



Deploy anywhere: orchestrating the DevOps toolchain with Jenkins Pipeline

Lin Li (Lily)
2017.11.11

◆ AGENDA



- Agile and DevOps Transformation
- DevOps Toolchain
- Implementation CI/CD with Jenkins Pipeline
- Q&A



Who are we: since 1939

Build, Operate, & Secure Enterprise Software



We build enterprise-grade scalable software with analytics built in



Hybrid IT

You need to bridge the gap between legacy infrastructure and the digital enterprise. Micro Focus solutions are easy to consume and deploy in any environment, reducing IT costs and time to value. We harden the latest technologies to make them work for you.



DevOps

As a modern DevOps enterprise, you need to accelerate time to market and increase quality. Micro Focus provides an easy-to-use, seamless tool set that scales across the SDLC. From on-premises to cloud and from mainframe to mobile, we address all your DevOps needs.



Security & Risk

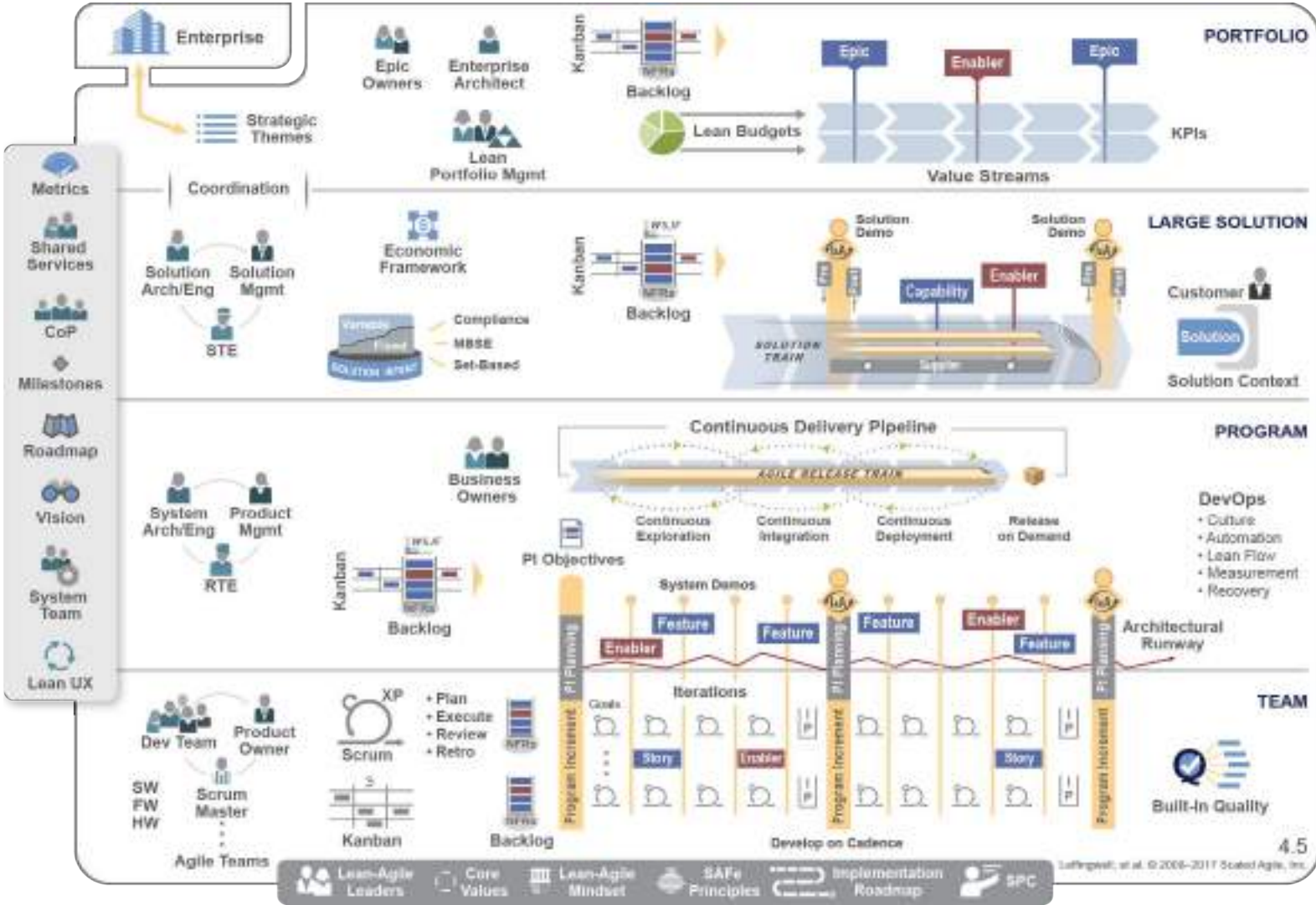
From compliance issues to the most advanced cyber threats, you need to safeguard your enterprise. Micro Focus protects what you value most: users, data, and applications.



Predictive Analytics


Everyone has data, everyone has analytics, but that's not enough. With predictive and proactive analytics, Micro Focus helps you to not only deliver insights, but drive greater intelligence and productivity across your enterprise.

