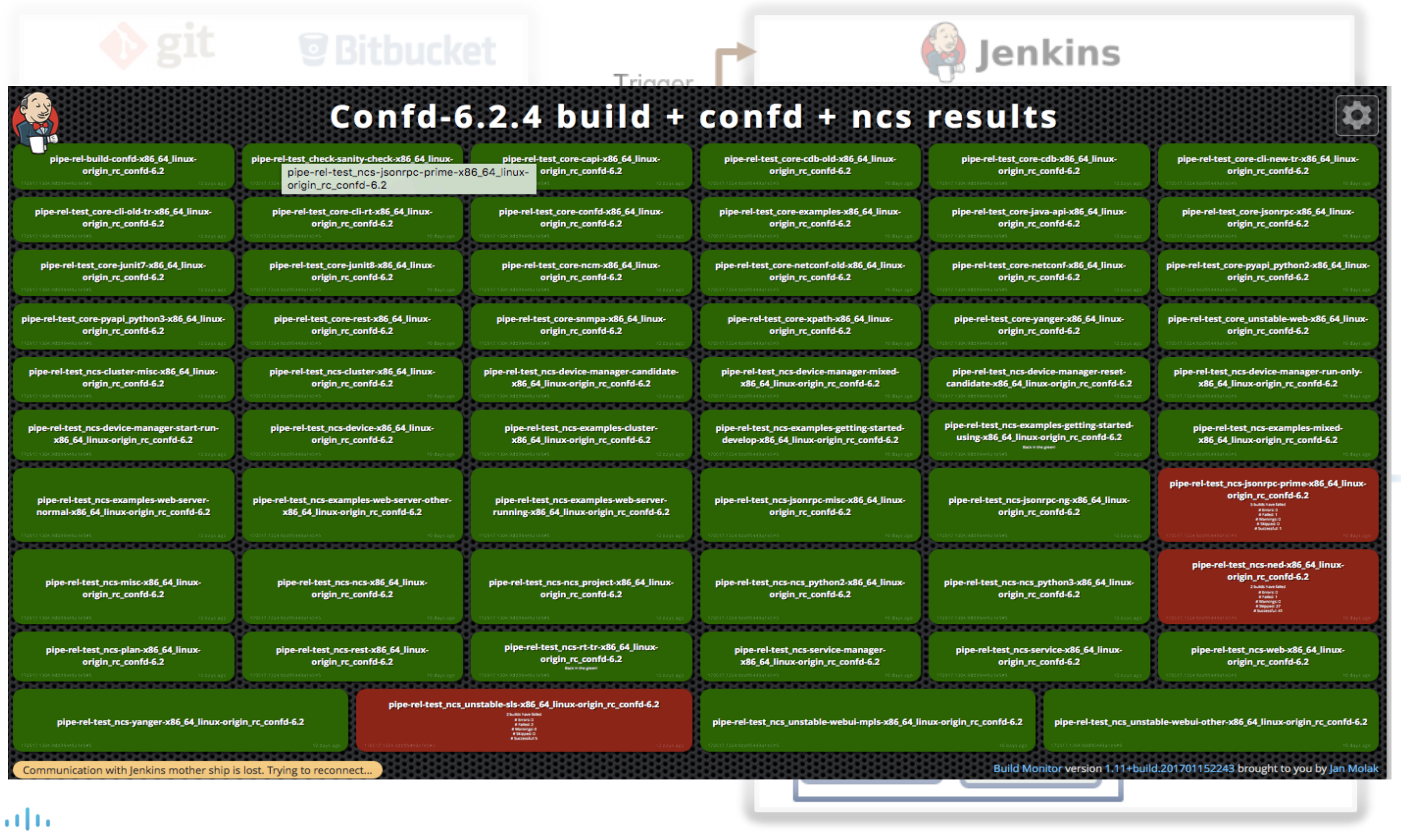
report

pipe-rel-report-test_core-x86_64_linux-origin_trunk
2 days ago 1 min 53 sec
pipe-rel-report-test_core_unstable-x86_64_linux-origin_trunk
2 days ago 4 sec
pipe-rel-report-test_ncs-x86_64_linux-origin_trunk
2 days ago 19 sec
pipe-rel-report-test_ncs_unstable-x86_64_linux-origin_trunk
2 days ago 5 sec
pipe-rel-report-test_confd-basic-x86_64_linux-origin_trunk
2 days ago 18 sec
pipe-rel-report-test_local-x86_64_linux-origin_trunk
2 days ago 39 sec

