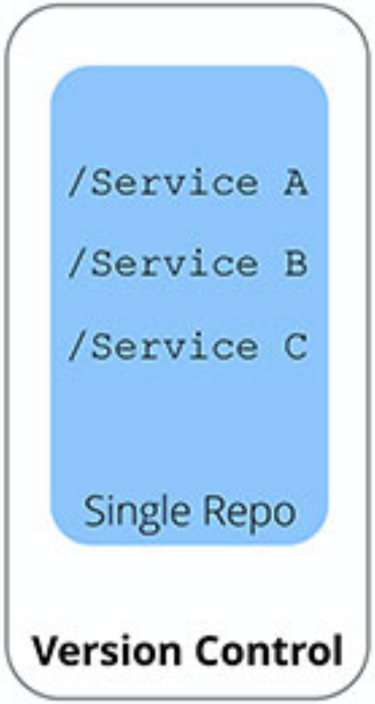
MONOLITHS

TO MICROSERVICES

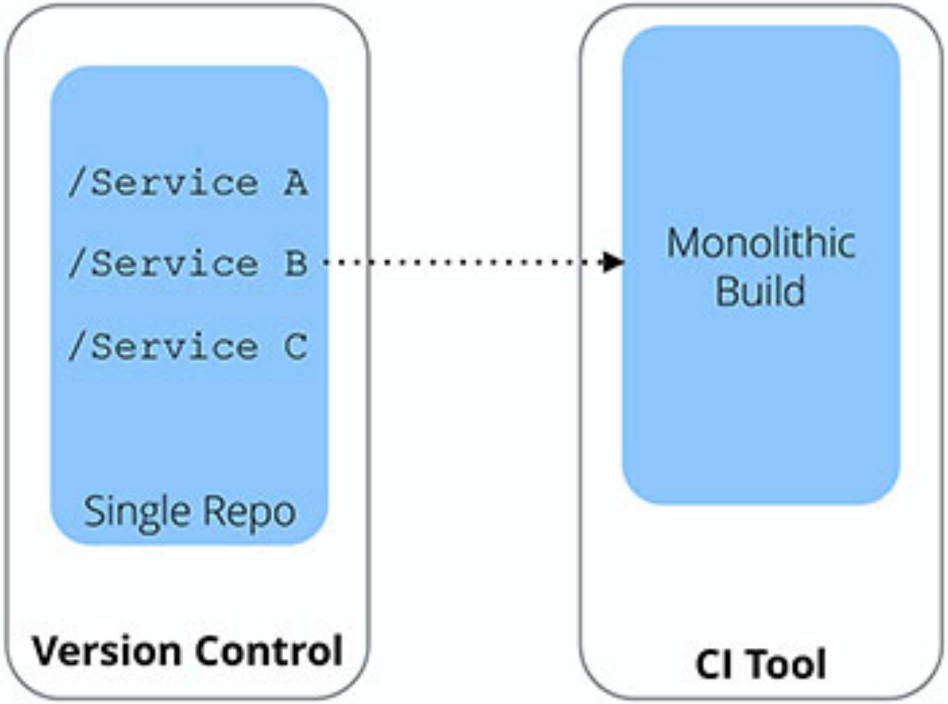
Sam Newman

Build & Deployment

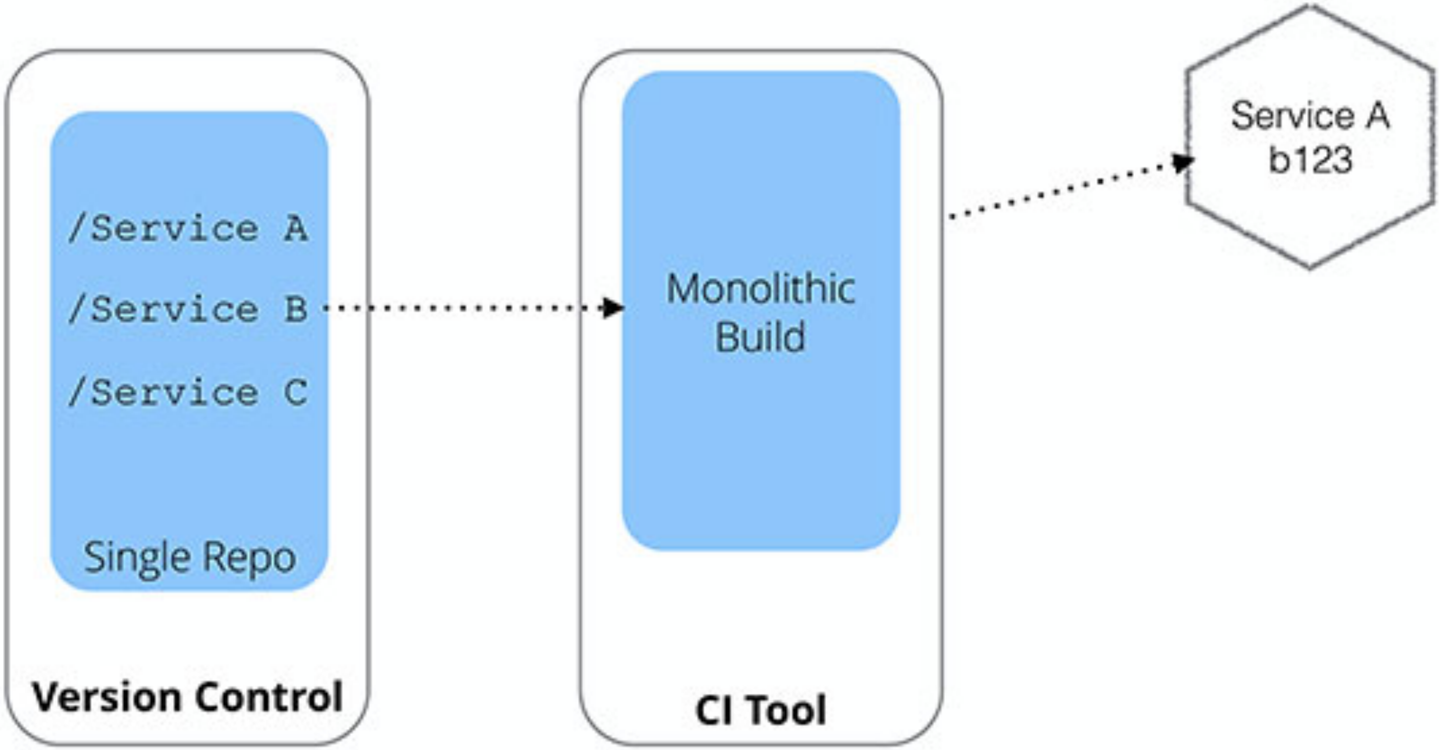
ONE GIANT BUILD



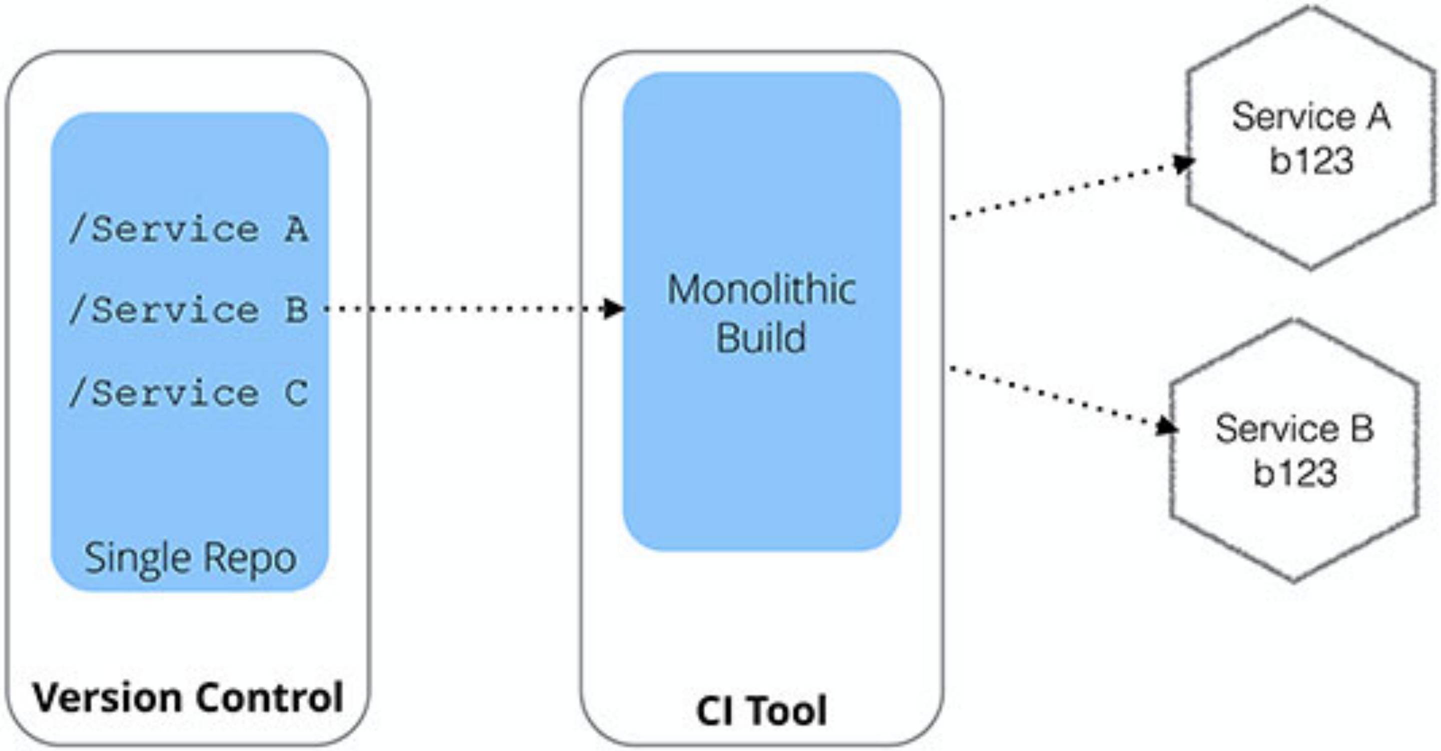
ONE GIANT BUILD



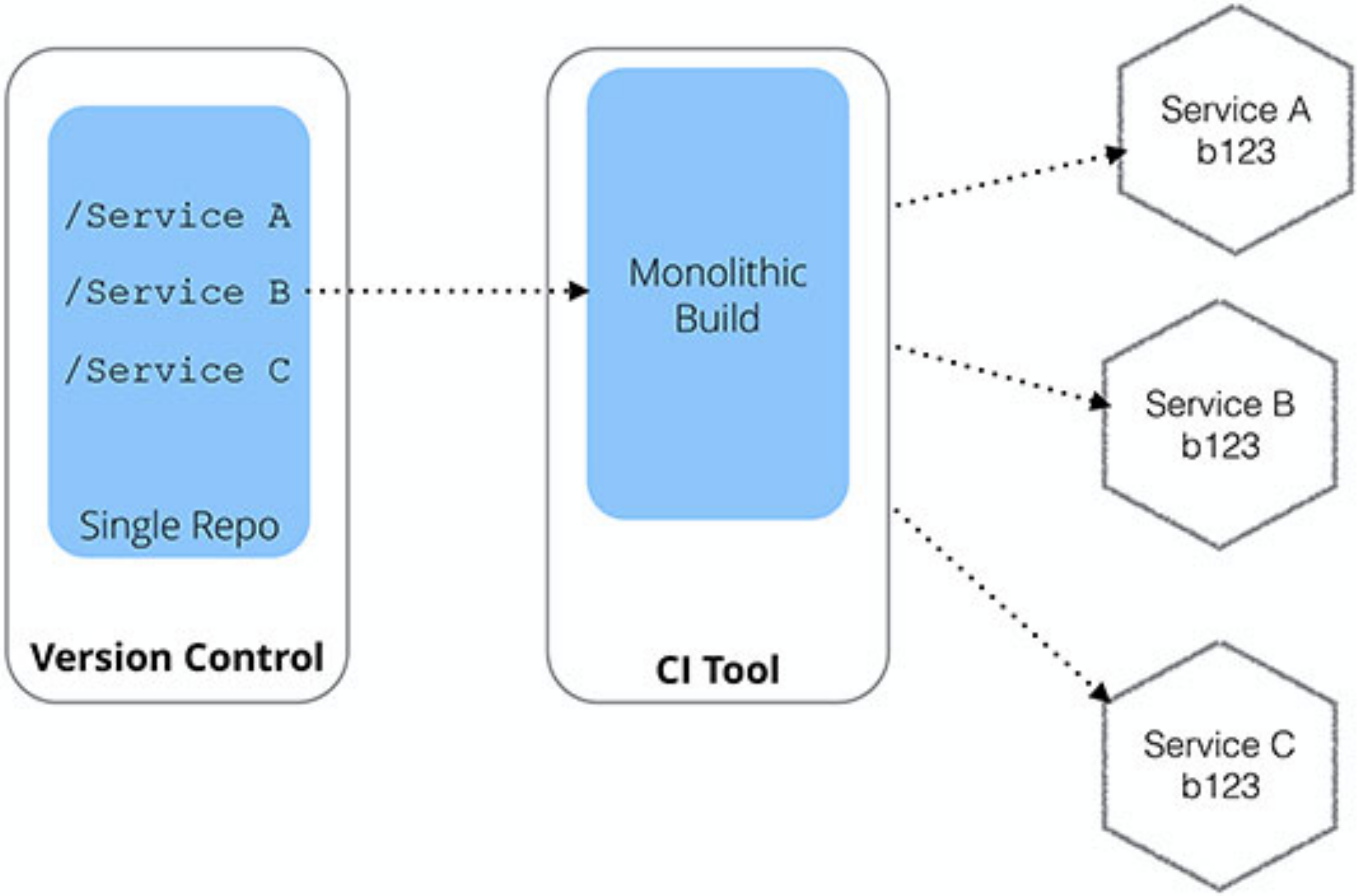
ONE GIANT BUILD



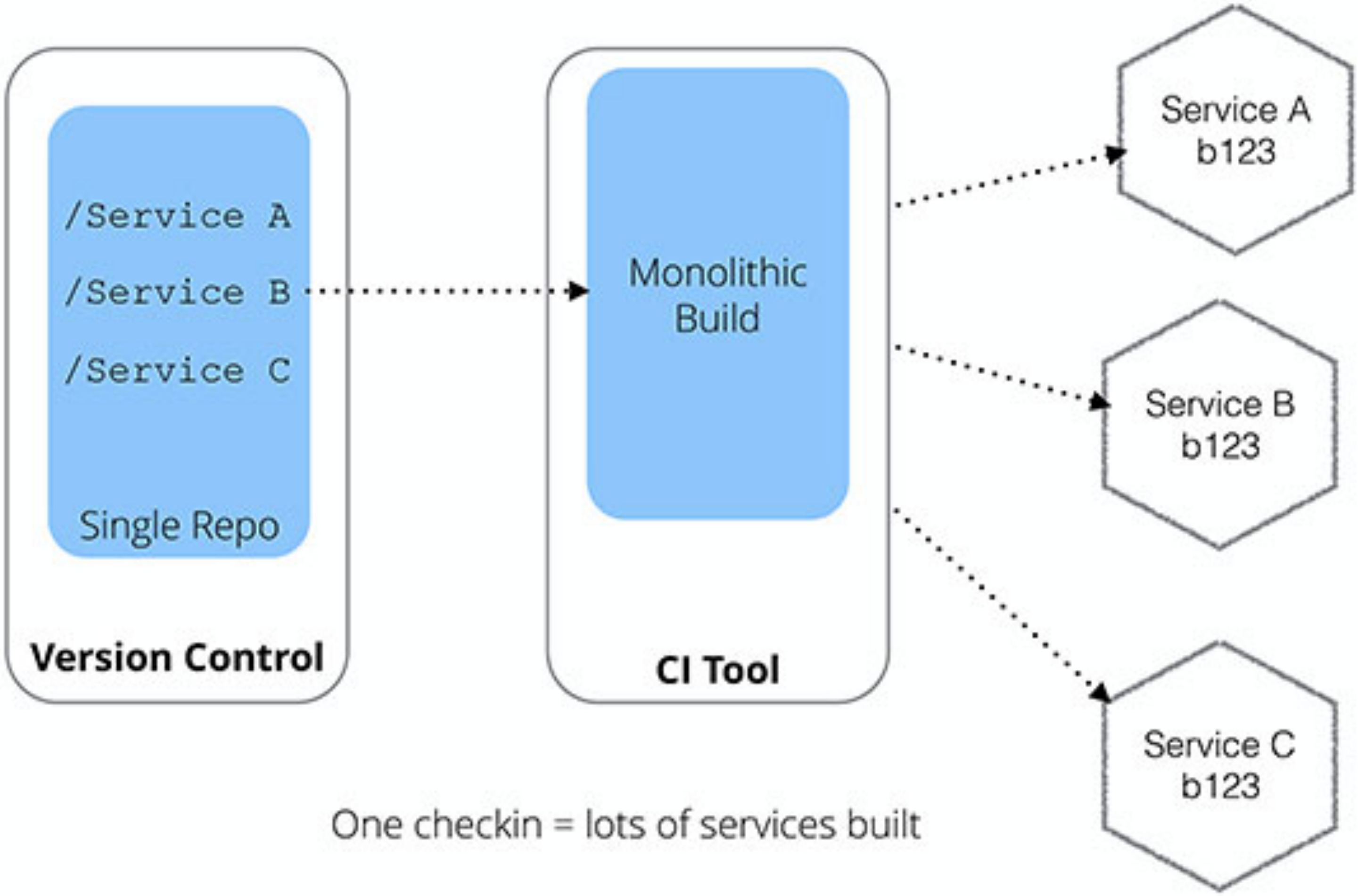
ONE GIANT BUILD



ONE GIANT BUILD



ONE GIANT BUILD



/Service A
/Service B
/Service C

Single Repo

Version Control

Monolithic Build

CI Tool

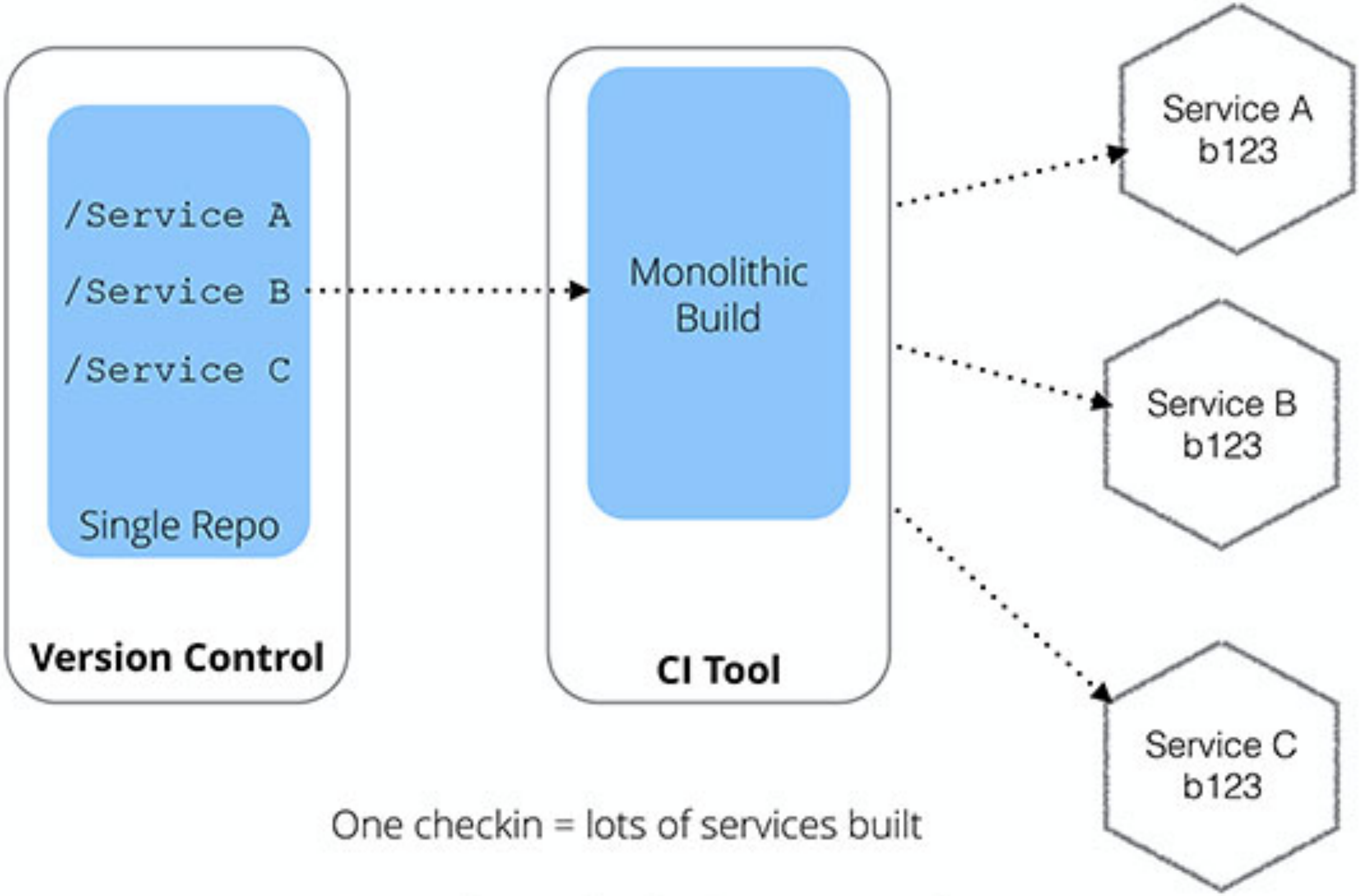
Service A b123

Service B b123

Service C b123

One checkin = lots of services built

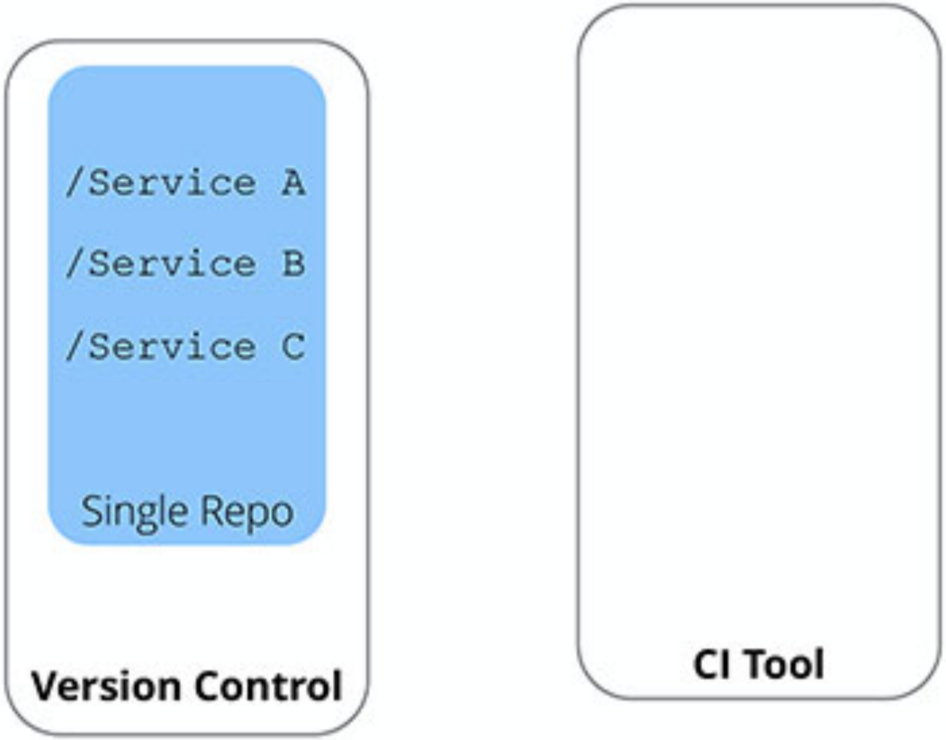
ONE GIANT BUILD



One checkin = lots of services built

Can't independently release in a safe manner

ONE BUILD PER SERVICE, SINGLE REPO



/Service A

/Service B

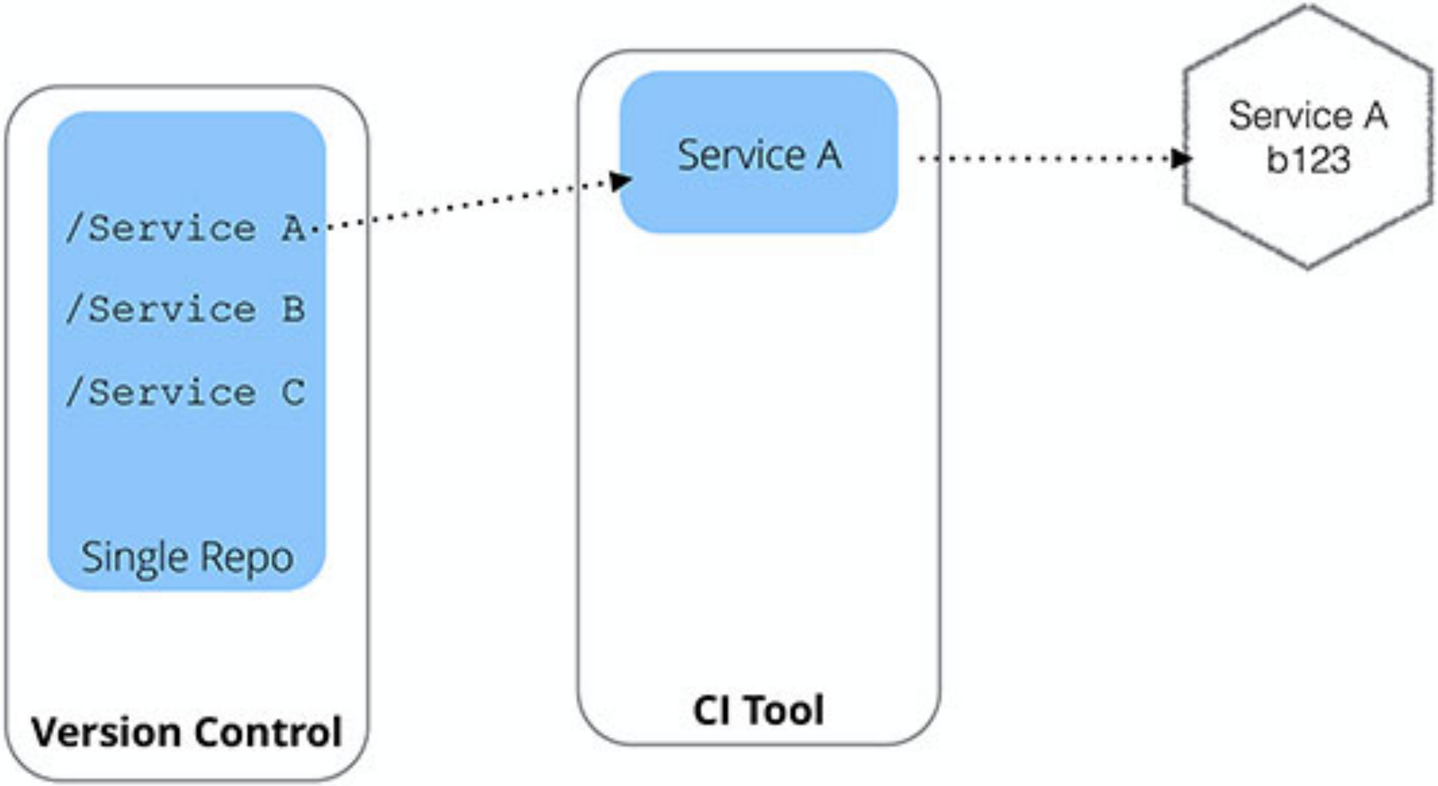
/Service C

Single Repo

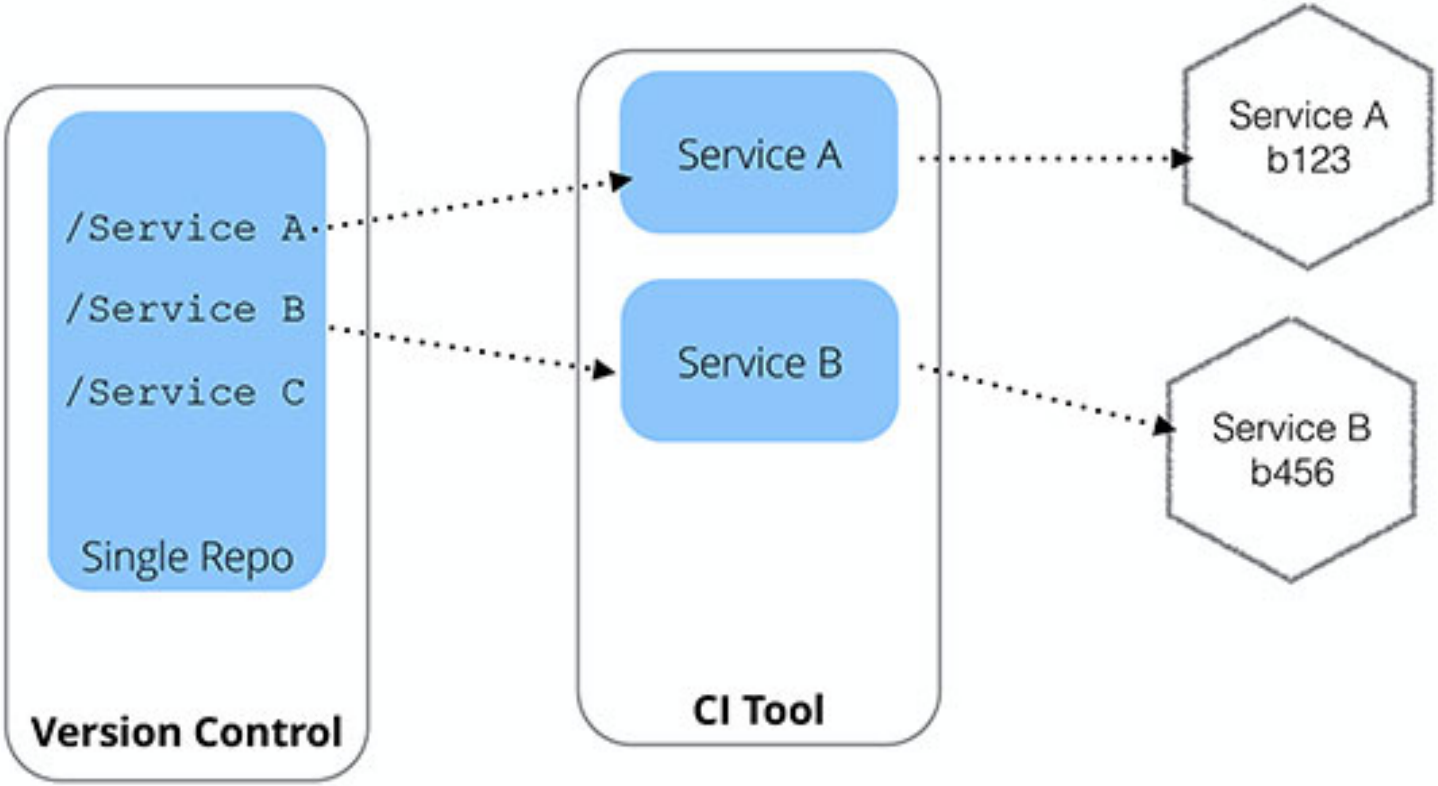
Version Control

CI Tool

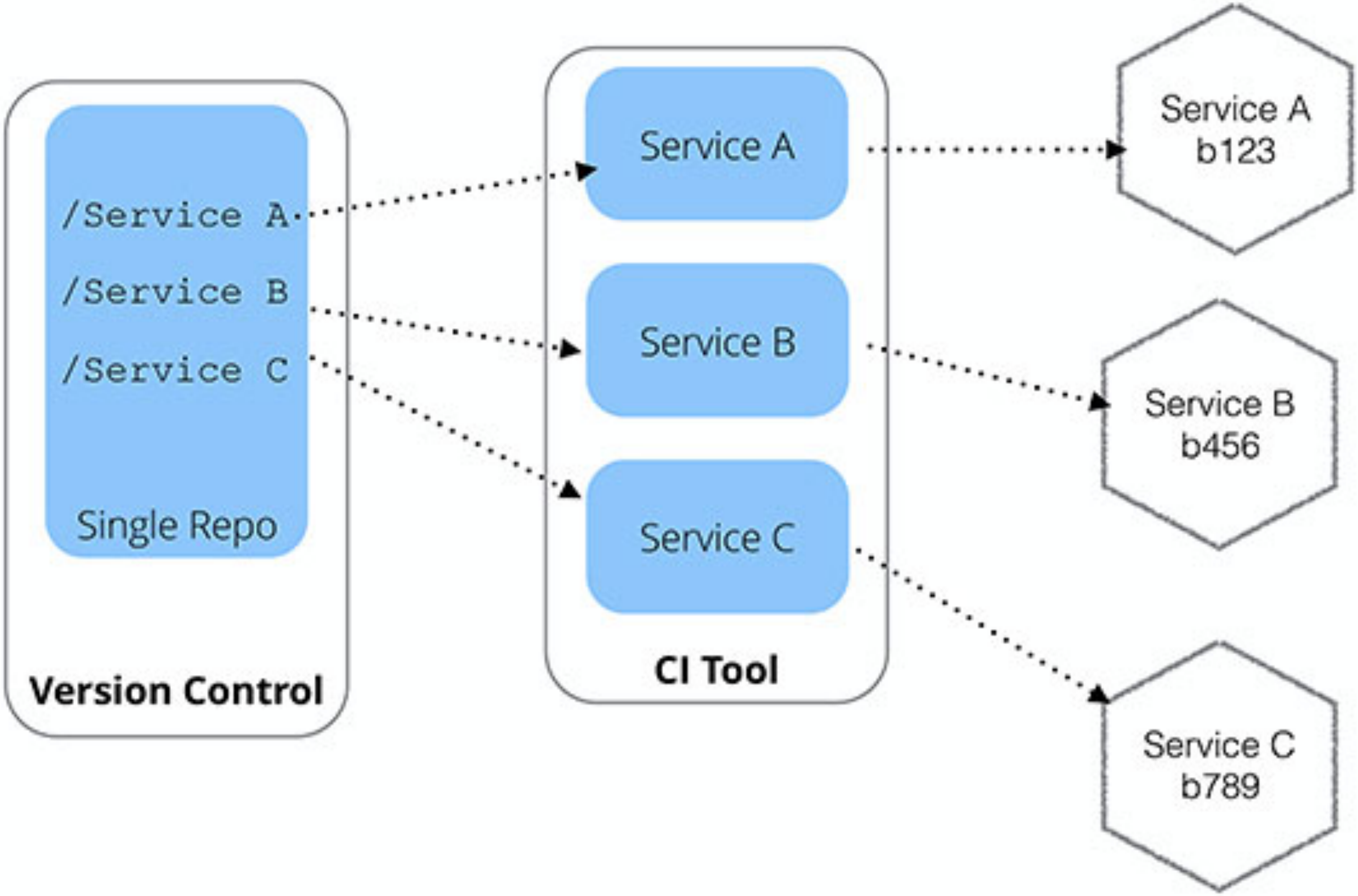
ONE BUILD PER SERVICE, SINGLE REPO



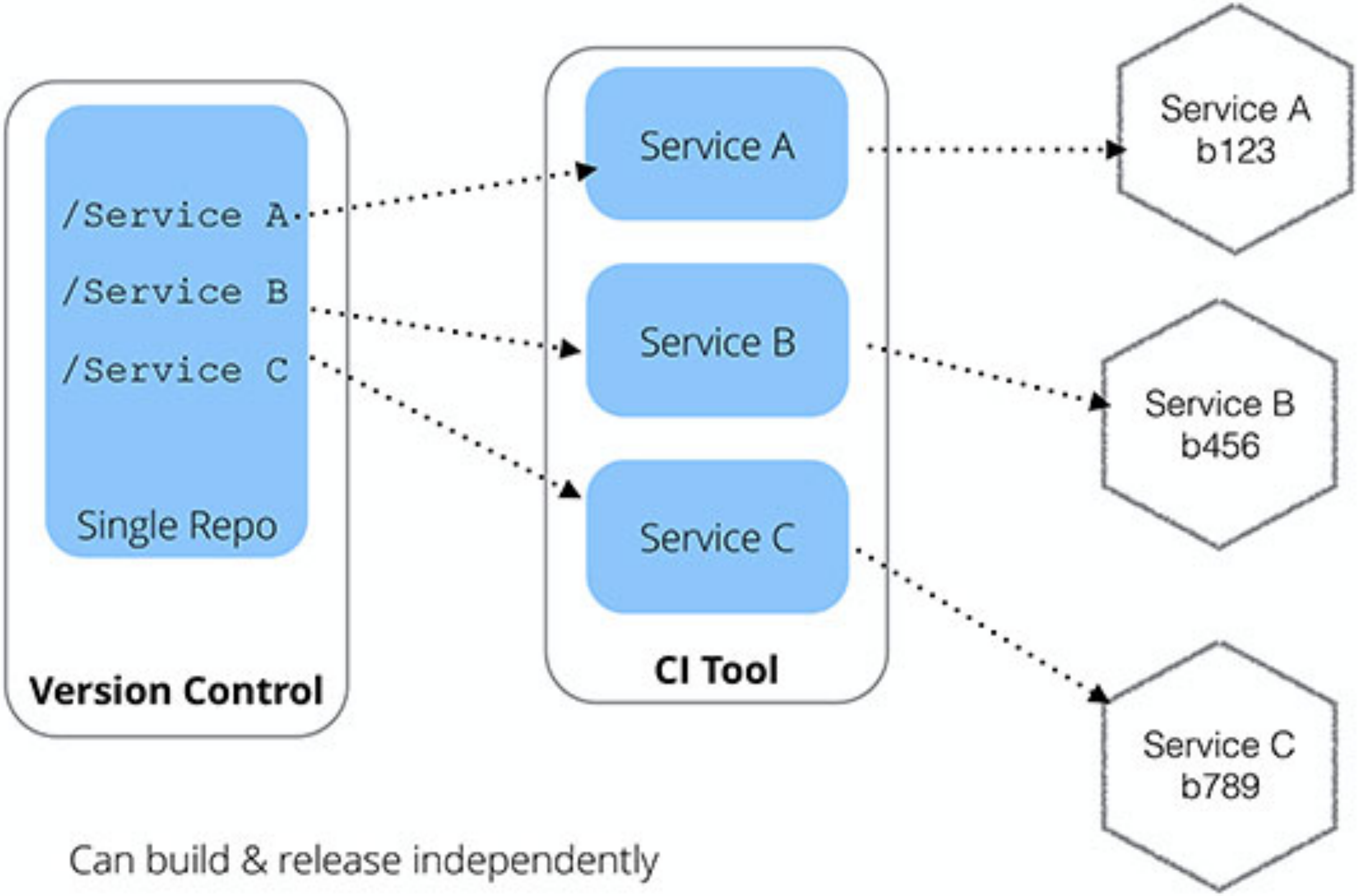
ONE BUILD PER SERVICE, SINGLE REPO



ONE BUILD PER SERVICE, SINGLE REPO

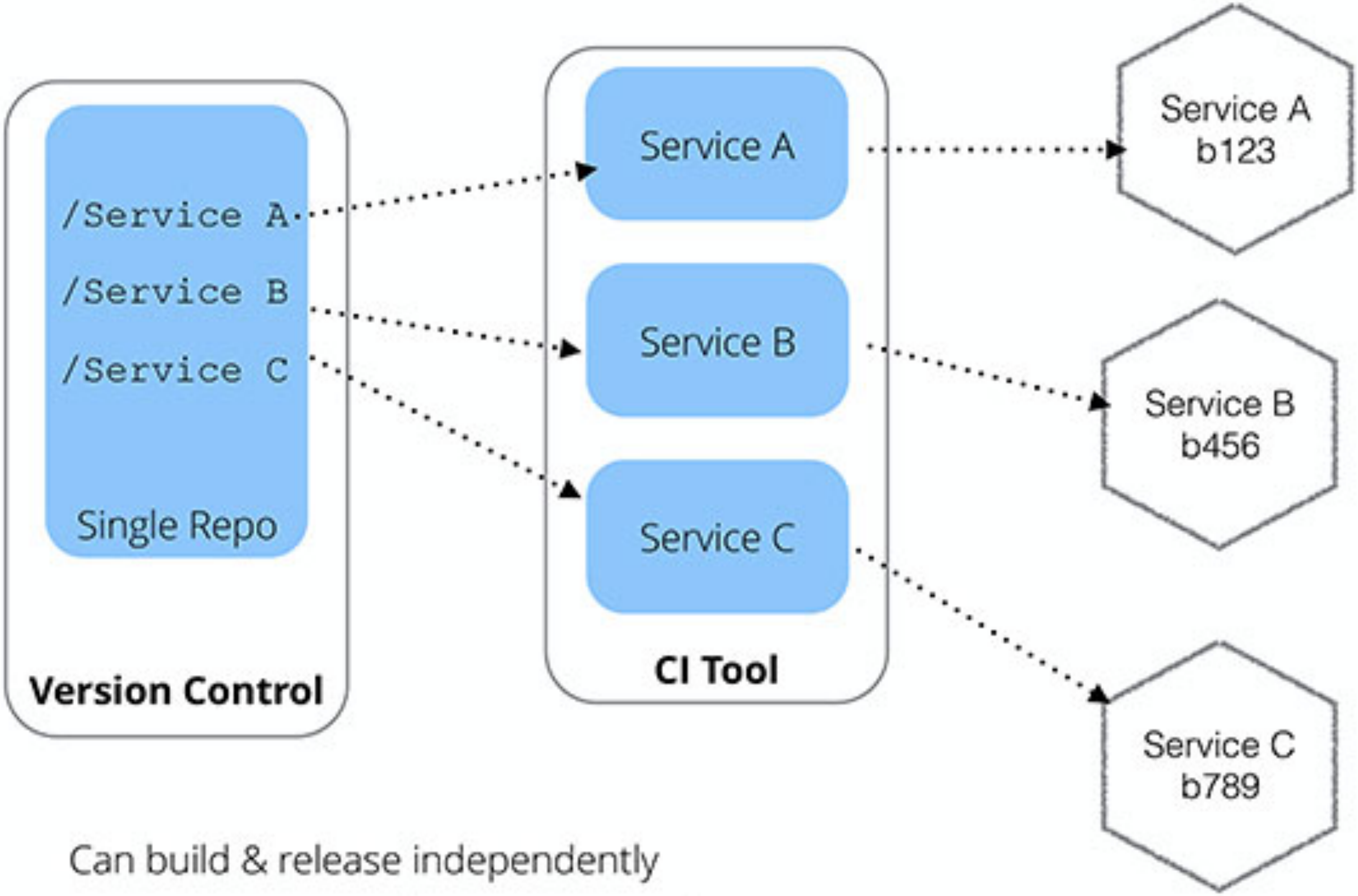


ONE BUILD PER SERVICE, SINGLE REPO