DevOps transformation alignment with Scaled Agile Framework




DevOps transformation: impacts the R&D & DevOps practices and operation

Suite Strategy




Agile Practices



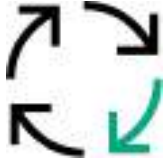
**Containers
Microservices**



**Cross Suite
Execution Dependencies**



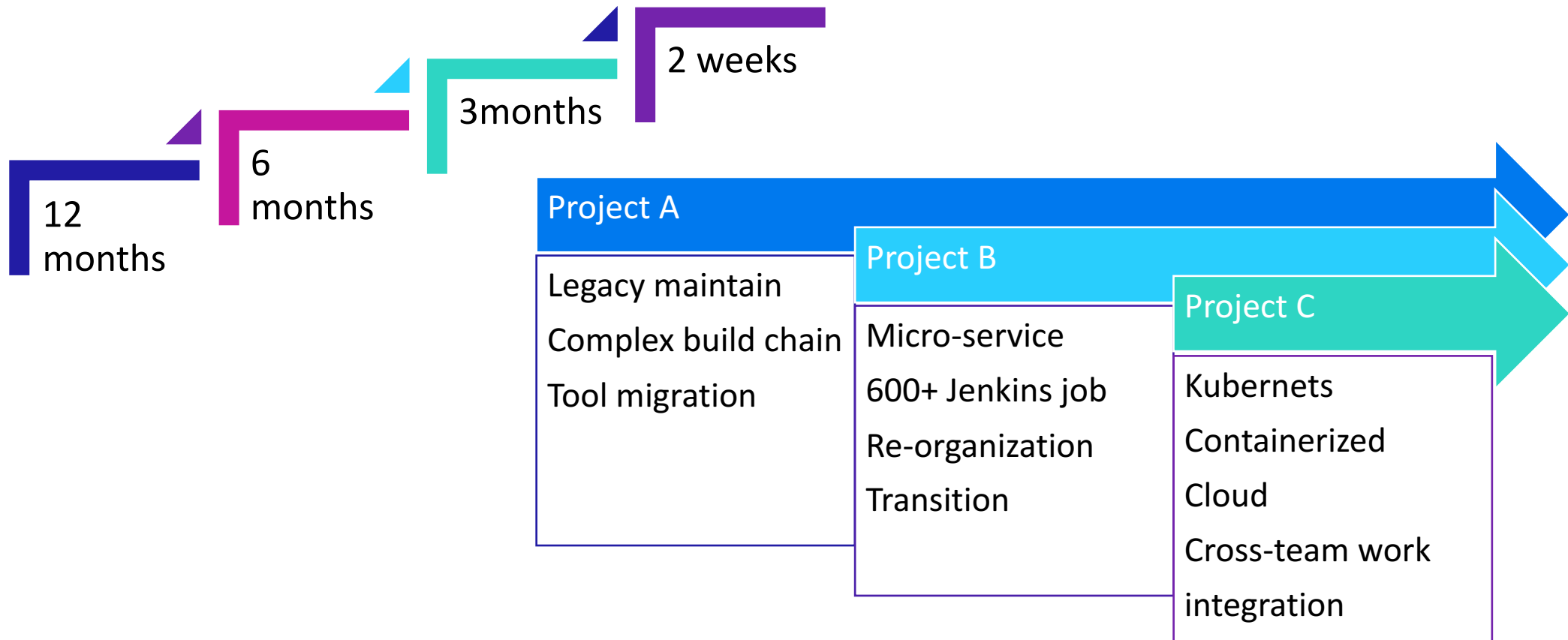
**Continuous Integration
Delivery & Demo**



