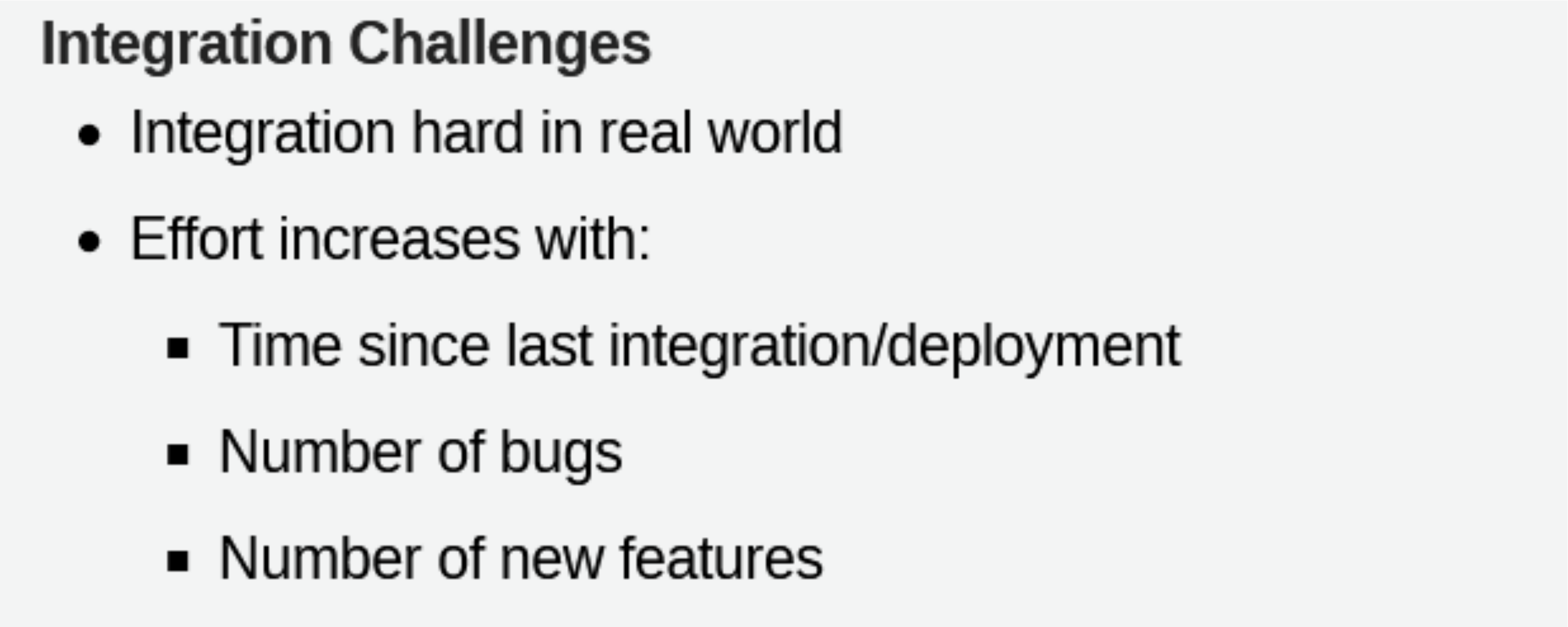
CP4A CI/CD Pipeline

Saif Ur Rehman

IBM Cloud

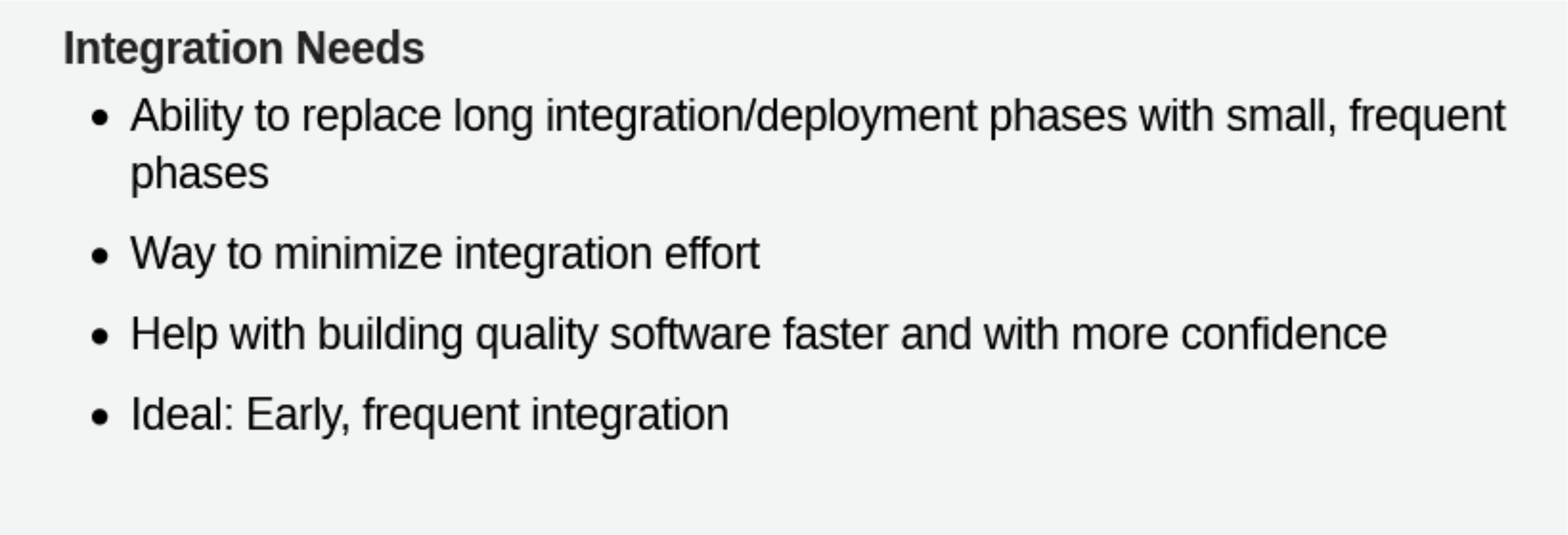
Integration



Integration Challenges

- Integration hard in real world
- Effort increases with:
  - Time since last integration/deployment
  - Number of bugs
  - Number of new features

