

MONOLITHS

TO MICROSERVICES

Sam Newman

Day Two

Overview

- Questions from day 1 & exercises
- Service collaboration options
- Build & Deployment
- Monitoring

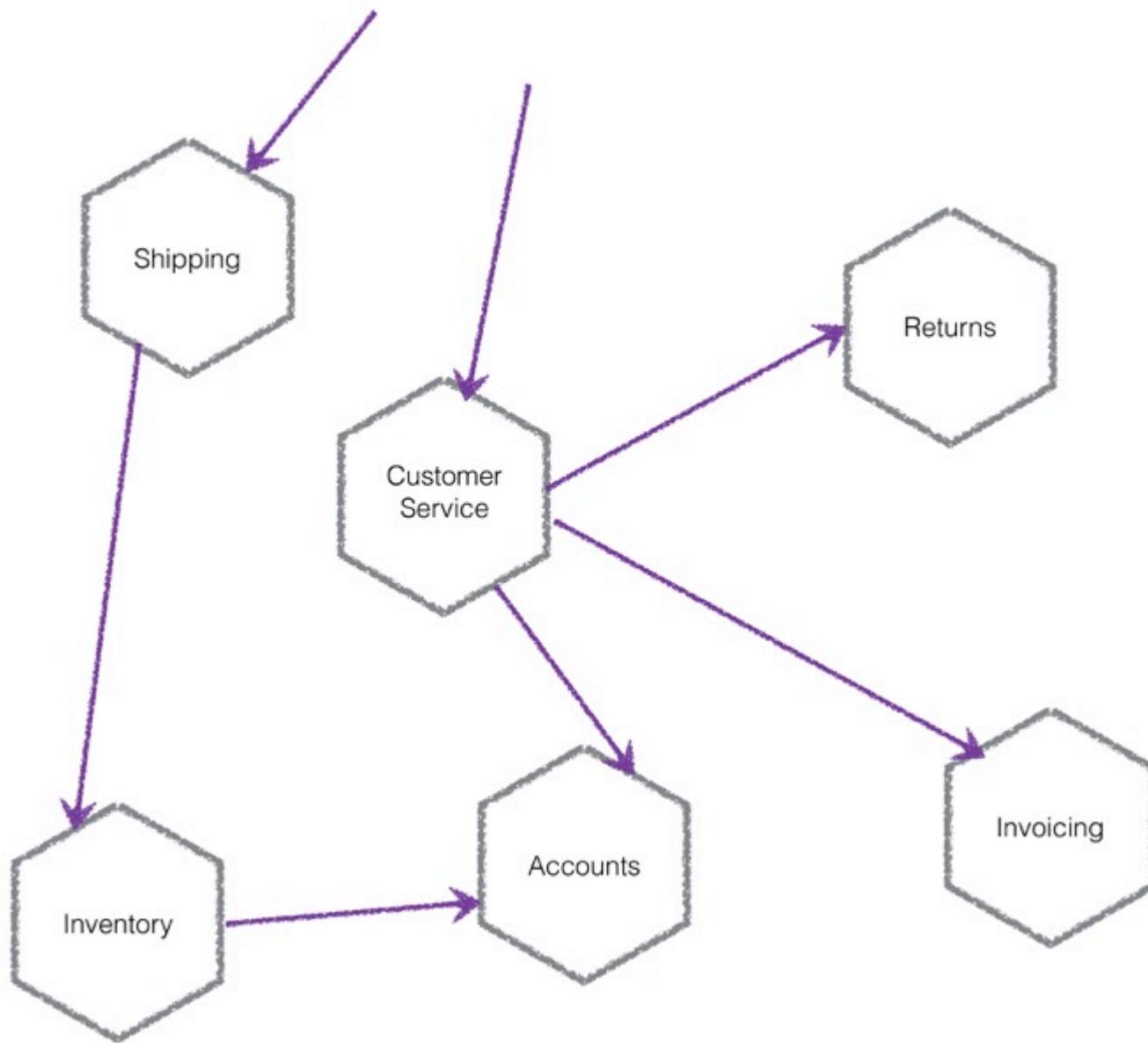
MONOLITHS