**Standards for Security,
Quality & Compliance**



A glance of Projects



Product Suite

Imperial Crew

180+ engineers
21 Scrum teams
3 ARTs



DARTH VADER
GRAND MOFF TARKIN
IMPERIAL OFFICER

MSE-6 DROID



STORMTROOPER
IMPERIAL GUARD
R2-Q5

Forward Systems

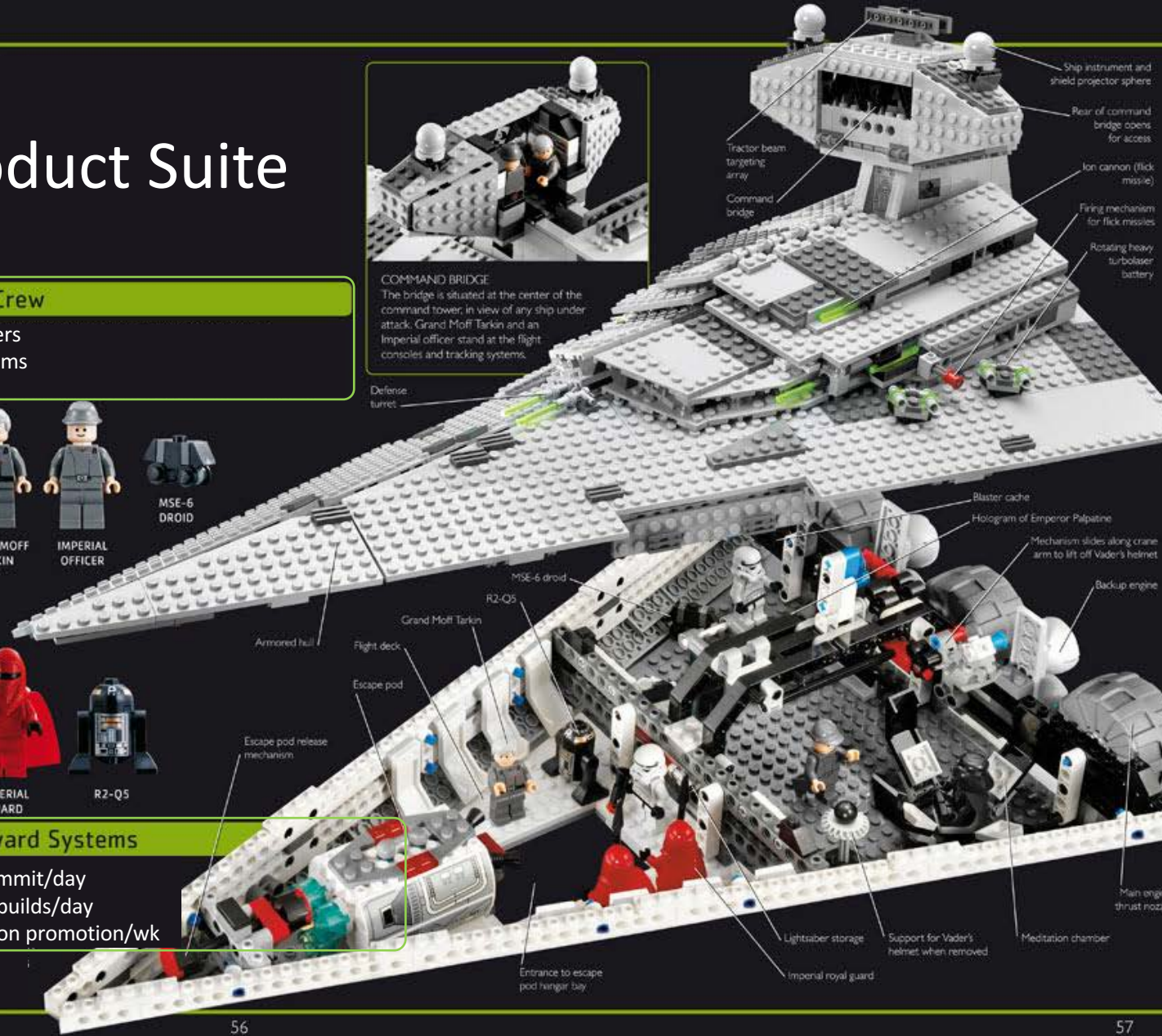
150+ commit/day
150+ CI builds/day
10 Version promotion/wk



COMMAND BRIDGE

The bridge is situated at the center of the command tower, in view of any ship under attack. Grand Moff Tarkin and an Imperial officer stand at the flight consoles and tracking systems.

Defense turret



Tractor beam targeting array
Command bridge

Ship instrument and shield projector sphere

Rear of command bridge opens for access

Ion cannon (flick missile)

Firing mechanism for flick missiles

Rotating heavy turbolaser battery



Chamber

6 products/20+ services
120+ Git repos
140 Docker image
145 running pods



Distress beacon

rocket thruster

Repulsor soft-landing coils

Fuel cell

Escape Pod

Service: 1448 UI test, 2000 API,
26788 UT/build
Suite: 442 suite case/build

Data File

Release: 2017.10
Feature scope: 6899
Capacity: 5815
No. of features: 217
Complexity level: 5
Challenging level: 5

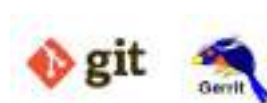
◆ AGENDA

“

- Agile and DevOps Transformation
- DevOps Toolchain
- Implementation CI/CD with Jenkins Pipeline
- Q&A

”

Orchestration of the DevOps toolchain



Choose appropriate tools for your program or organization

- Goals and feature matrix
- Integration with other tools
- Extensibility and flexibility
- User adoption
- Learning, implementation and maintaining
- Support and community activity
- scope
- Pricing
- POC, demo & trial farm