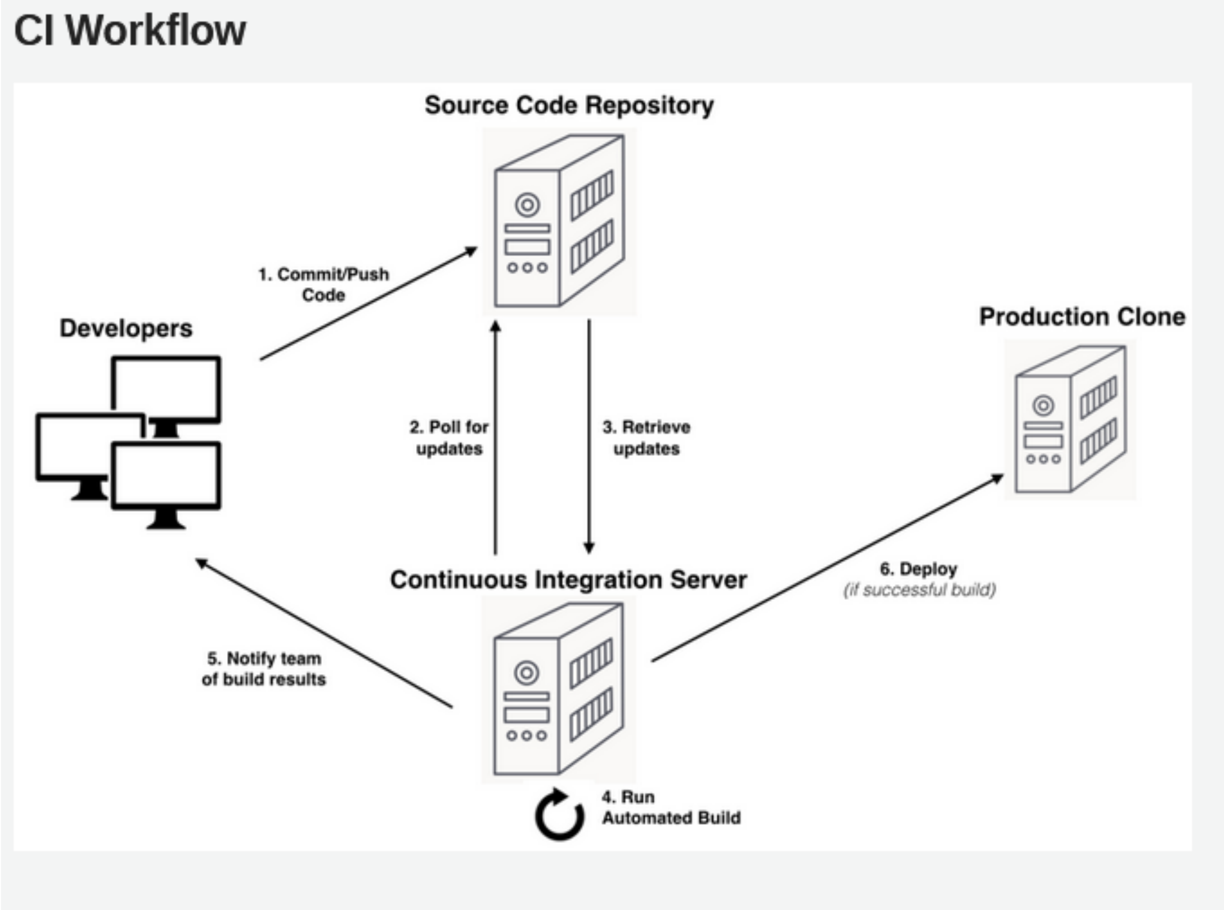
Integration



Integration Needs

- Ability to replace long integration/deployment phases with small, frequent phases
- Way to minimize integration effort
- Help with building quality software faster and with more confidence
- Ideal: Early, frequent integration

Continuous Integration (CI)

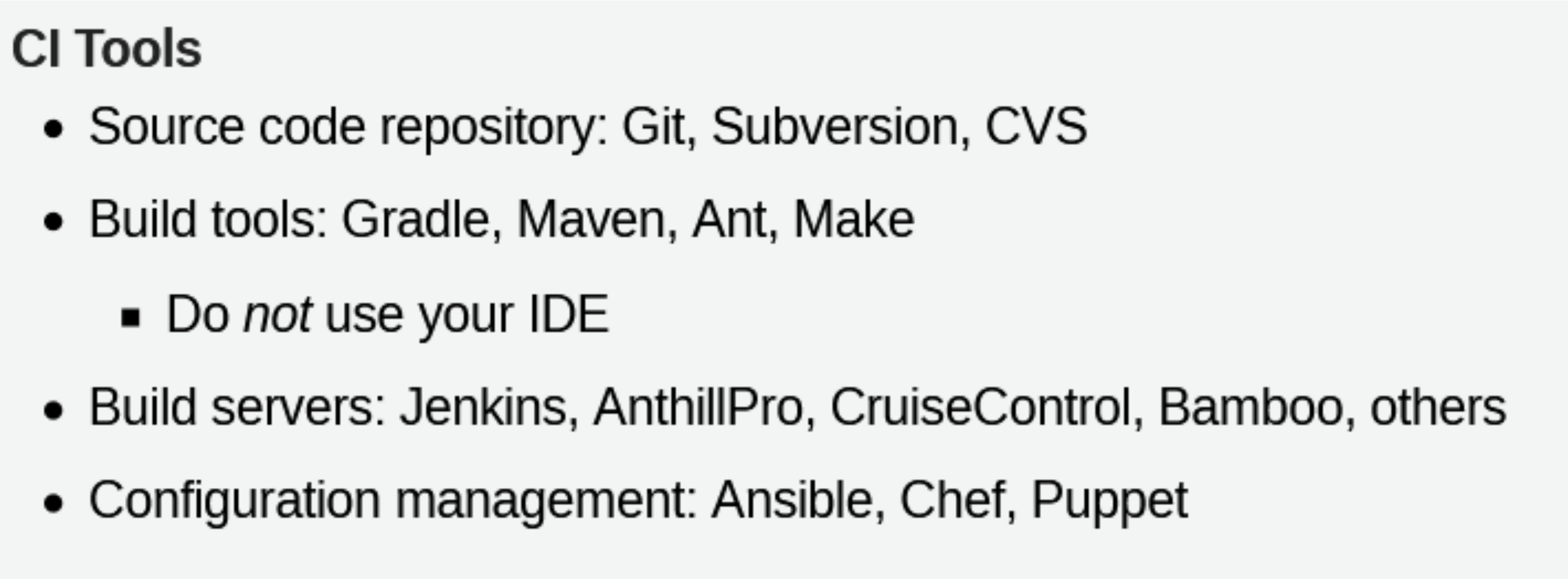


Continuous Integration (CI)

CI Benefits

Rapid feedback	•After build executes, team members notified about status •Reduces time to discover and fix new defects
Reduced risk	•Integrating many times a day reduces risks in project •Bugs detected and fixed sooner •Software health measurable using unit testing, code inspection reports
Team ownership	•No longer "us" vs "them" •Everyone receives regular reports on build status •Enables greater project visibility; everyone can spot trends and make effective decisions •Creates confidence to add features to project •Everyone on board with current project health
Building of deployable software	•Build process must generate deployable software •Goal is to create software that can be deployed at any time •Does not mean you *must* deploy software, but it is good release candidate •Many development teams struggle with this scenario
Automated process	•Automating build saves time, costs, effort •Process runs the same every time •Developers freed from repetitive processes, can do more high-value work