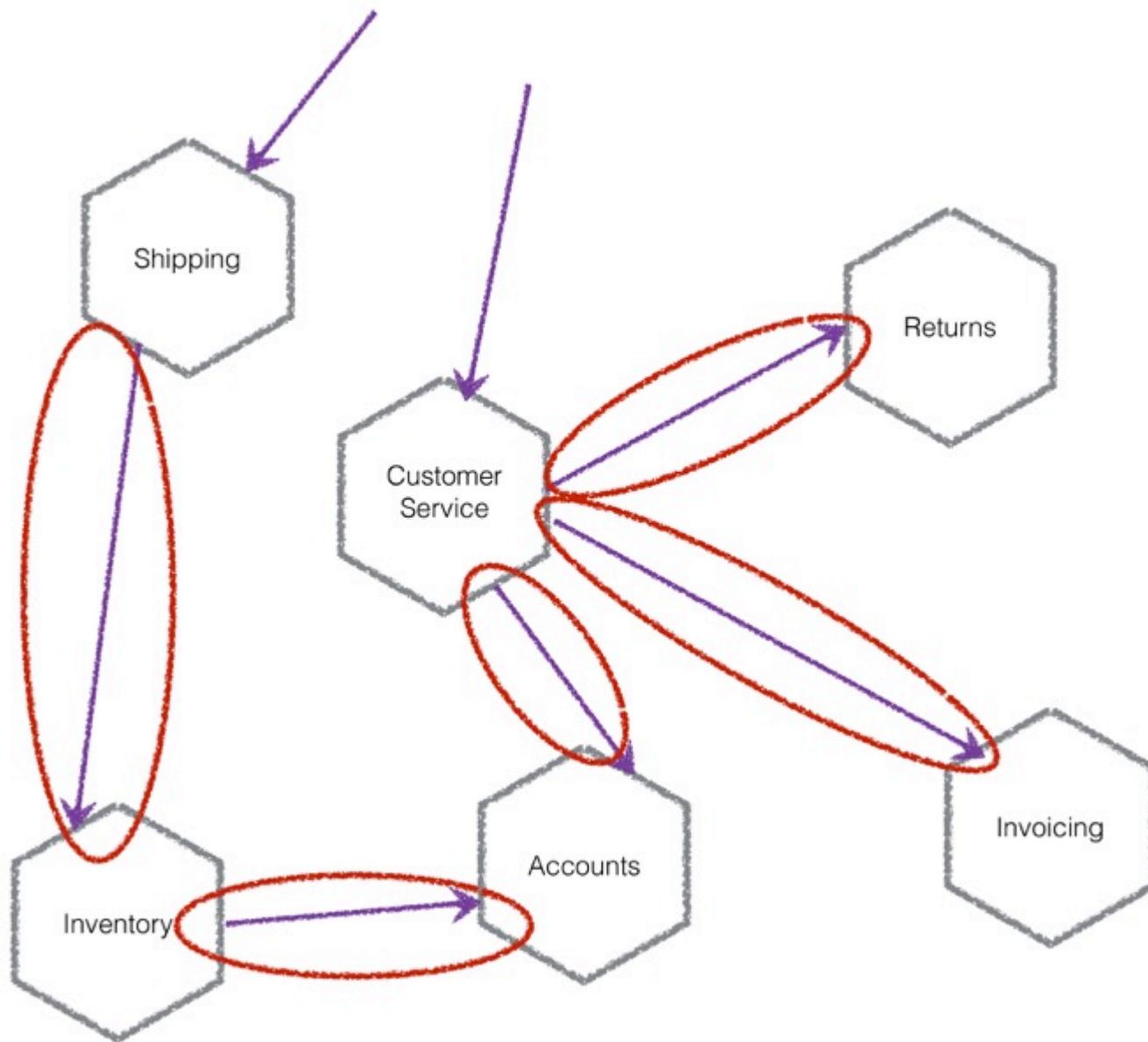
TO MICROSERVICES

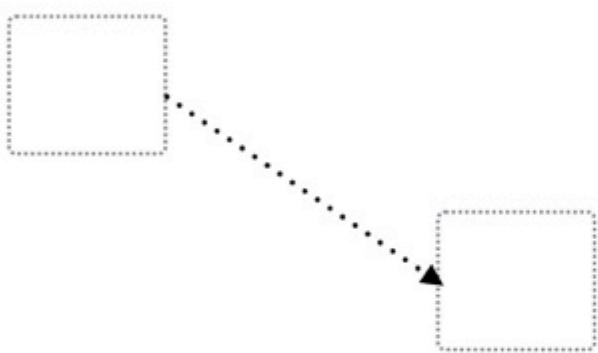
Sam Newman

Service Interactions



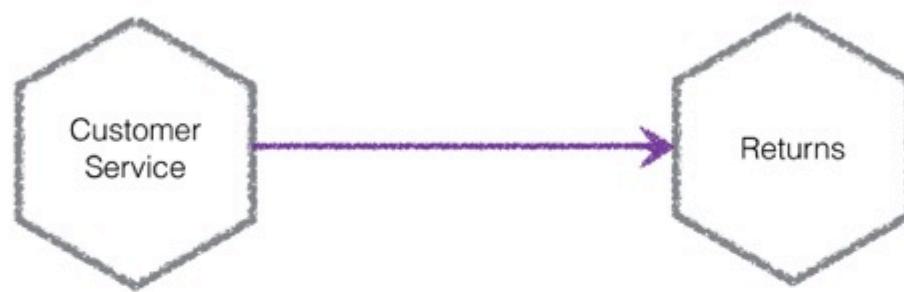






Cost of change
is low

Easy to reason
about



Changing a call ?