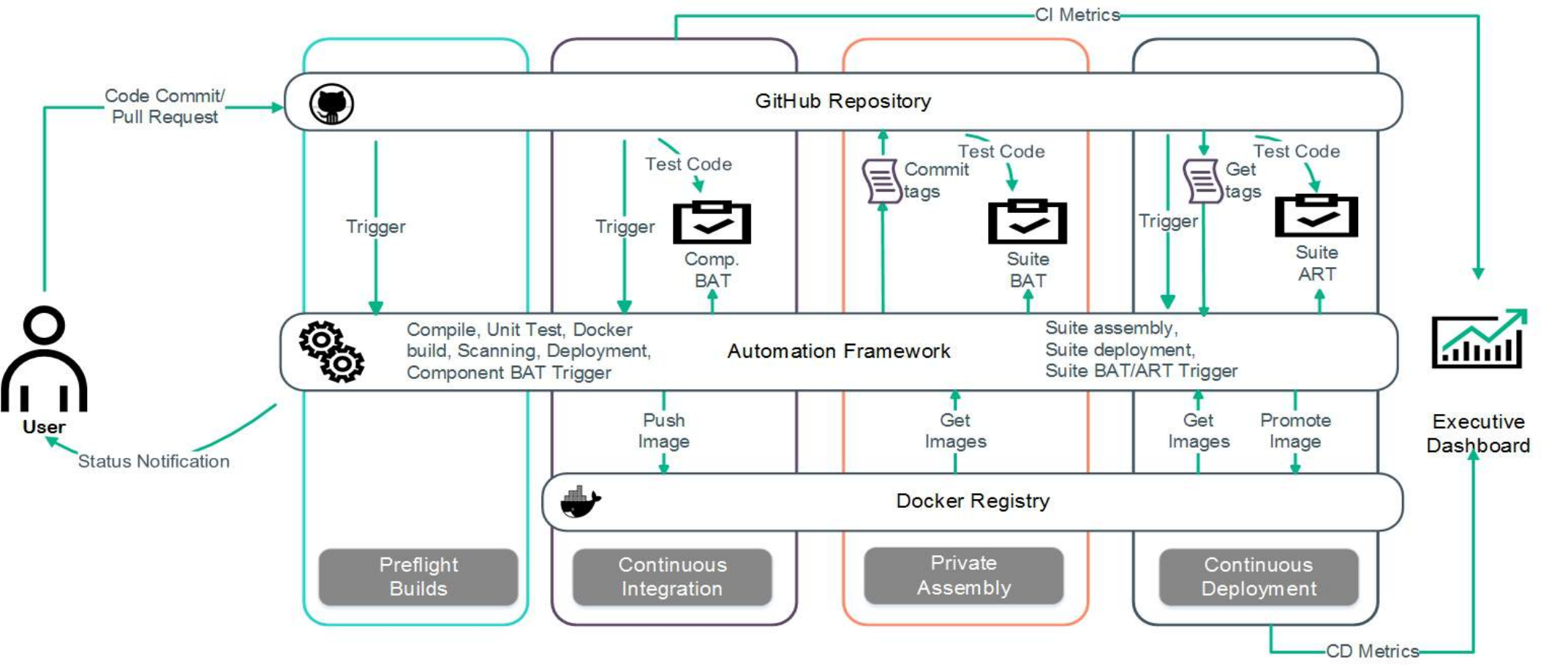
◆ AGENDA



- Agile and DevOps Transformation
- DevOps Toolchain
- Implementation CI/CD with Jenkins Pipeline
- Q&A



CI/CD process overview

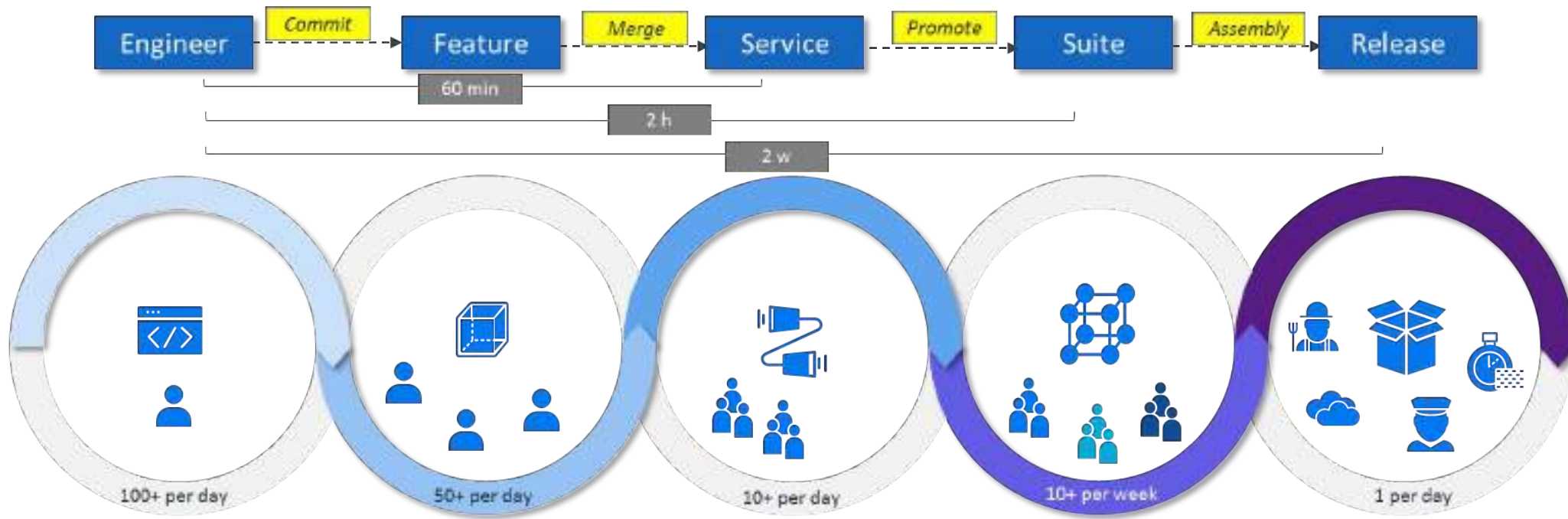


Innovations of CI/CD



- Service definition: architecture design and process
- A good architecture impacts process and DevOps activities