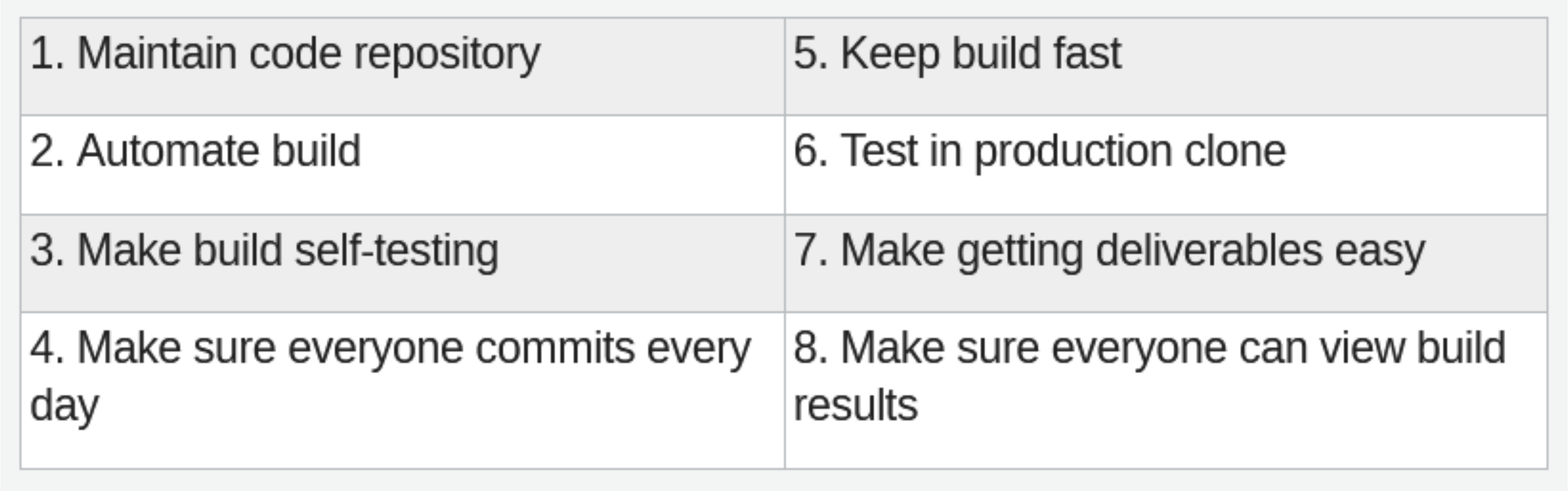
Continuous Integration (CI)



CI Tools

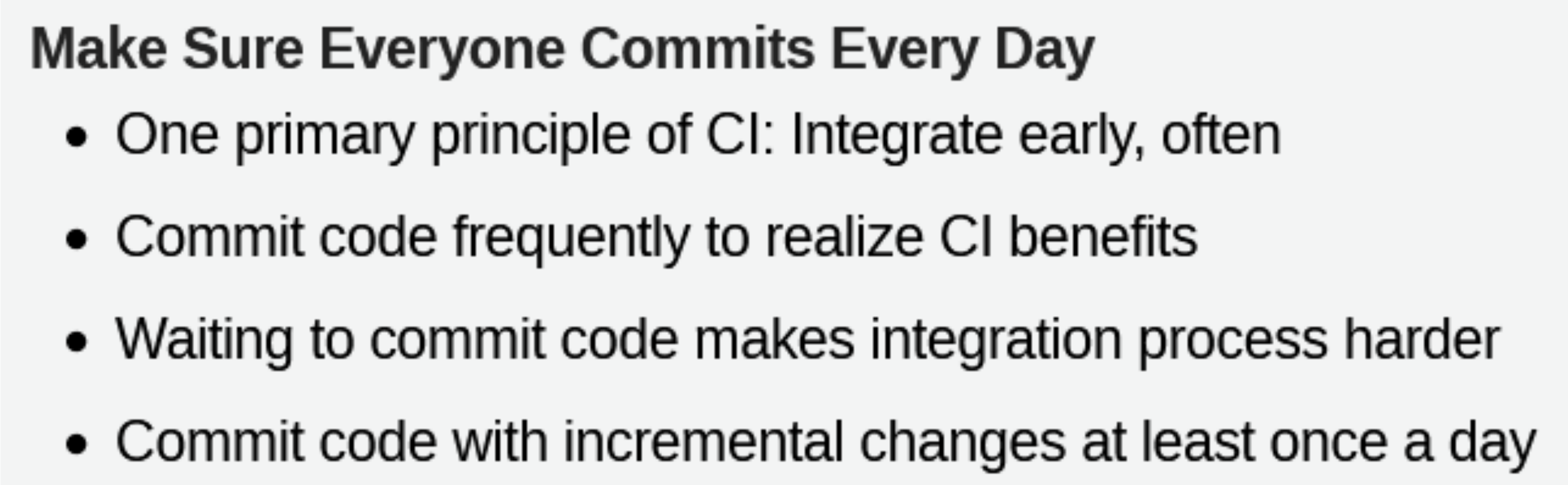
- Source code repository: Git, Subversion, CVS
- Build tools: Gradle, Maven, Ant, Make
  - Do *not* use your IDE
- Build servers: Jenkins, AnthillPro, CruiseControl, Bamboo, others
- Configuration management: Ansible, Chef, Puppet

CI Best Practices



1. Maintain code repository	5. Keep build fast
2. Automate build	6. Test in production clone
3. Make build self-testing	7. Make getting deliverables easy
4. Make sure everyone commits every day	8. Make sure everyone can view build results