Artifacts

- ~2500 test cases are run on ~120 jenkins jobs.
- **If not paralallelized, execution of the full pipeline takes ~48 hours.**
- Test execution time for full test pipeline is **~2 hours.**

NSO Integration and Delivery pipeline - (Dashboards)



Confd-6.2.4 build + confd + ncs results

pipe-rel-build-confd-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_check-sanity-check-x86_64_linux-
pipe-rel-test_ncs-jsonrpc-prime-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-capi-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-cdb-old-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-cdb-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-cli-new-tr-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-cli-old-tr-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-cli-rt-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-confd-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-examples-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-java-api-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-jsonrpc-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-junit7-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-junit8-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-ncm-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-netconf-old-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-netconf-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-pyapi_python2-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-pyapi_python3-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-rest-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-snmpa-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-xpath-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core-yanger-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_core_unstable-web-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-cluster-misc-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-cluster-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-device-manager-candidate-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-device-manager-mixed-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-device-manager-reset-candidate-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-device-manager-run-only-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-device-manager-start-run-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-device-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-examples-cluster-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-examples-getting-started-develop-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-examples-getting-started-using-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-examples-mixed-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-examples-web-server-normal-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-examples-web-server-other-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-examples-web-server-running-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-jsonrpc-misc-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-jsonrpc-ng-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-jsonrpc-prime-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-misc-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-ncs-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-ncs_project-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-ncs_python2-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-ncs_python3-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-ned-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-plan-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-rest-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-rt-tr-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-service-manager-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-service-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-web-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs-yanger-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs_unstable-sls-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs_unstable-webui-mpls-x86_64_linux-origin_rc_confd-6.2
pipe-rel-test_ncs_unstable-webui-other-x86_64_linux-origin_rc_confd-6.2

Communication with Jenkins mother ship is lost. Trying to reconnect...

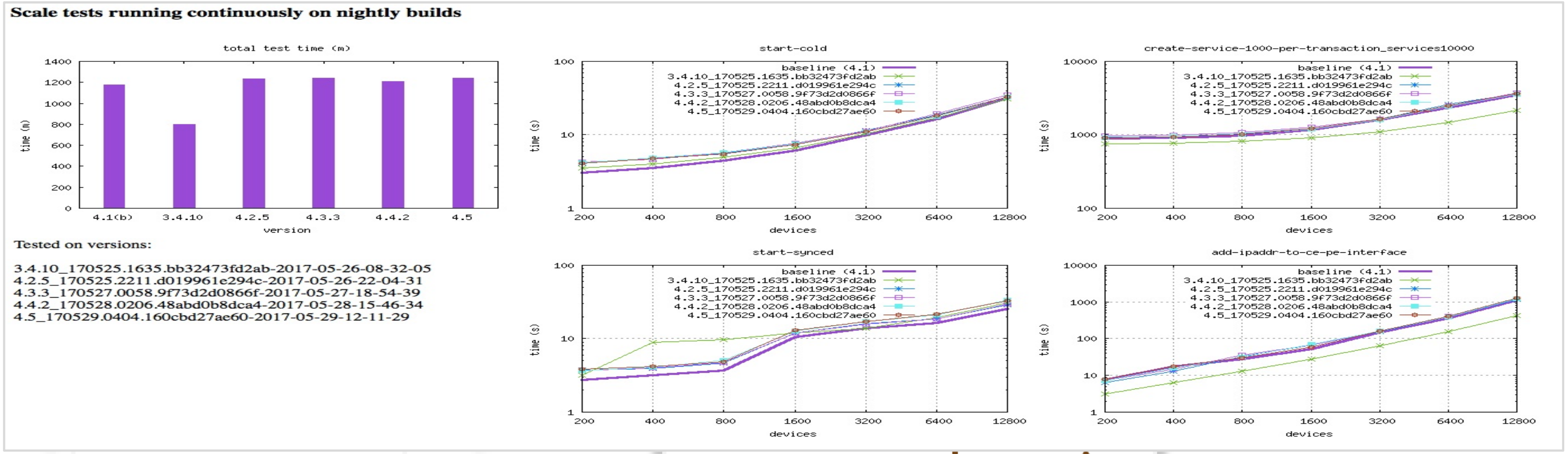
Build Monitor version 1.11+build.201701152243 brought to you by Jan Molak