• Multiple leave CI/CD pipelines and assembly



- Service integration contract test
- Promotion and revert
- Version mapping manifest cross leaves

Operation as a service

- automation testing
- monitoring
- easily start up as a service in Kubernetes cluster
- fork shared service cross-team and contribute back to central repository

Pipeline as code: define process and rules in code



- Pipeline as code: define process and rules in code
 - Each component can build, test, deploy,
 - split source code to separate Git repository which has its own Jenkinsfile in root directory

Reusable Pipeline

- load from local file

```
stage('stage #1: update suitekit'){
  steps {
    script{
      function = load 'deploy/common.groovy'
      suitekitDir = "suitectl-${SUITEKIT_VERSION}-linux"
      function.updateSuitekit(SUITEKIT_VERSION, "${env.WORKSPACE}")
    }
  }
}
```

- Apply From : URL
- Shared Library functionality
 - @Library('somelib')
 - import com.mycorp.pipeline.somelib.UsefulClass

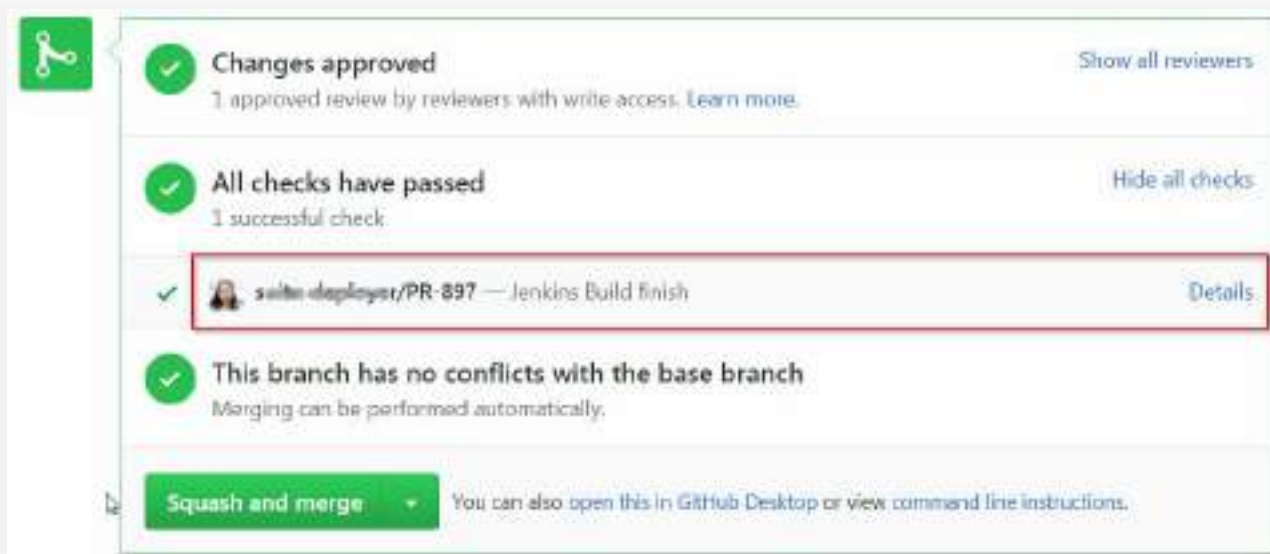
Unify build chain

- Use Maven to unify build chain for different technical stacks or compiling tools of each micro services: C++, Java, npm, Docker, yaml etc.
- Simplify the logic programming and configuration of Jenkins pipeline

Pre-flight build & Private CI/CD

- Pre-flight build: keep quality gate before code changes be merges into branches
- Private CI/CD: share computing and storage resource pool

Integration with GitHub



A screenshot of a GitHub pull request interface. It shows a list of status checks, all of which are green with checkmarks. The checks include: 'Changes approved' (1 approved review), 'All checks have passed' (1 successful check), 'Jenkins Build finish' (highlighted with a red box), and 'This branch has no conflicts with the base branch'. At the bottom, there is a green 'Squash and merge' button and a note about opening the PR in GitHub Desktop or viewing command line instructions.

Changes approved [Show all reviewers](#)
1 approved review by reviewers with write access. [Learn more.](#)

All checks have passed [Hide all checks](#)
1 successful check

✓ [s-wildt-deployer/PR-897](#) — Jenkins Build finish [Details](#)






This branch has no conflicts with the base branch
Merging can be performed automatically.

[Squash and merge](#) You can also open this in [GitHub Desktop](#) or view [command line instructions](#).