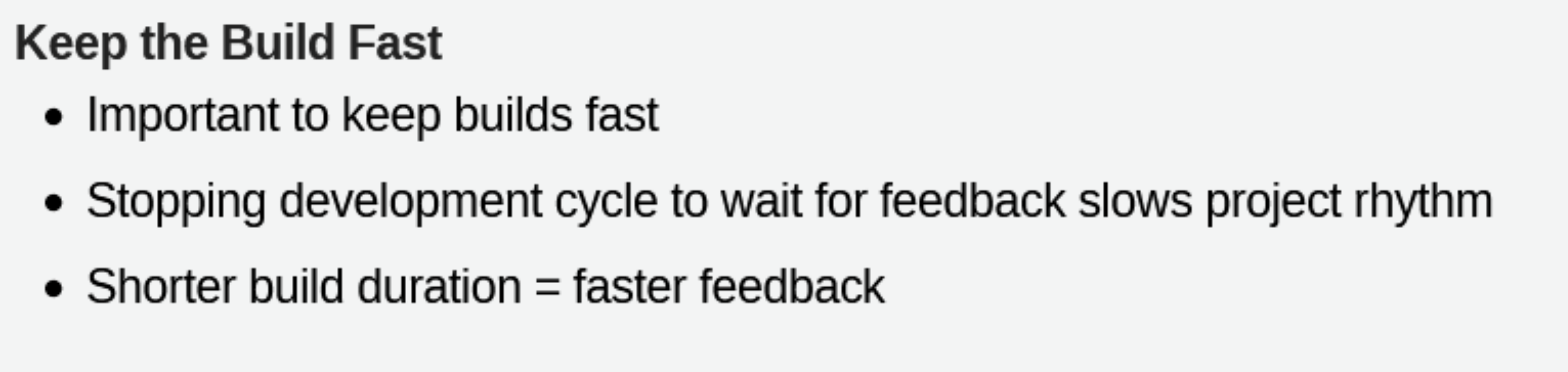
CI Best Practices



Make Sure Everyone Commits Every Day

- One primary principle of CI: Integrate early, often
- Commit code frequently to realize CI benefits
- Waiting to commit code makes integration process harder
- Commit code with incremental changes at least once a day

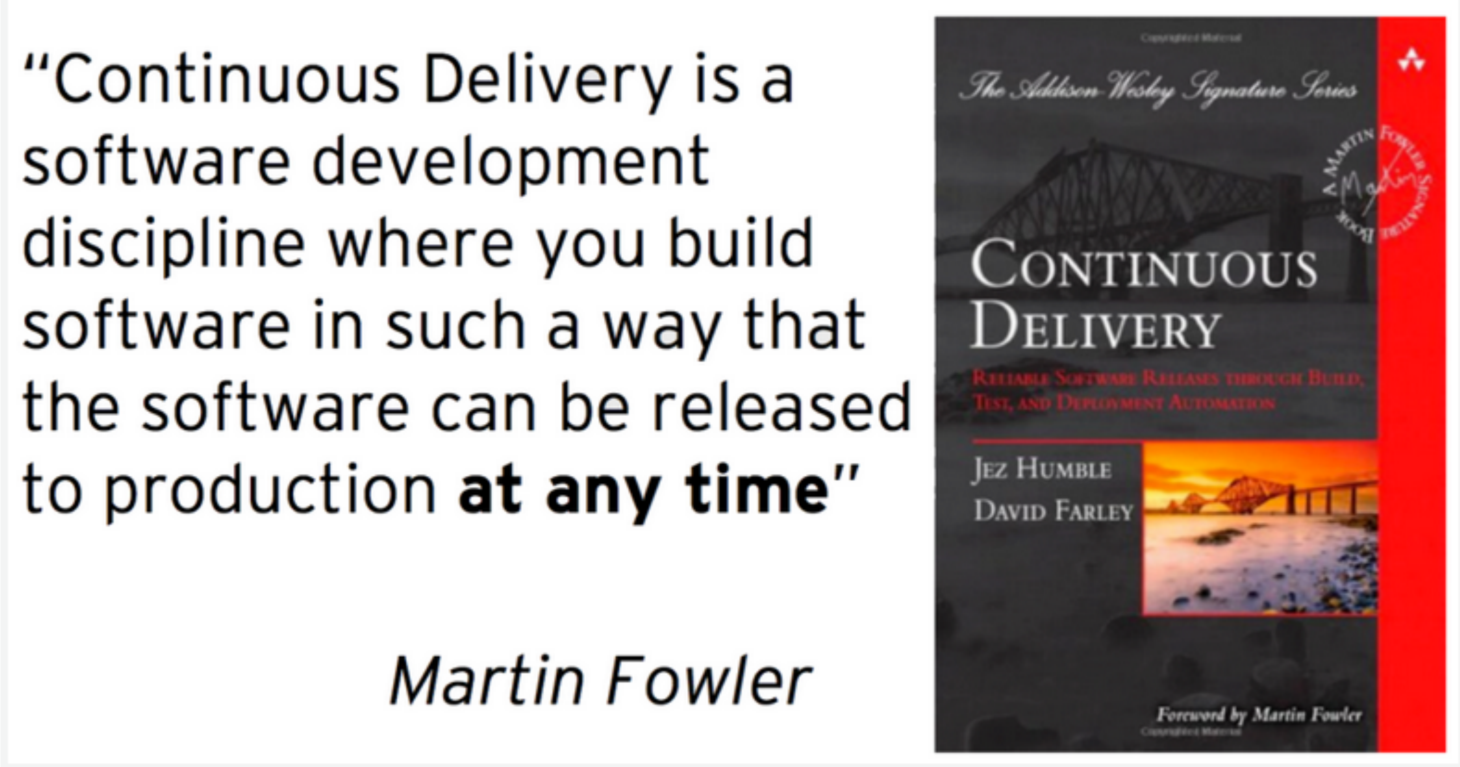
CI Best Practices



Keep the Build Fast

- Important to keep builds fast
- Stopping development cycle to wait for feedback slows project rhythm
- Shorter build duration = faster feedback

Continuous Delivery (CD)



"Continuous Delivery is a software development discipline where you build software in such a way that the software can be released to production **at any time**"