Team Responsibilites

- ➢ Check in frequently
- ➢ Don’t check in broken code
- ➢ Don’t check in untested code
- ➢ Don’t check in when the build is broken
- ➢ Don’t go home after checking in until the system builds

Many teams develop rituals around these policies, meaning the teams effectively manage themselves.

NSO Integration and Delivery pipeline (CI/CD) - (Dashboards)

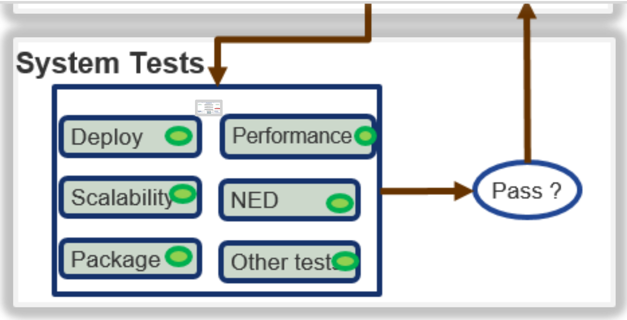


Scale tests running continuously on nightly builds

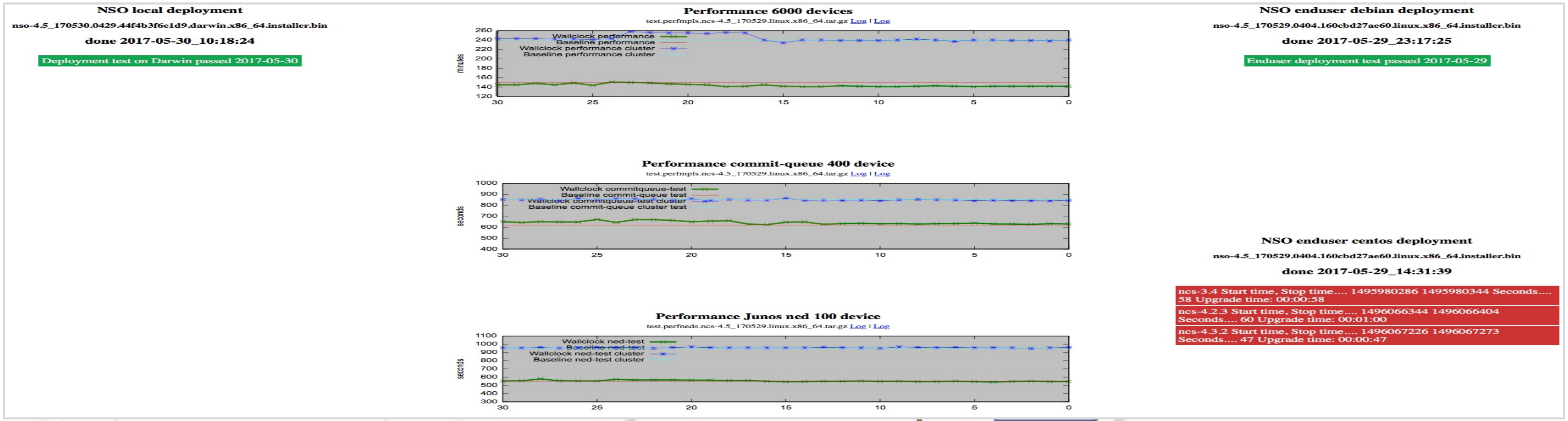
Tested on versions:

3.4.10_170525.1635.bb32473fd2ab-2017-05-26-08-32-05
4.2.5_170525.2211.d019961e294c-2017-05-26-22-04-31
4.3.3_170527.0058.9f73d2d0866f-2017-05-27-18-54-39
4.4.2_170528.0206.48abd0b8dca4-2017-05-28-15-46-34
4.5_170529.0404.160cbd27ae60-2017-05-29-12-11-29

Feedback Mechanism:
- ✓ Results displayed on a dashboard for better visibility.
- ✓ Performance degradation is clearly visible.