Can build & release independently

ONE BUILD PER SERVICE, SINGLE REPO

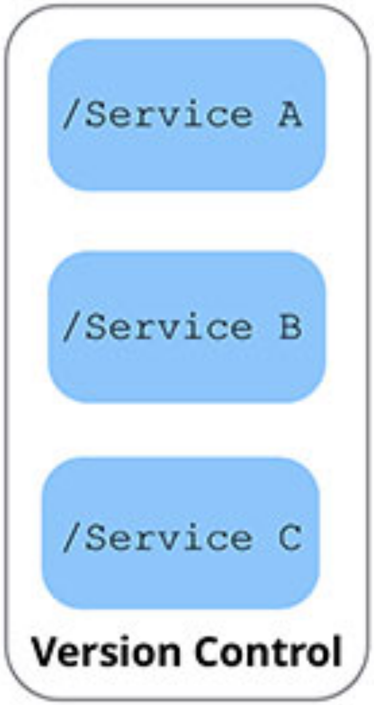


Can build & release independently

Can encourage making cross-repo changes

Ownership

ONE BUILD & REPO PER SERVICE



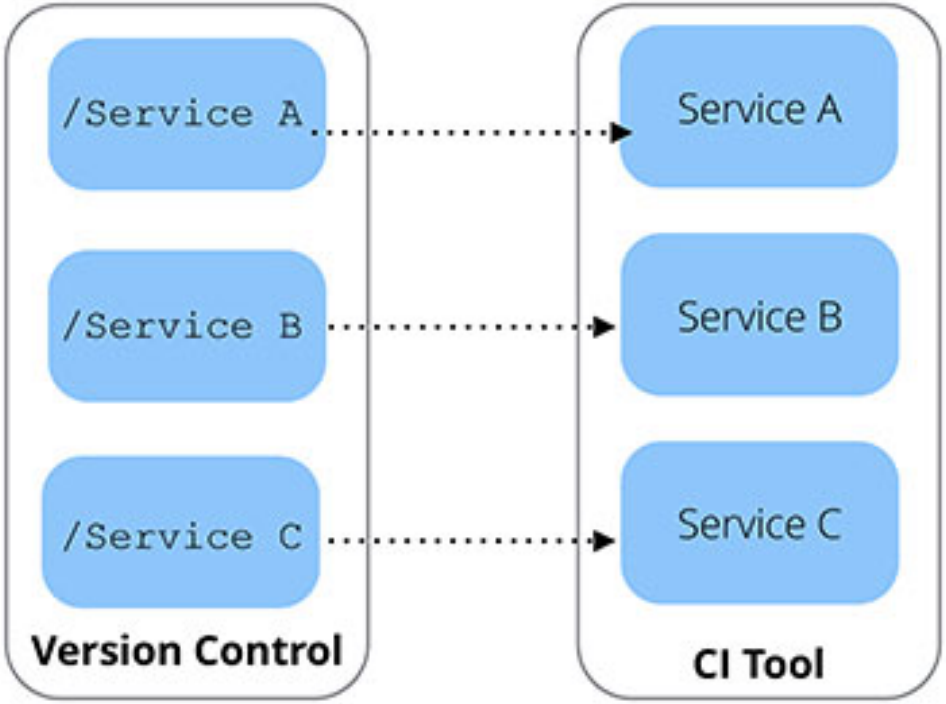
/Service A

/Service B

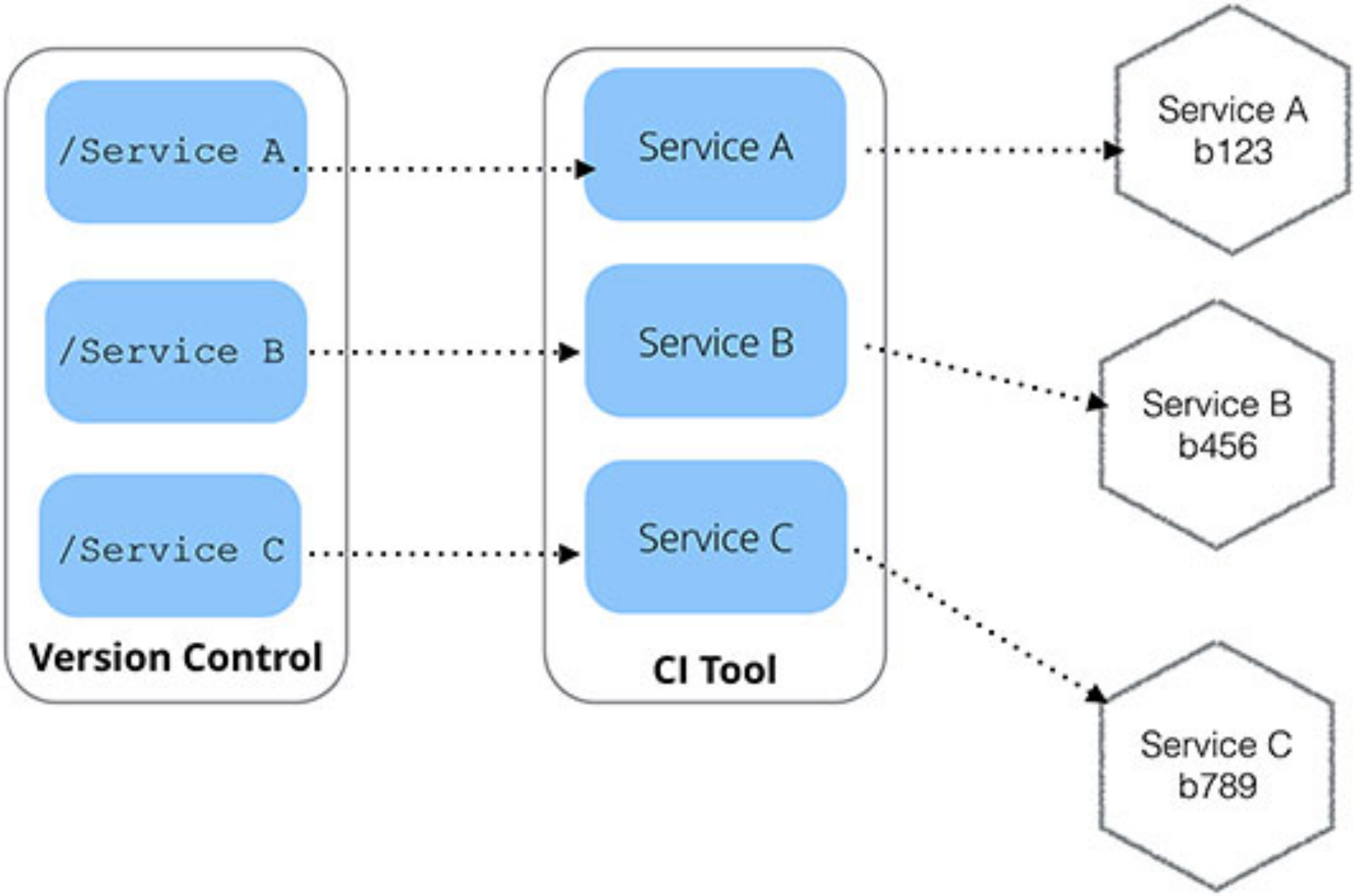
/Service C

Version Control

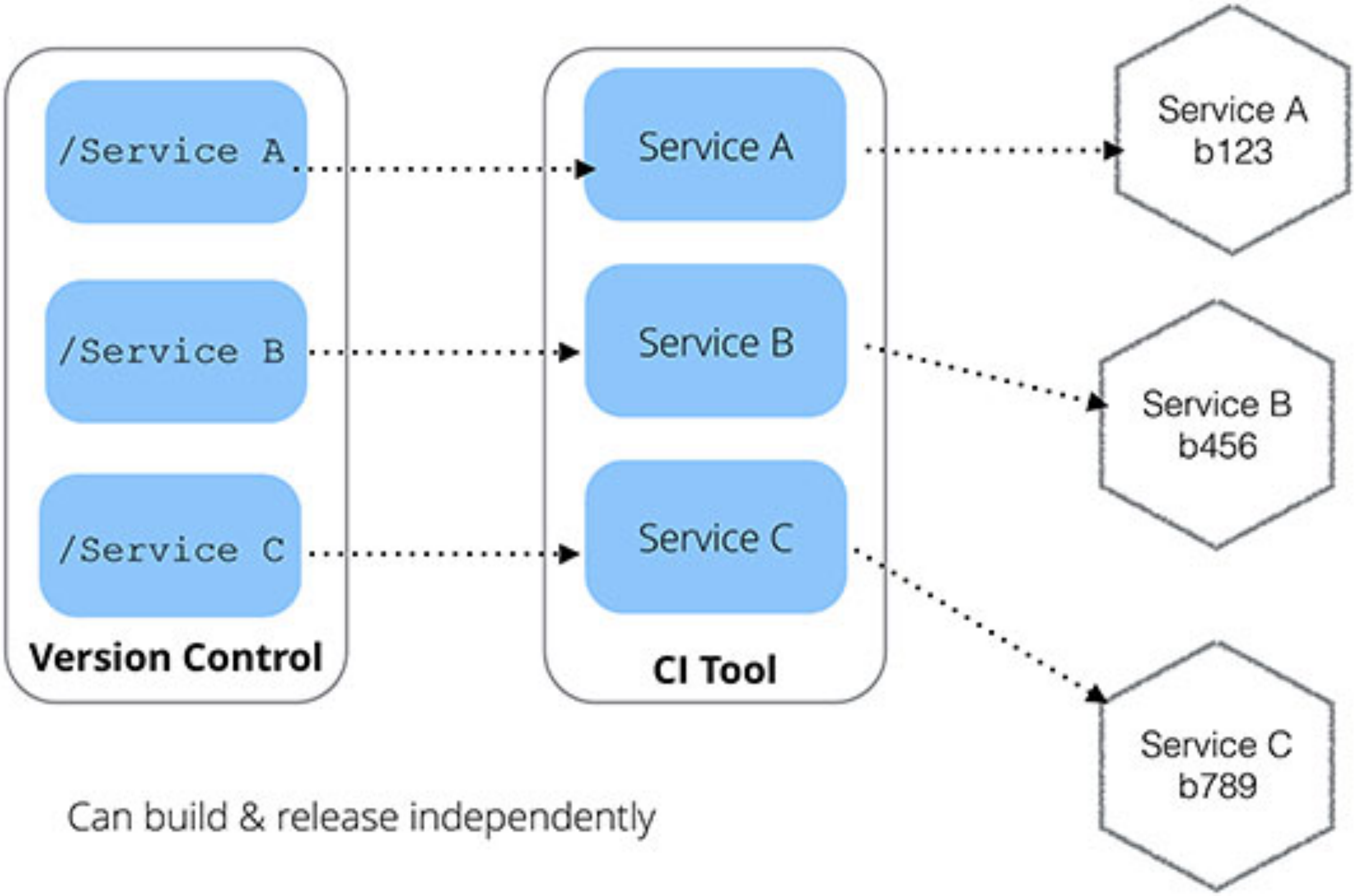
ONE BUILD & REPO PER SERVICE



ONE BUILD & REPO PER SERVICE

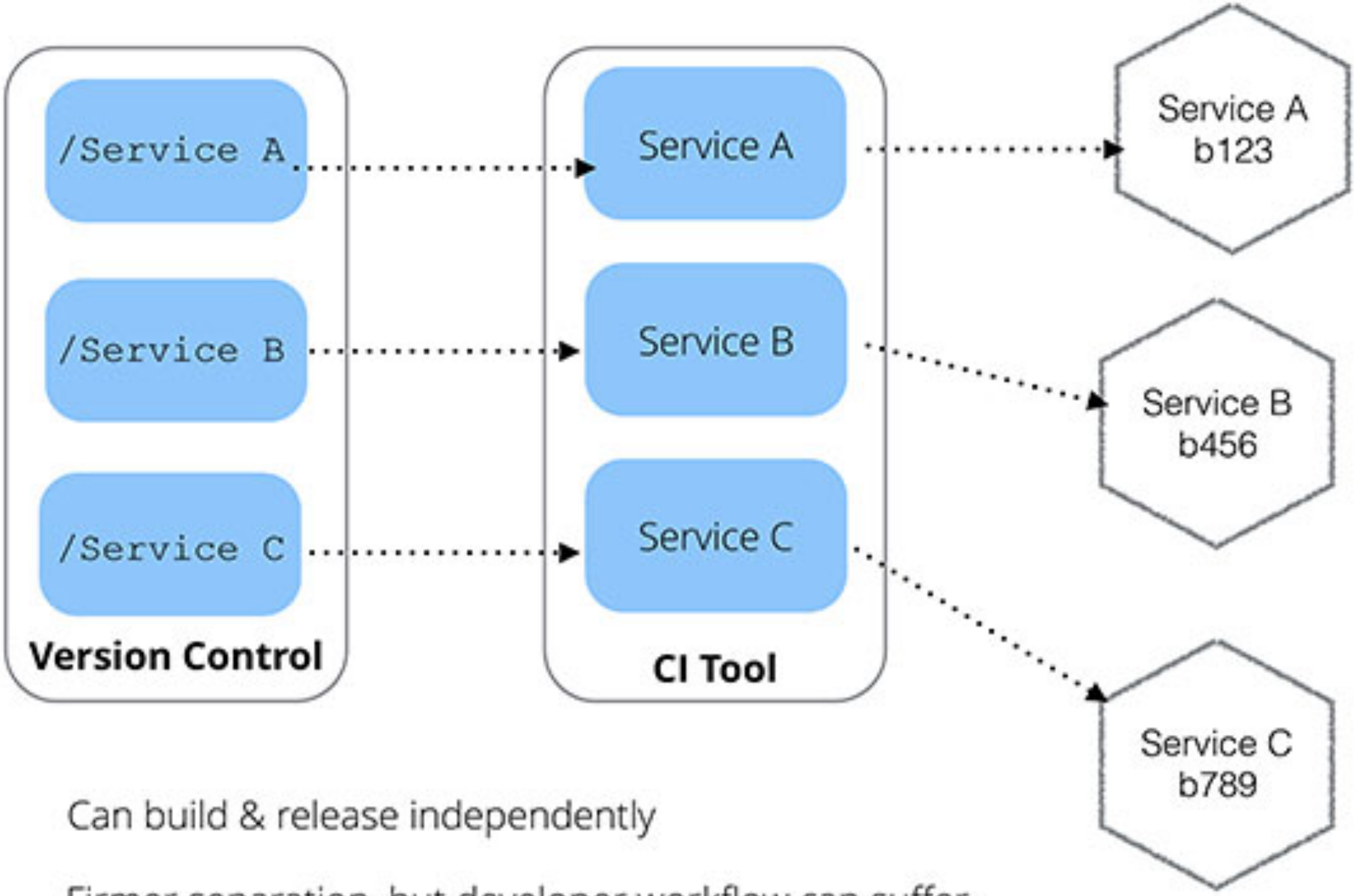


ONE BUILD & REPO PER SERVICE



Can build & release independently

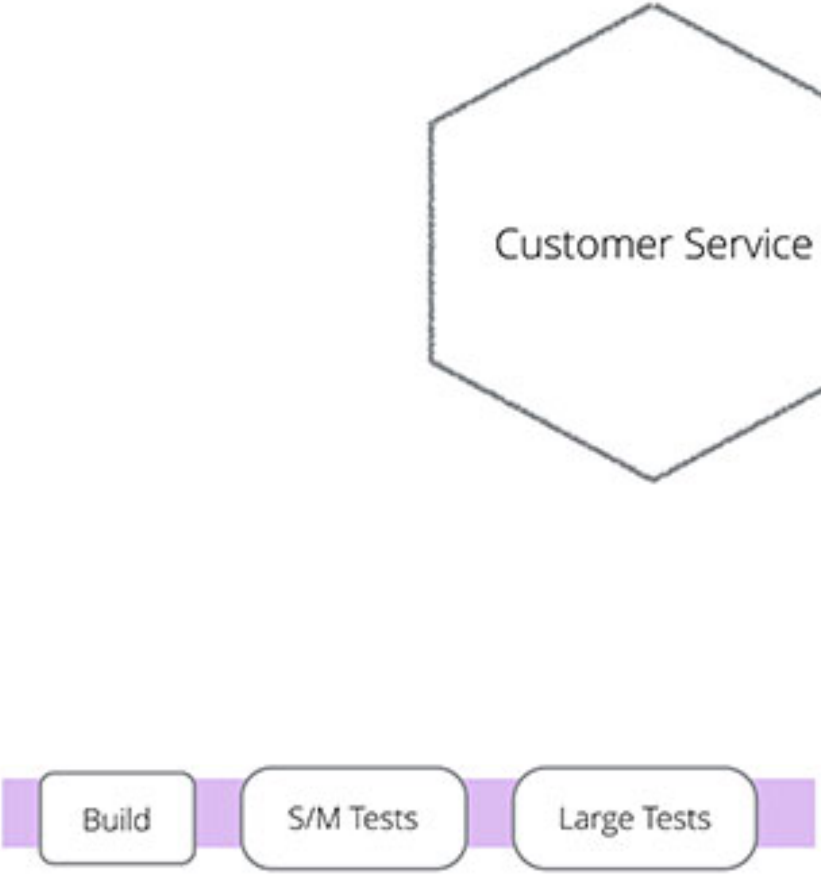
ONE BUILD & REPO PER SERVICE



Can build & release independently

Firmer separation, but developer workflow can suffer

Customer Service

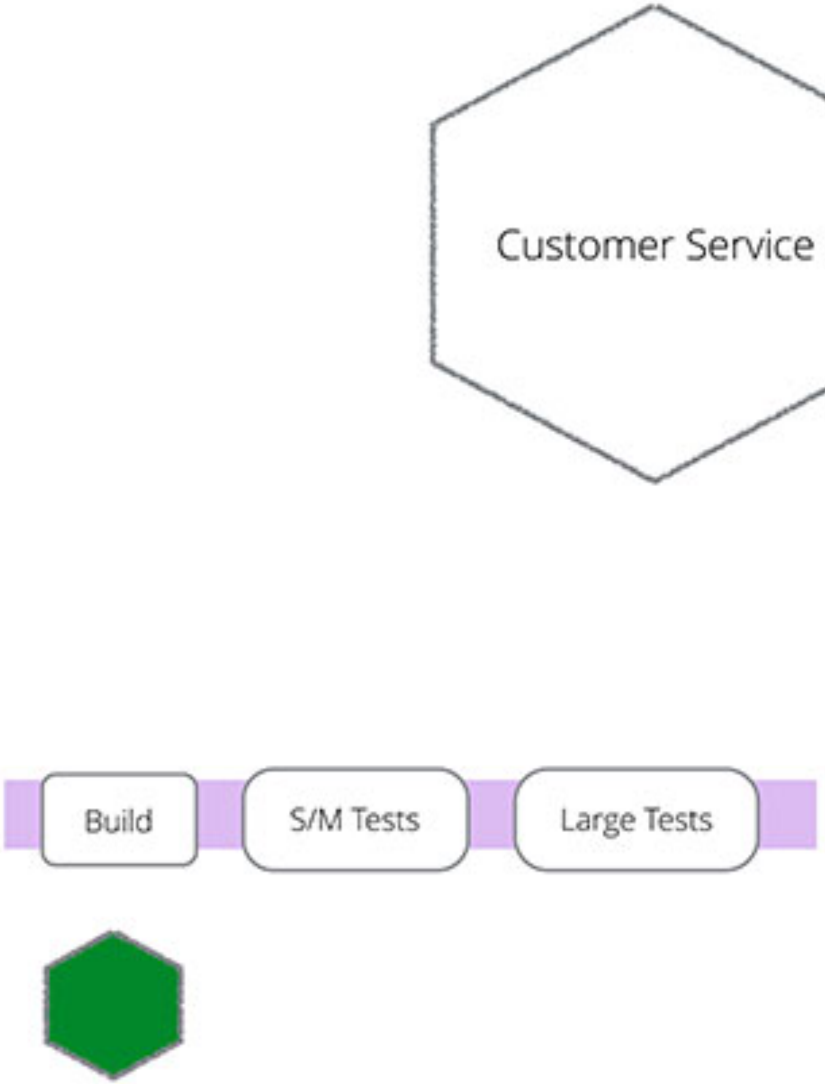


Customer Service

Build

S/M Tests

Large Tests

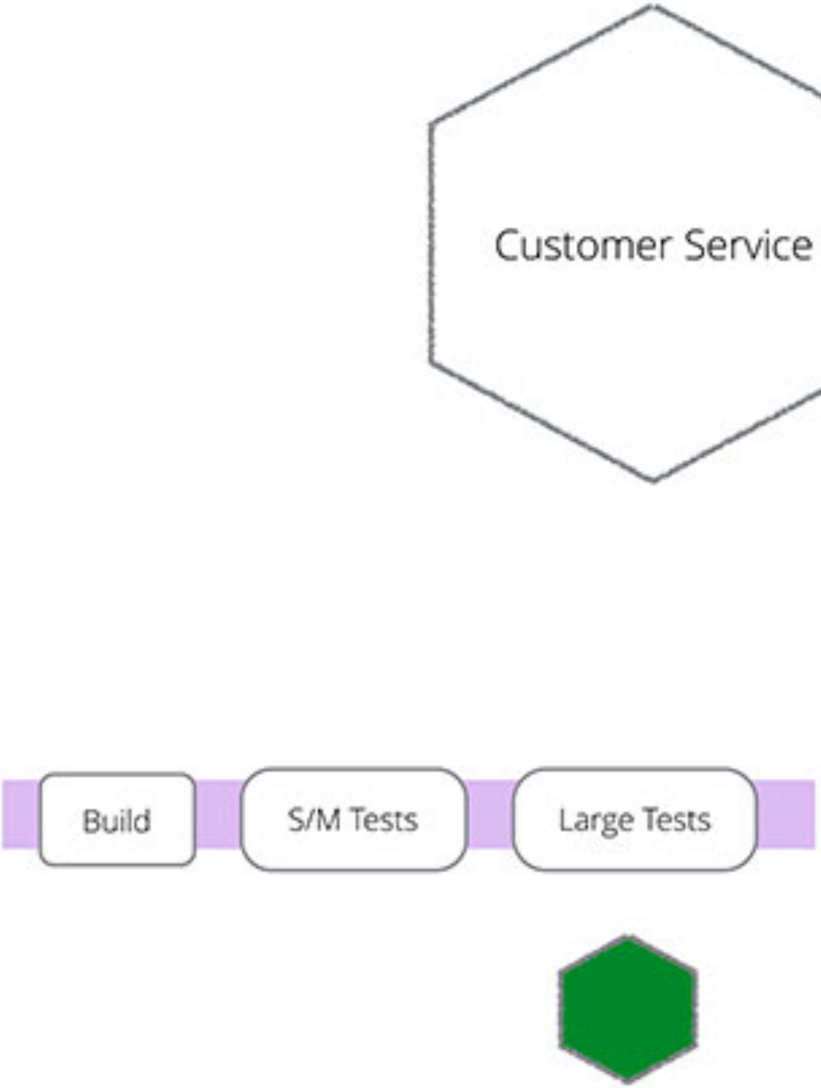


Customer Service

Build

S/M Tests

Large Tests

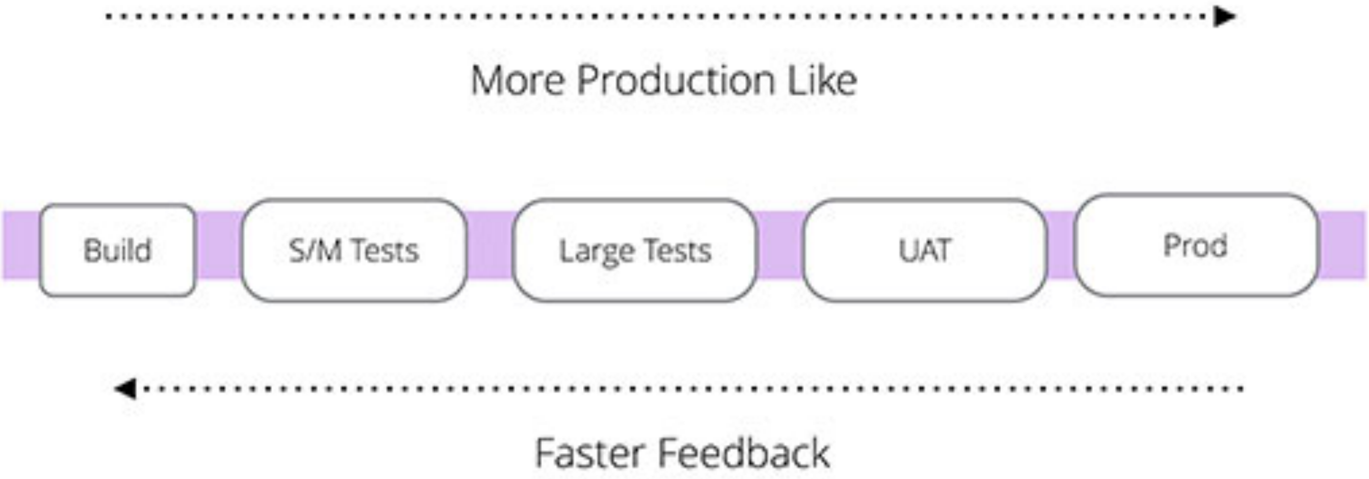


Customer Service

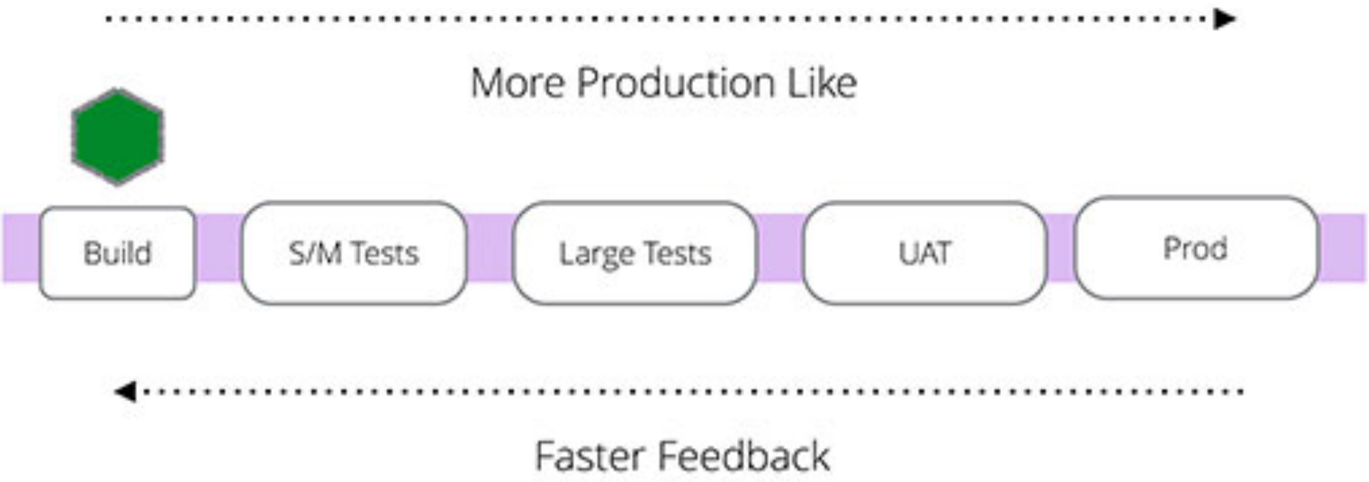
Build

S/M Tests

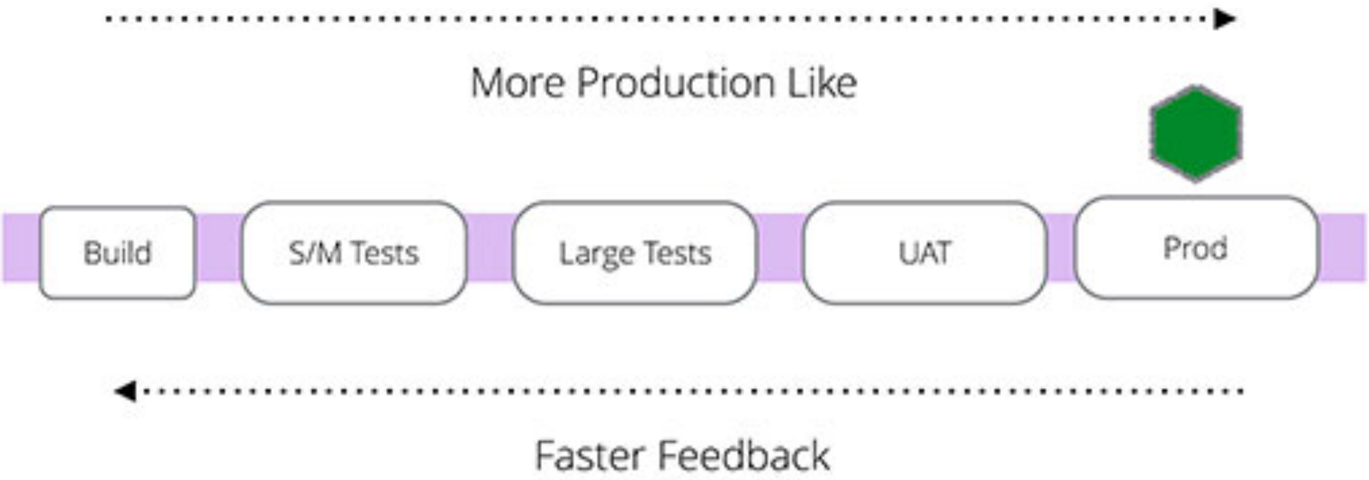
Large Tests



Artifacts should be built once, and moved through environments

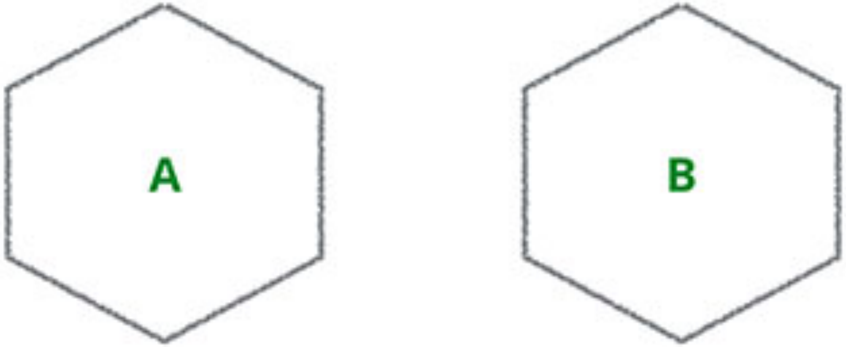