Changing a call ?
potential API breakage



Changing a call ?

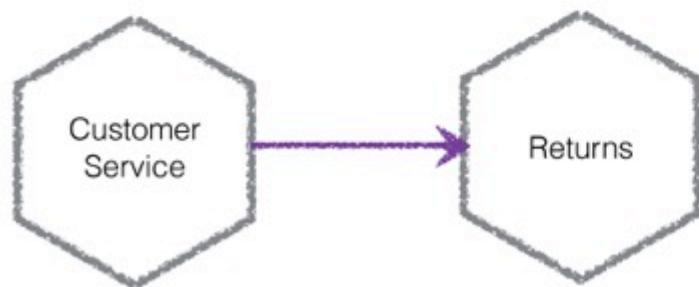
potential API breakage

two deployments to rollout a change

Are calls between services like
calls inside a process boundary?

PERFORMANCE IMPLICATIONS

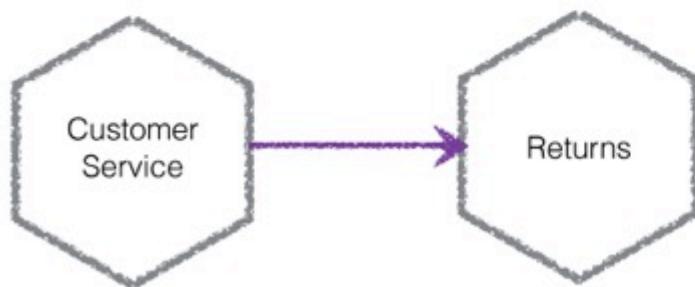




PERFORMANCE IMPLICATIONS



Per-call overhead is very low

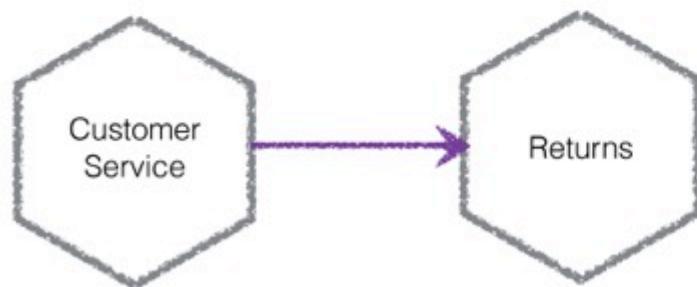


PERFORMANCE IMPLICATIONS



Per-call overhead is very low

Movement of data by reference

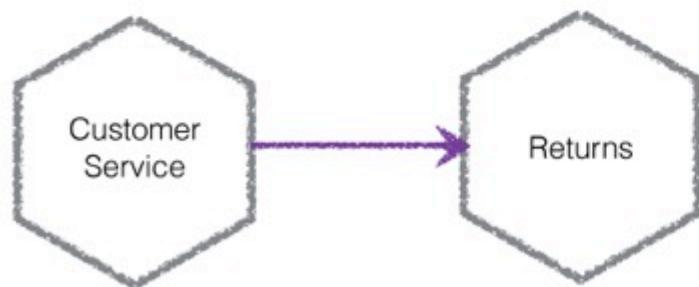


PERFORMANCE IMPLICATIONS



Per-call overhead is very low

Movement of data by reference



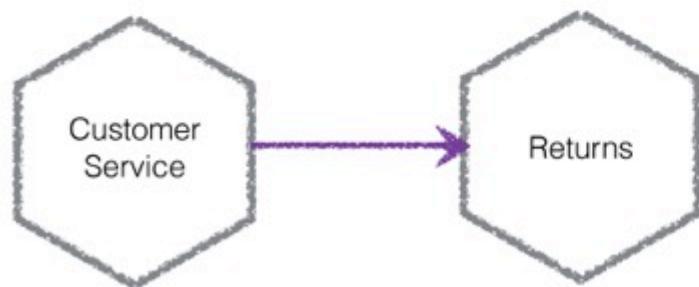
Call overhead can be very high

PERFORMANCE IMPLICATIONS



Per-call overhead is very low