NSO Integration and Delivery pipeline (CI/CD) - (Dashboards)



NSO local deployment

nso-4.5_170530.0429.44f4b3f6e1d9.darwin.x86_64.installer.bin

done 2017-05-30_10:18:24

Deployment test on Darwin passed 2017-05-30

Performance 6000 devices

test.perfmpls.ncs-4.5_170529.linux.x86_64.tar.gz Log | Log

minutes
Wallclock performance
Baseline performance
Wallclock performance cluster
Baseline performance cluster

Performance commit-queue 400 device

test.perfmpls.ncs-4.5_170529.linux.x86_64.tar.gz Log | Log

seconds
Wallclock commitqueue-test
Baseline commit-queue test
Wallclock commitqueue-test cluster
Baseline commit-queue cluster test

Performance Junos ned 100 device

test.perfneds.ncs-4.5_170529.linux.x86_64.tar.gz Log | Log

NSO enduser debian deployment

nso-4.5_170529.0404.160cbd27ae60.linux.x86_64.installer.bin

done 2017-05-29_23:17:25

Enduser deployment test passed 2017-05-29

NSO enduser centos deployment

nso-4.5_170529.0404.160cbd27ae60.linux.x86_64.installer.bin

done 2017-05-29_14:31:39

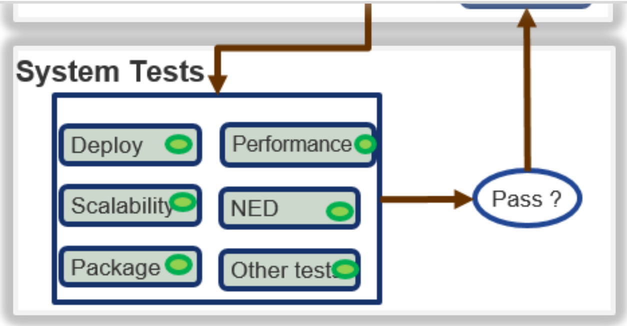
ncs-3.4 Start time, Stop time.... 1495980286 1495980344 Seconds.... 58 Upgrade time: 00:00:58

ncs-4.2.3 Start time, Stop time.... 1496066344 1496066404 Seconds.... 60 Upgrade time: 00:01:00

ncs-4.3.2 Start time, Stop time.... 1496067226 1496067273 Seconds.... 47 Upgrade time: 00:00:47

Performance Test

- Executed nightly.
- Performance degradation is visible.