Commits on Oct 18, 2017



A screenshot of a GitHub commit history list for the date Oct 18, 2017. It displays three commit entries, each with a commit message, author, and commit hash. The first commit is 'promote from smartanalysis LULU-200711-ec-001 (#959)' by smaci, committed with jun-wan, and is marked as successful. The second is 'change backup info for service in H_Slave (#953)' by pei-hua, committed with jun-wan, and is marked as failed. The third is 'promote idm to 1.2.8 version (#956)' by zhang-jun-juan, committed with cxu, and is marked as successful.

 promote from smartanalysis LULU-200711-ec-001 (#959) smaci committed with jun-wan a day ago ✓	 9714a2f	
 change backup info for service in H_Slave (#953) pei-hua committed with jun-wan a day ago ✗	 9e24b0f	
 promote idm to 1.2.8 version (#956) zhang-jun-juan committed with cxu a day ago ✓	 2ff1313	

Integration with GitHub

The image displays two screenshots of the GitHub interface for the repository 'suite-configure'.

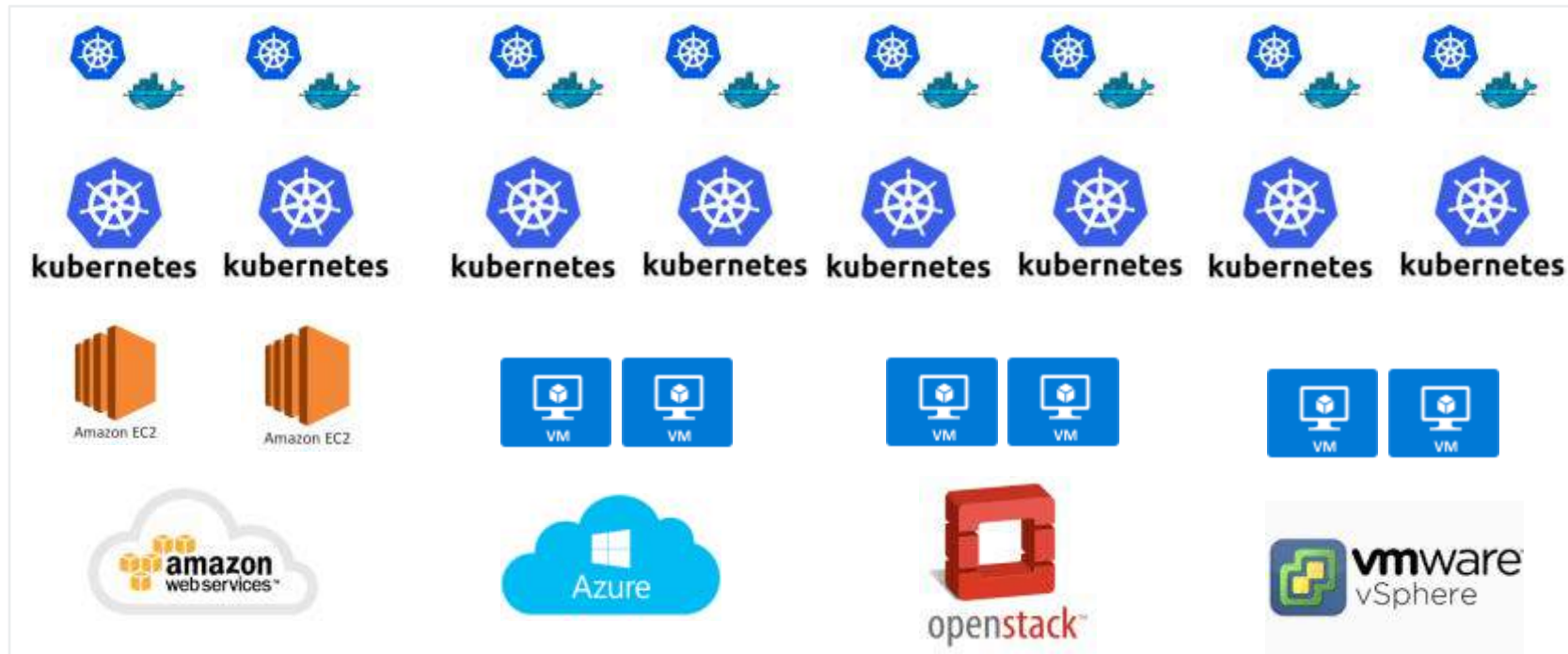
The left screenshot shows the 'Branches (6)' tab. It lists the following branches:

- 2017.07.HE
- master
- new_ui_design
- [blurred]
- [blurred]
- [blurred]

The right screenshot shows the 'Pull Requests (30)' tab, which is highlighted with a red box. It displays a table of pull requests:

S	W	Name ↑	Last Success	Last Failure
🟢	☀️	PR-577	3 hr 25 min - [blurred]	N/A
🟡	☀️	PR-576	6 hr 25 min - [blurred]	N/A
🟢	☀️	PR-574	12 hr - # 1-s - [blurred]	N/A
🟢	☁️	PR-573	1 day 2 hr - [blurred]	1 day 4 hr - [blurred]
🟡	☔️	PR-572	12 hr - # 3-s - [blurred]	12 hr - [blurred]
🟡	☀️	PR-571	1 day 6 hr - [blurred]	N/A
🟡	☀️	PR-570	1 day 4 hr - [blurred]	N/A
🟡	☁️	PR-569	1 day 8 hr - [blurred]	1 day [blurred]