Martin Fowler

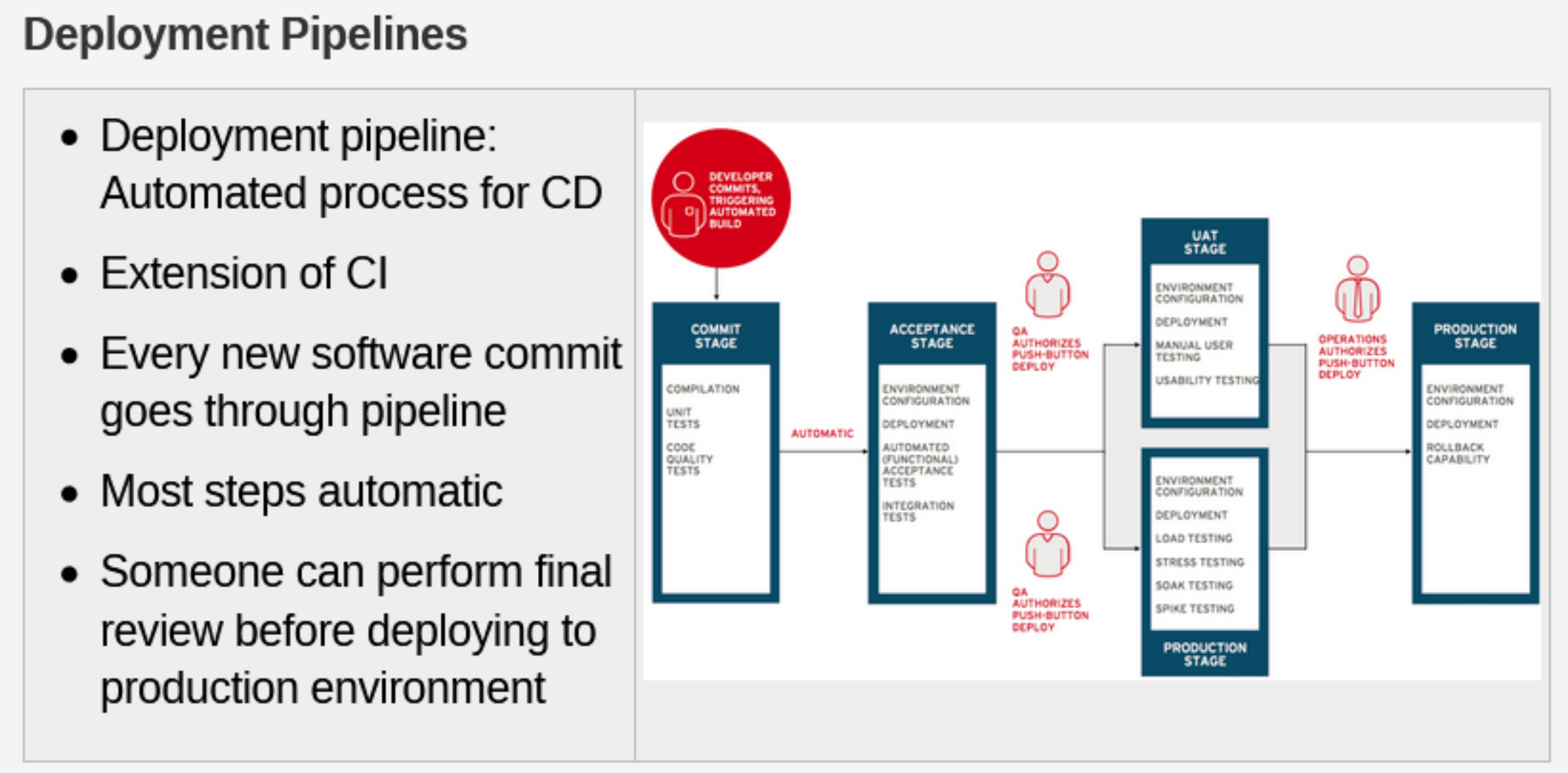
The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fowler

Continuous Delivery (CD)

CD Benefits

Empower teams	•Developers, QA, operations personnel can deploy application version they want into environment of choice •Testers can select older software versions to verify changes in newer versions •Support staff can deploy released application version into environment to test for defects •Operations staff can select good build and deploy to production •Can perform releases at push of button
Reduce errors	•Errors can easily appear in software •Can be in source code or configuration files •Having everything versioned eliminates need for manual configuration •Having everything automated gives teams repeatable process •Not subject to manual configuration risks
Promote deployment flexibility	•Making deployment with CD is simple task: •Provision environment •Deploy code •Make configuration changes •All part of automated process •Gives teams flexibility to deploy application to any environment with push of button

Continuous Delivery (CD)

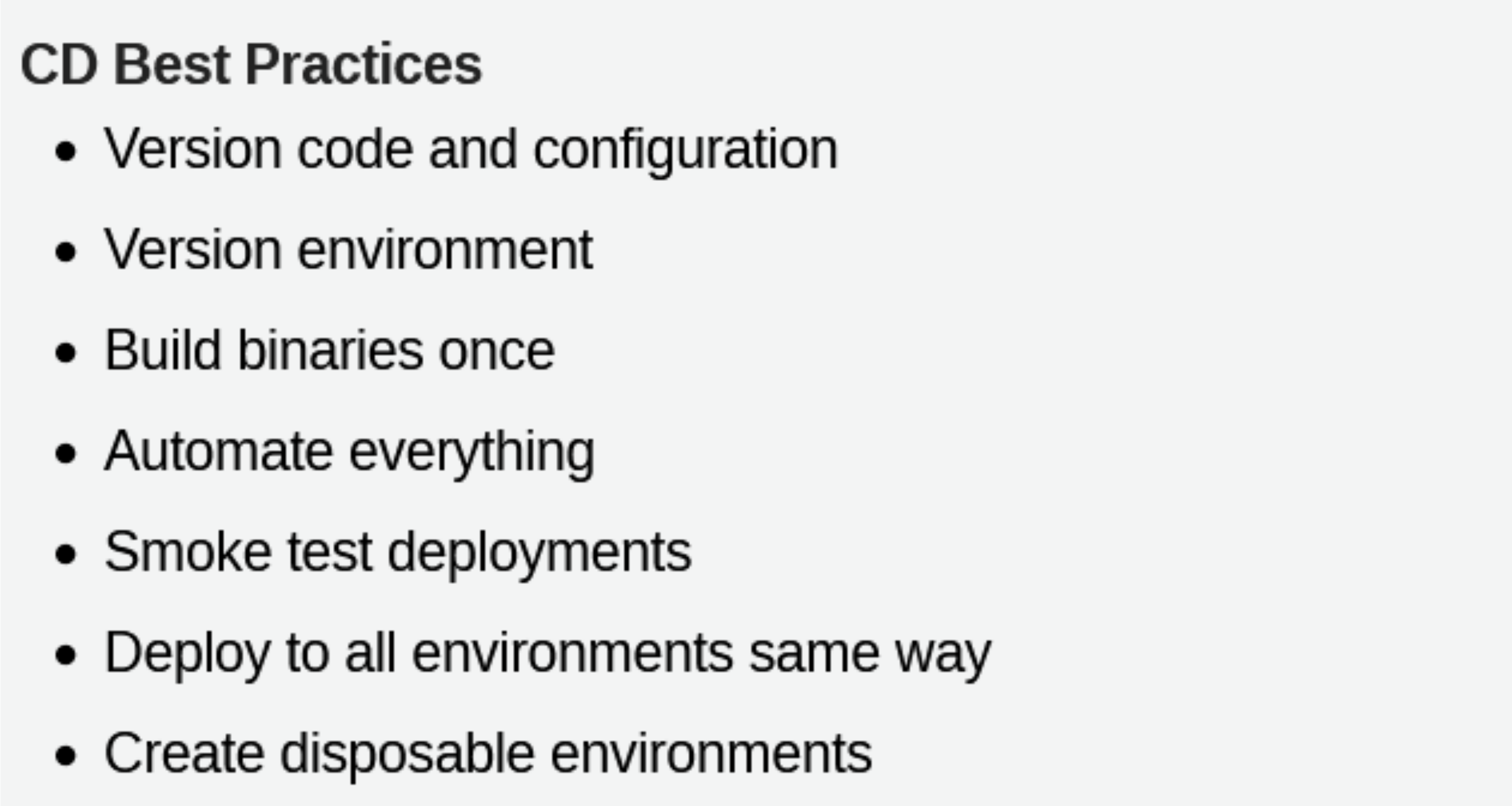


Deployment Pipelines

- Deployment pipeline: Automated process for CD
- Extension of CI
- Every new software commit goes through pipeline
- Most steps automatic
- Someone can perform final review before deploying to production environment