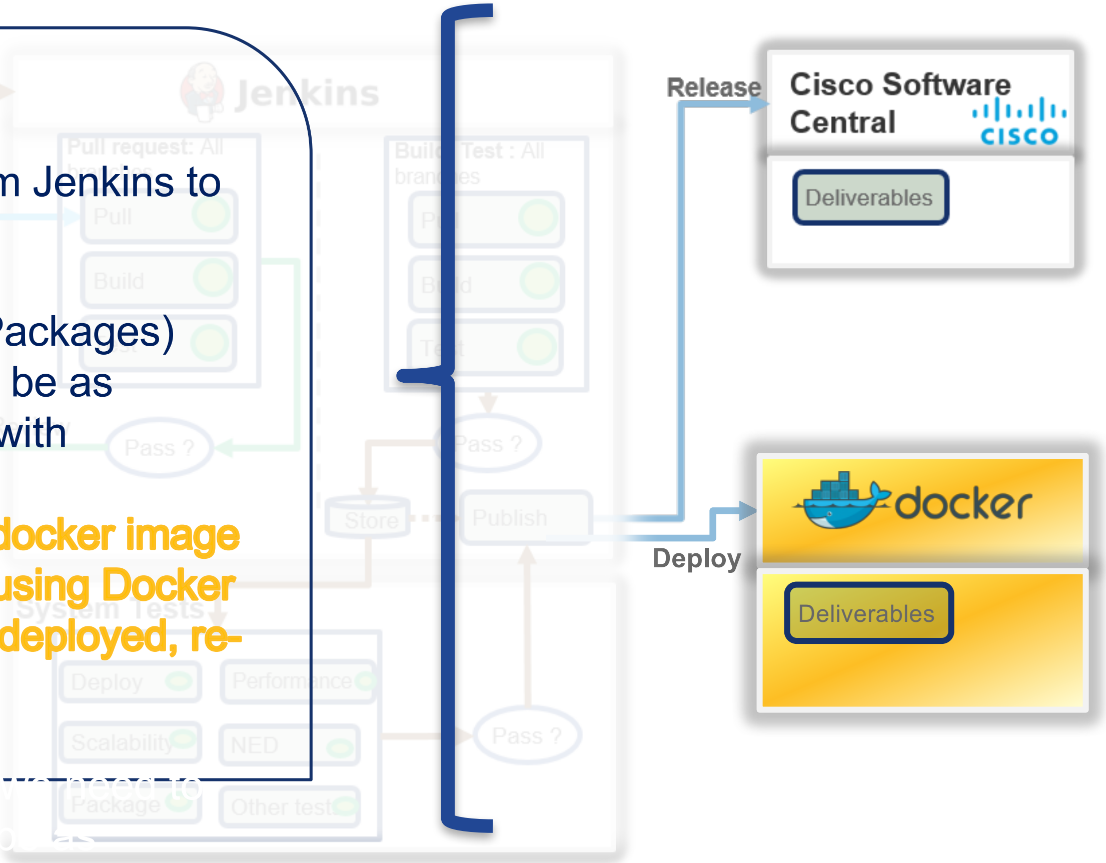
Deployment Tests

- ➢ Executed nightly.
- ➢ Failed upgrade versions visible on the dashboards.

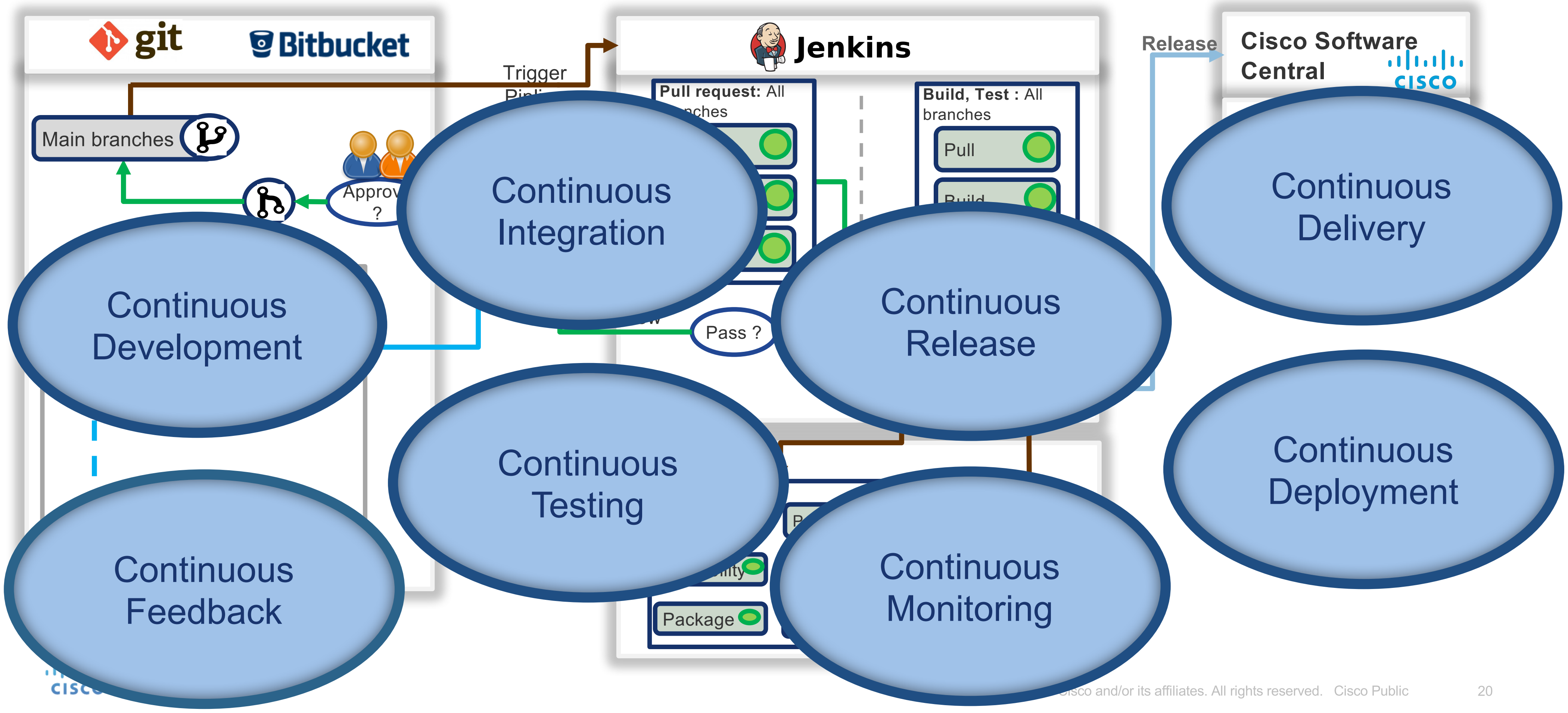
NSO Integration and Delivery pipeline (CD)