Artifacts should be built once, and moved through environments



Artifacts should be built once, and moved through environments

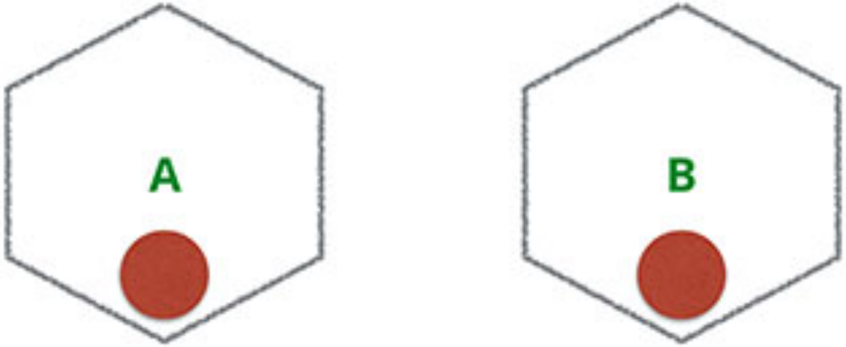
SHARED DEPENDENCIES



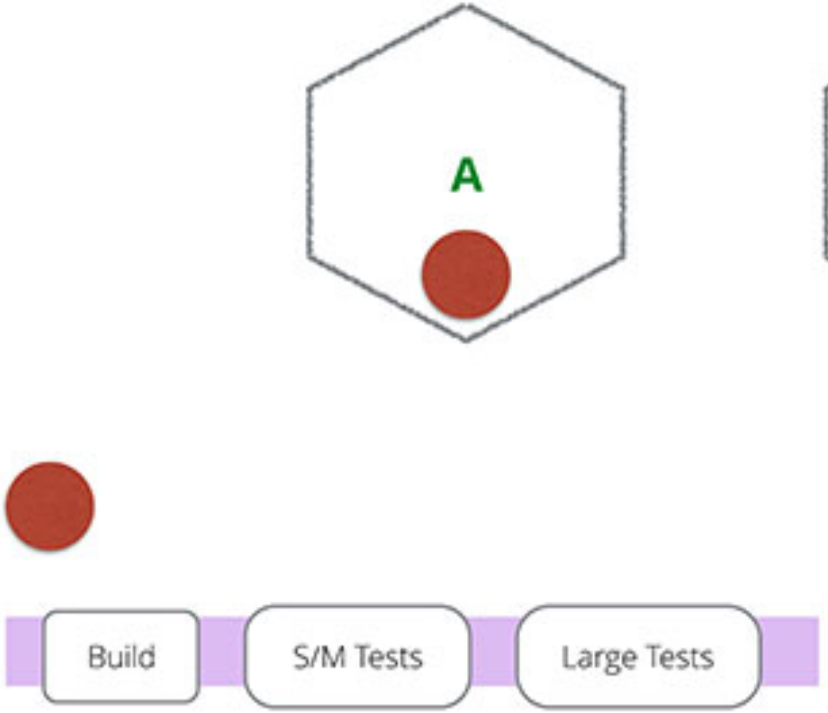
A

B

SHARED DEPENDENCIES



SHARED DEPENDENCIES



A

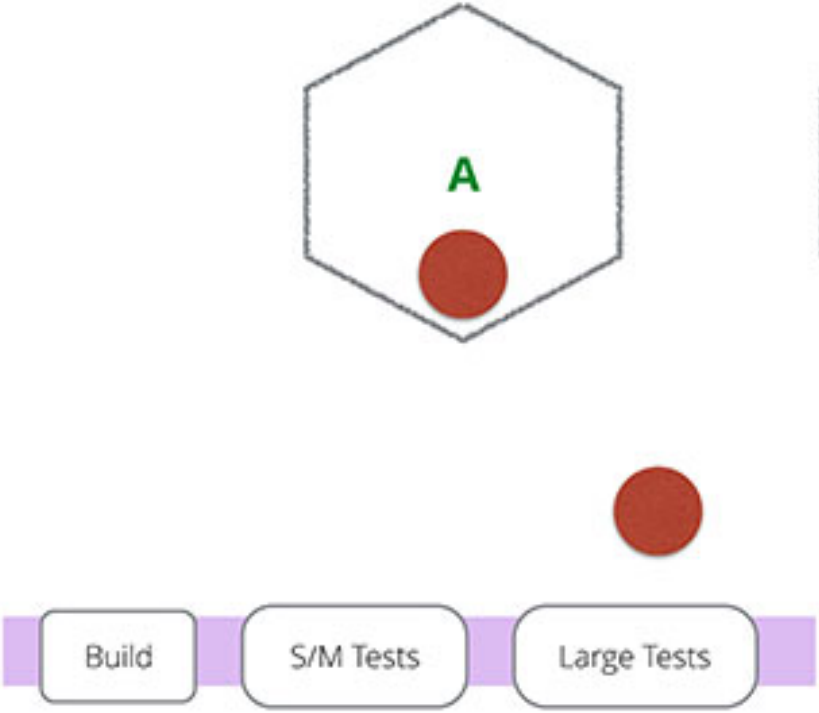
B

Build

S/M Tests

Large Tests

SHARED DEPENDENCIES



FIXED DEPENDENCIES

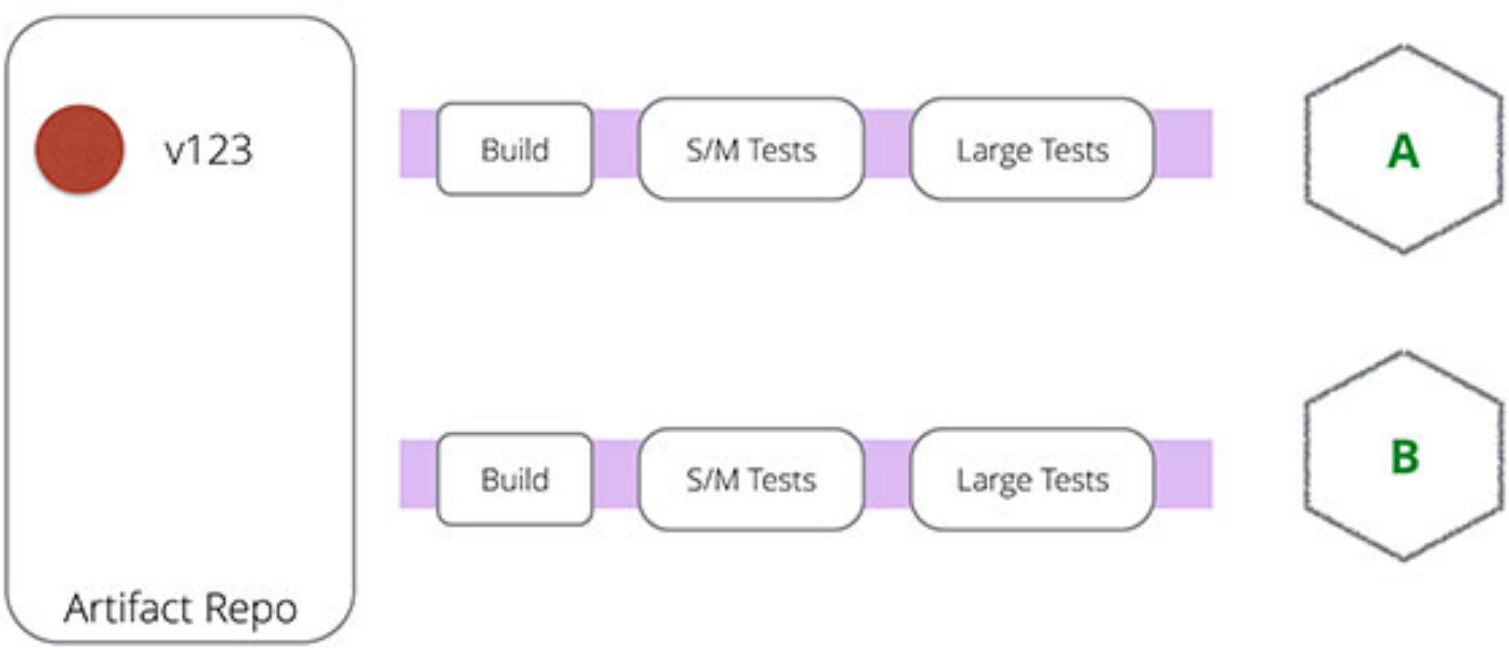
v123

Artifact Repo

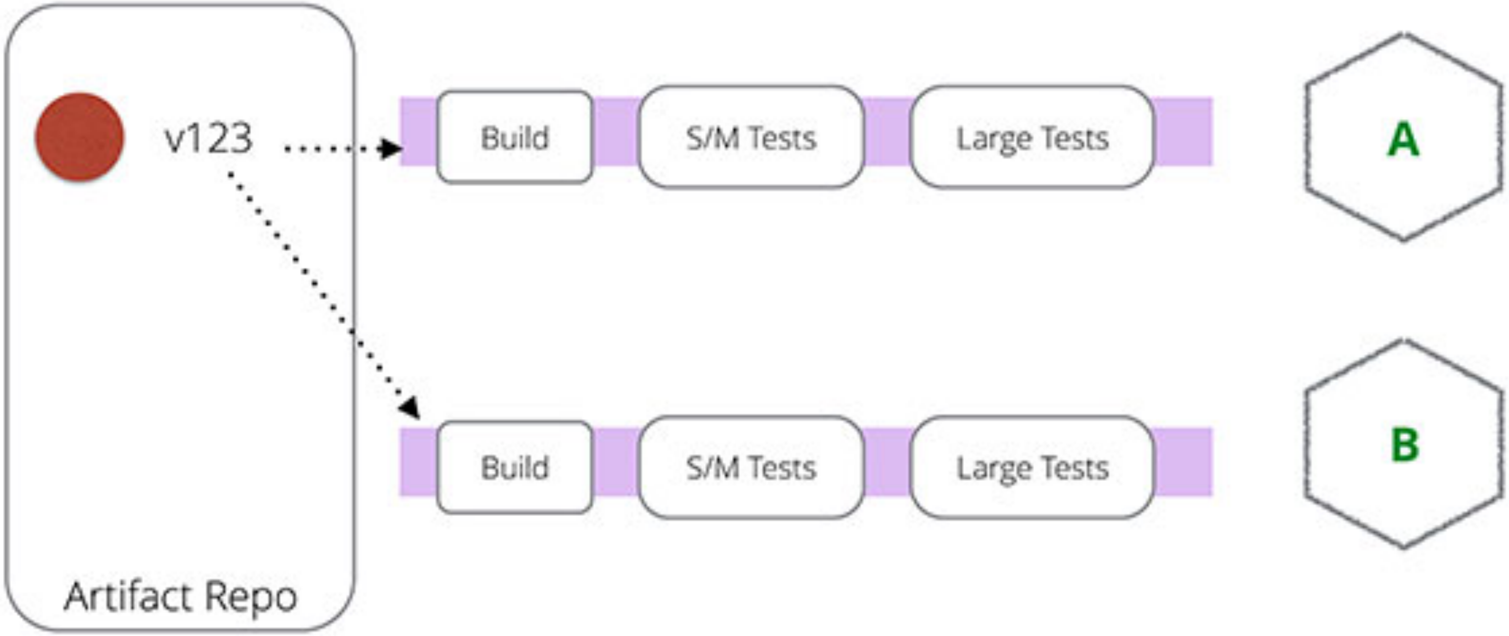
A

B

FIXED DEPENDENCIES



FIXED DEPENDENCIES



v123

Build

S/M Tests

Large Tests

A

Build

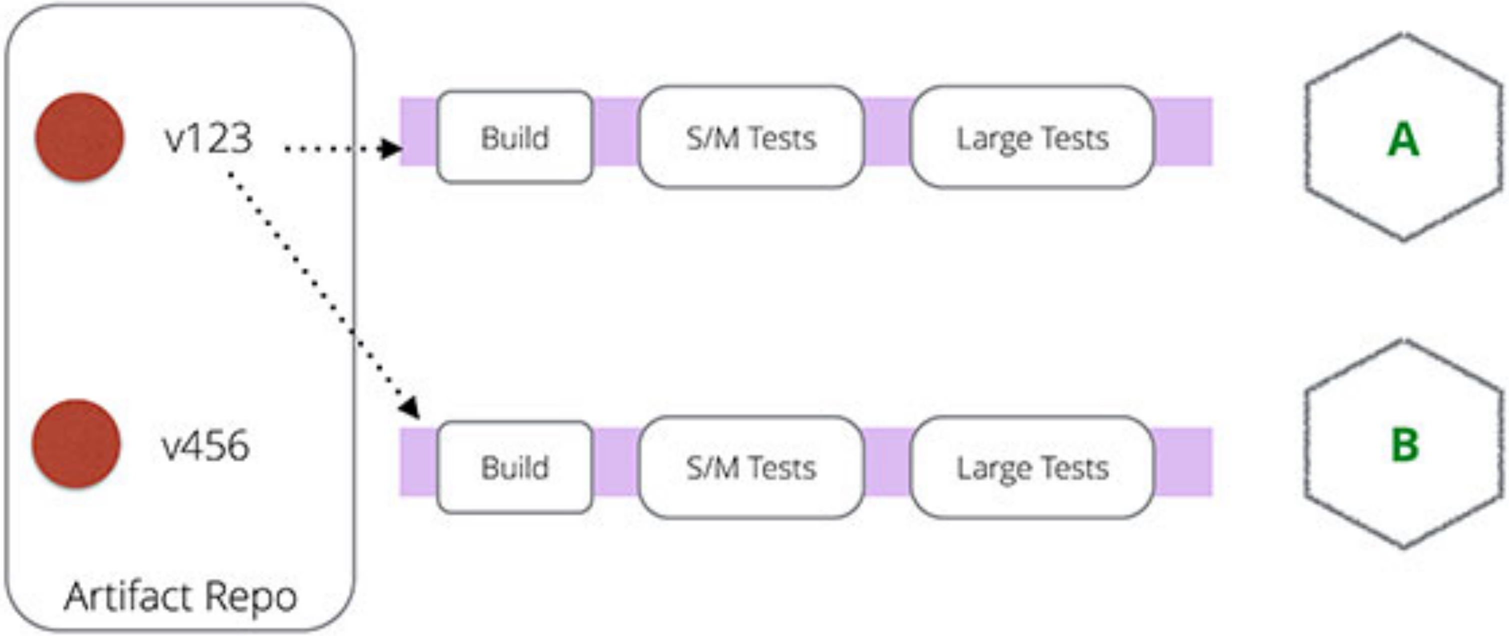
S/M Tests

Large Tests

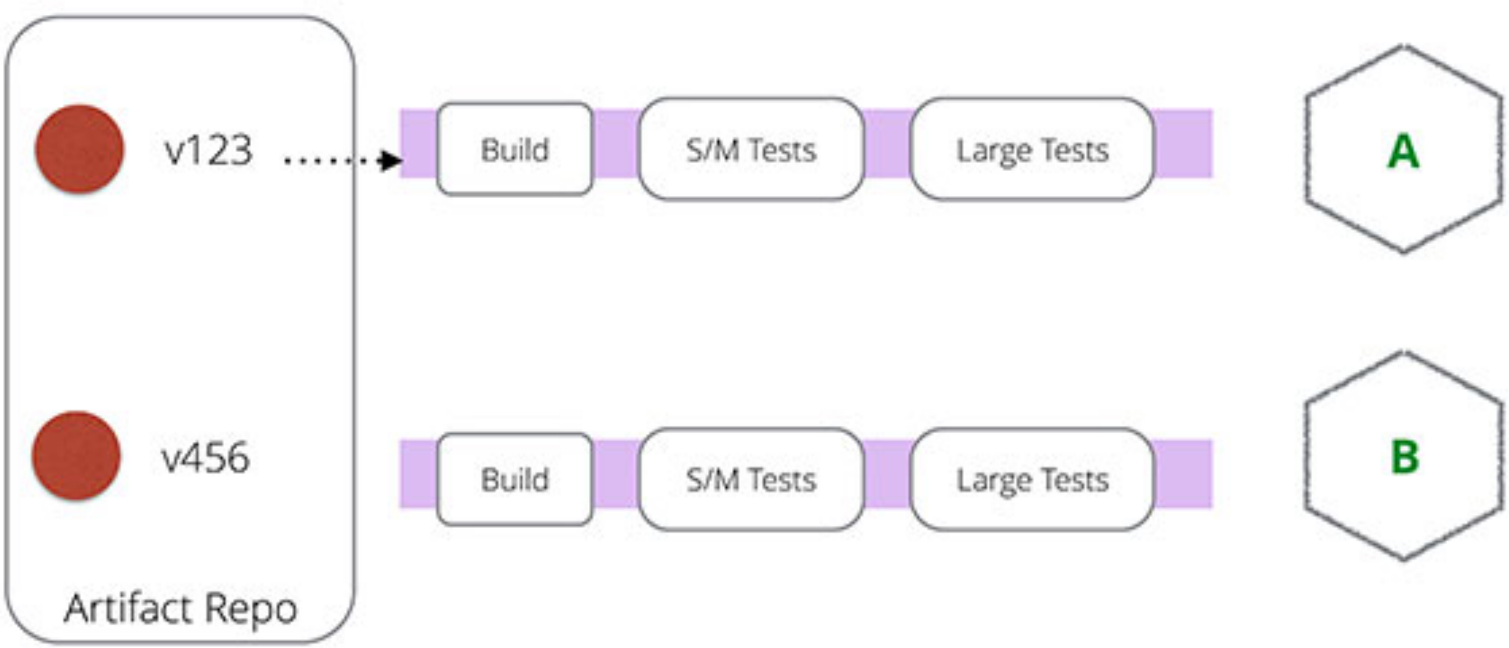
B

Artifact Repo

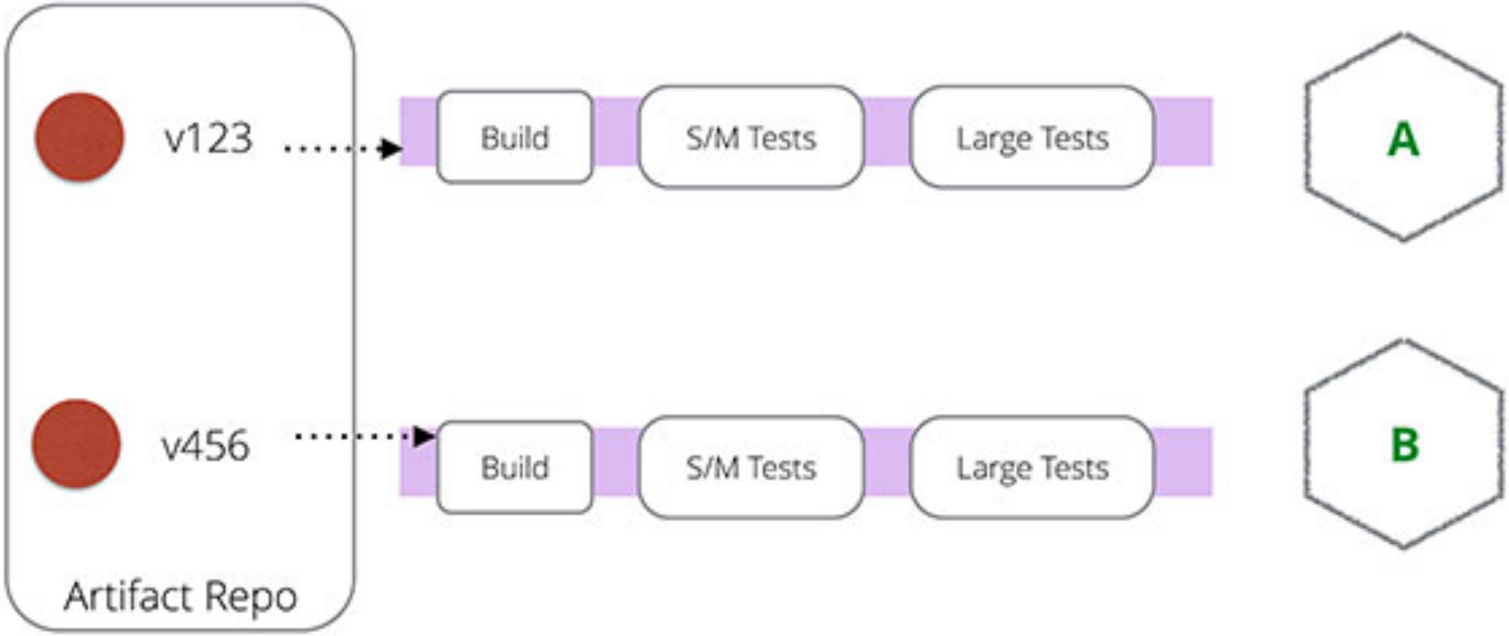
FIXED DEPENDENCIES



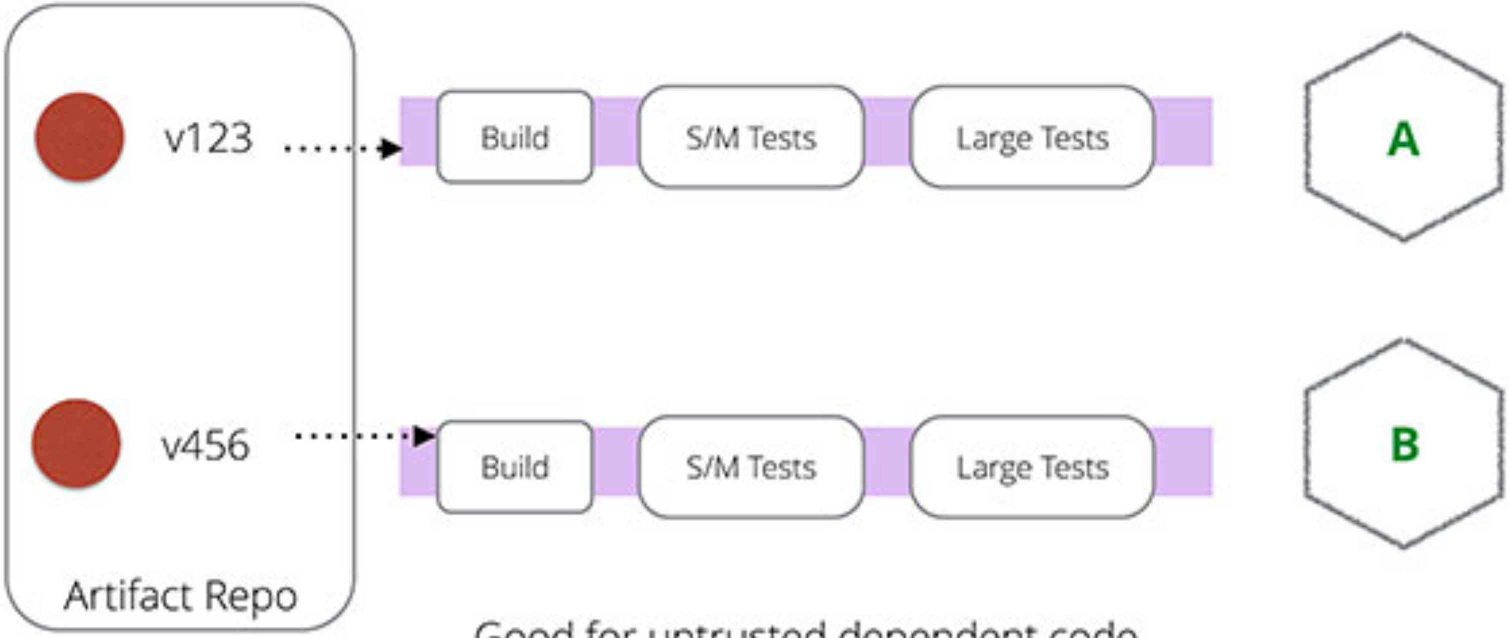
FIXED DEPENDENCIES



FIXED DEPENDENCIES

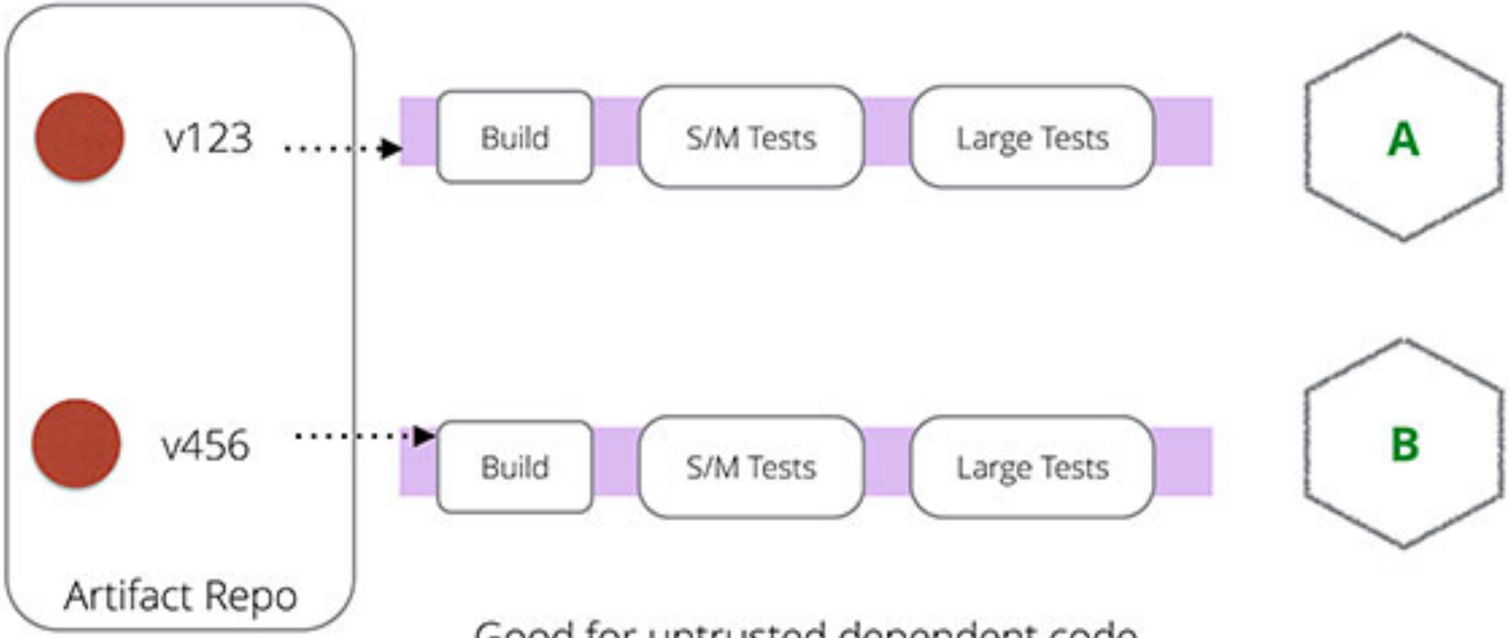


FIXED DEPENDENCIES



Good for untrusted dependent code

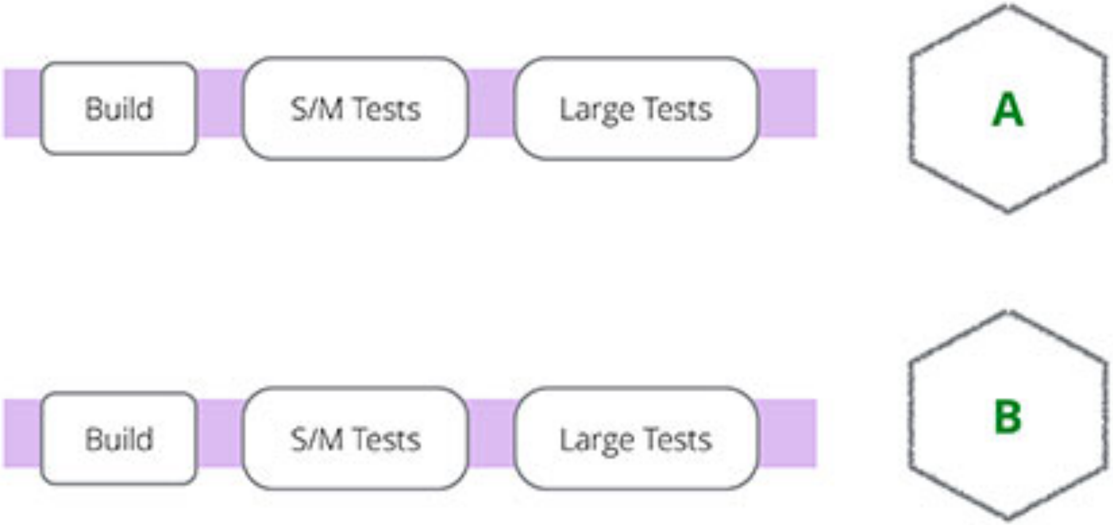
FIXED DEPENDENCIES



Good for untrusted dependent code

And for avoiding hyperactive builds

FLUID DEPENDENCIES



Build

S/M Tests

Large Tests

A

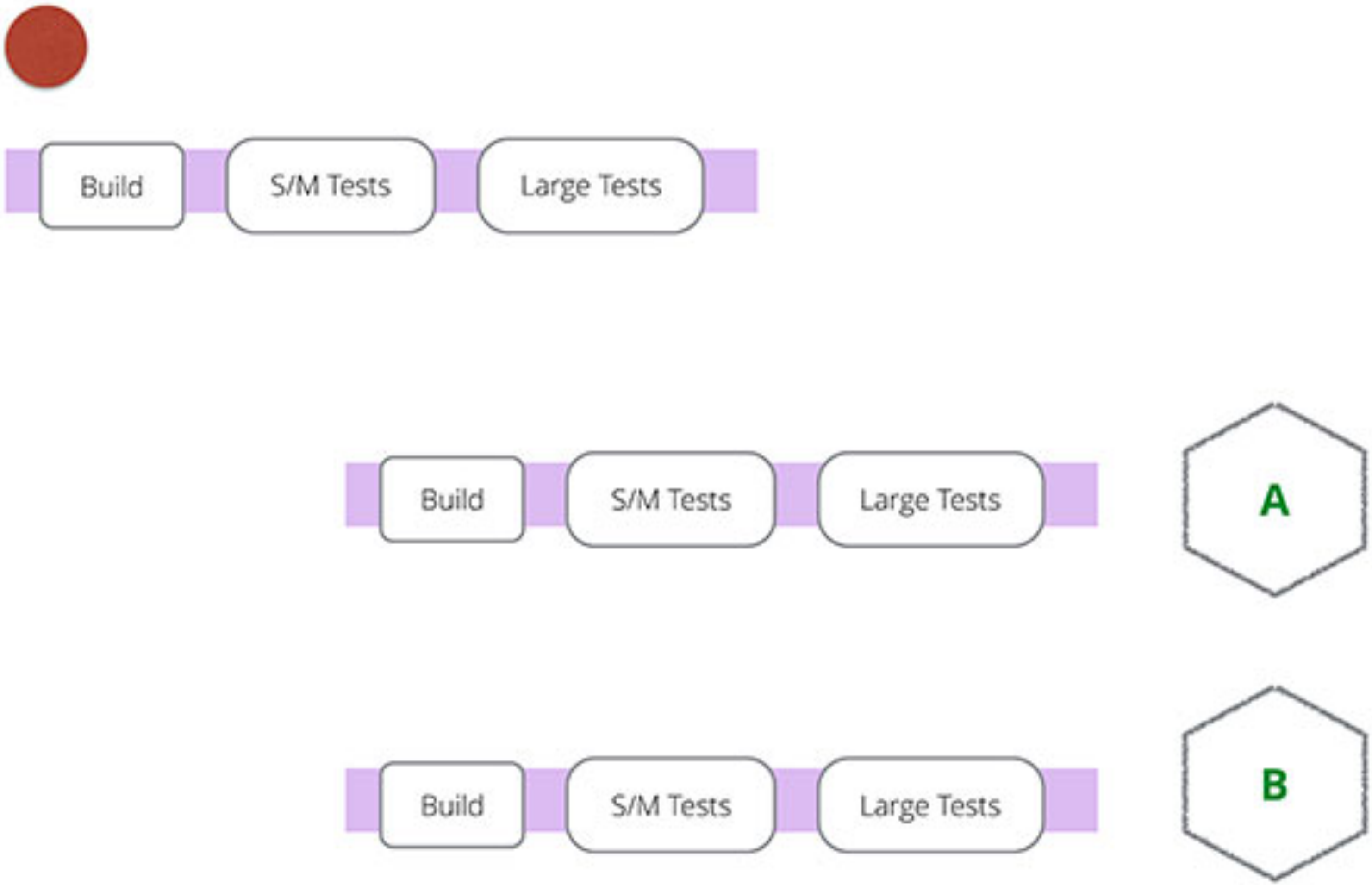
Build