DEVELOPER COMMITS, TRIGGERING AUTOMATED BUILD
COMMIT STAGE
COMPILATION
UNIT TESTS
CODE QUALITY TESTS
AUTOMATIC
ACCEPTANCE STAGE
ENVIRONMENT CONFIGURATION
DEPLOYMENT
AUTOMATED (FUNCTIONAL) ACCEPTANCE TESTS
INTEGRATION TESTS
QA AUTHORIZES PUSH-BUTTON DEPLOY
QA AUTHORIZES PUSH-BUTTON DEPLOY
UAT STAGE
ENVIRONMENT CONFIGURATION
DEPLOYMENT
MANUAL USER TESTING
USABILITY TESTING
ENVIRONMENT CONFIGURATION
DEPLOYMENT
LOAD TESTING
STRESS TESTING
SOAK TESTING
SPIKE TESTING
PRODUCTION STAGE
OPERATIONS AUTHORIZES PUSH-BUTTON DEPLOY
PRODUCTION STAGE
ENVIRONMENT CONFIGURATION
DEPLOYMENT
ROLLBACK CAPABILITY

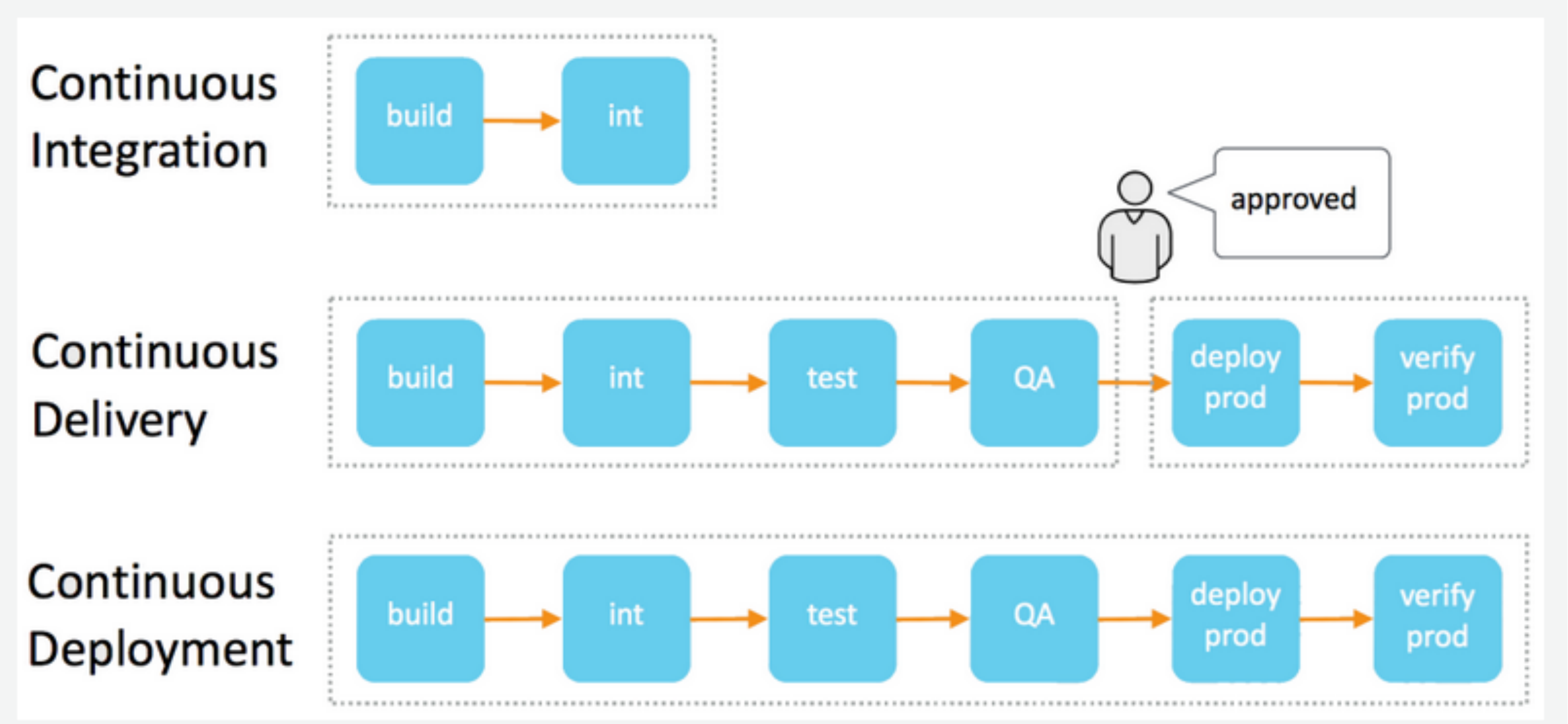
Continuous Delivery (CD)



CD Best Practices

- Version code and configuration
- Version environment
- Build binaries once
- Automate everything
- Smoke test deployments
- Deploy to all environments same way
- Create disposable environments

DevOps Terminology



Tekton

1. What are pipelines
2. Technology choices in CP4Apps and OpenShift
3. Pipeline structure (steps, tasks, pipelines)
4. Putting it all together
5. How pipelines provide control and governance
6. Which pipelines and tasks are shipped with CP4Apps
7. Customizing pre-built pipelines, adding tasks

Modernize your DevOps Toolchain

Kubernetes-native pipelines for CI/CD

Pre-built tasks & pipelines for build and deploy

Integrates easily with git events

Leverages the power of Kubernetes to manage your toolchain

Cloud Pak for Applications provides pre-built pipelines in a modern DevOps Toolchain

Tekton provides Kubernetes-style resources for declaring CI/CD concepts

Why Pipelines built using Tekton?