Delivery Management

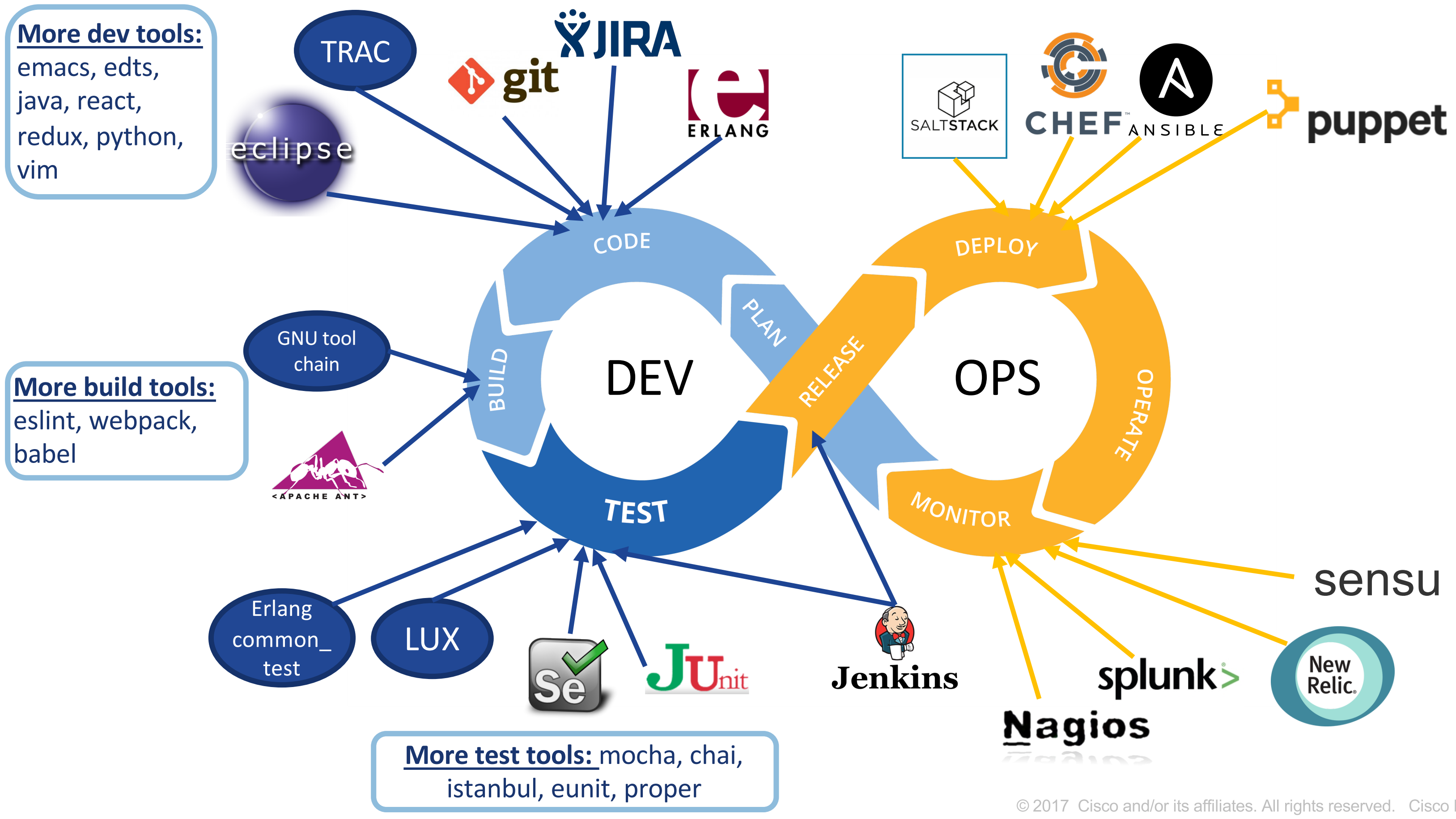
- ➢ Automated upload of deliverables from Jenkins to Cisco Software Central.
- ➢ Custom script using Python
- ➢ ~300 deliverables (NSO, NEDs and Packages) published every month. This needs to be as automated as possible not to burden with unnecessary workload.
- ➢ **Jenkins can push working images to docker image registry. And can trigger Deployment using Docker Cloud. Containers can be monitored, deployed, re-deployed, recovered by Docker.**