S/M Tests

Large Tests

B

FLUID DEPENDENCIES



Build

S/M Tests

Large Tests

Build

S/M Tests

Large Tests

A

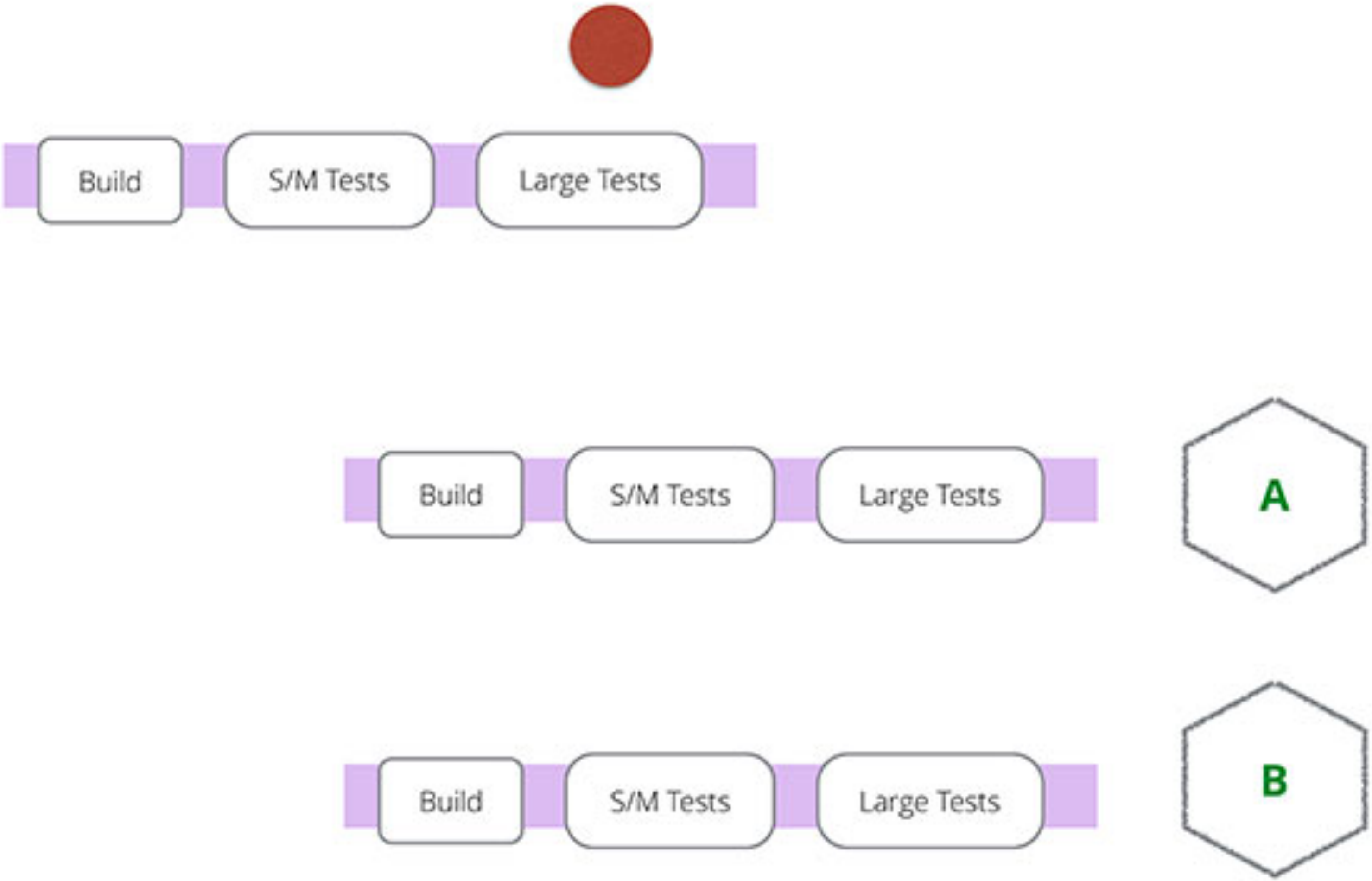
Build

S/M Tests

Large Tests

B

FLUID DEPENDENCIES



Build

S/M Tests

Large Tests

Build

S/M Tests

Large Tests

A

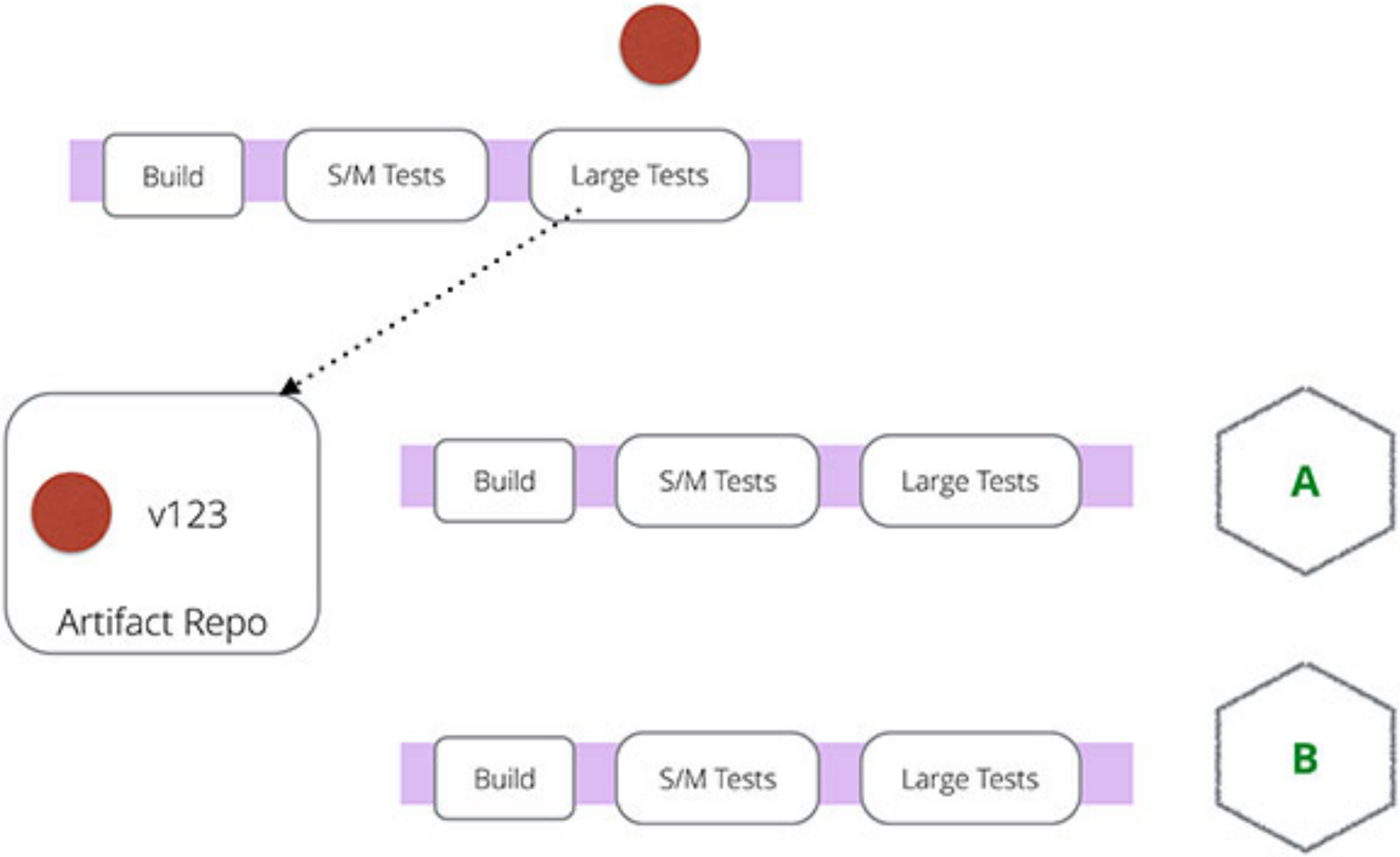
Build

S/M Tests

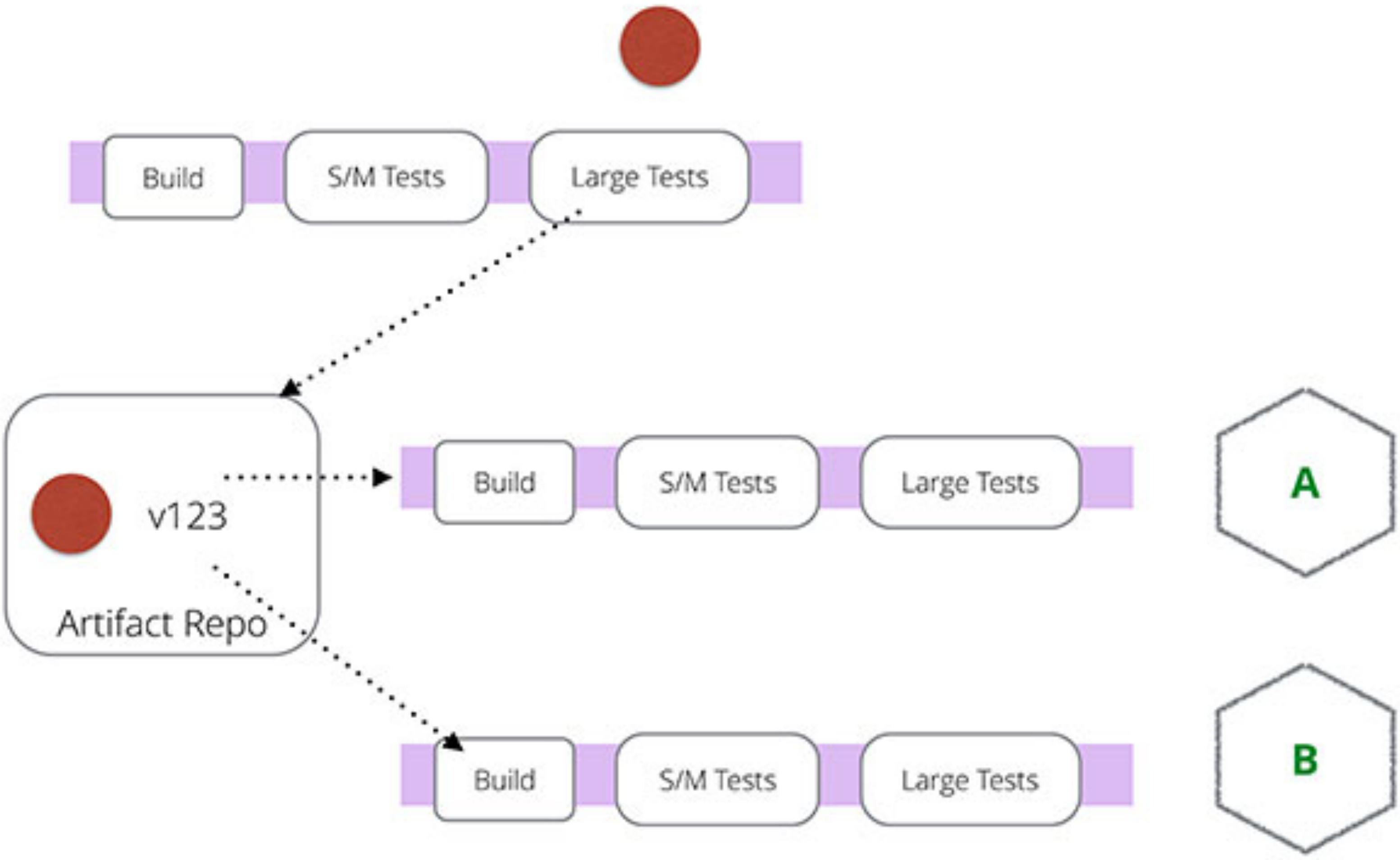
Large Tests

B

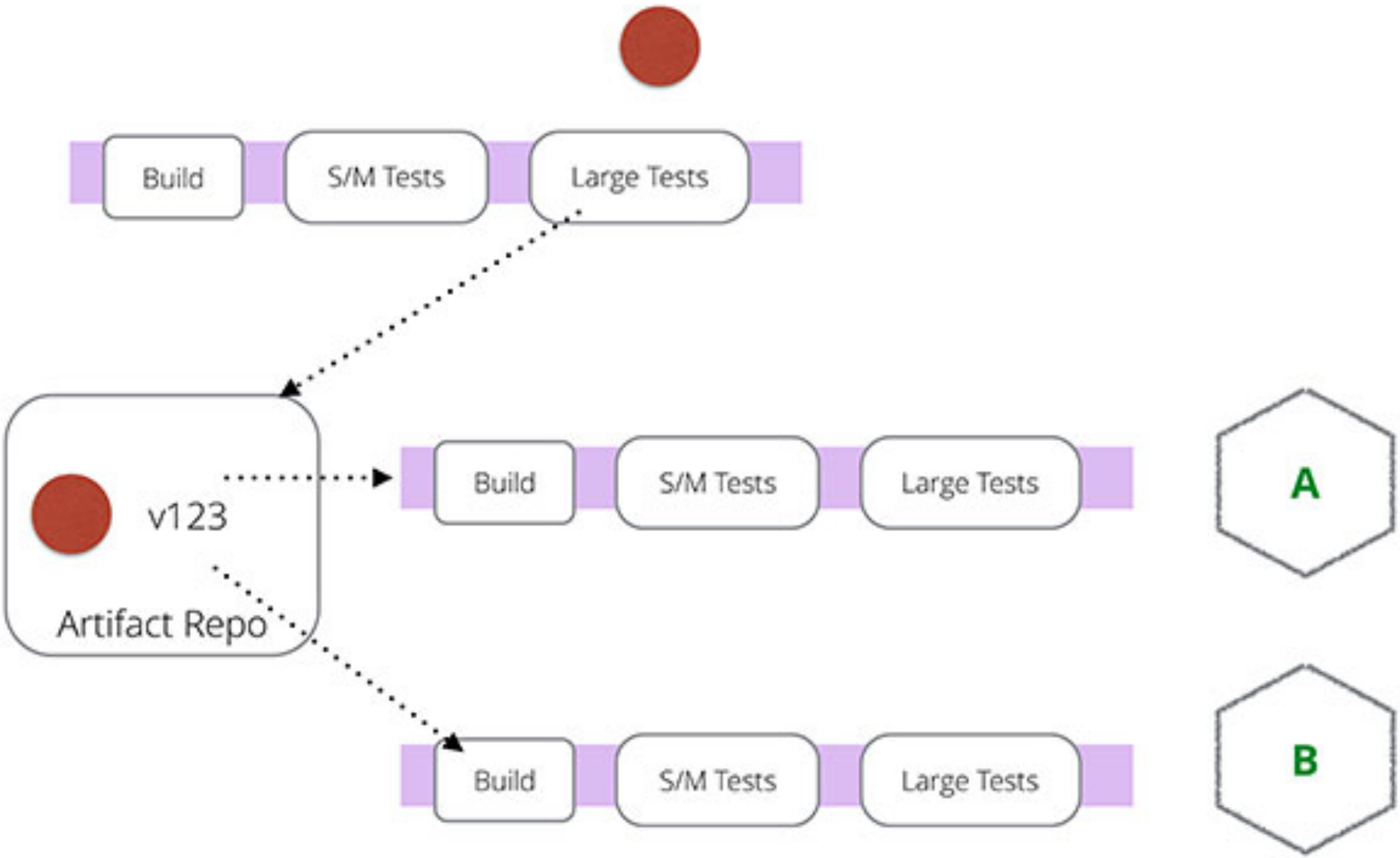
FLUID DEPENDENCIES



FLUID DEPENDENCIES

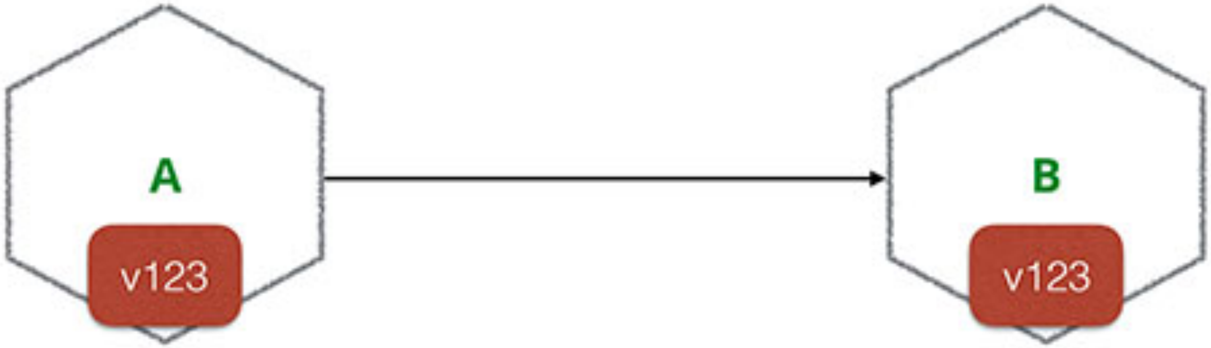


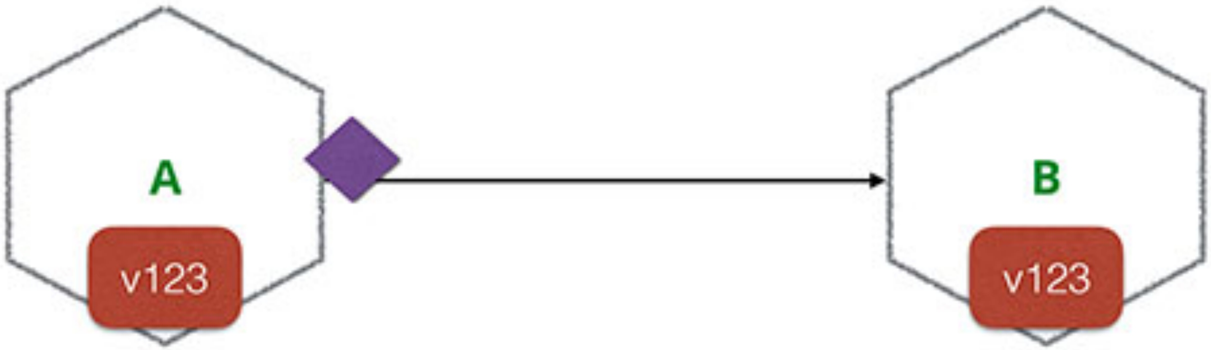
FLUID DEPENDENCIES

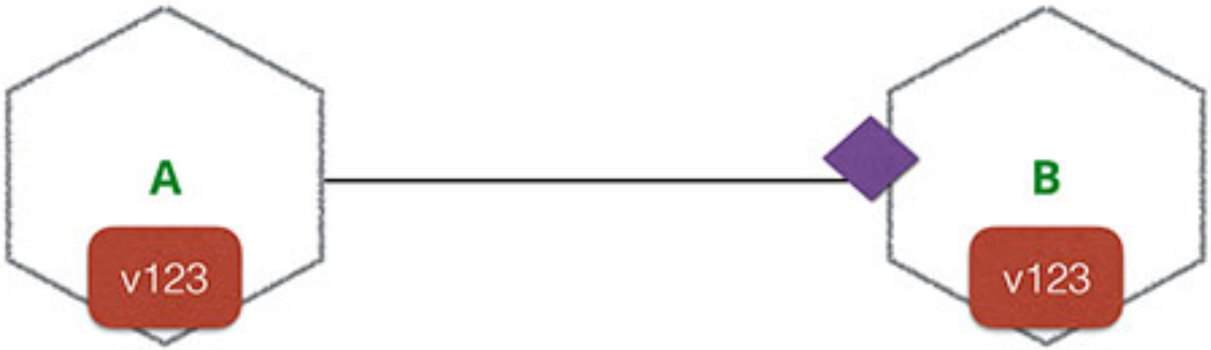


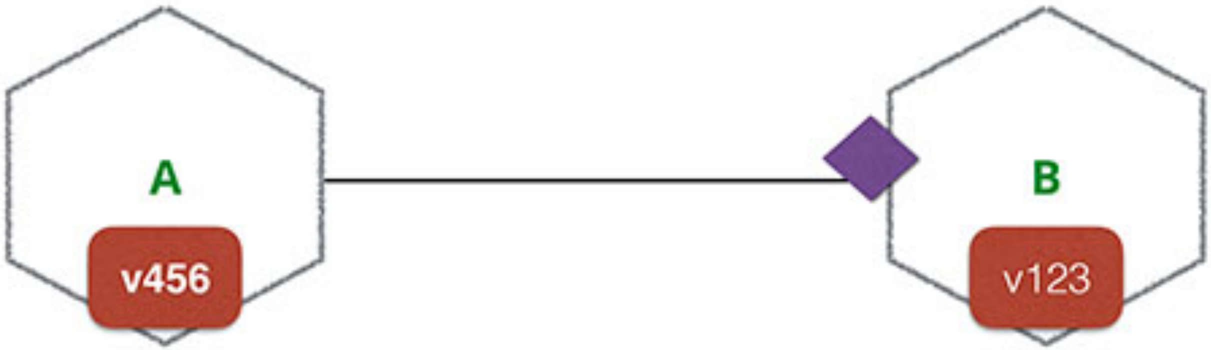
Great for continuously integrating changes

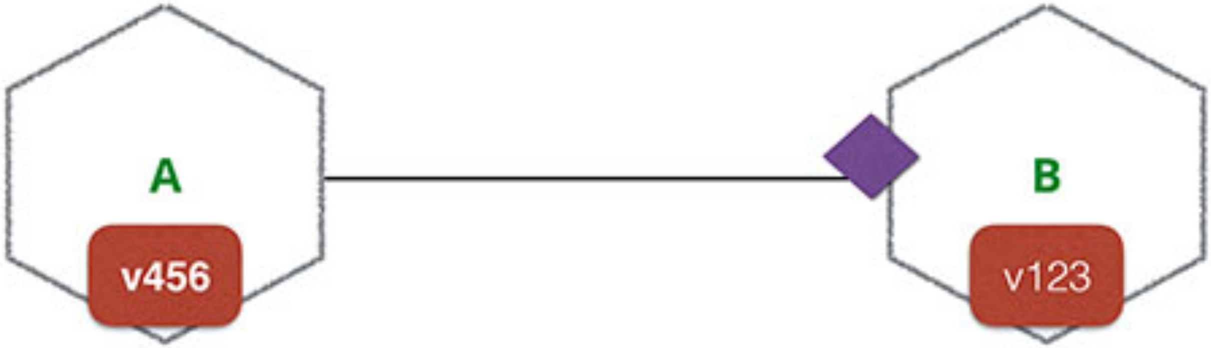
Shared Code?


A
v123
B
v123


A
v123
B
v123


A
v123
B
v123


A
v456
B
v123



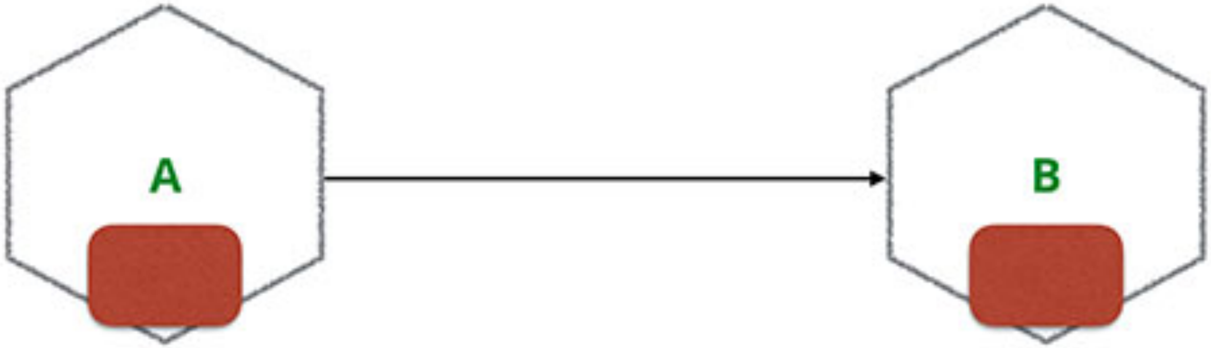
Can we still communicate?

Independent Deployability

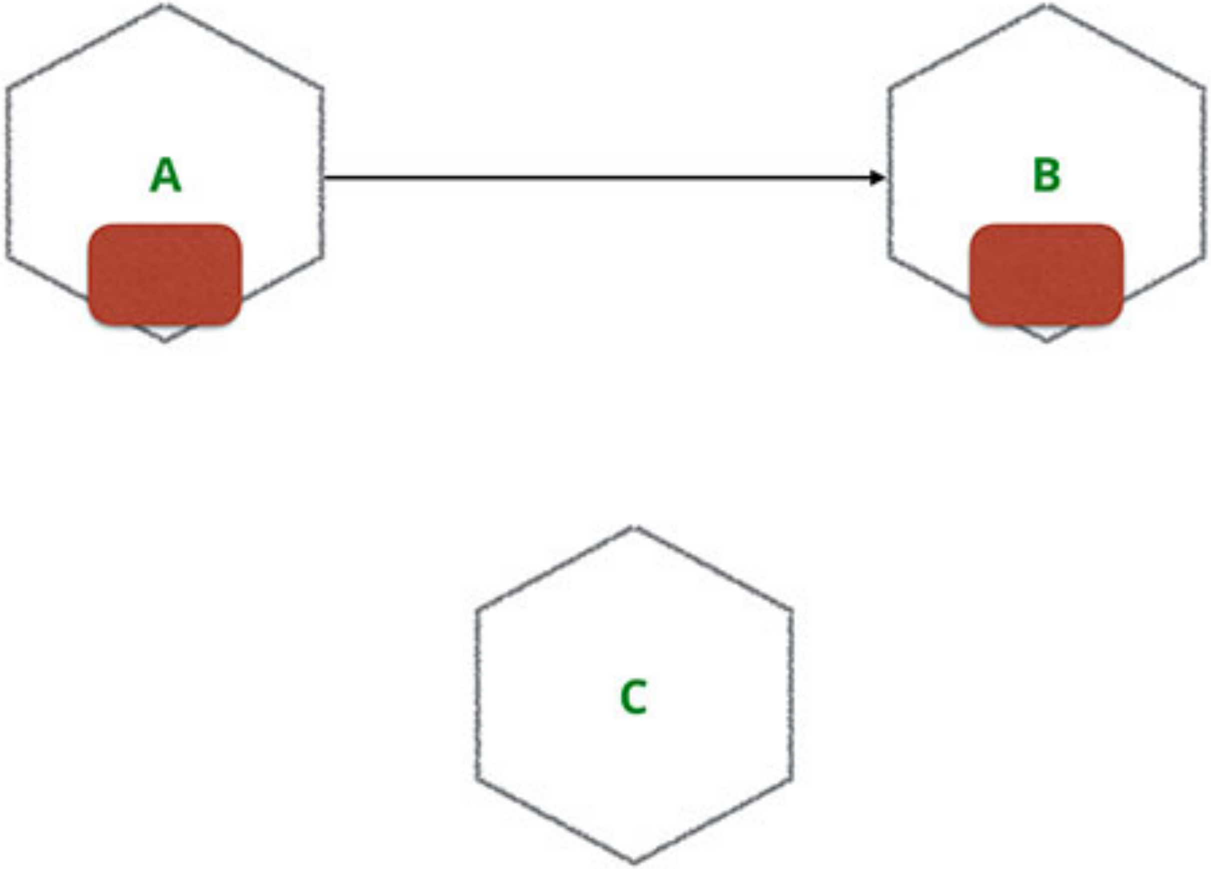
Independent Deployability

Avoid shared code if it leads to the need for lock-step release

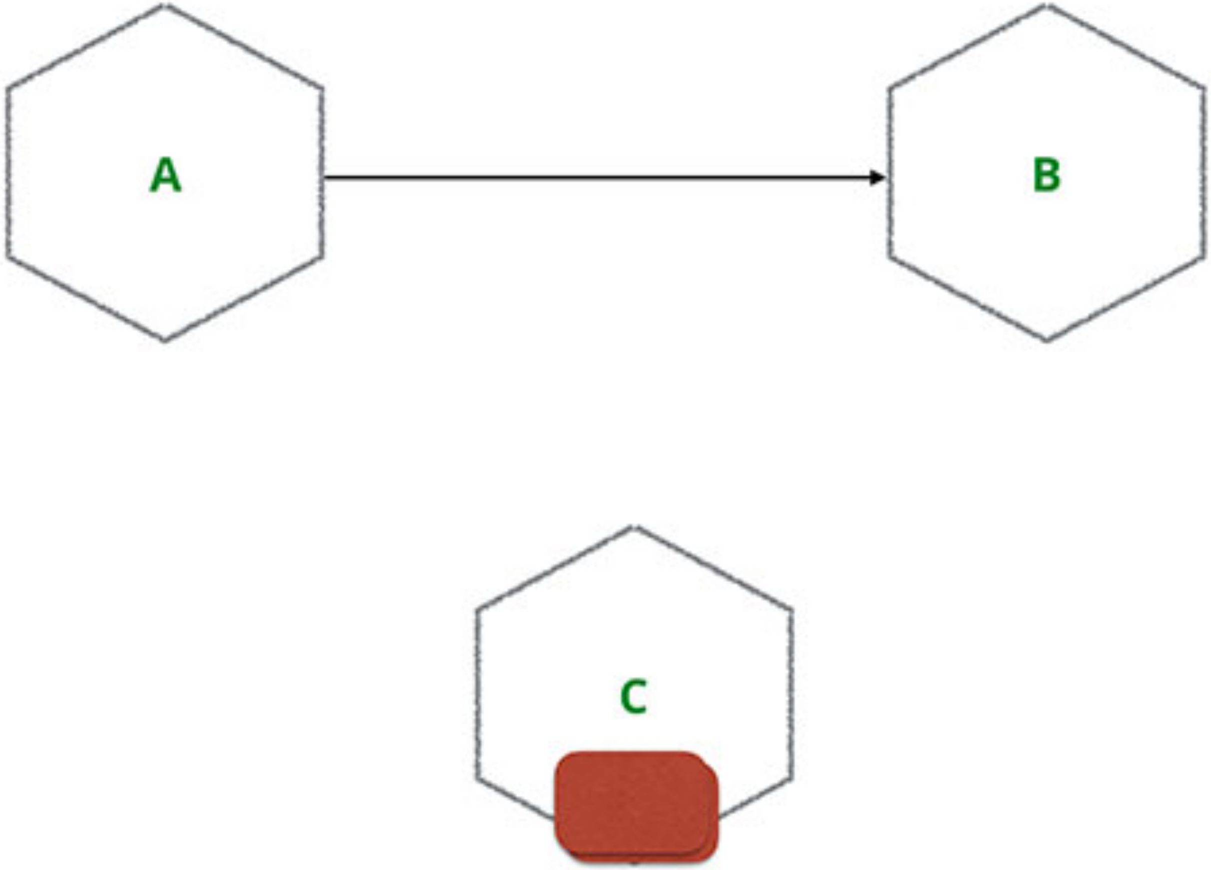
Consider service reuse rather than library reuse



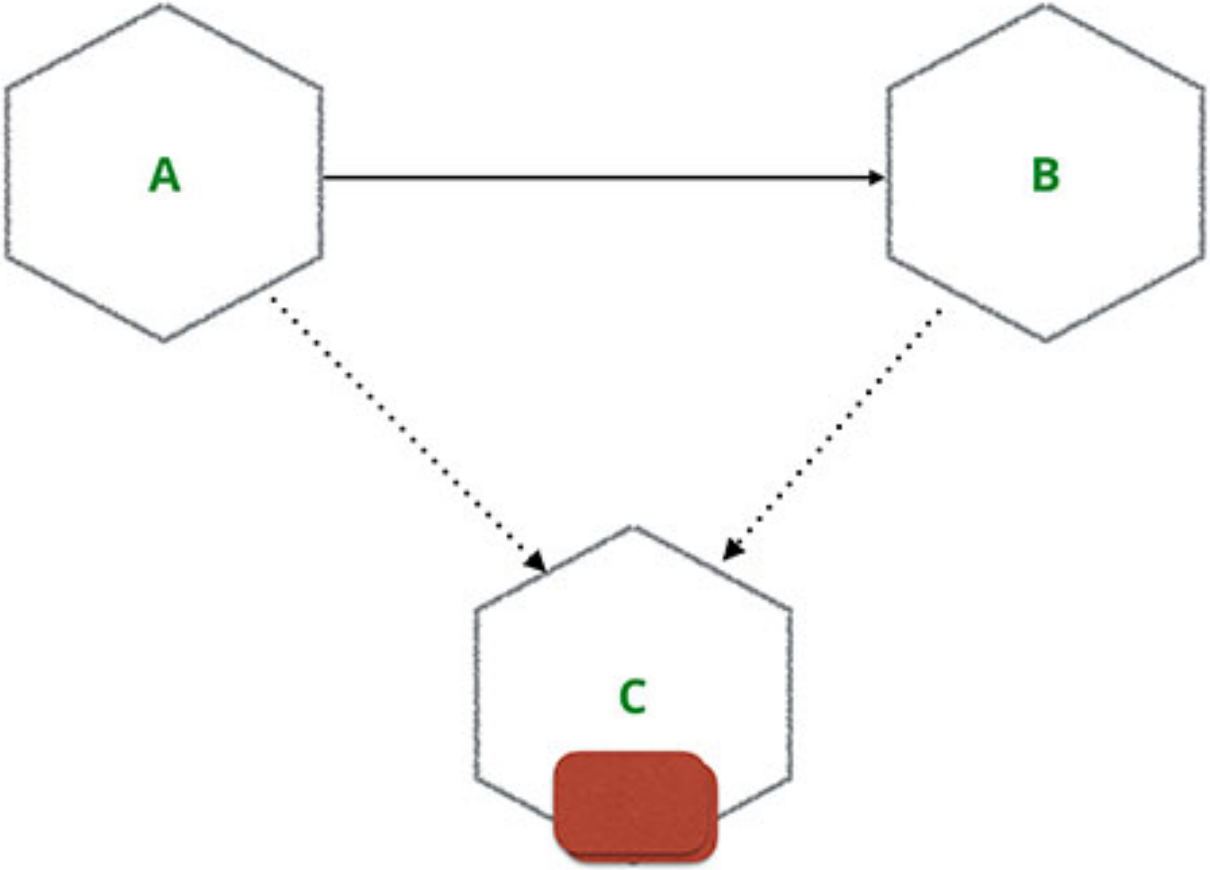
Consider service reuse rather than library reuse