Description	**Value**
Runs serverless (no babysitting !)	Don’t worry about a Jenkins farm
Containers as building blocks	Not a monolith
Standard CRDs	It’s just standard Kubernetes
Build images with Kubernetes tools	Appsody uses buildah for image build
Deploy across multiple worker nodes	Scale and deploy automatically
Portable to any Kubernetes	No vendor lock-in
Part of CD Foundation	Governed open source w/ broad industry contribution

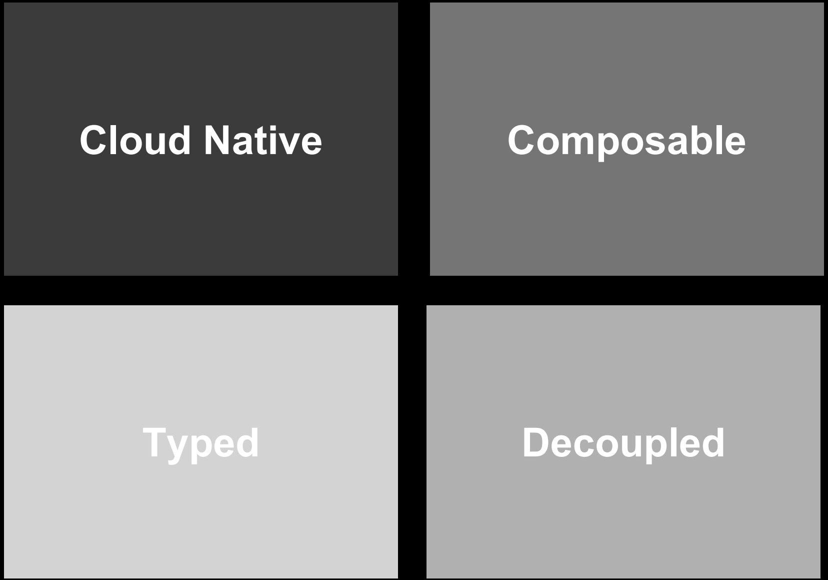
Tekton, a technical description

Cloud Native: Run on Kubernetes, has Kubernetes clusters as a first-class type, use containers as their building blocks

Composable: Tekton concepts build upon each other

Decoupled: The Tasks which make up a Pipeline can be run in isolation. One Pipeline can be used to deploy to any k8s cluster

Typed: Typed resources make it possible to swap out implementations



Cloud Native

Composable

Typed

Decoupled

Tekton Concept: Step

- The smallest building block
- Specify images, commands, arguments
- Is a container

```
steps:
  - name: echo
    image: ubuntu
    command:
      - echo
    args:
      - "hello world"
```

Tekton CRD: Task

- Sequence of **Steps**
- Steps run in sequential order
- Reusable
- Perform a specific task
- Runs on the same k8s node

```
apiVersion: tekton.dev/v1alpha1
kind: Task
metadata:
  name: echo-hello-world
spec:
  steps:
    - name: echo
      image: ubuntu
      command:
        - echo
      args:
        - "hello world"
```

Tekton CRD: Pipeline

- Creates an ordering of **Tasks**
  - Sequentially
  - Concurrently

Links input and output

Execute **Tasks** on different nodes

Natural Kubernetes experience

- oc apply which will invoke the operator

```
apiVersion: tekton.dev/v1alpha1
kind: Pipeline
metadata:
  name: tutorial-pipeline
spec:
  - name: build-app
    taskRef:
      name: build-push
    resources:
      outputs:
        - name: image
          resource: my-image
  - name: deploy-app
    taskRef:
      name: deploy-kubectl
    resources:
      inputs:
        - name: image
      resource: my-image
      from:
        - build-app
```

Tekton CRD: Pipeline Run

- An instance of Pipeline execution

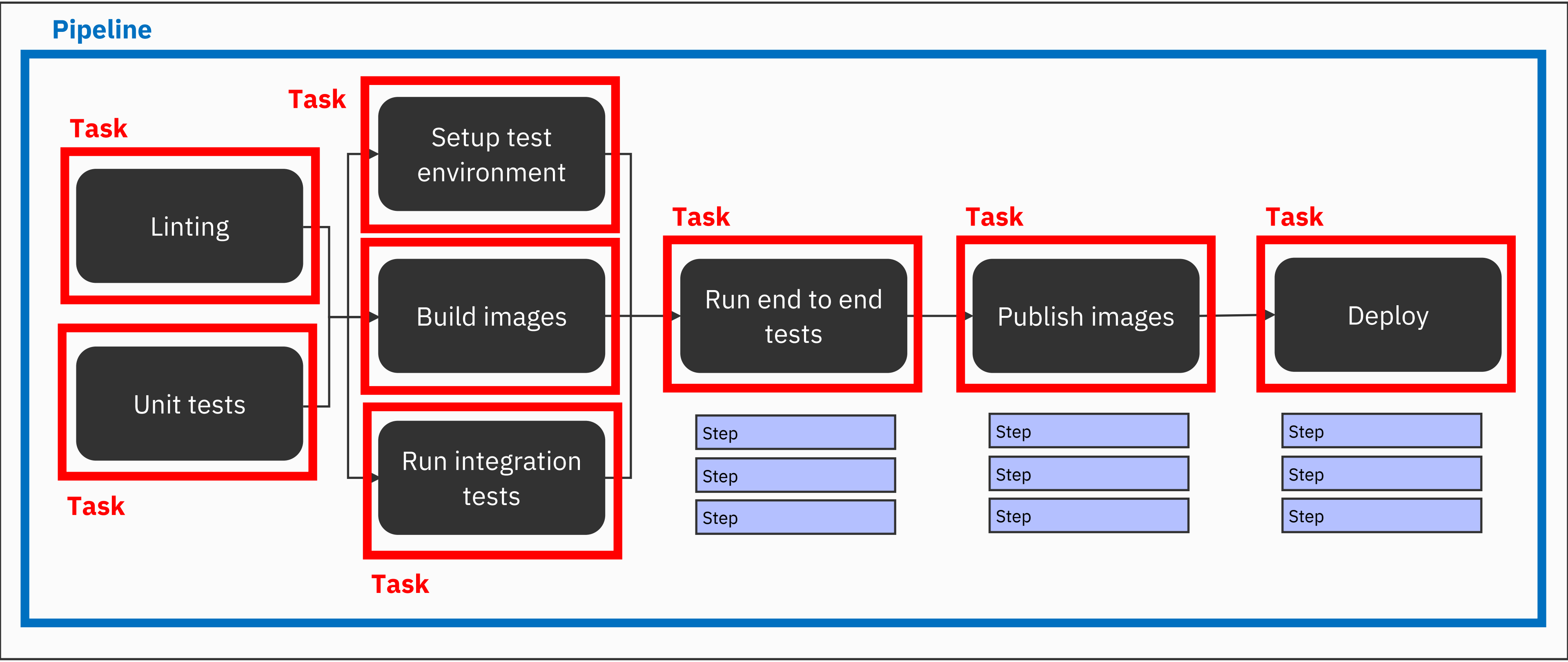
- Names pipeline to execute
- Applied manually
  - Useful for testing or one-off execution

- Created dynamically
- Tekton webhook for Github can dynamically create pipeline run.