NSO Integration and Delivery pipeline



DevOps - CI, CD tool kit



More dev tools: emacs, edts, java, react, redux, python, vim

More build tools: eslint, webpack, babel

More test tools: mocha, chai, istanbul, eunit, proper

Practices

- ✓ Automate everything: build, test and deployment.
- ✓ Keep absolutely everything in the source code management system.
- ✓ Commit your code to the repository frequently.
- ✓ Don’t commit directly to the delivery branch; use a feature branch and PR workflow.
- ✓ Use a CI tool that integrates tightly with your source code repository.
- ✓ Have small steps (test suites), with clear error messages.
- ✓ Don’t ignore failing test cases, even on the feature branches.
- ✓ Automated feedback on the entire process.

Challenges

- ➢ Initial setup (Development / Build / QA / Integration environments)
- ➢ Setting up a CI server requires that the build, unit test, and executable packaging processes all be automated.
- ➢ Extra cost: hardware & software
- ➢ Continuous maintenance.
- ➢ A number of new tools and processes must be mastered.
- ➢ Requires mastering a build scripting language, a unit testing platform, and potentially a setup/install platform as well.
- ➢ Coaching developers on value of testing.
- ➢ Establishing a solid CI practice takes a lot of work and technical knowledge.

cisco

Conclusion

- Continuous Integration is relatively easy.
  - It is all about communication
- Continuous Delivery is challenging.
  - Some things are hard to test automatically
  - You need dedicated test-writing people / mindset.
- Continuous Deployment might not be a best fit for mission critical applications.

References

- Git https://git-scm.com
- Bitbucket https://www.atlassian.com/software/bitbucket/server
- Jenkins https://jenkins.io
- DevOps https://devops.com
- ThoughtWorks https://www.thoughtworks.com/continuous-integration