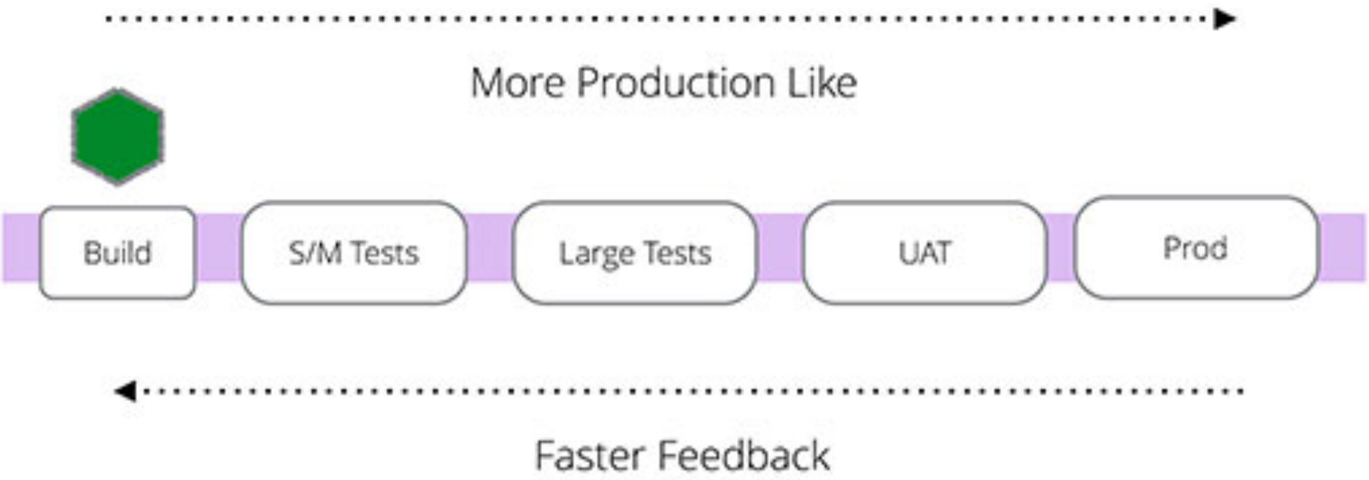


Consider service reuse rather than library reuse



Consider service reuse rather than library reuse




More Production Like

Build

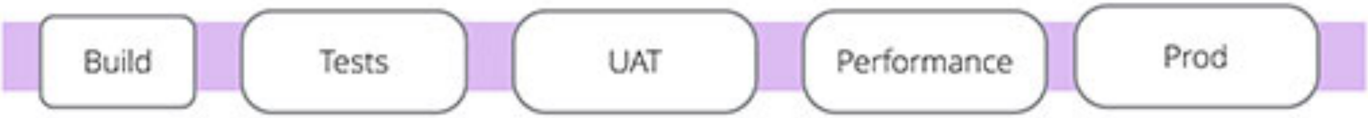
S/M Tests

Large Tests

UAT

Prod

Faster Feedback



Build

Tests

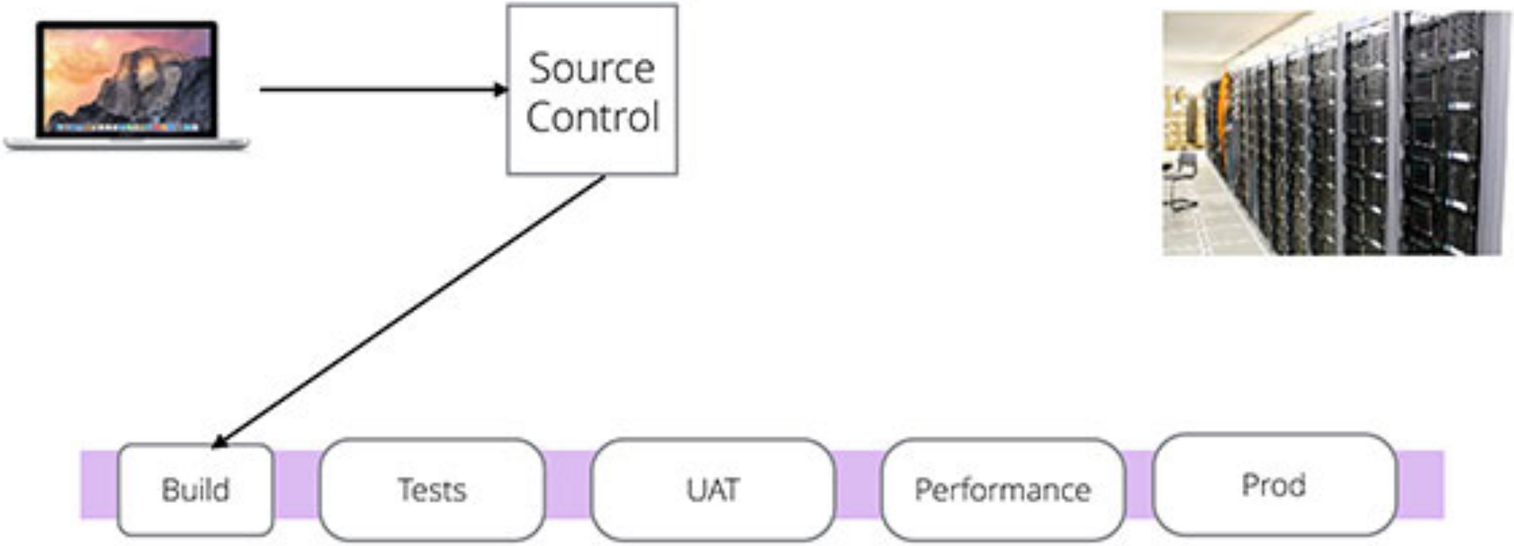
UAT

Performance

Prod

Source Control

Build → Tests → UAT → Performance → Prod



Source Control

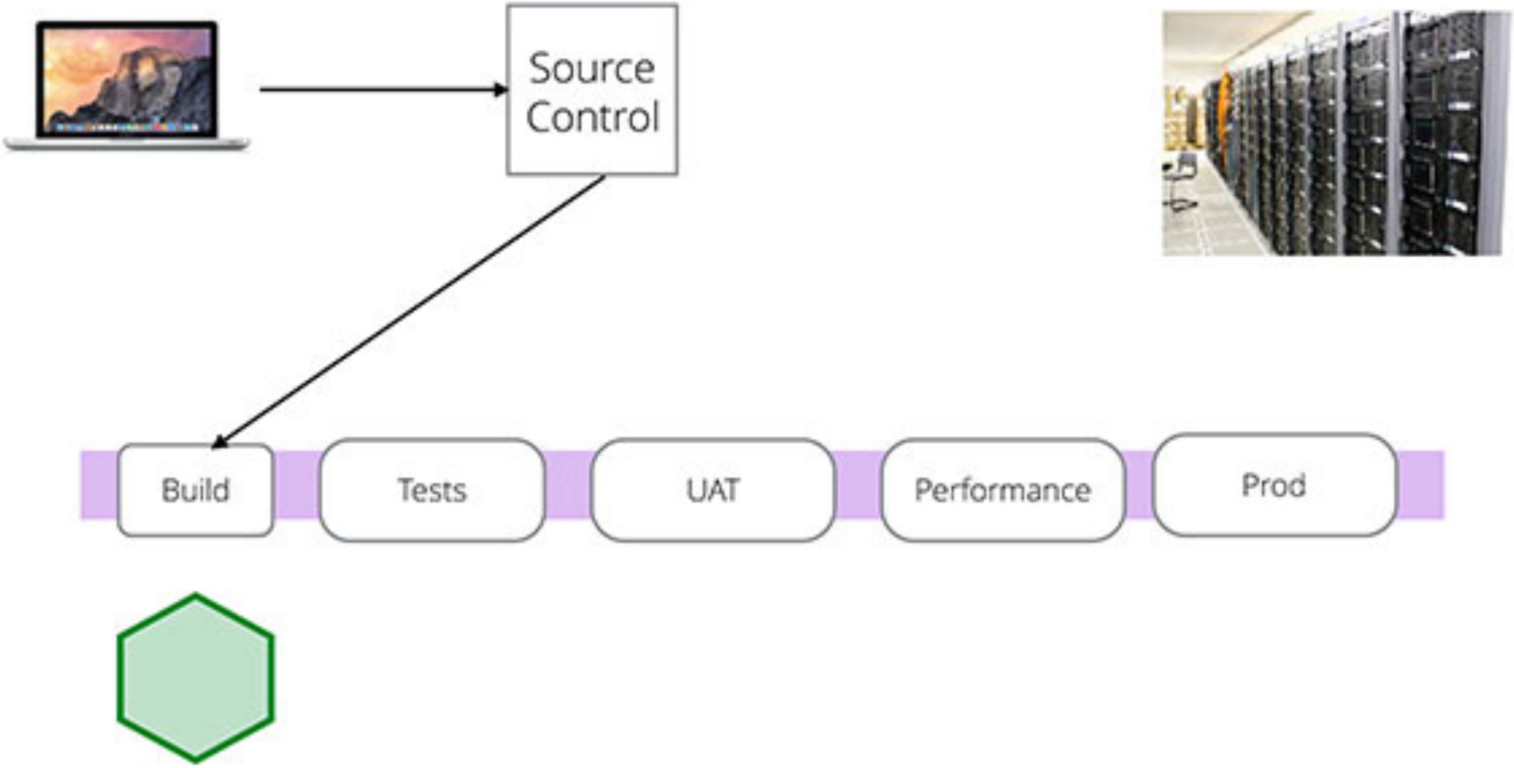
Build

Tests

UAT

Performance

Prod



Source Control

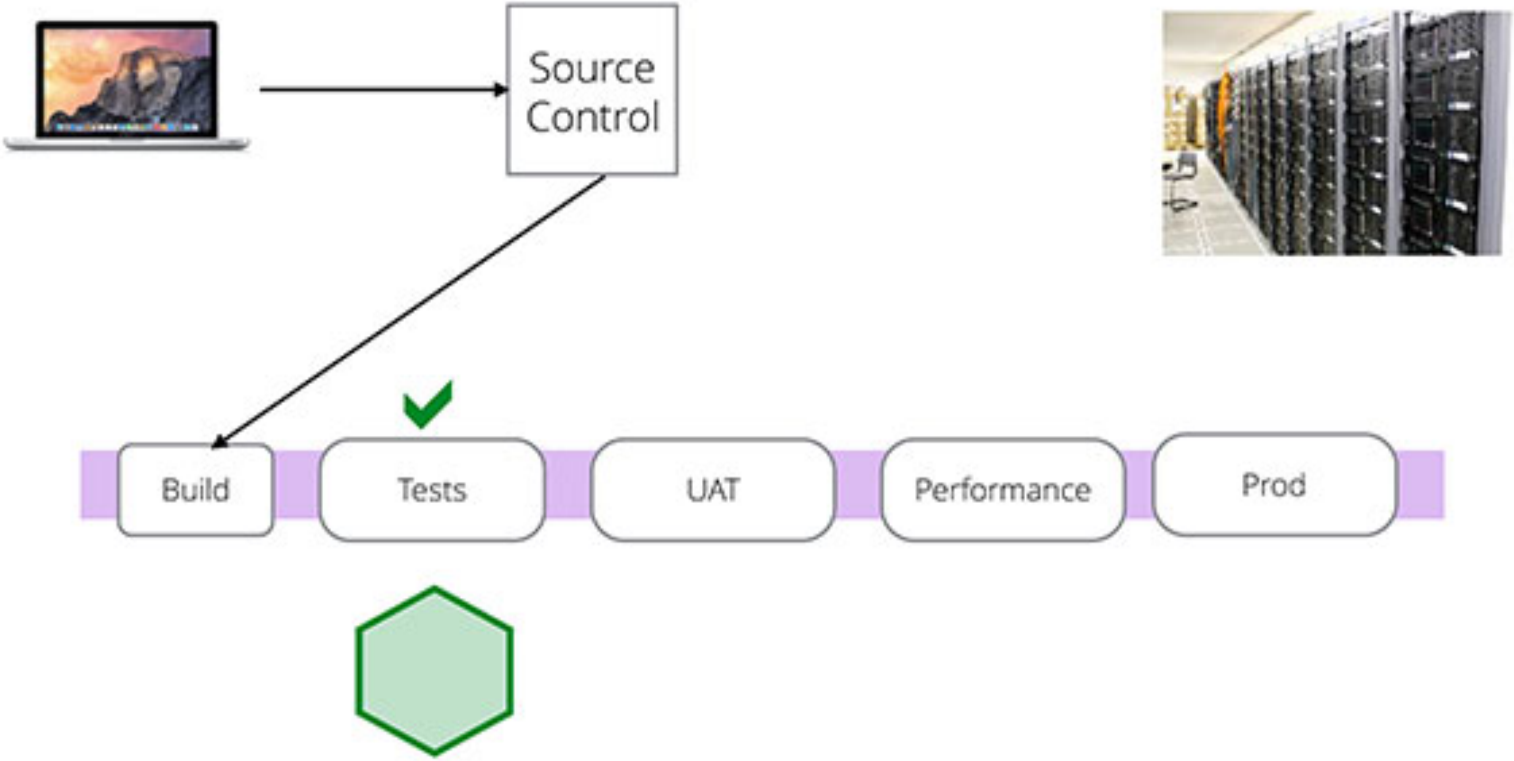
Build

Tests

UAT

Performance

Prod



Source Control

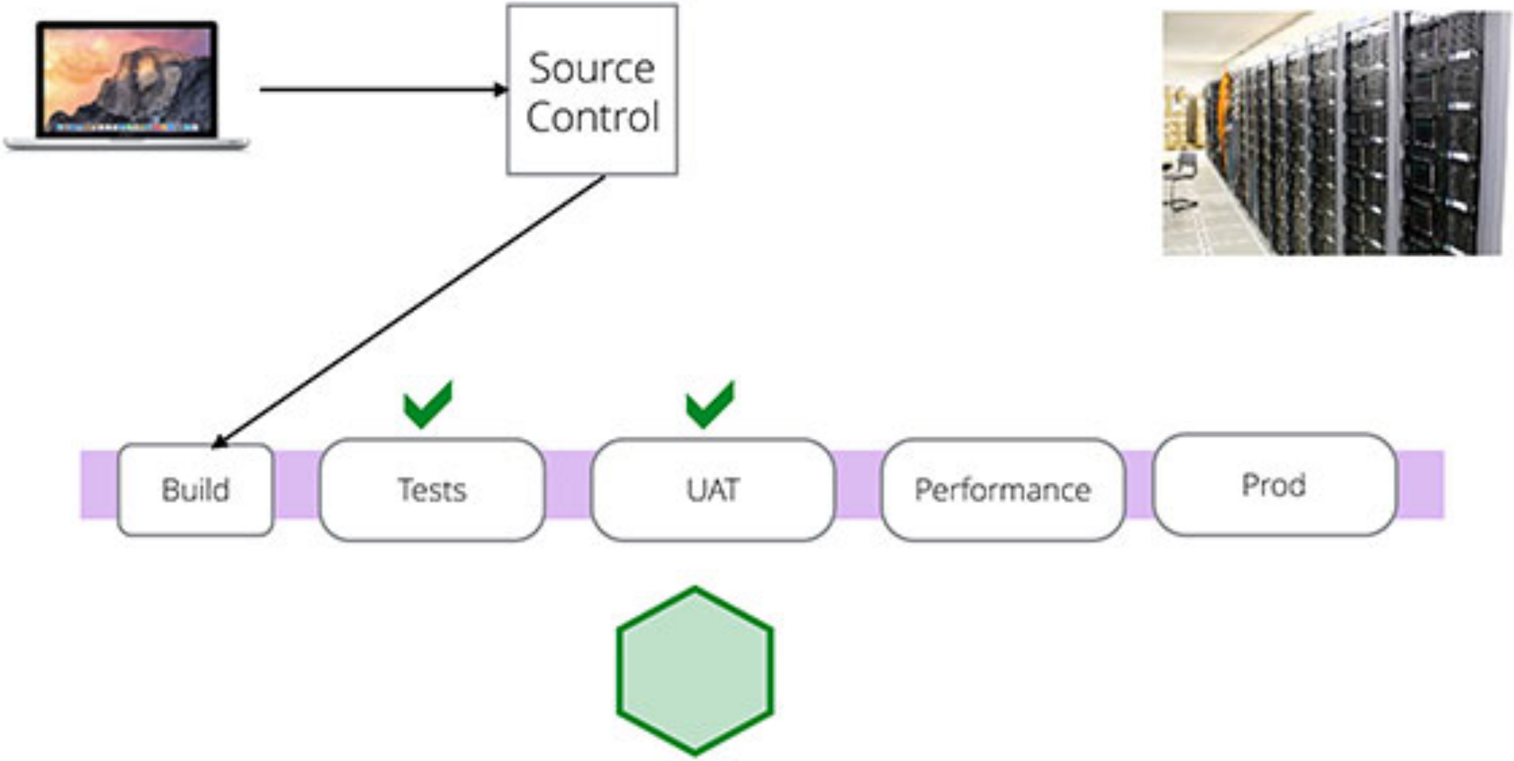
Build

Tests

UAT

Performance

Prod



Source Control

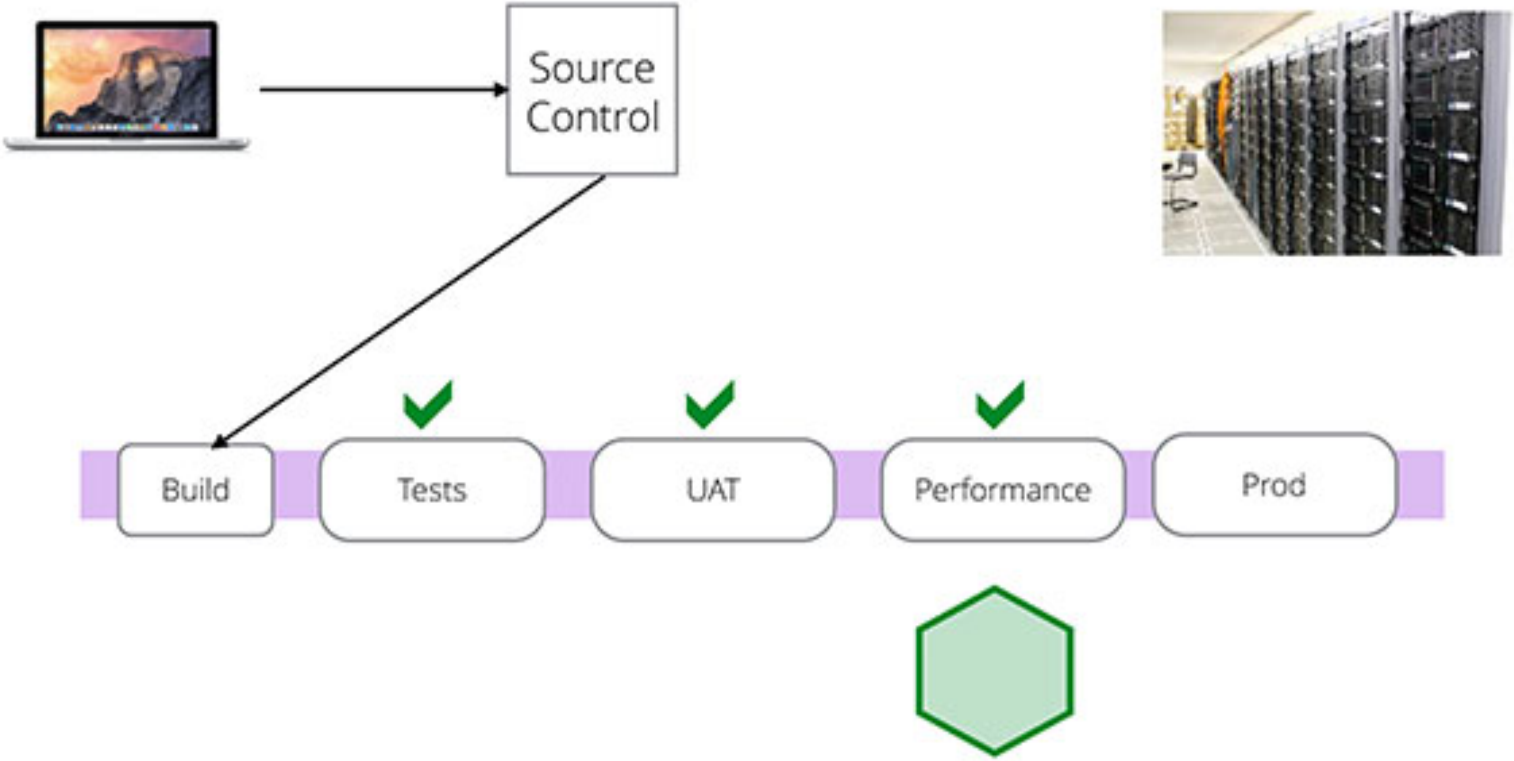
Build

Tests

UAT

Performance

Prod



Source Control

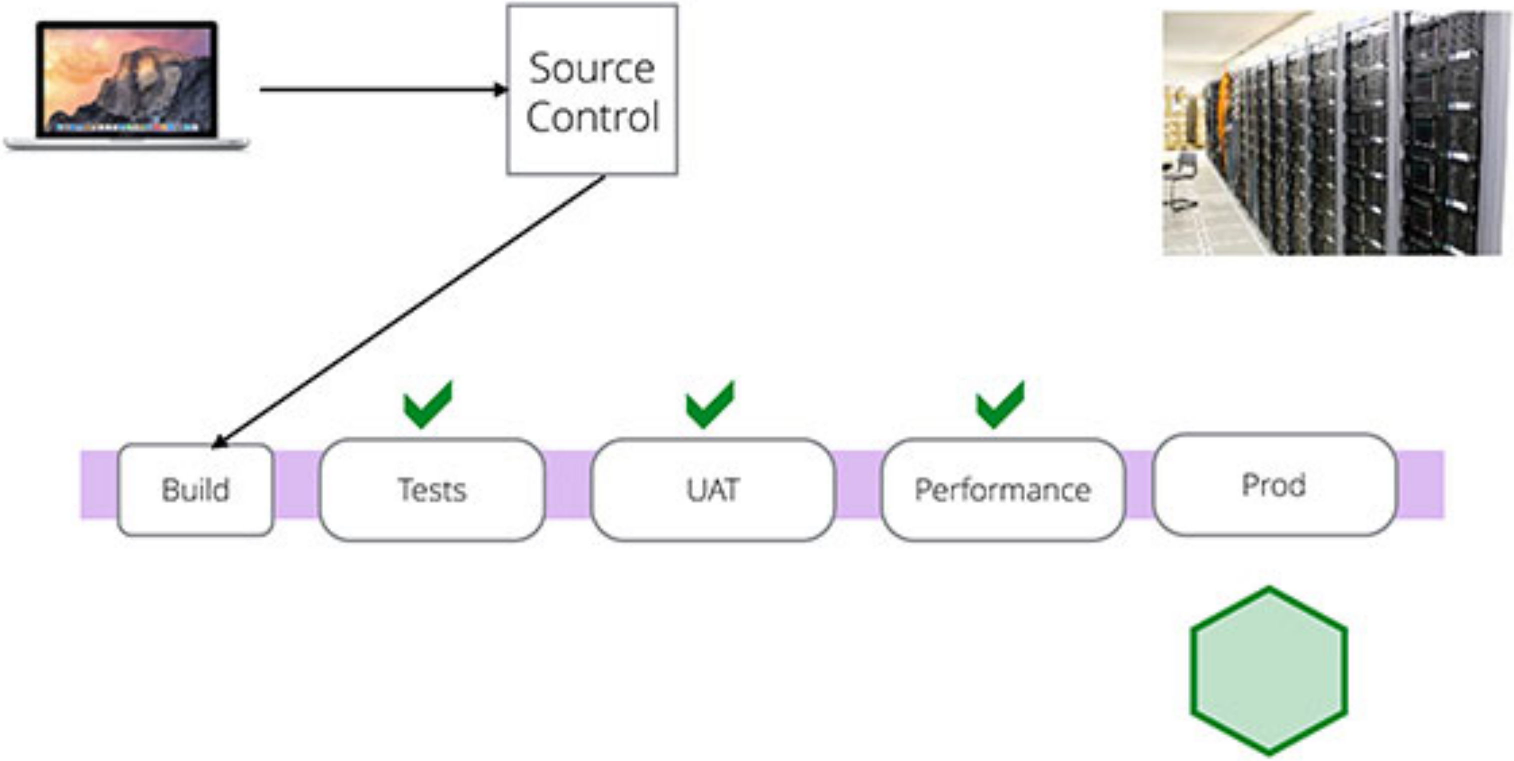
Build

Tests

UAT

Performance

Prod



Source Control

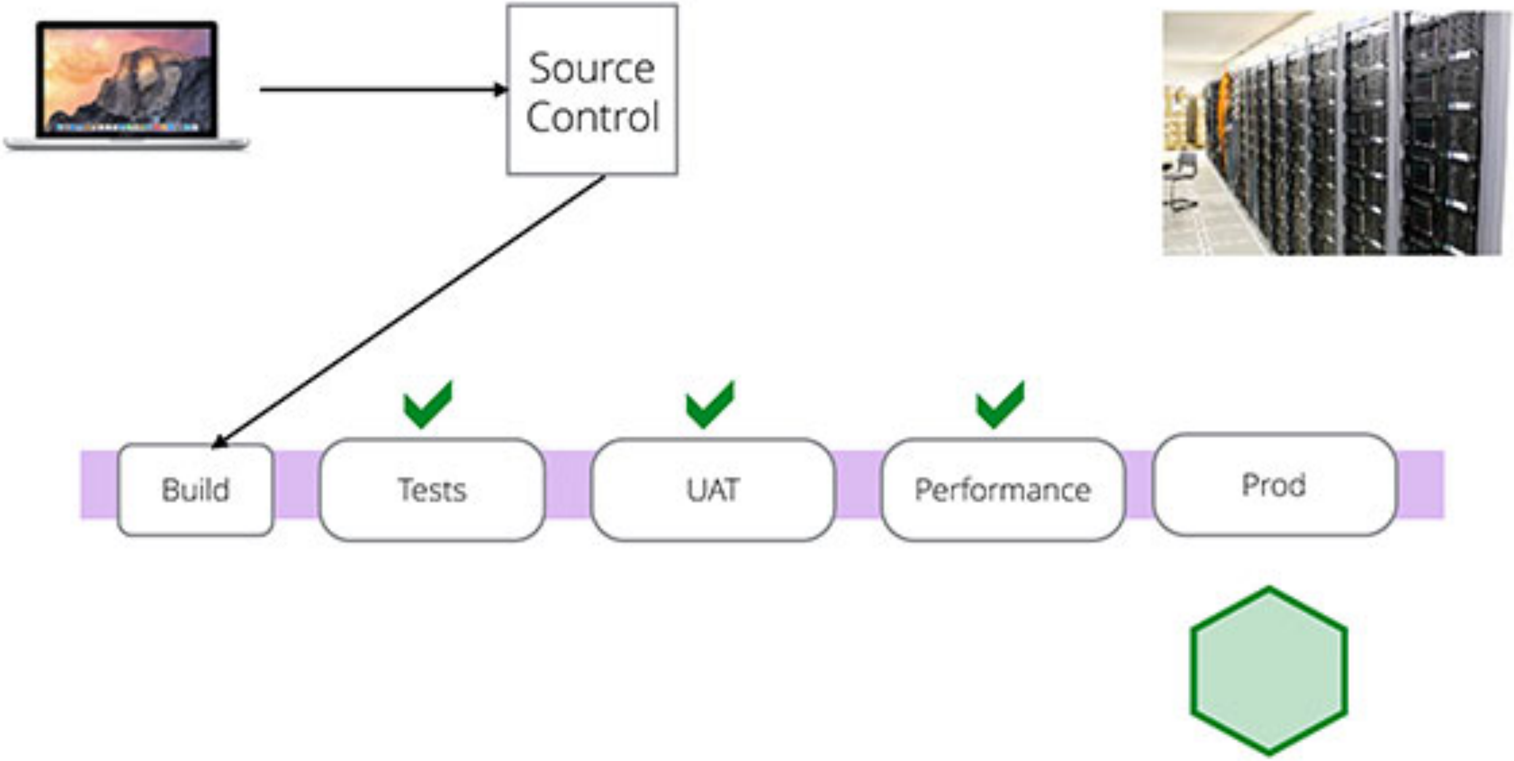
Build

Tests

UAT

Performance

Prod



Source Control

Build

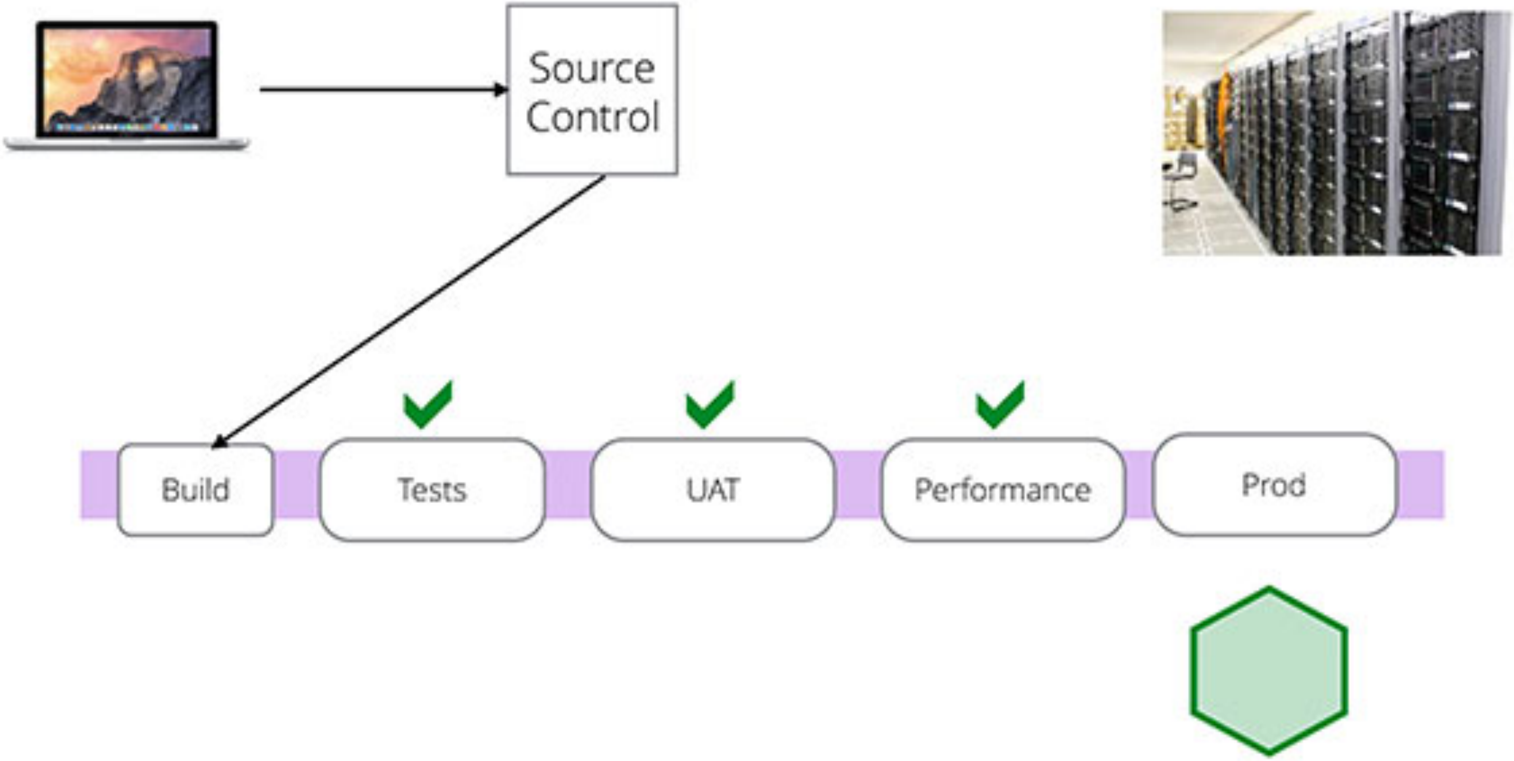
Tests

UAT

Performance

Prod

One Artifact For All Environments



Source Control

Build

Tests

UAT

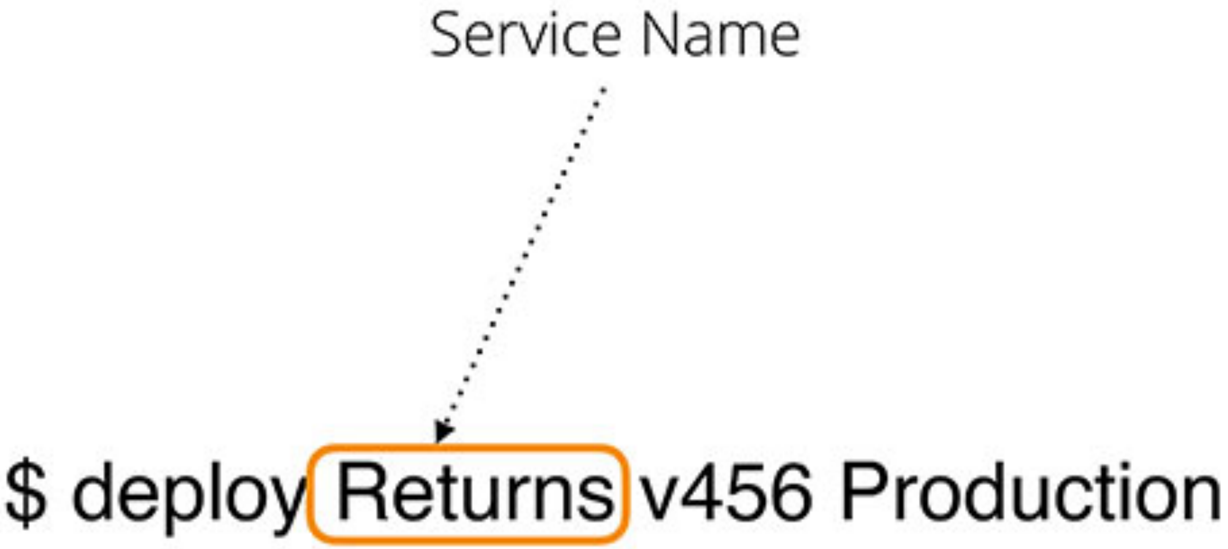
Performance

Prod

One Artifact For All Environments

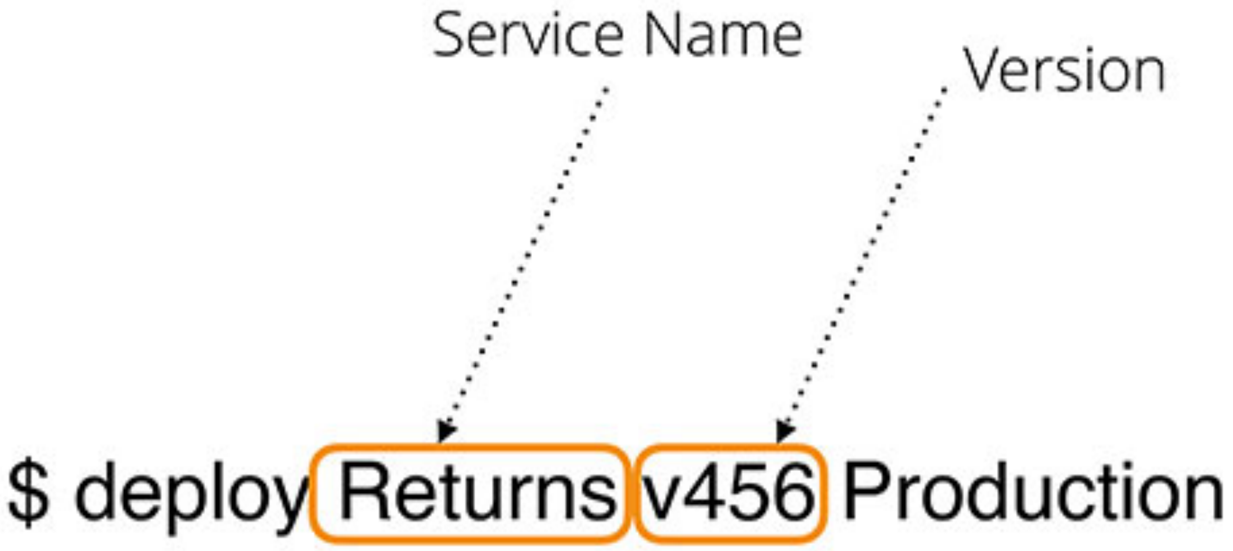
Same Deployment Process Everywhere

```
$ deploy Returns v456 Production
```



Service Name

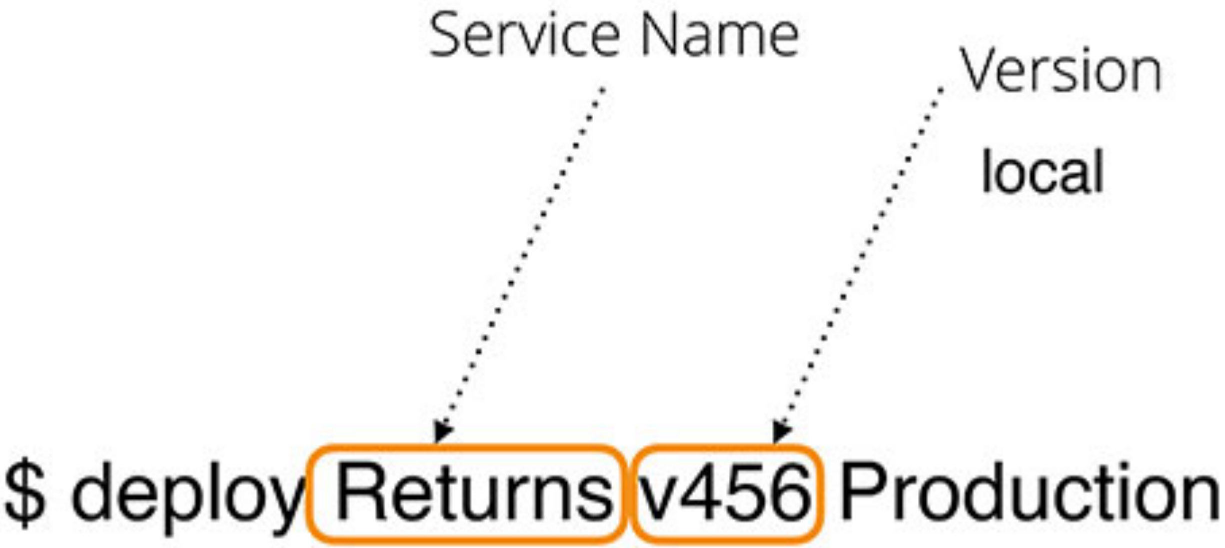
```
$ deploy Returns v456 Production
```