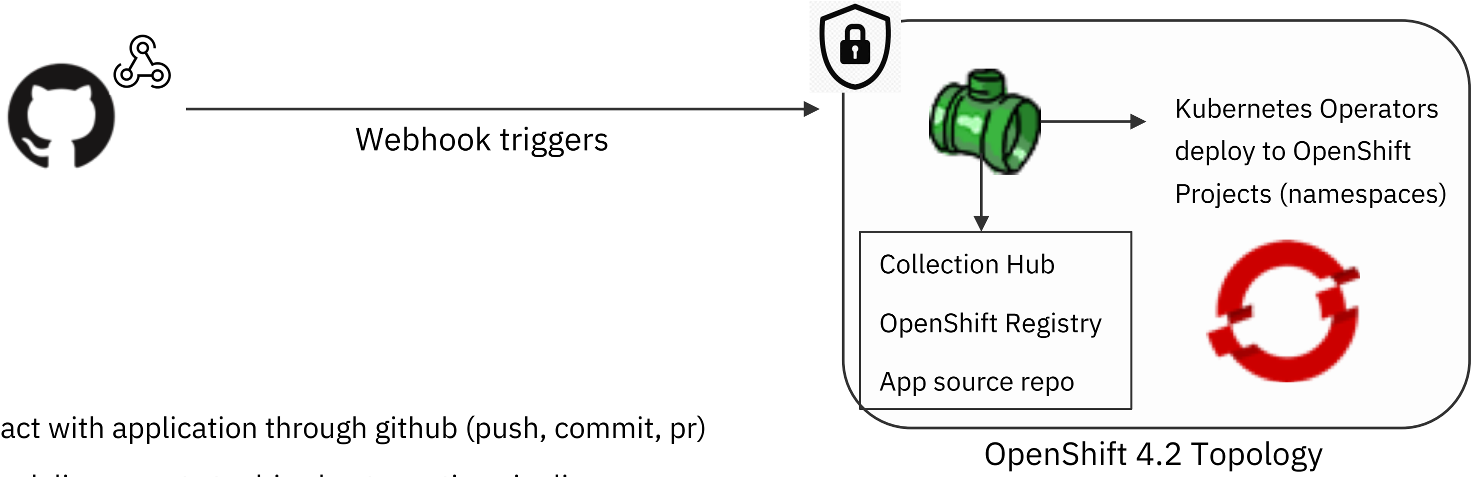
- Can inline Pipelines and Pipeline Tasks.

```yaml
apiVersion: tekton.dev/v1alpha1
kind: PipelineRun
metadata:
  name: tutorial-pipeline-run-1
spec:
  serviceAccountName: tutorial-service
  pipelineRef:
    name: tutorial-pipeline
  resources:
    - name: source-repo
      resourceRef:
        name: my-git
    - name: web-image
      resourceRef:
        name: my-image
```

Putting it all together



How the pipelines control build and deployment on OpenShift with CP4Apps



Process:

1. Developers interact with application through github (push, commit, pr)
2. Github webhooks deliver events to drive best practice pipelines
3. Images are built server side with pipelines using enterprise governed stacks
4. Appsody Operator deploys microservice using best practices as a result of the deployed pipeline

Which pre-built pipeline tasks ship with CP4Apps v4.0

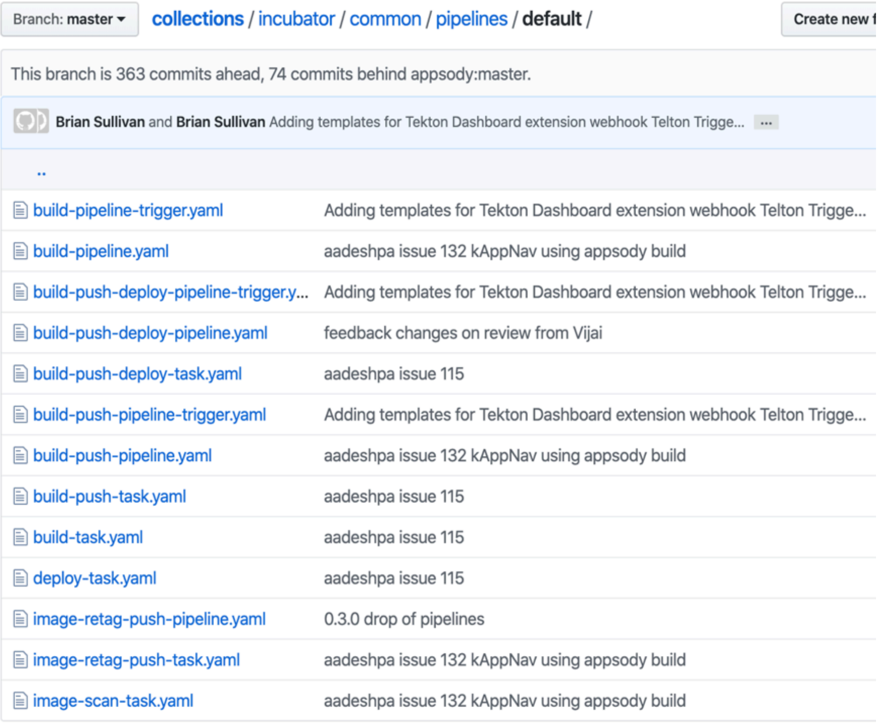
Primary Tasks

Build

Deploy

Retag

VA Scan



Branch: master ▾ collections / incubator / common / pipelines / **default** / Create new f

This branch is 363 commits ahead, 74 commits behind appsody:master.

Brian Sullivan and **Brian Sullivan** Adding templates for Tekton Dashboard extension webhook Telton Trigge... …

..

build-pipeline-trigger.yaml	Adding templates for Tekton Dashboard extension webhook Telton Trigge...
build-pipeline.yaml	aadeshpa issue 132 kAppNav using appsody build
build-push-deploy-pipeline-trigger.y...	Adding templates for Tekton Dashboard extension webhook Telton Trigge...
build-push-deploy-pipeline.yaml	feedback changes on review from Vijai
build-push-deploy-task.yaml	aadeshpa issue 115
build-push-pipeline-trigger.yaml	Adding templates for Tekton Dashboard extension webhook Telton Trigge...
build-push-pipeline.yaml	aadeshpa issue 132 kAppNav using appsody build
build-push-task.yaml	aadeshpa issue 115
build-task.yaml	aadeshpa issue 115
deploy-task.yaml	aadeshpa issue 115
image-retag-push-pipeline.yaml	0.3.0 drop of pipelines
image-retag-push-task.yaml	aadeshpa issue 132 kAppNav using appsody build
image-scan-task.yaml	aadeshpa issue 132 kAppNav using appsody build

NOTE:

Find more tasks in the Kabanero Community

https://github.com/kabanero-io/kabanero-pipelines

Find more tasks in the

Tekton Community

https://github.com/tektoncd/catalog

Community content is not supported by CP4Apps

IBM