Deploy anywhere

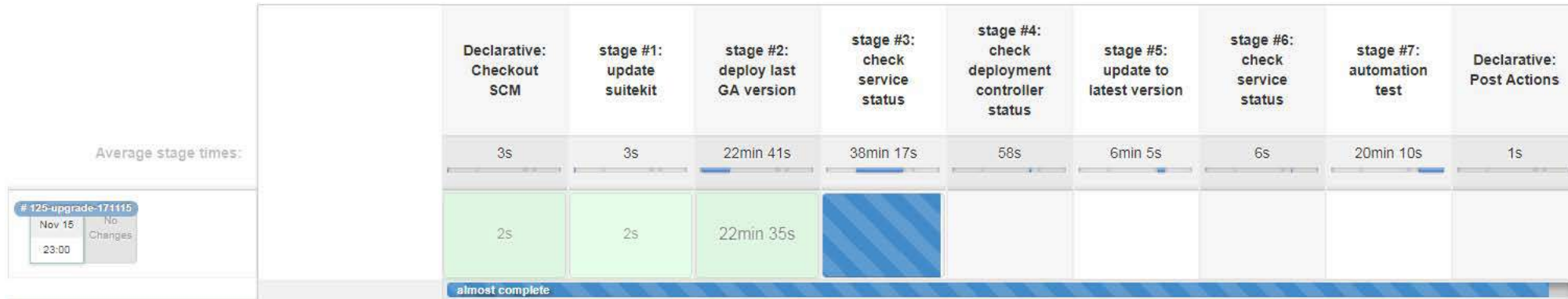


- Public, Private Cloud
- Lab, On-premises
- Dev, QA, staging, demo
- Small/medium/large profile, modes

Unify configuration, environment, and method in each different layers

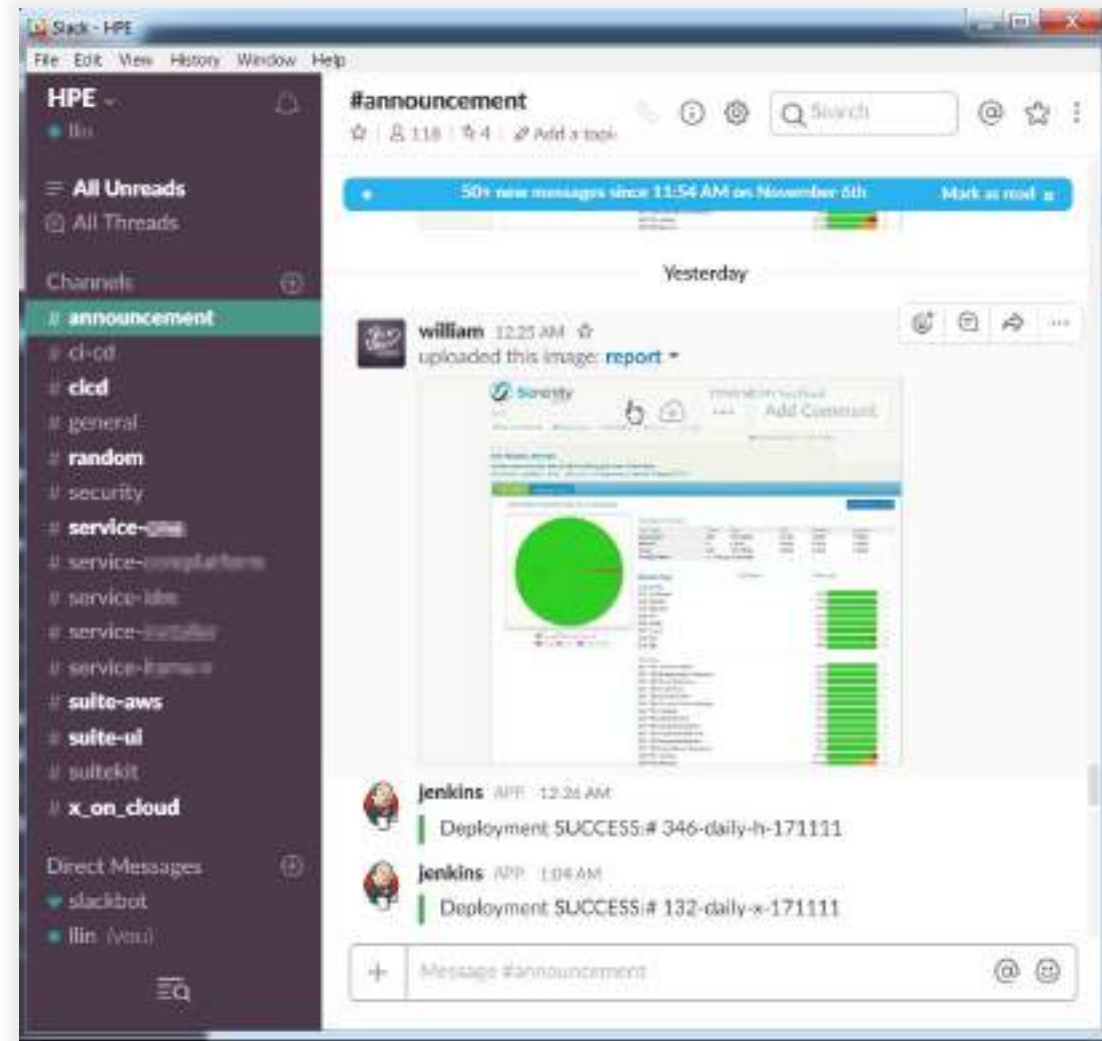
- Infrastructure provisioning: Terraform, Cloud Formation, Ansible
- Pre-check
- Kubernetes cluster: Container Delivery Foundation
- Installation: Deploy kit (Go + Ansible)