Service Name

Version

```
$ deploy Returns v456 Production
```

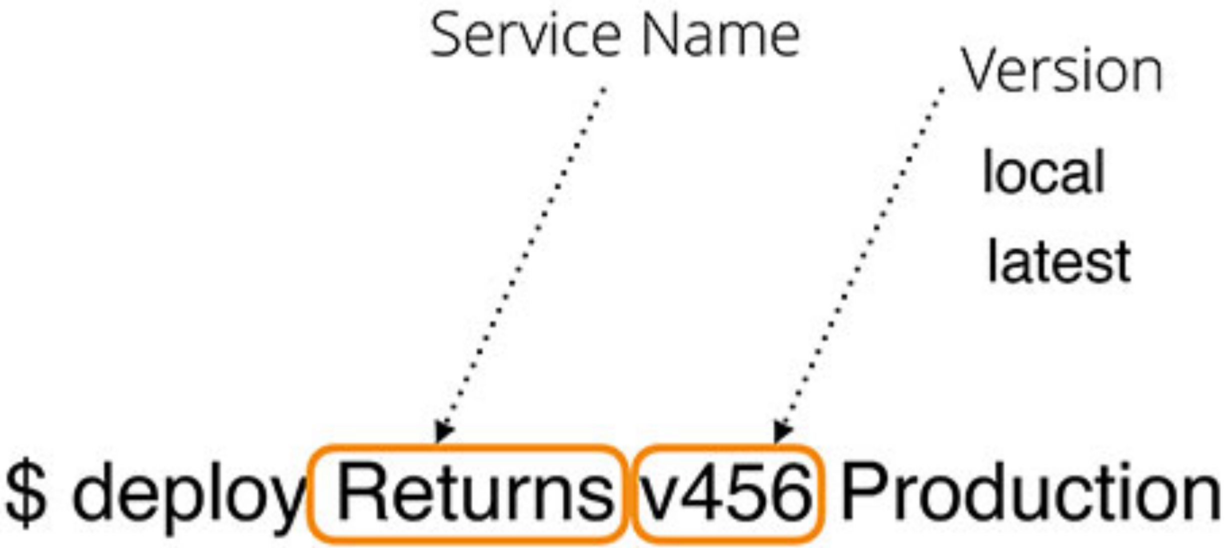


Service Name

Version

local

```
$ deploy Returns v456 Production
```



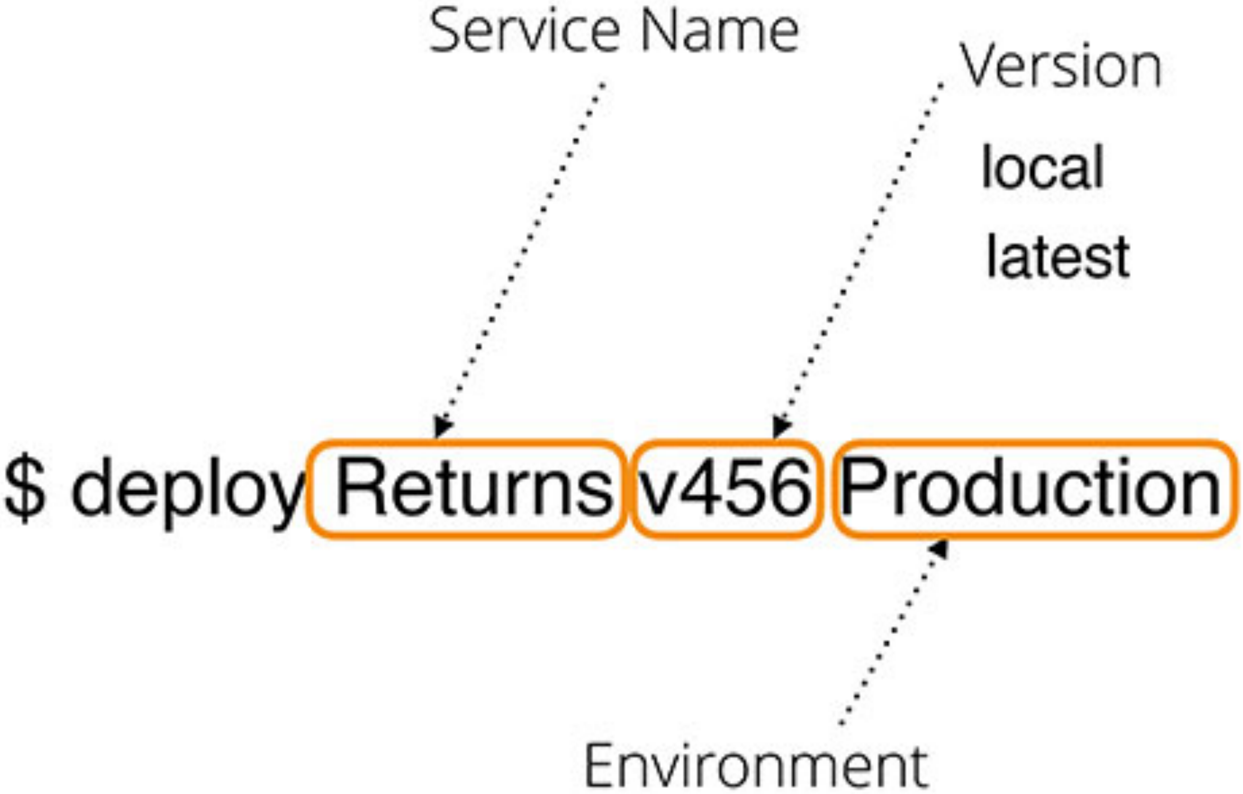
Service Name

Version

local

latest

```
$ deploy Returns v456 Production
```



Service Name

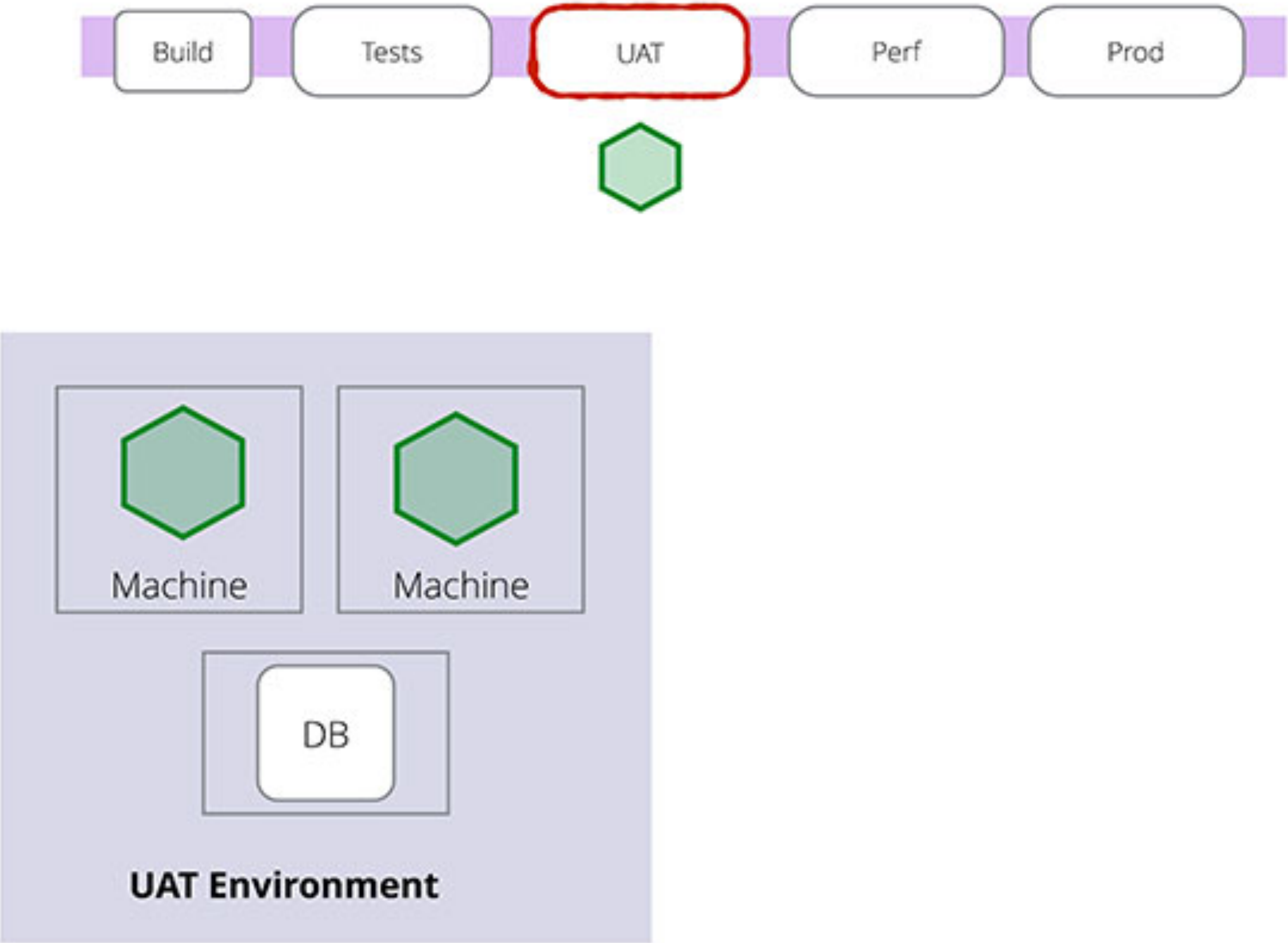
Version

local

latest

```
$ deploy Returns v456 Production
```

Environment



Build

Tests

UAT

Perf

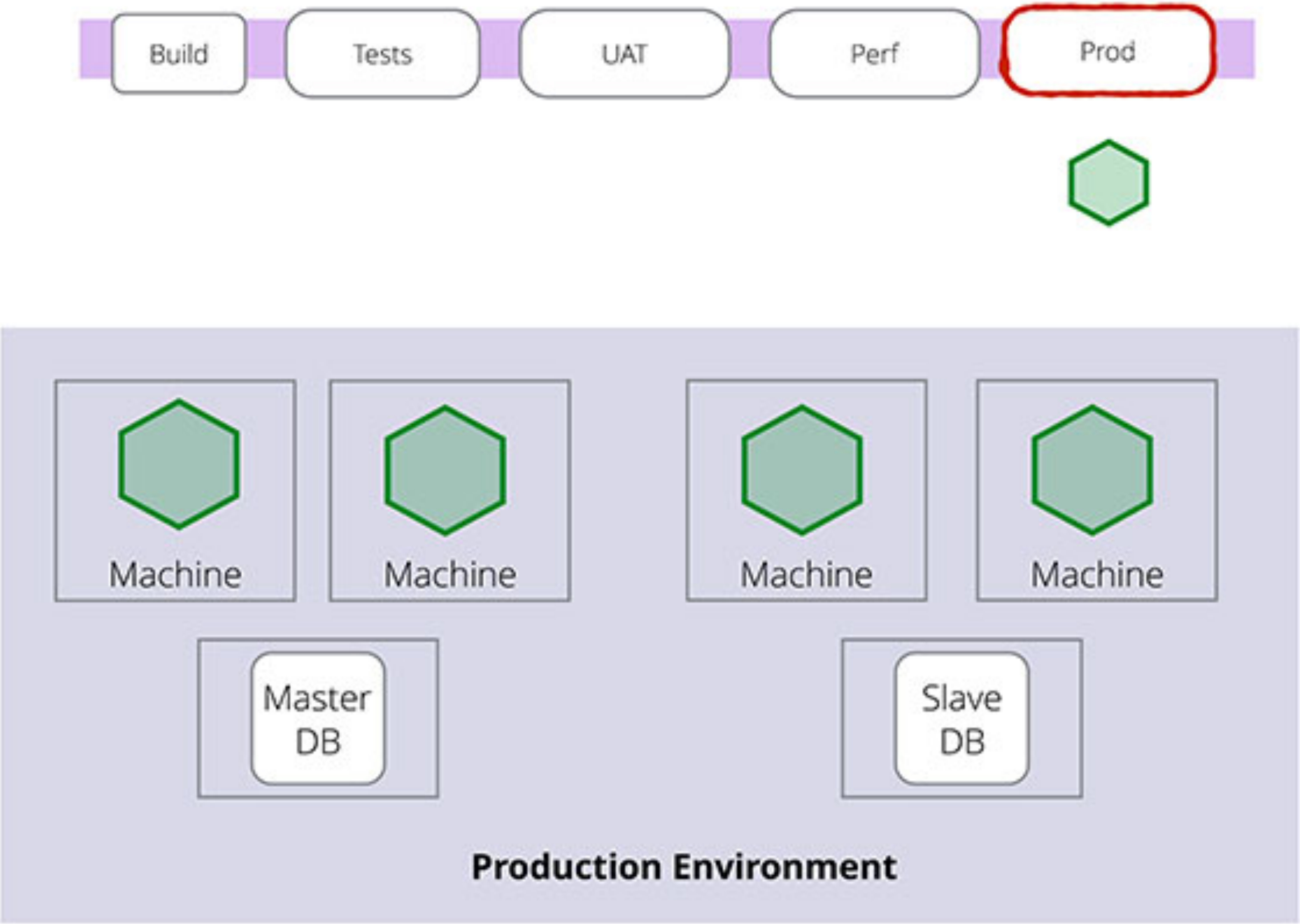
Prod

Machine

Machine

DB

UAT Environment


Build
Tests
UAT
Perf
Prod
Machine
Machine
Machine
Machine
Master DB
Slave DB
Production Environment

Same Artifact

Same Artifact

Different Topology

Core Principles?

Core Principles?

Independent Deployability

Core Principles?

Independent Deployability

One Artifact For All Environments

Core Principles?

Independent Deployability

One Artifact For All Environments

Same Deployment Process Everywhere

What do we want from an artifact?

What do we want from an artifact?

What do we want from an artifact?

Easy to create

What do we want from an artifact?

Easy to create

Easy to deploy

What do we want from an artifact?

Easy to create

Easy to deploy

Abstract out the tech stack

What do we want from an artifact?

Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Tarballs

Tarballs

Giant bundles of stuff

Tarballs

Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Tarballs

✓ Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Tarballs

✓ Easy to create

✗ Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Tarballs

✓ Easy to create

✗ Easy to deploy

? Abstract out the tech stack

Good for dev, good for ops

Tarballs

✓ Easy to create

✗ Easy to deploy

? Abstract out the tech stack

✗ Good for dev, good for ops

Stack-specific

Stack-specific

nuget

pip

jar

gems

Stack-specific

Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Stack-specific

✓ Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Stack-specific

✓ Easy to create

? Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Stack-specific

✓ Easy to create

? Easy to deploy

✗ Abstract out the tech stack

Good for dev, good for ops

Stack-specific

✓ Easy to create

? Easy to deploy

✗ Abstract out the tech stack

✗ Good for dev, good for ops

OS-Specific

OS-Specific

```
$ sudo apt-get install myservice
```

OS-Specific

```
$ sudo apt-get install myservice
```

```
$ deploy Returns v456 Production
```

OS-Specific

Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

OS-Specific

✗ Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

OS-Specific

✗ Easy to create

✓ Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

OS-Specific

✗ Easy to create

✓ Easy to deploy

✓ Abstract out the tech stack

Good for dev, good for ops

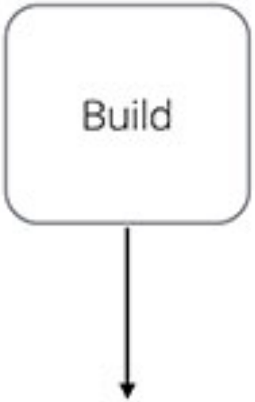
OS-Specific

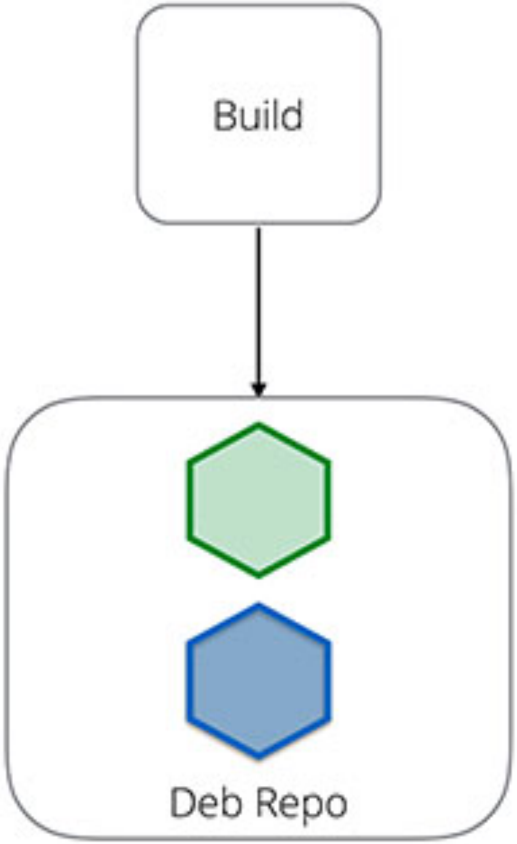
✗ Easy to create

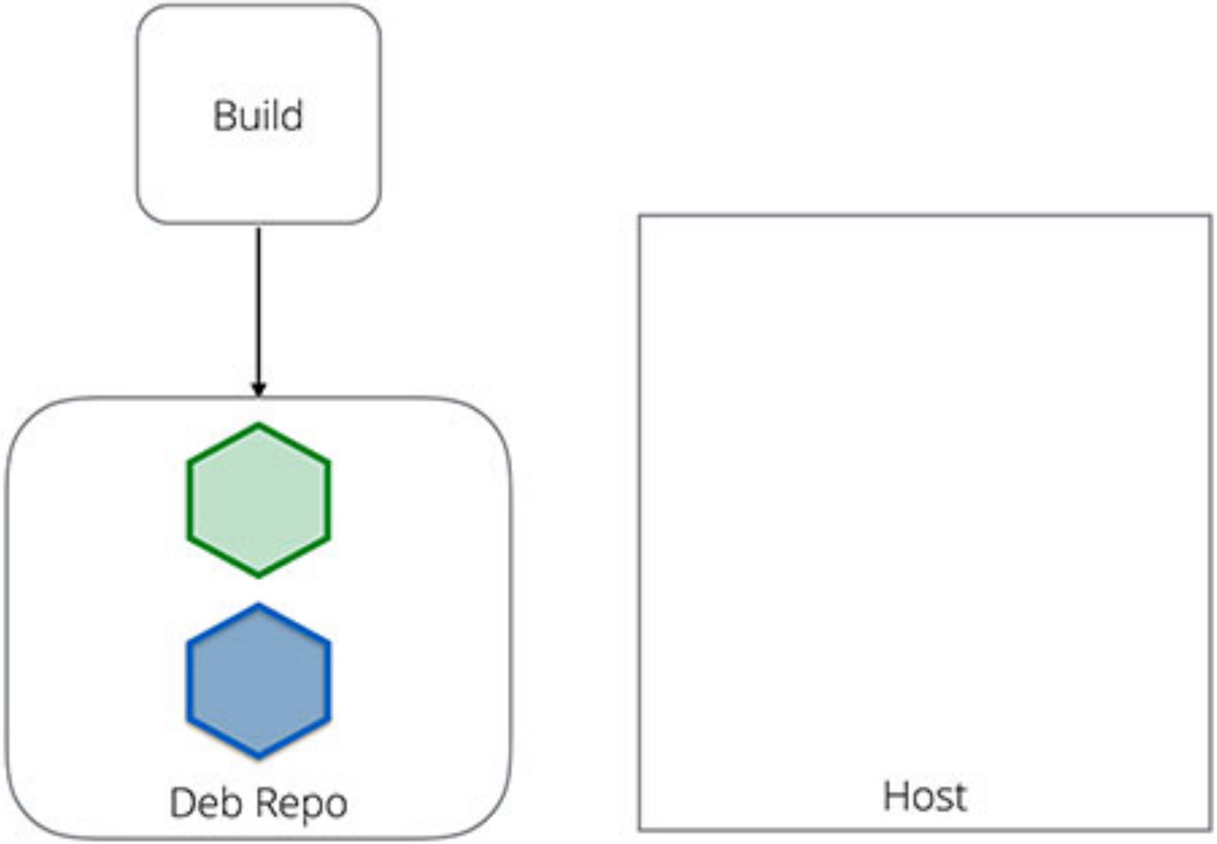
✓ Easy to deploy

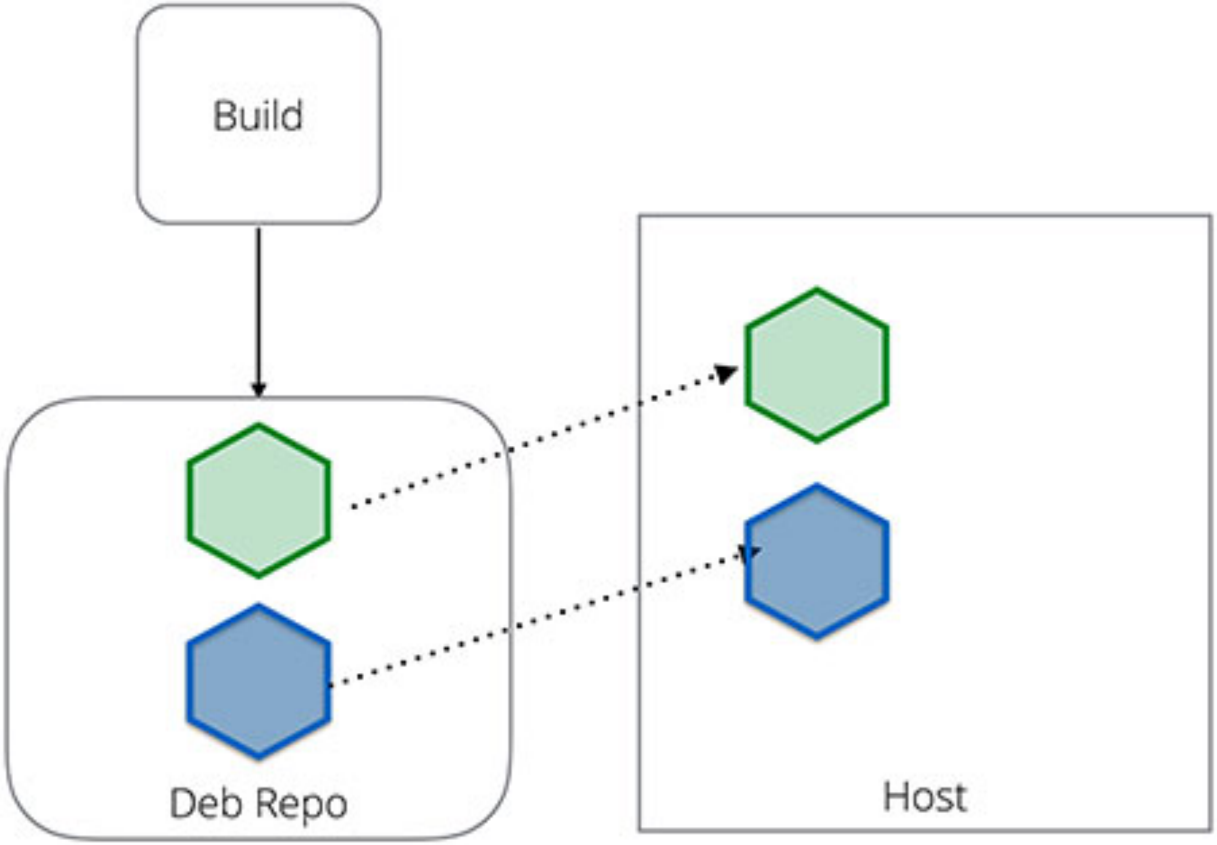
✓ Abstract out the tech stack

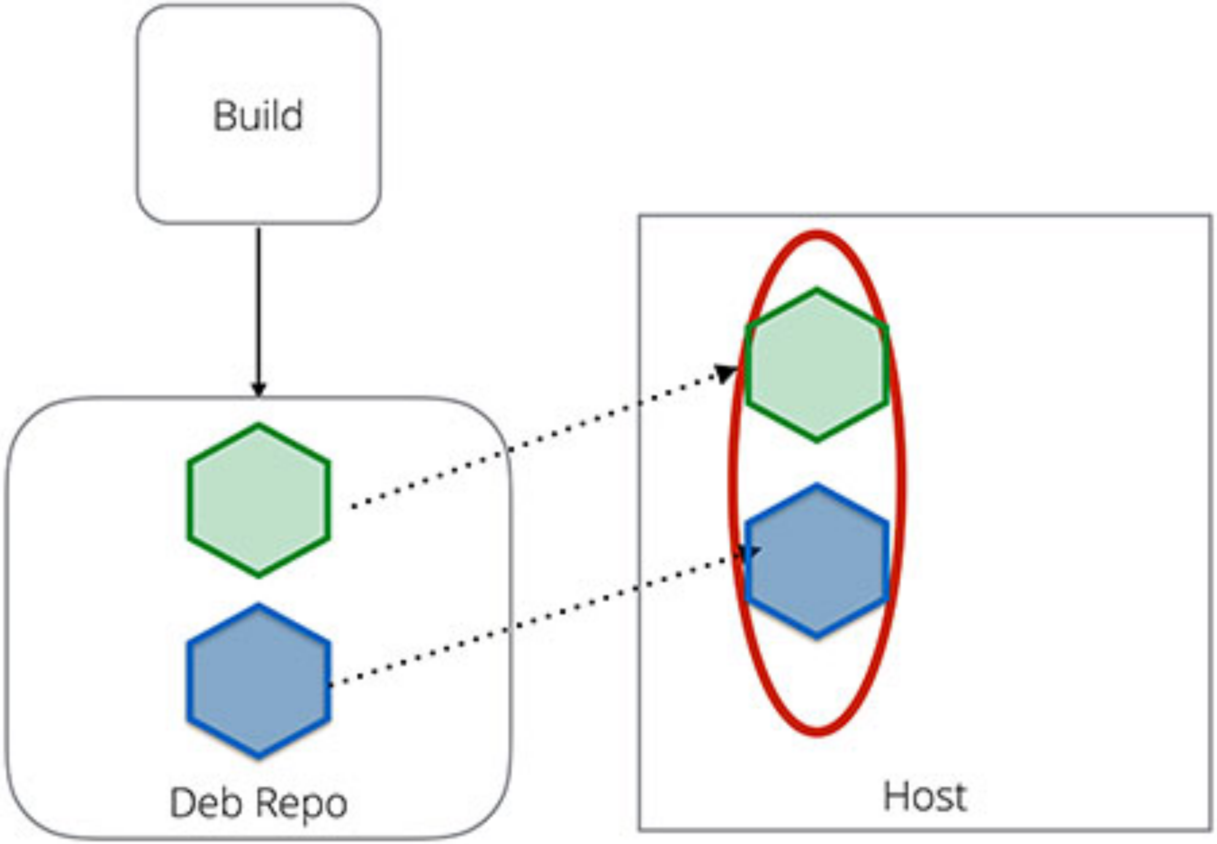
? Good for dev, good for ops

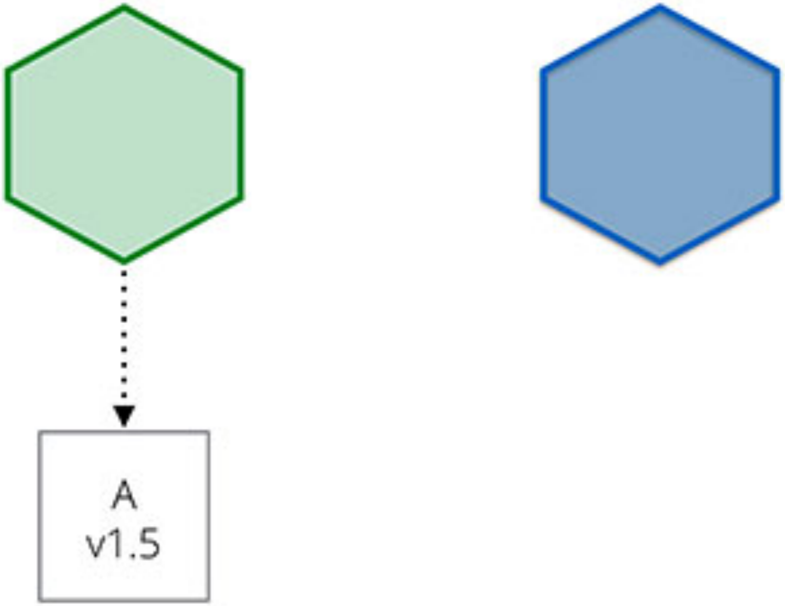

Build

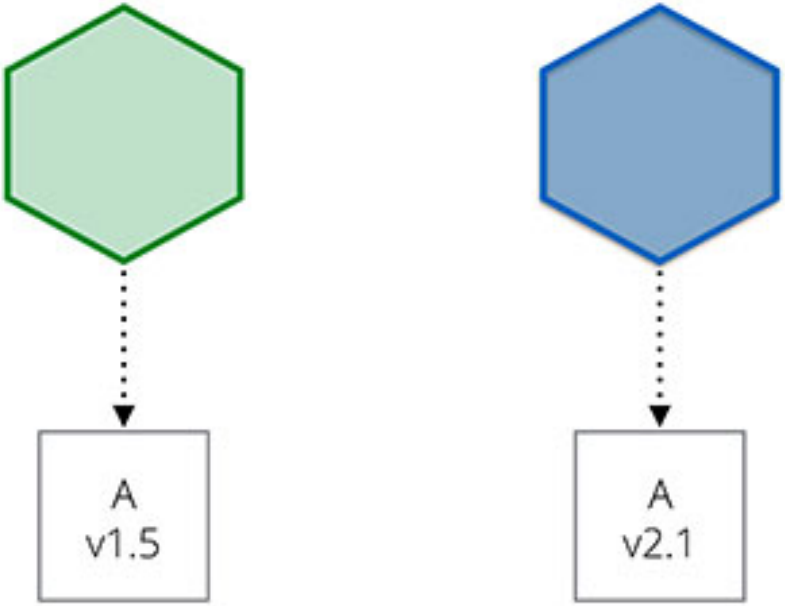

Build
Deb Repo


Build
Deb Repo
Host


Build
Deb Repo
Host

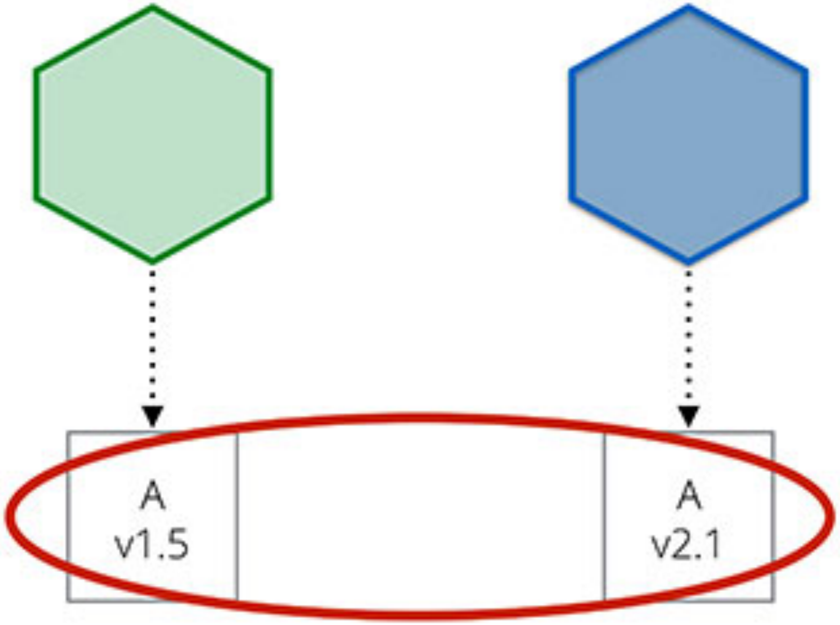

Build
Deb Repo
Host

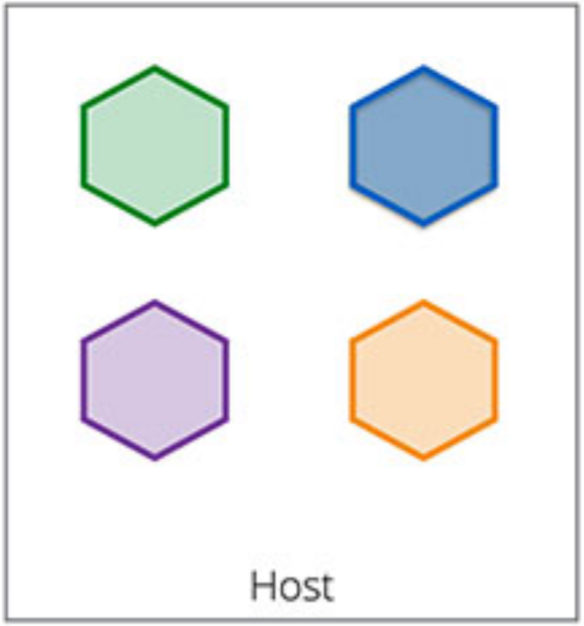

A
v1.5

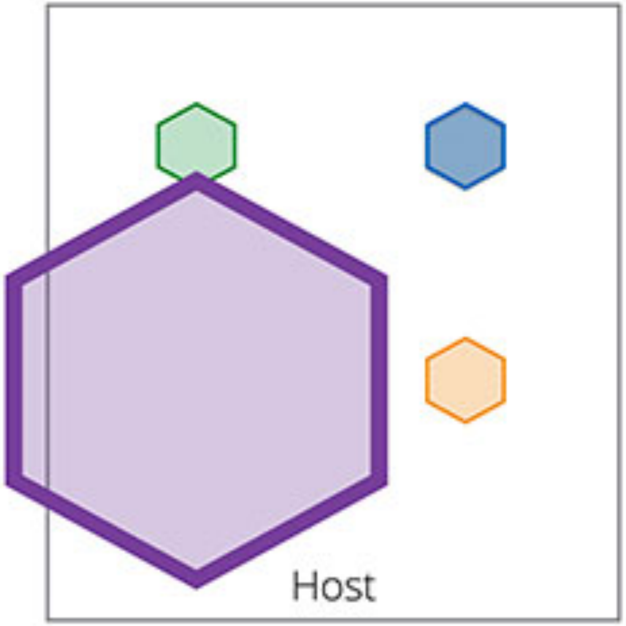


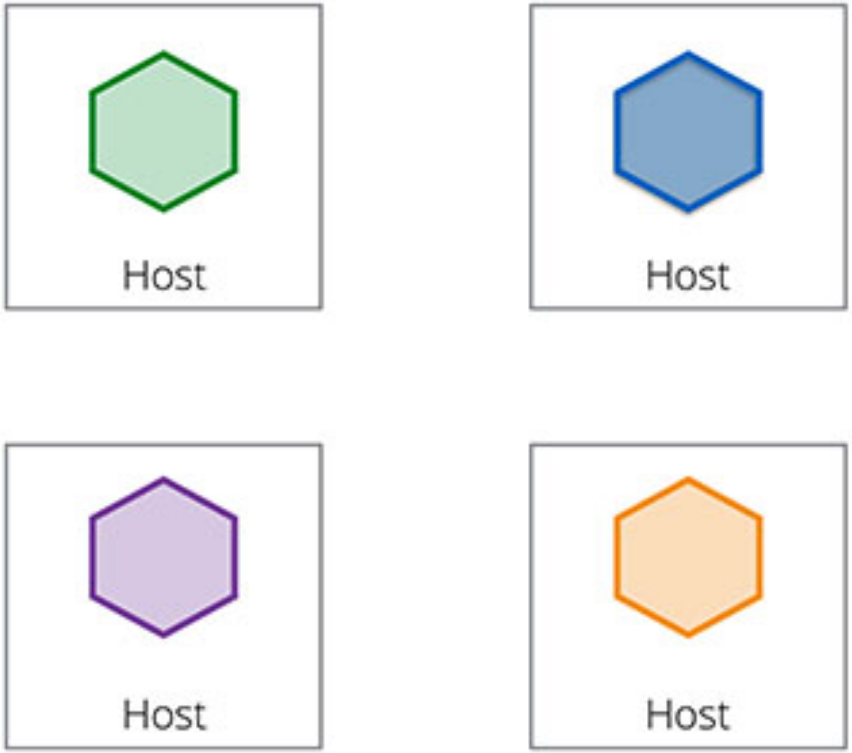
A
v1.5

A
v2.1


A
v1.5
A
v2.1


Host


Host



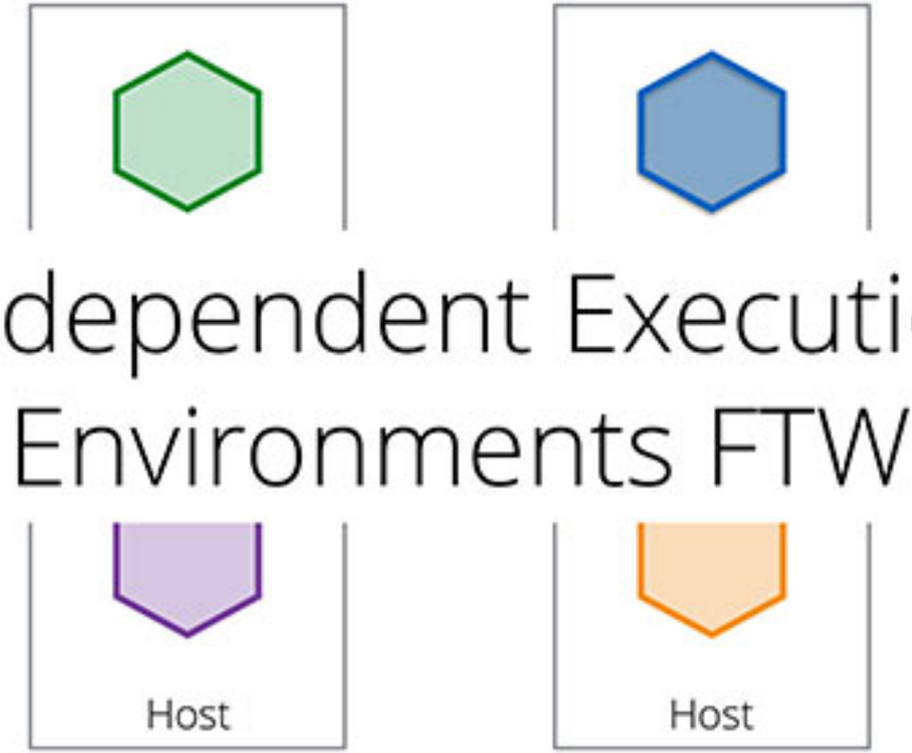
Host

Host

Host

Host

Independent Execution Environments FTW!