Thank You

Srilakshmi Kanda
Test Engineer

Q&A

Backup Slides

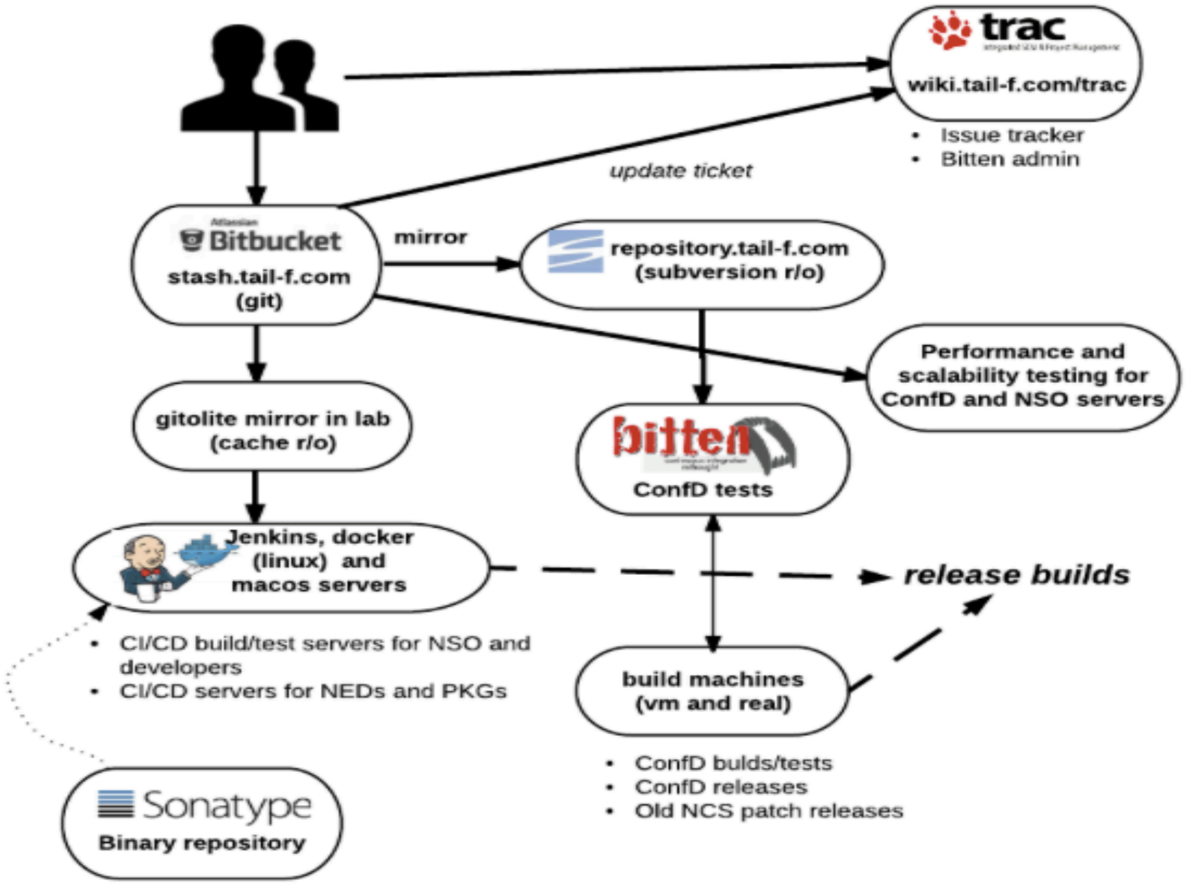
Continuous Release and Continuous Monitoring

- **Continuous Release**
  - Businesses require well-defined release planning and management processes that drive release roadmaps, project plans and delivery schedules as well as end-to-end traceability across those processes.
  - Challenges – Release Management, Release Coordination, Release Automation.
- **Continuous Monitoring and Feedback**
  - Customer Feedback comes in different forms, such as tickets opened by customers, formal change requests, informal complaints etc. Feedback also comes from monitoring data. This data comes from the servers running the applications from development, QA and Production, or from metric tools.
  - Challenges – Continuous Test Policing

Continuous Integration <> Continuous Delivery

- CD = CI + fully automated test suite
- Not every change is a release
  - Manual trigger
  - Trigger on a key file (version)
  - Tag releases !
- CD – It is all about testing !
  - Challenges – Release Management, Release Coordination, Release Automation.

Overview of the CI environment on Tail-f:



trac
wiki.tail-f.com/trac
update ticket
Bitbucket
stash.tail-f.com (git)
mirror
repository.tail-f.com (subversion r/o)
Performance and scalability testing for ConfD and NSO servers
gitolite mirror in lab (cache r/o)
bitten
ConfD tests
Jenkins, docker (linux) and macos servers
release builds
build machines (vm and real)
Sonatype
Binary repository

- Issue tracker
- Bitten admin

- CI/CD build/test servers for NSO and developers
- CI/CD servers for NEDs and PKGs

- ConfD bulds/tests
- ConfD releases
- Old NCS patch releases

- Webui NodeJS/NPM proxies
- Webui development
- Docker proxies
- Internal docker images.
- Cisco SSL
- Internal images