Orchestrate with Jenkins: reuse tools, steps and stages

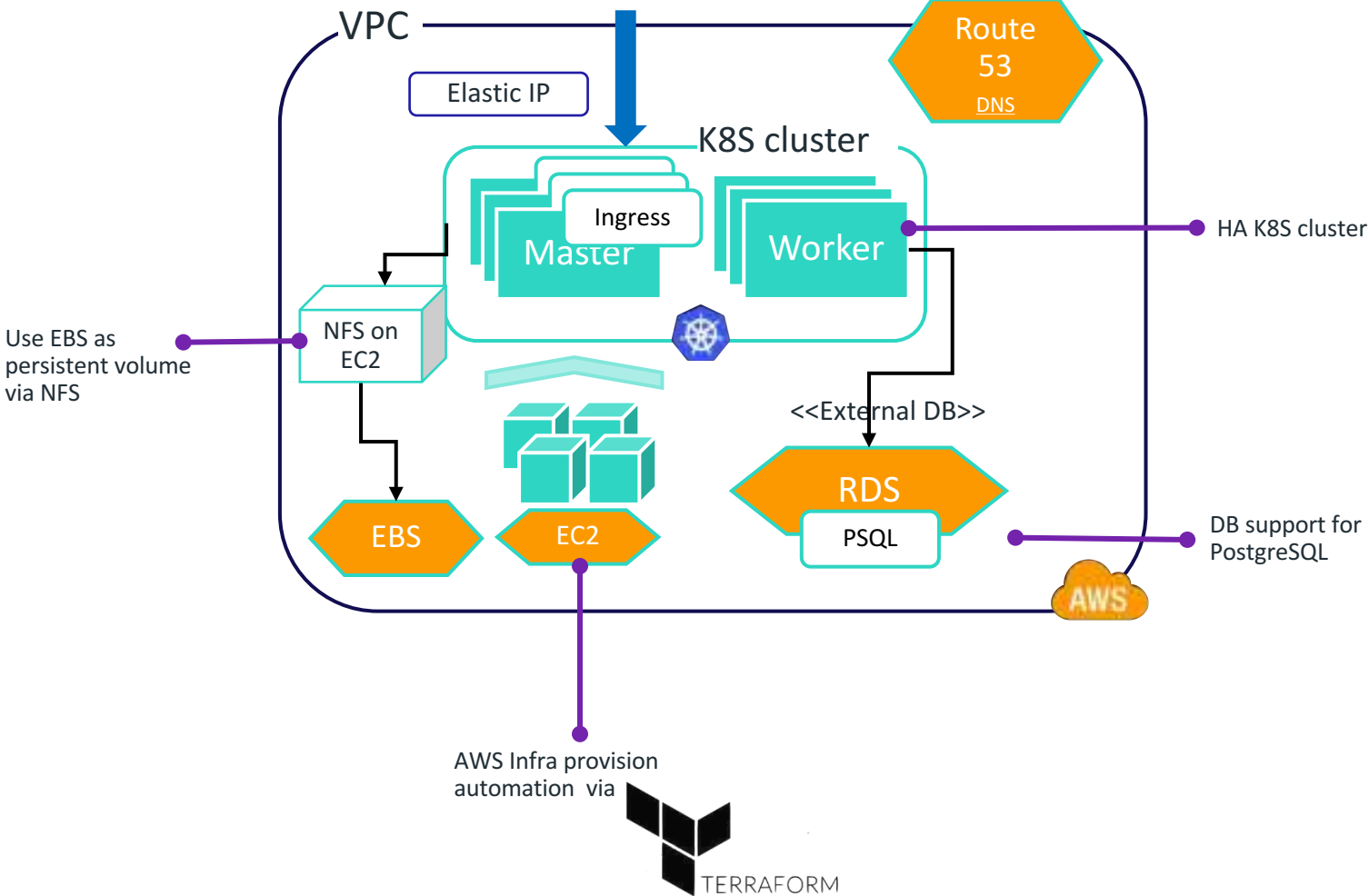


Integration with Slack

```
post {  
  success {  
    slackSend channel: '#cicd',  
              color: 'good',  
              message: "say something."  
  }  
}
```



Infra as code





Q&A
Thank You.