Custom Images

Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Custom Images

? Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Custom Images

? Easy to create

? Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Custom Images

? Easy to create

? Easy to deploy

✓ Abstract out the tech stack

Good for dev, good for ops

Custom Images

? Easy to create

? Easy to deploy

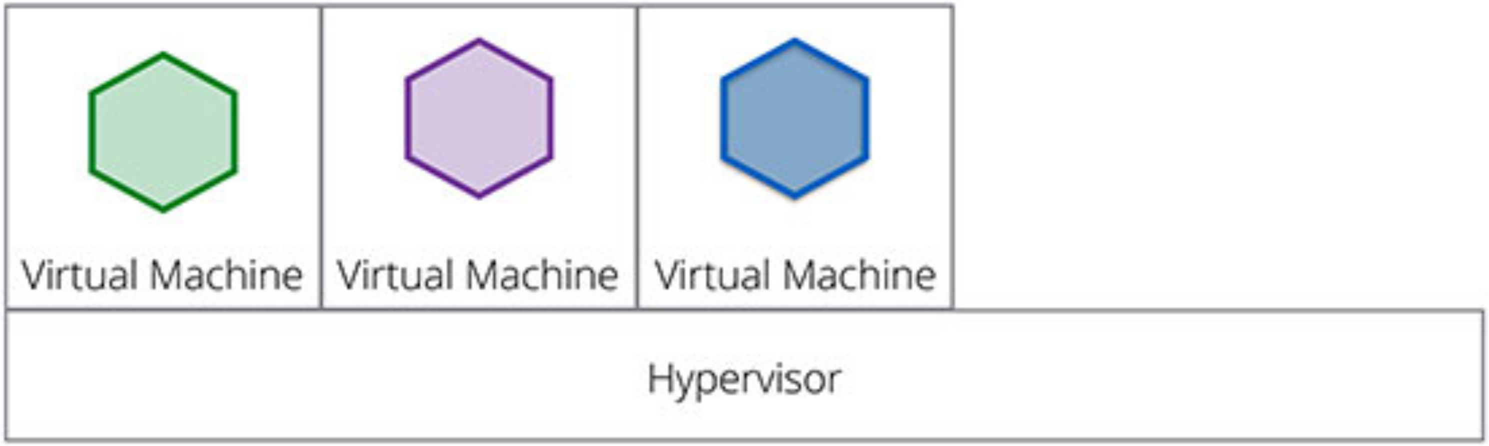
✓ Abstract out the tech stack

? Good for dev, good for ops

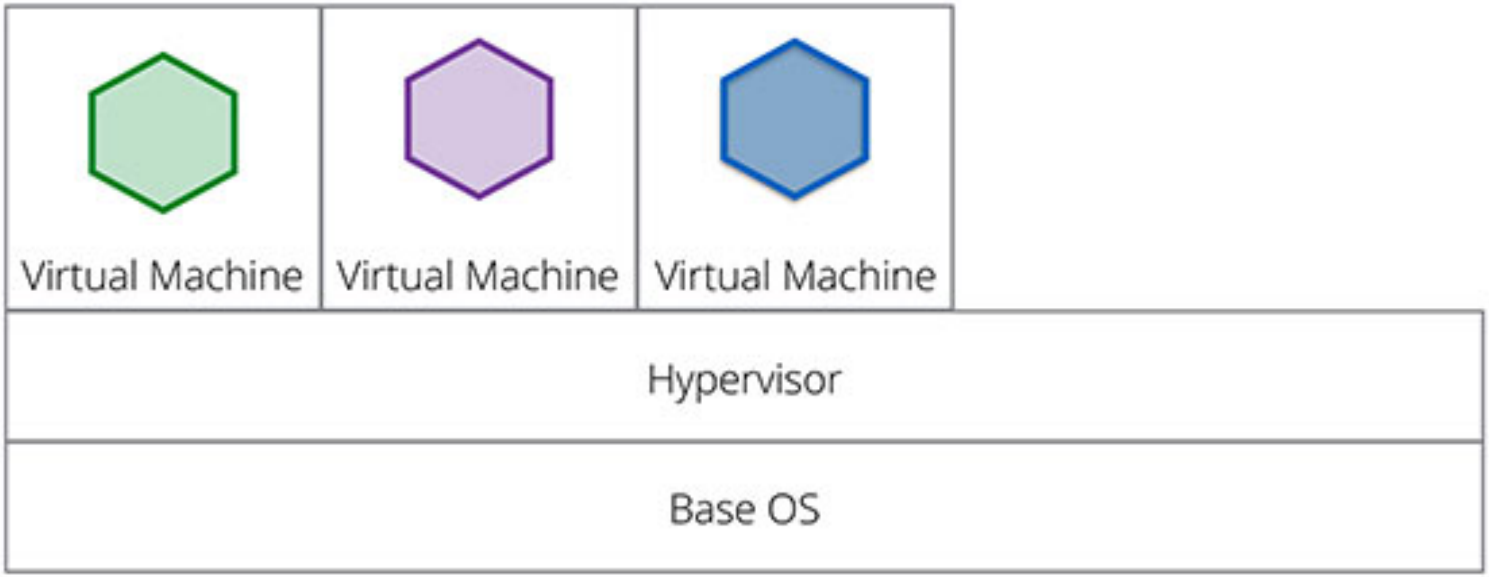
TYPE 2 VIRTUALISATION

Virtual Machine
Virtual Machine
Virtual Machine

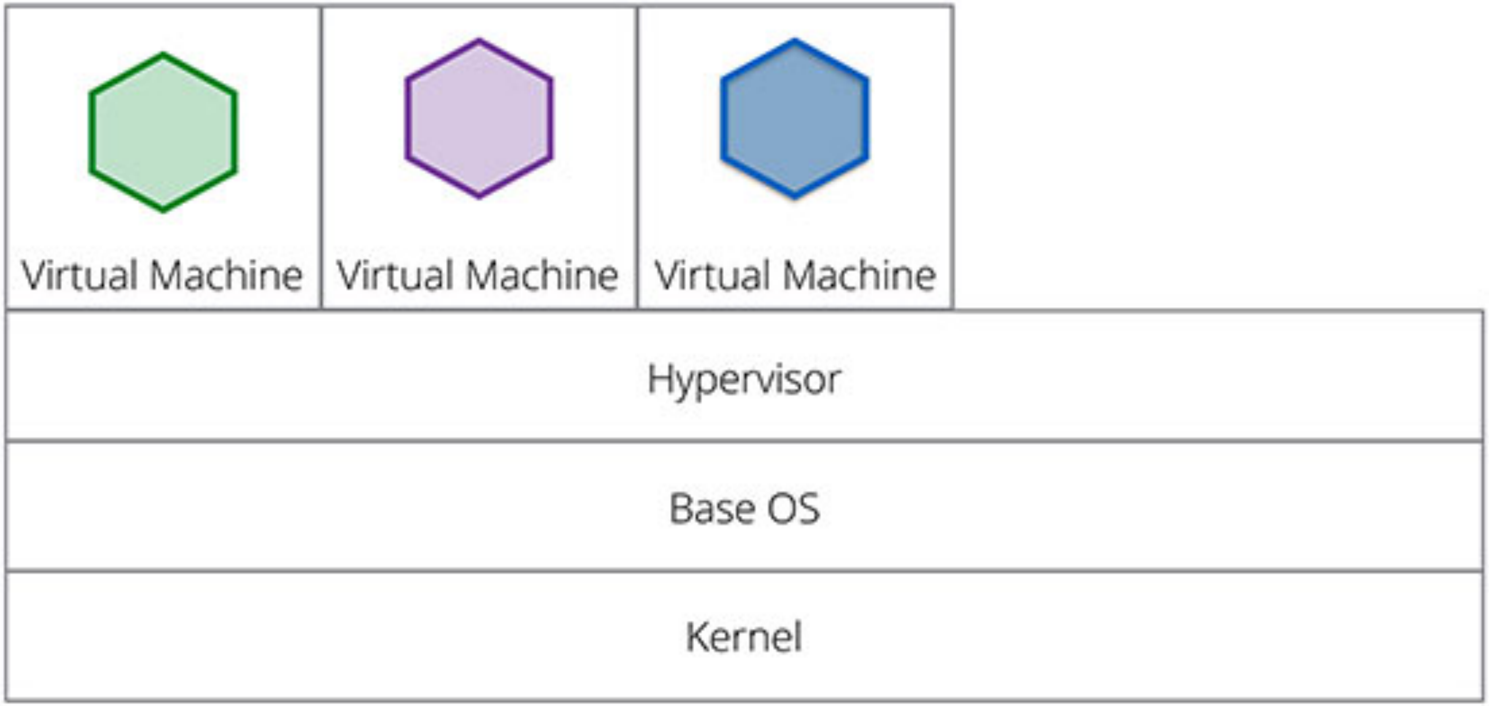
TYPE 2 VIRTUALISATION



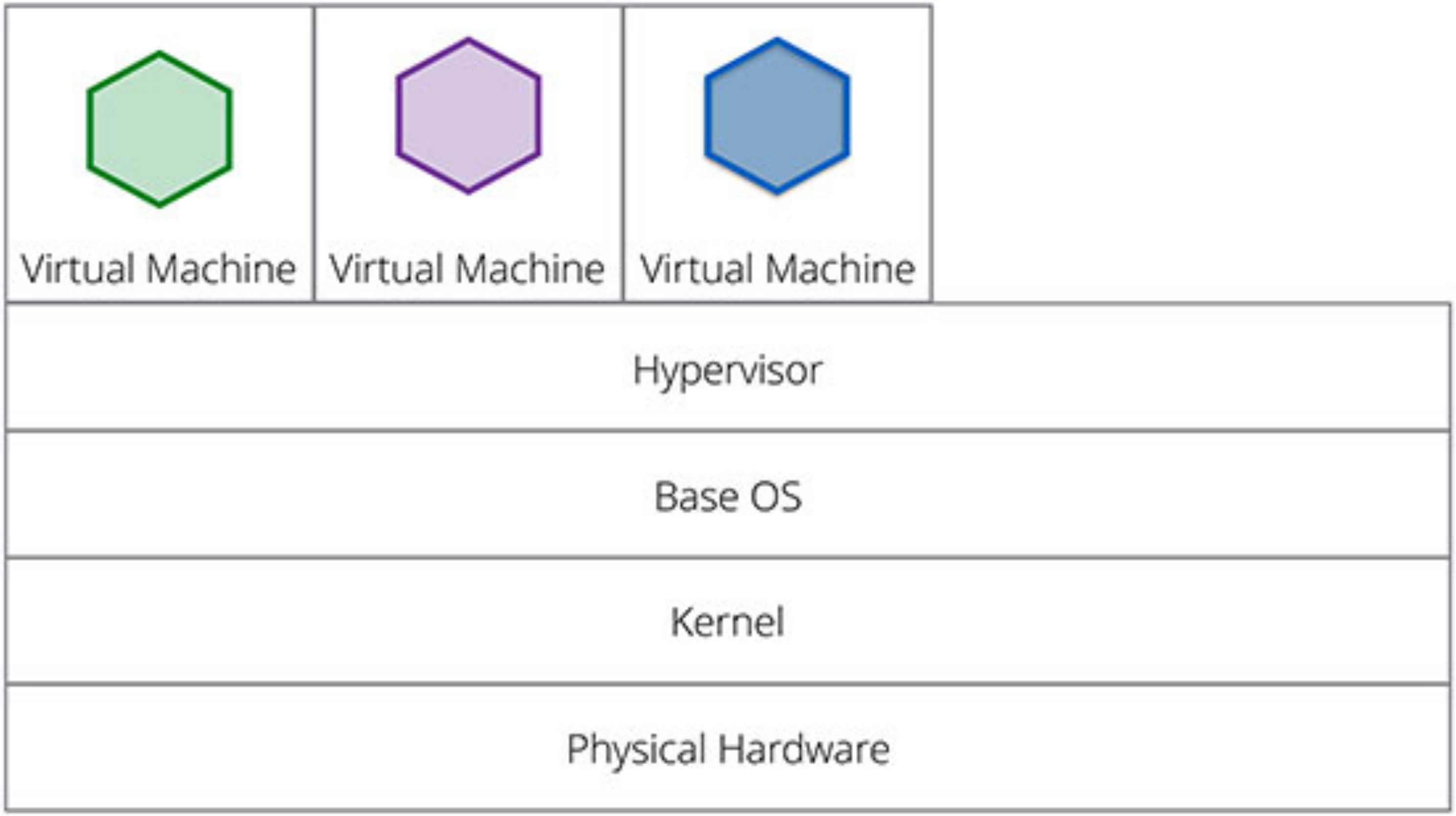
TYPE 2 VIRTUALISATION



TYPE 2 VIRTUALISATION



TYPE 2 VIRTUALISATION



Virtual Machine

Virtual Machine

Virtual Machine

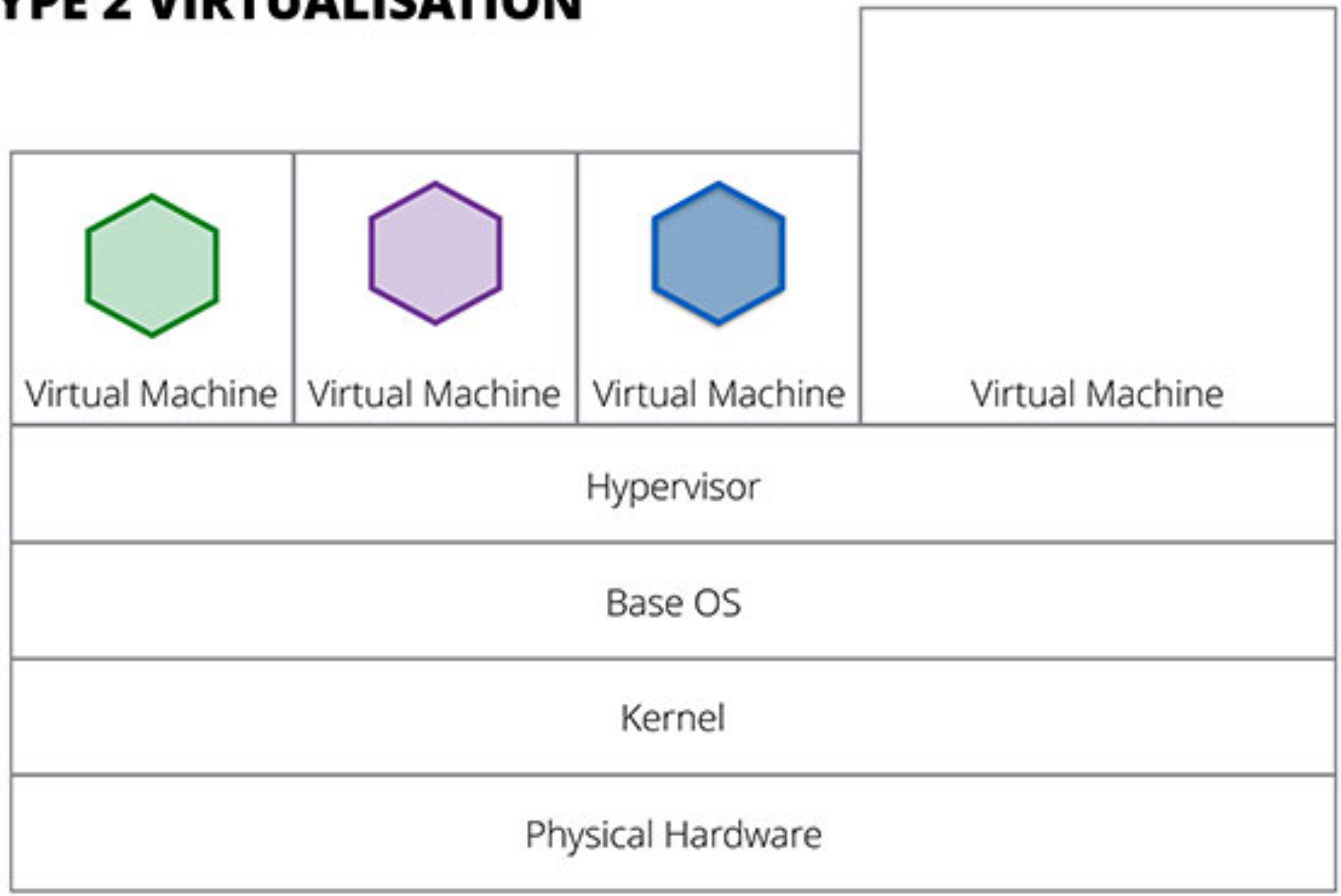
Hypervisor

Base OS

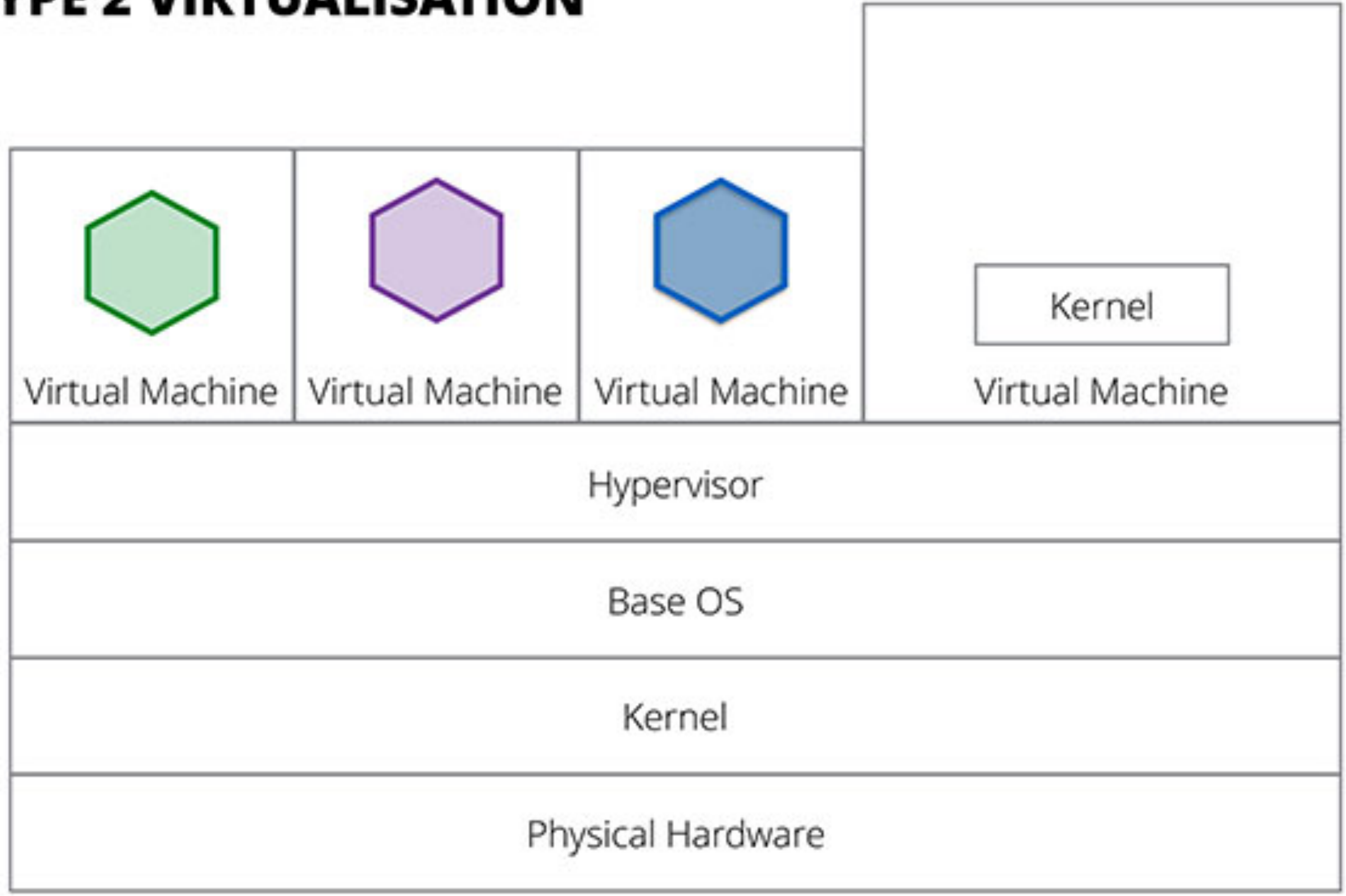
Kernel

Physical Hardware

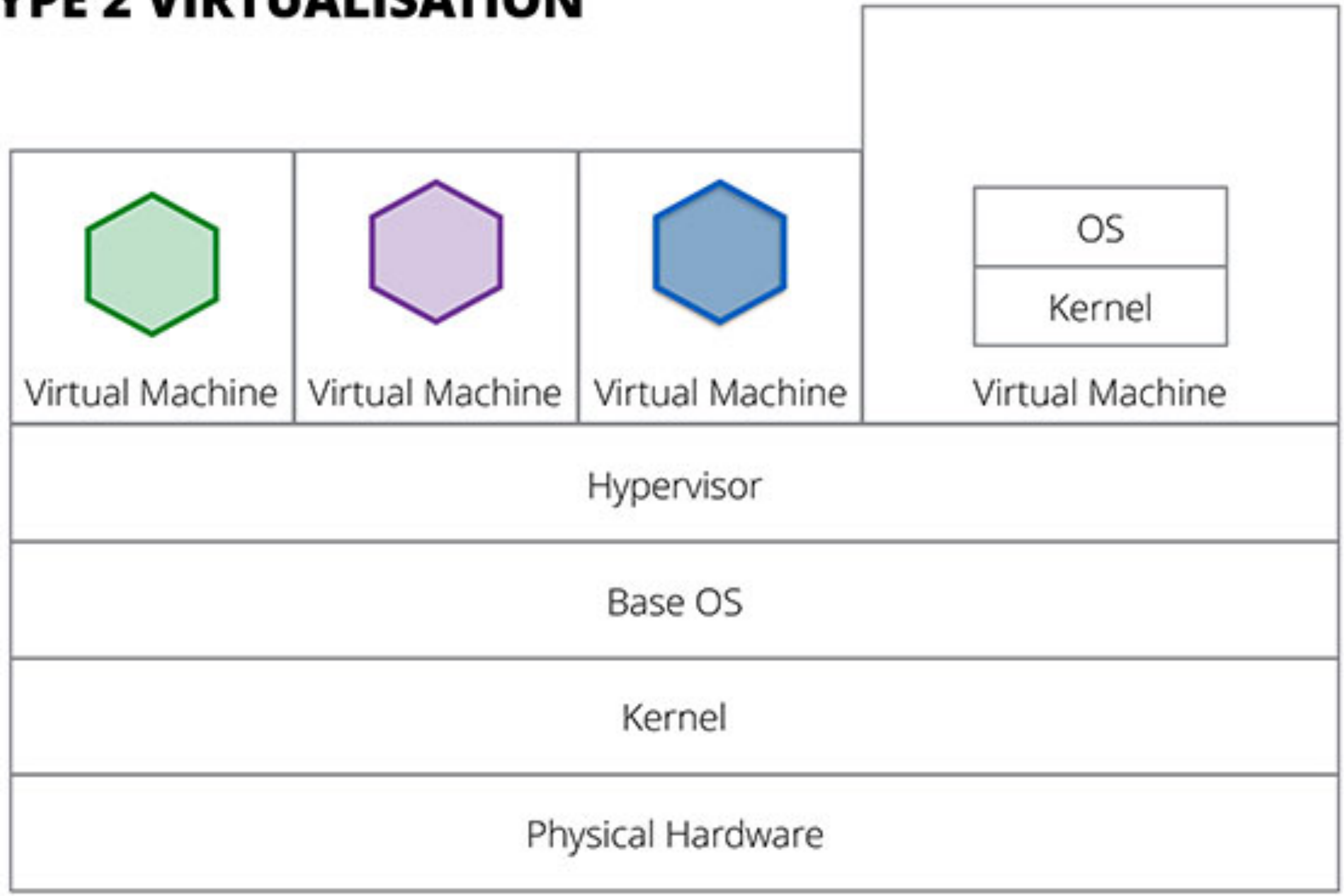
TYPE 2 VIRTUALISATION



TYPE 2 VIRTUALISATION



TYPE 2 VIRTUALISATION



Virtual Machine

Virtual Machine

Virtual Machine

OS

Kernel

Virtual Machine

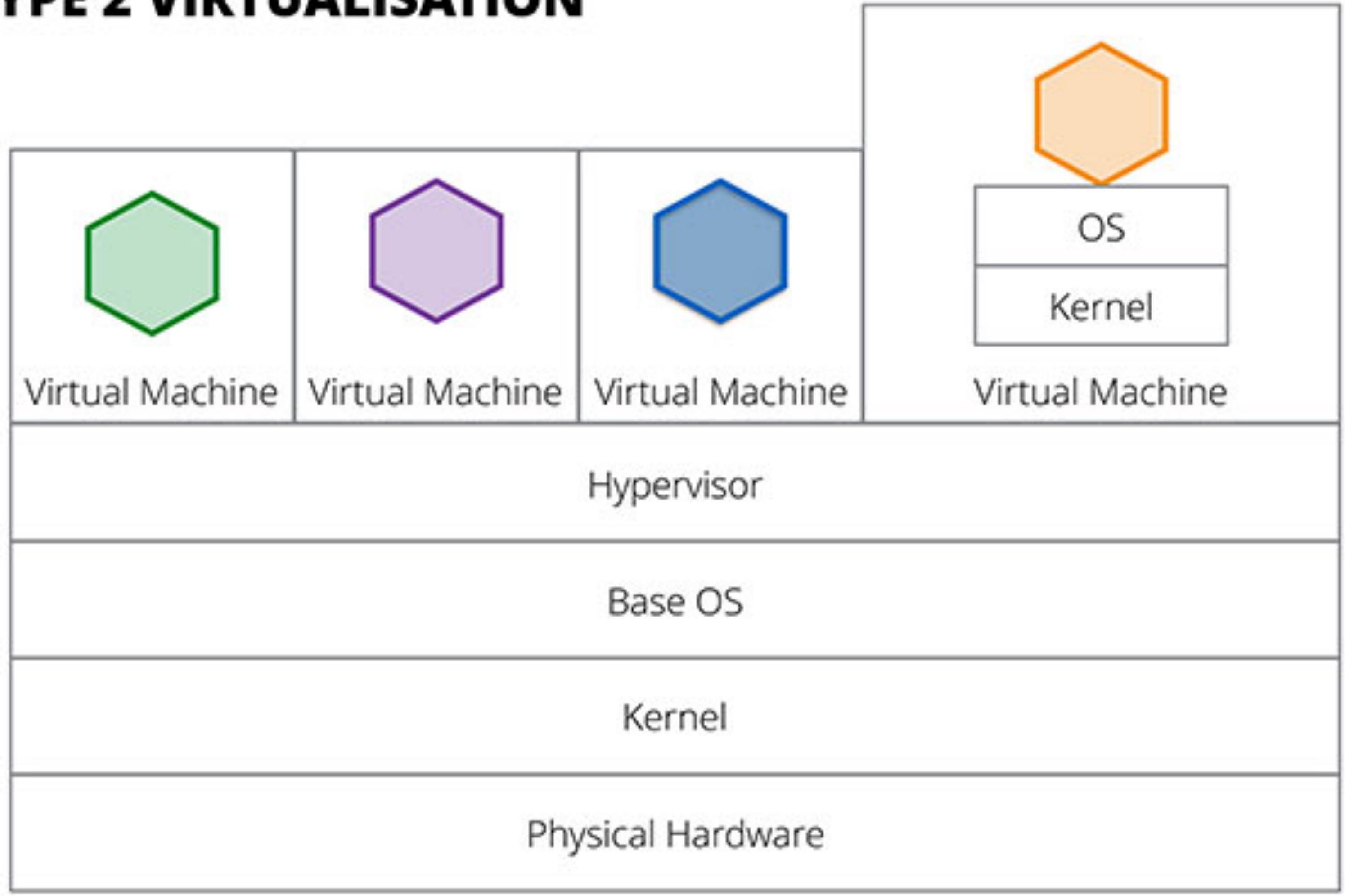
Hypervisor

Base OS

Kernel

Physical Hardware

TYPE 2 VIRTUALISATION



Virtual Machine

Virtual Machine

Virtual Machine

OS

Kernel

Virtual Machine

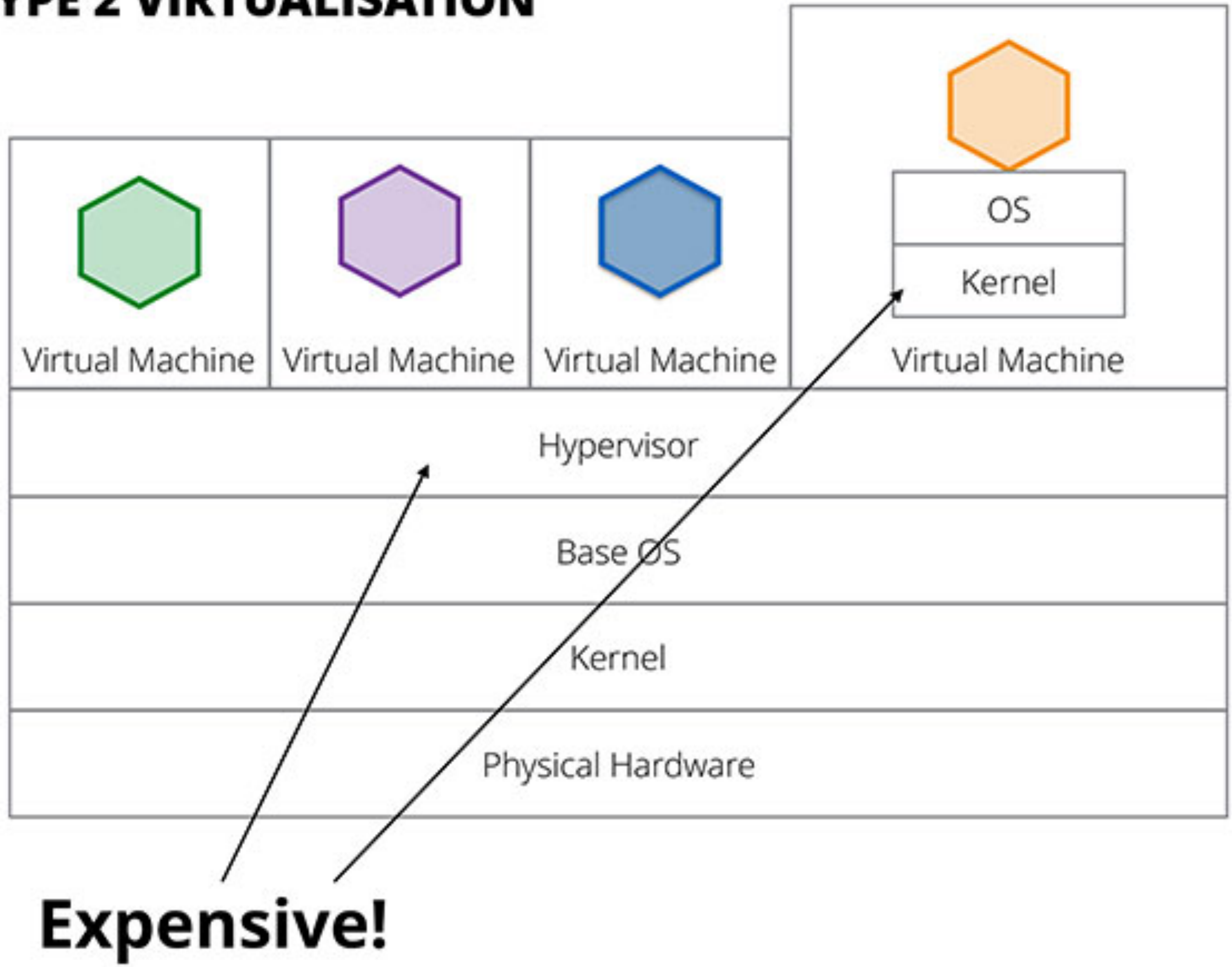
Hypervisor

Base OS

Kernel

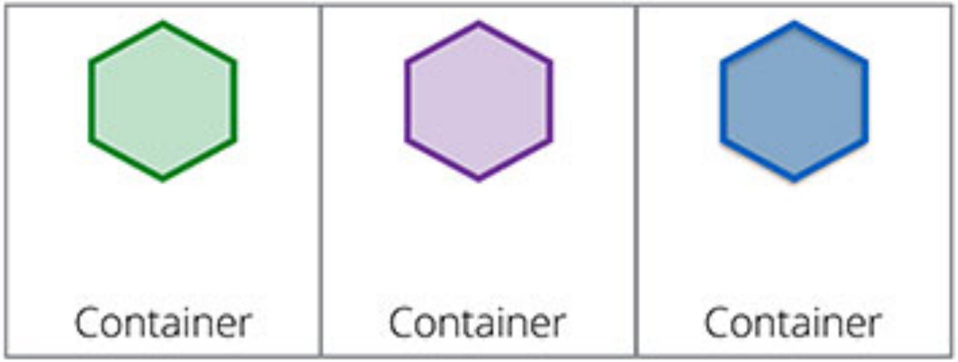
Physical Hardware

TYPE 2 VIRTUALISATION

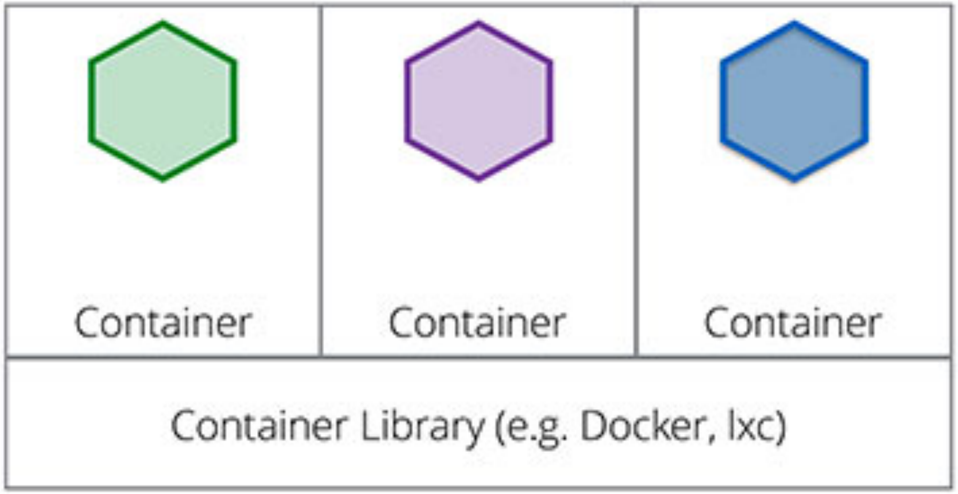


Expensive!

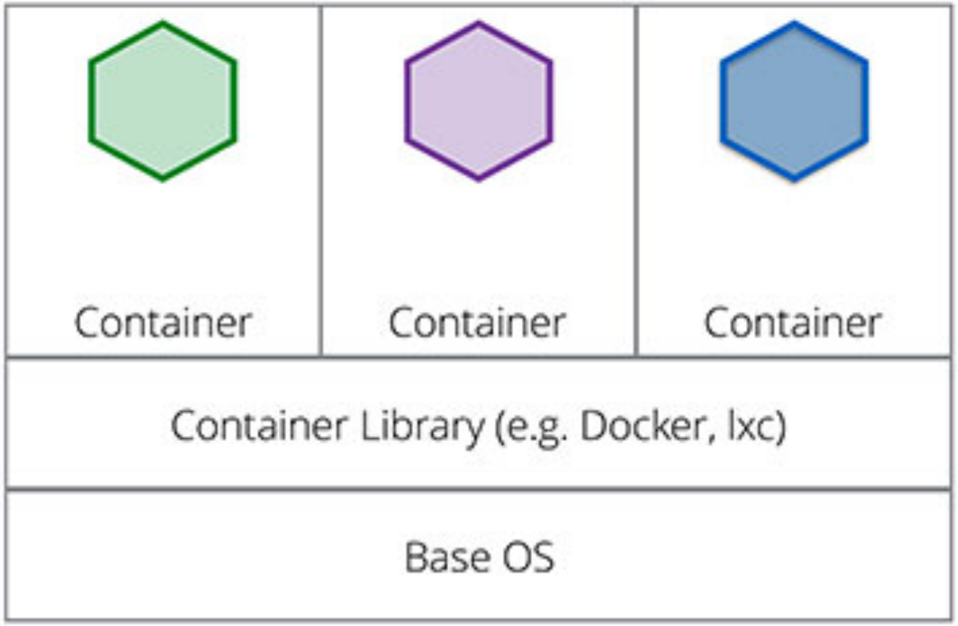
CONTAINERISATION



CONTAINERISATION



CONTAINERISATION



Container

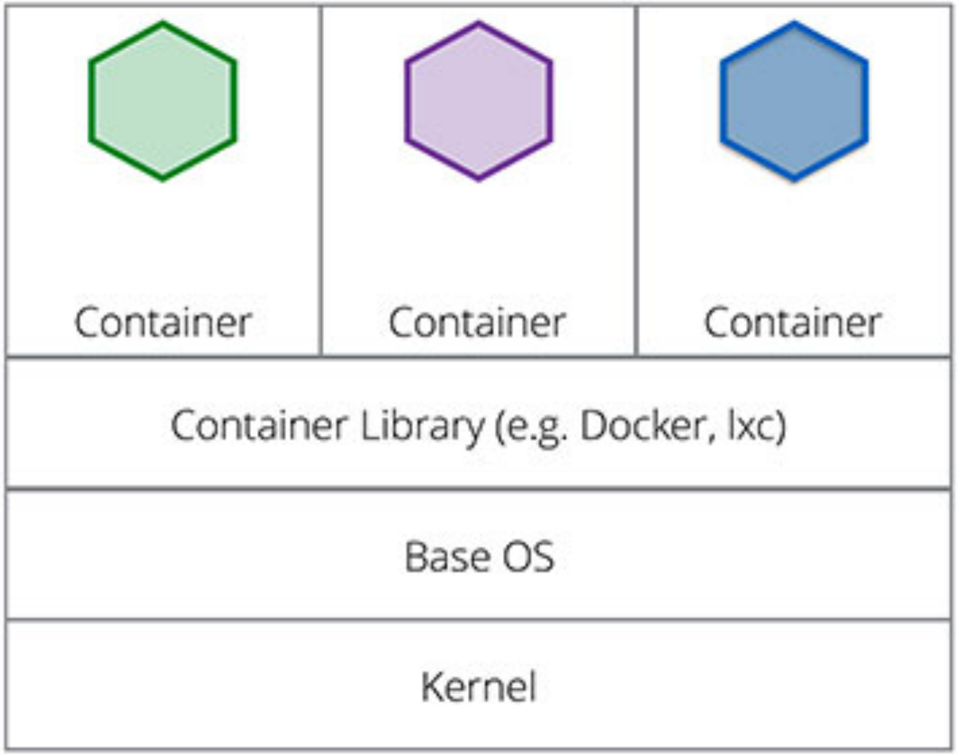
Container

Container

Container Library (e.g. Docker, lxc)

Base OS

CONTAINERISATION



Container

Container

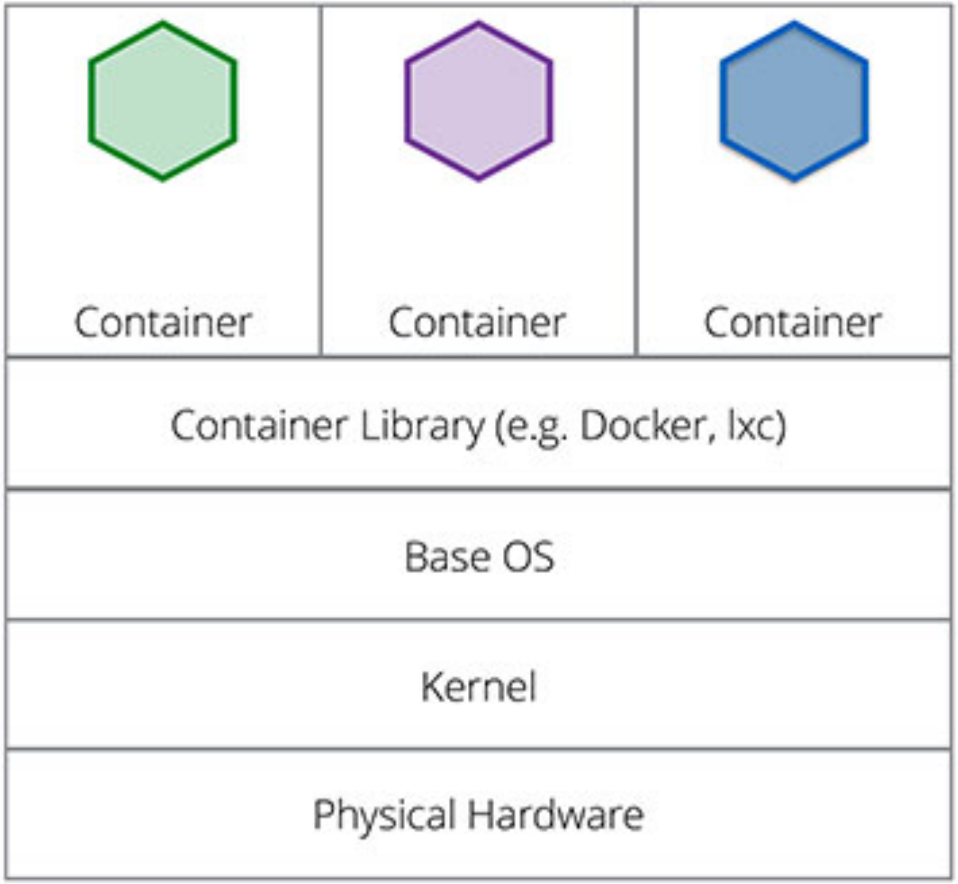
Container

Container Library (e.g. Docker, lxc)

Base OS

Kernel

CONTAINERISATION



Container

Container

Container

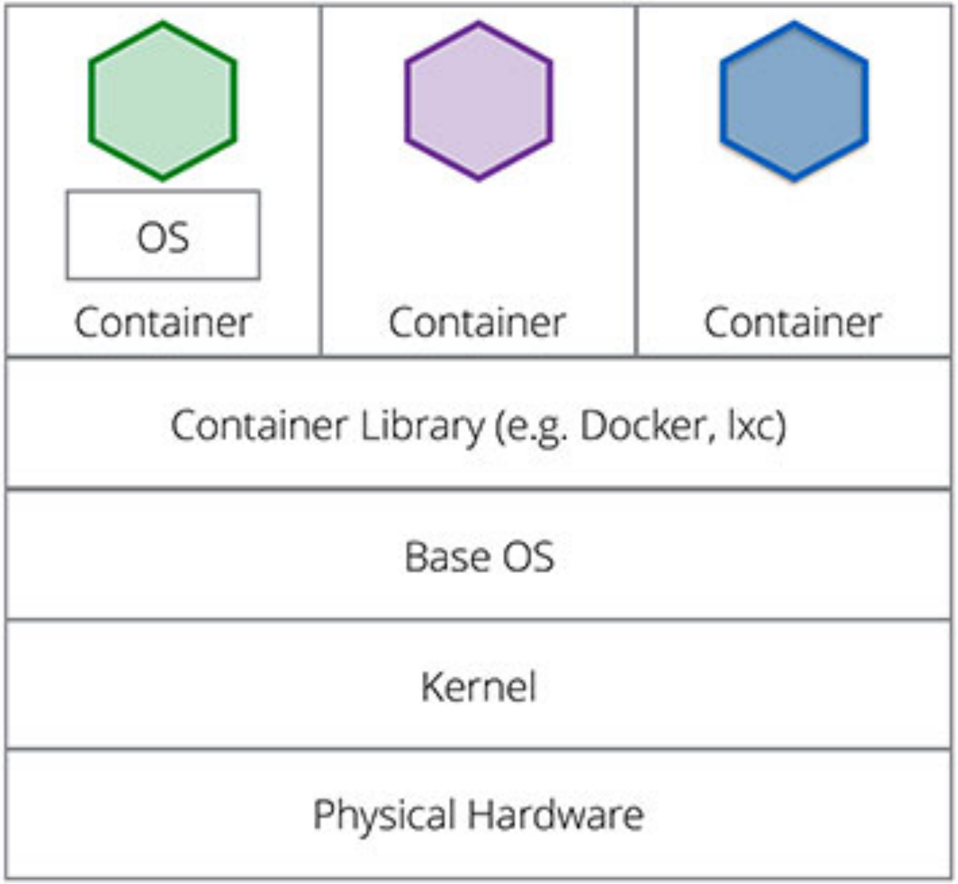
Container Library (e.g. Docker, lxc)

Base OS

Kernel

Physical Hardware

CONTAINERISATION



OS

Container

Container

Container

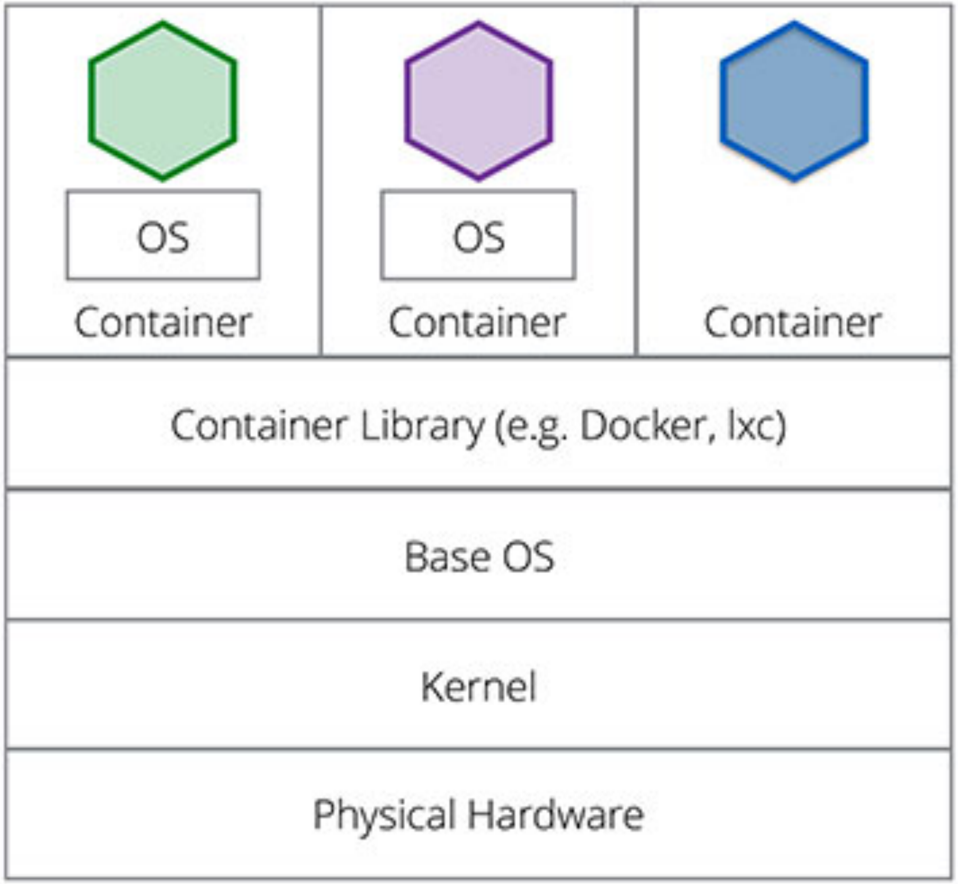
Container Library (e.g. Docker, lxc)

Base OS

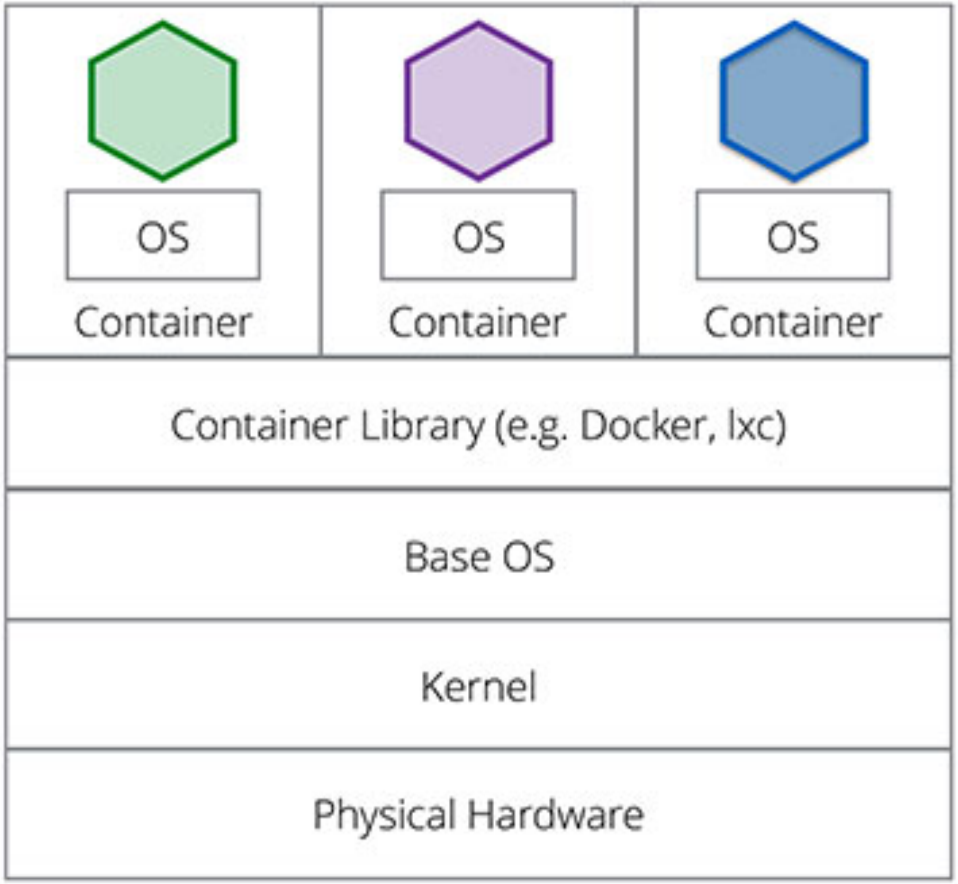
Kernel

Physical Hardware

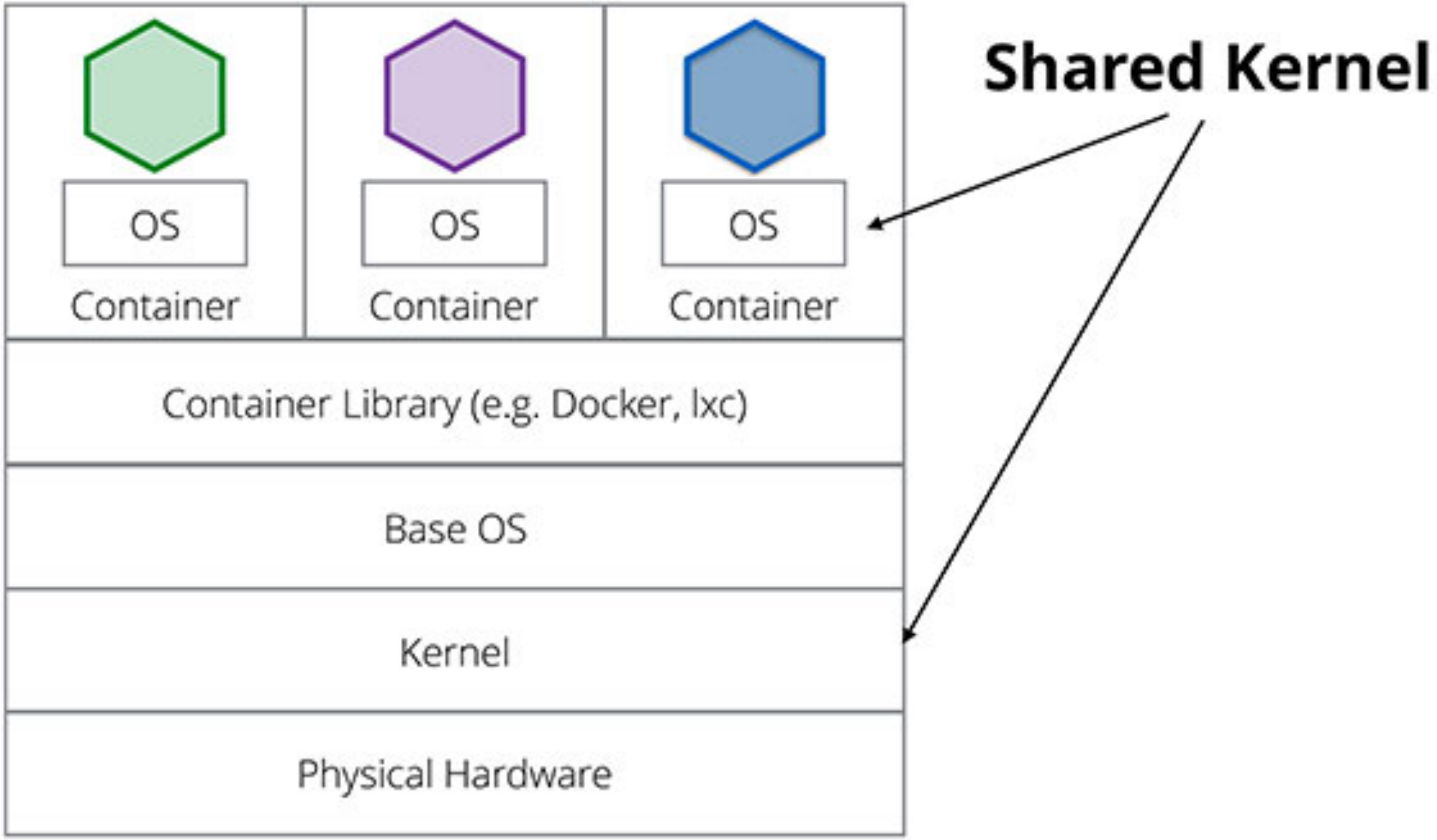
CONTAINERISATION



CONTAINERISATION



CONTAINERISATION



Shared Kernel

OS

Container

OS

Container

OS

Container

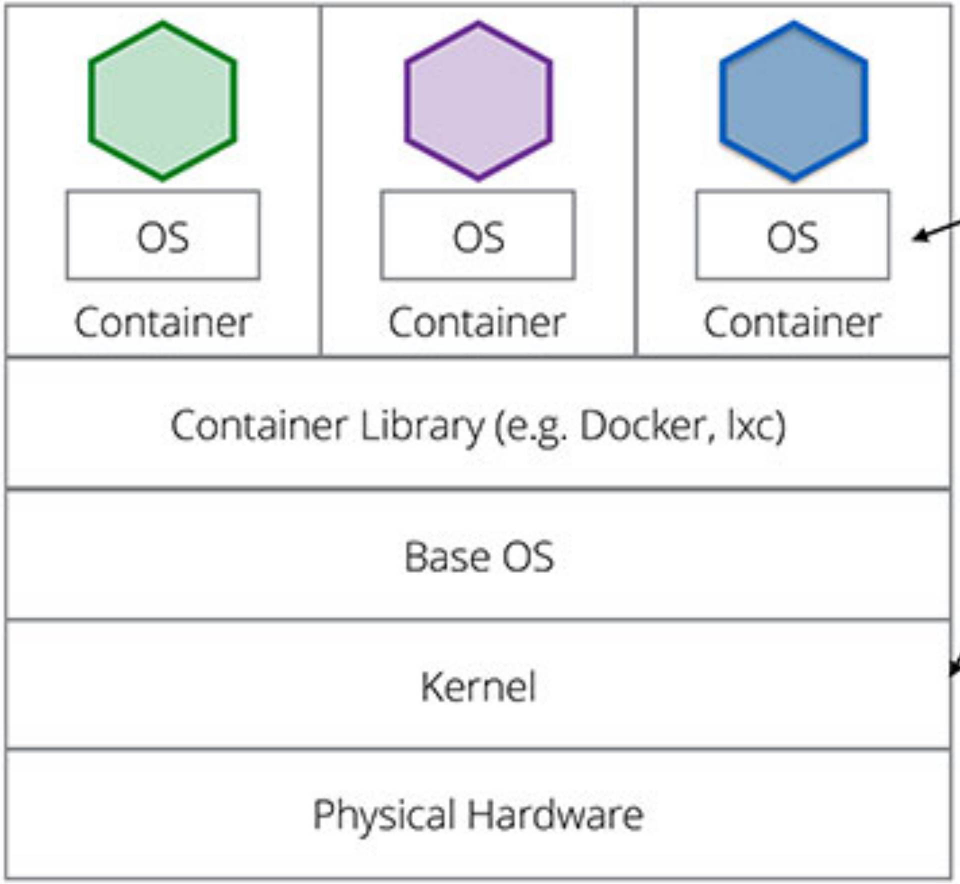
Container Library (e.g. Docker, lxc)

Base OS

Kernel

Physical Hardware

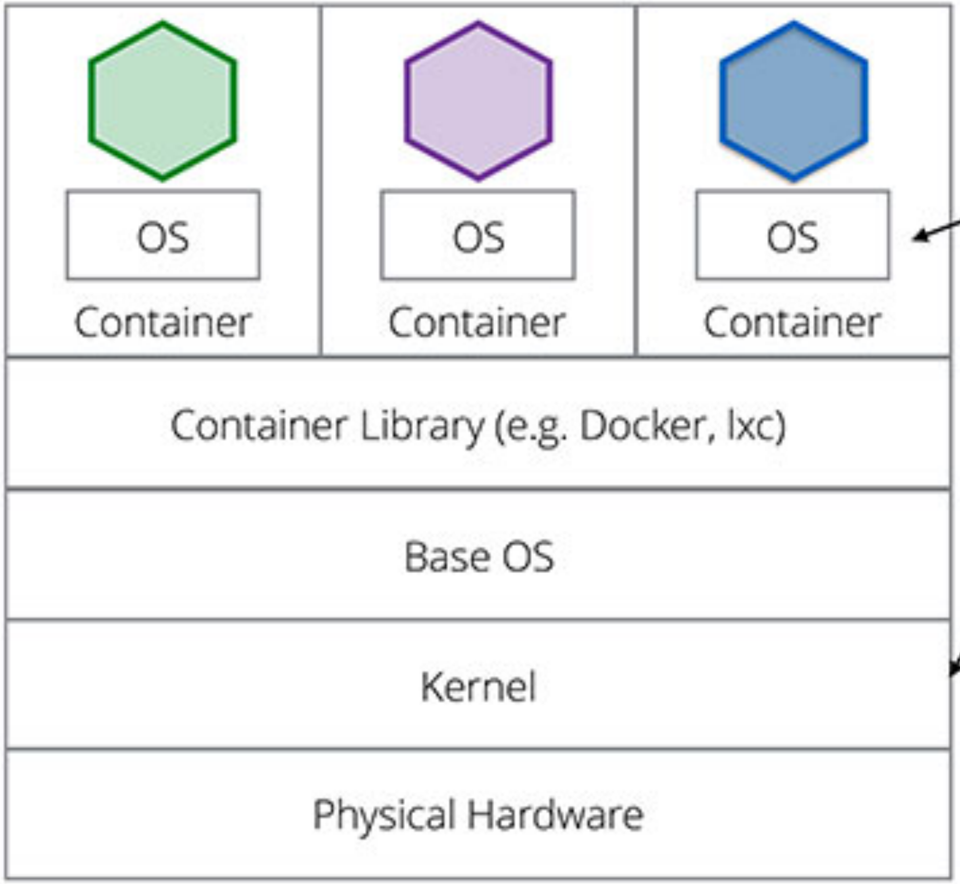
CONTAINERISATION



Shared Kernel

Much faster provisioning

CONTAINERISATION



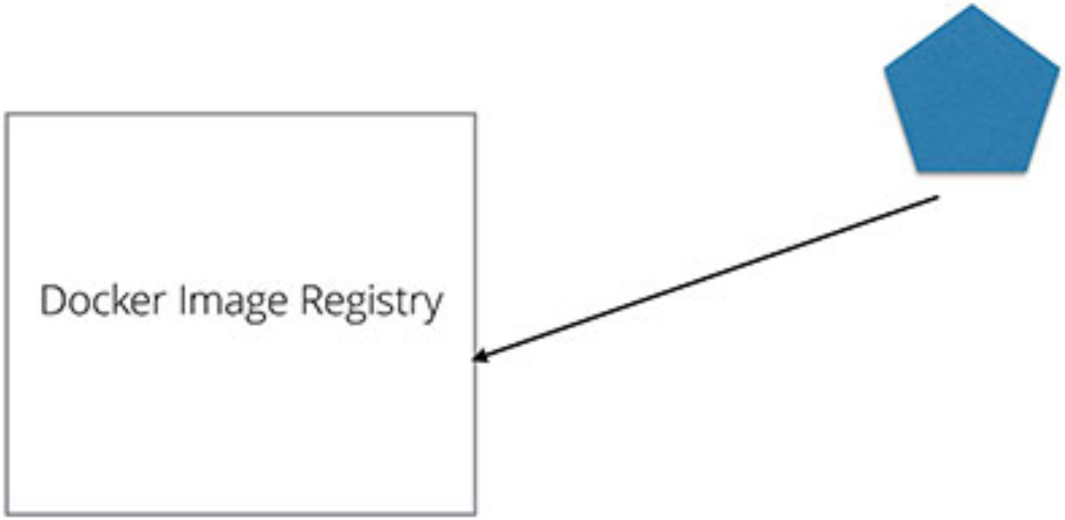
Shared Kernel

Much faster provisioning

Poorer Isolation

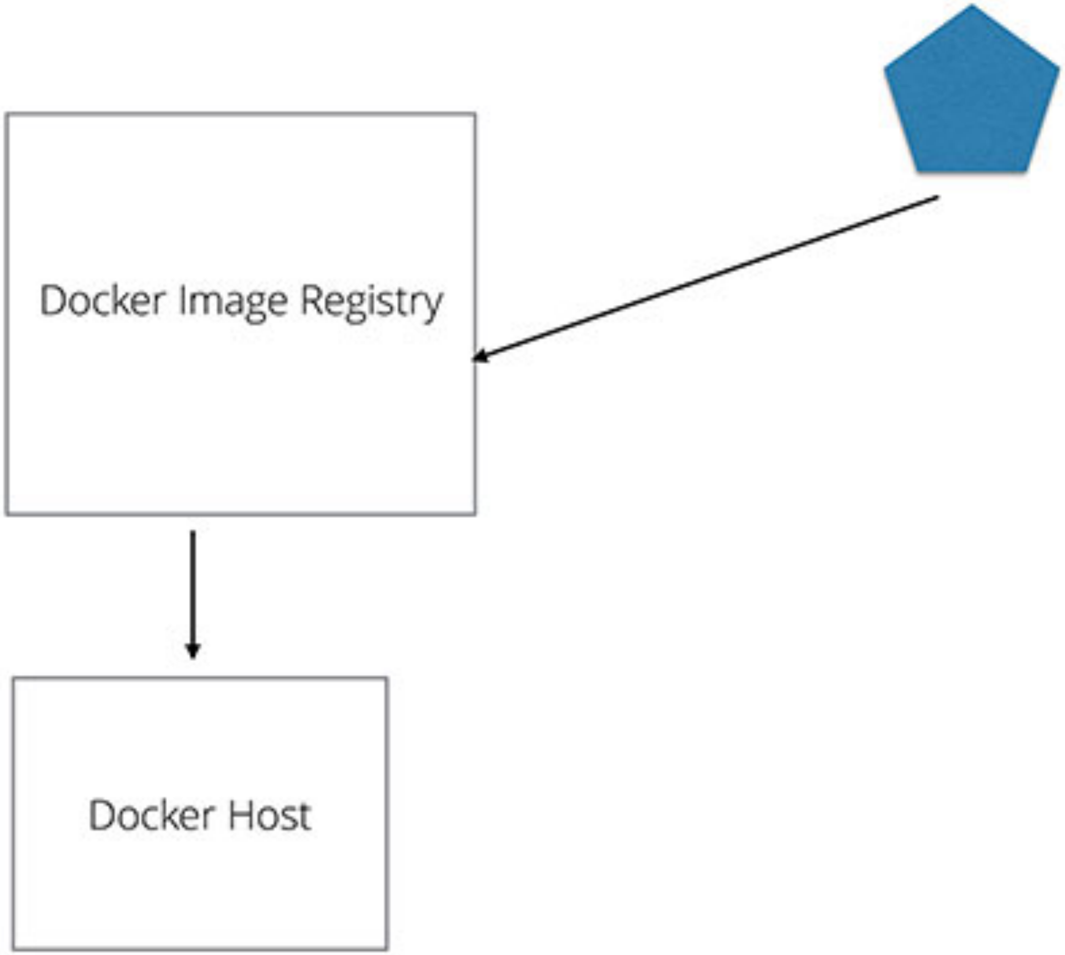
DOCKER

DOCKER



Docker Image Registry

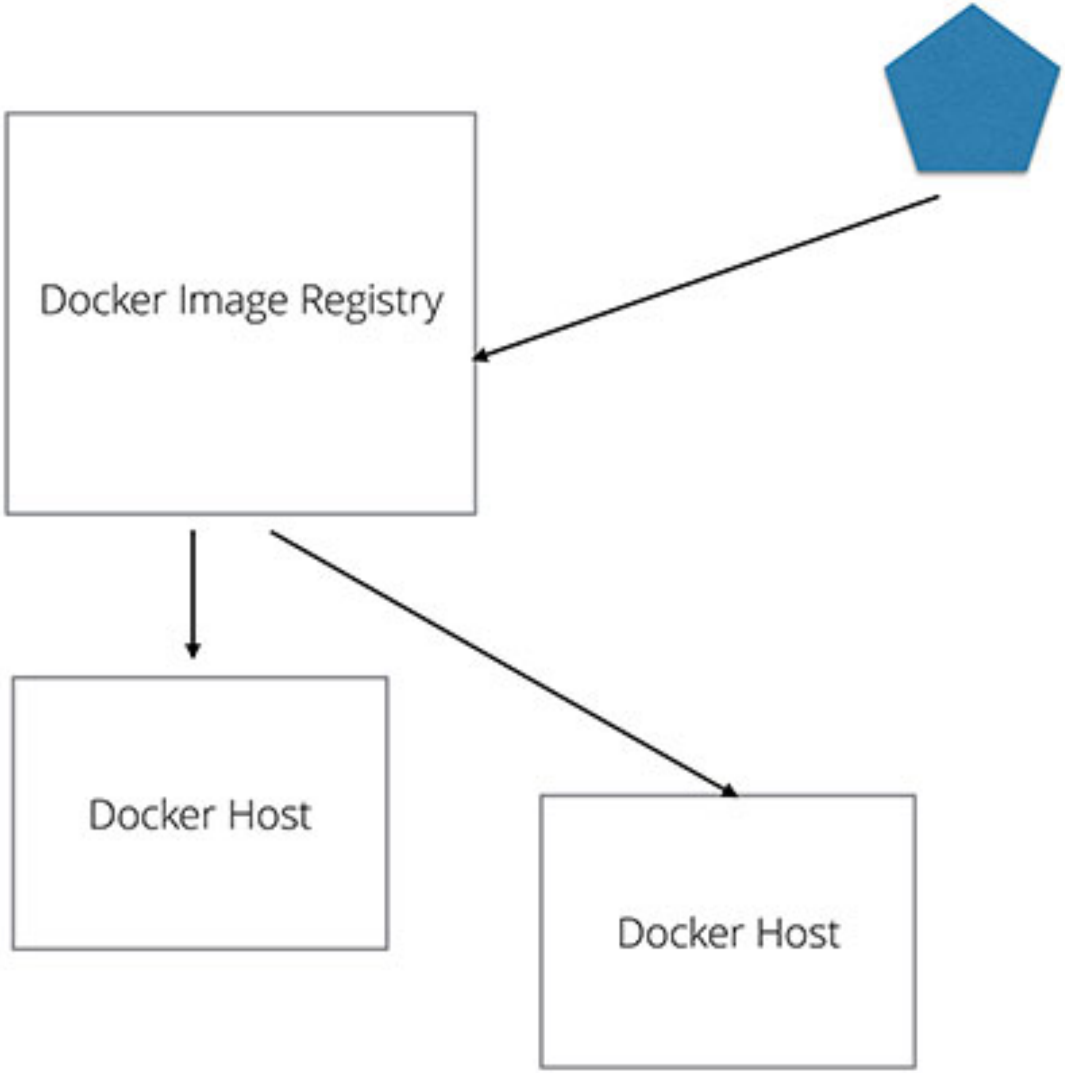
DOCKER



Docker Image Registry

Docker Host

DOCKER

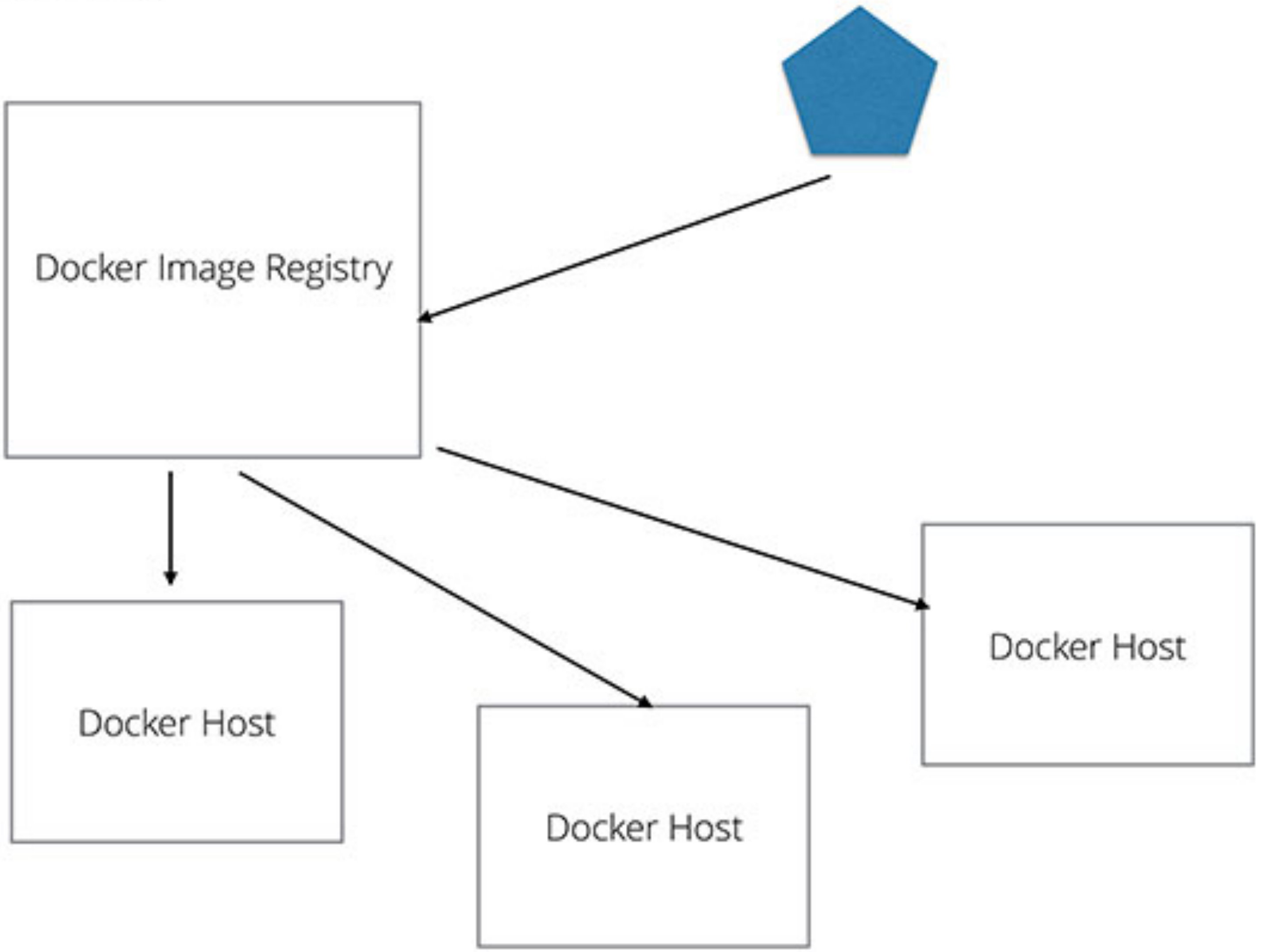


Docker Image Registry

Docker Host

Docker Host

DOCKER



docker

Cost of isolated hosts is reduced...

Cost of isolated hosts is reduced…

…in terms of effort…

Cost of isolated hosts is reduced…

…in terms of effort…

…and computing resources

Docker!

Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Docker!

? Easy to create

Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Docker!

? Easy to create

✓ Easy to deploy

Abstract out the tech stack

Good for dev, good for ops

Docker!

? Easy to create

✓ Easy to deploy

✓ Abstract out the tech stack

Good for dev, good for ops

Docker!

? Easy to create

✓ Easy to deploy

✓ Abstract out the tech stack

? Good for dev, good for ops

So, ditch what you have for a new platform?

Docker Swarm

MESOS

kubernetes

by Google

RENA
MONROVIA

Summary

Keep builds separate

Summary

Keep builds separate

Be careful about shared code

Summary

Keep builds separate

Be careful about shared code

Think of adopting new artifact choices

Summary

Keep builds separate

Be careful about shared code

Think of adopting new artifact choices

In the longer term, consider new platforms