Movement of data by reference

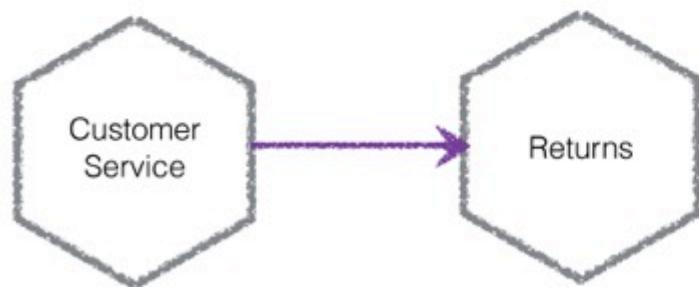


Call overhead can be very high

Data moved by marshalling, or handing off to an external store

HANDLING ERRORS

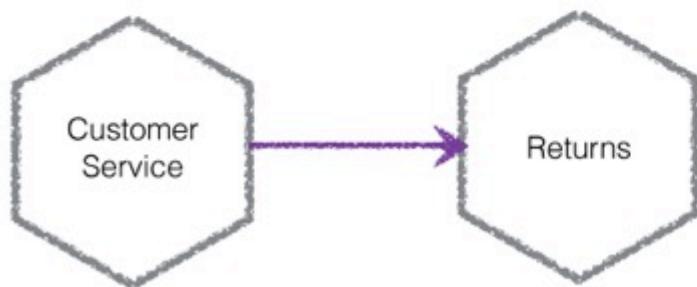




HANDLING ERRORS



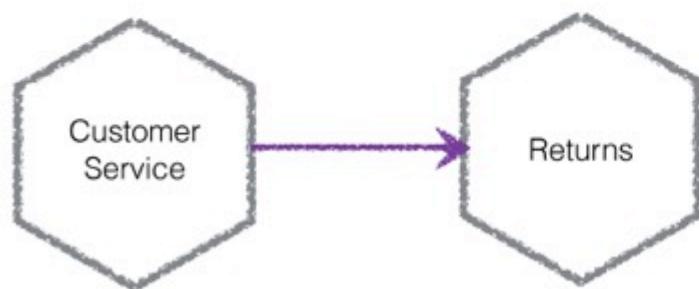
Errors straightforward



HANDLING ERRORS



Errors straightforward

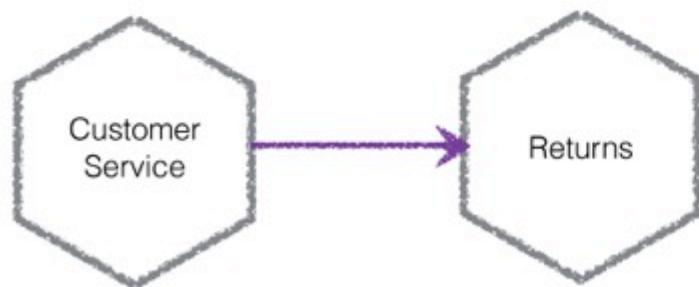


Timeouts

HANDLING ERRORS



Errors straightforward



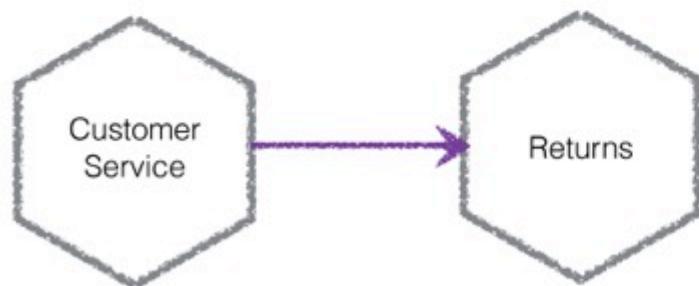
Timeouts

Downstream outage

HANDLING ERRORS



Errors straightforward

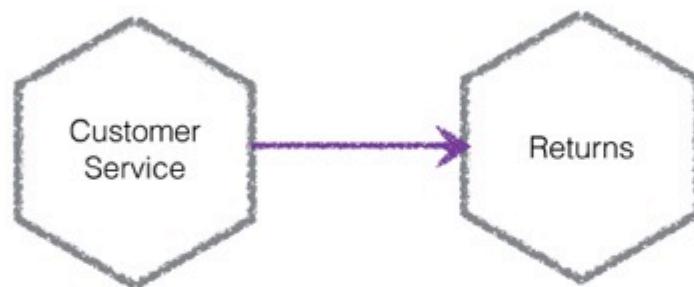


Timeouts

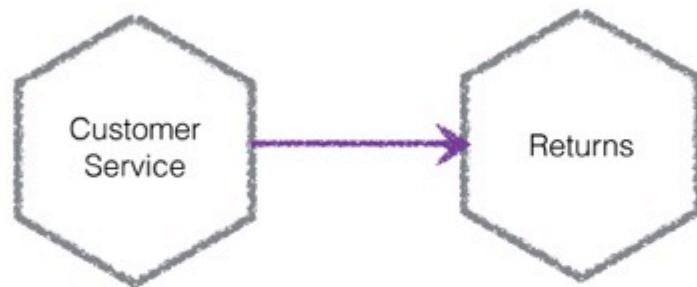
Downstream outage

Difference between client and server errors

4XX vs 5XX status codes

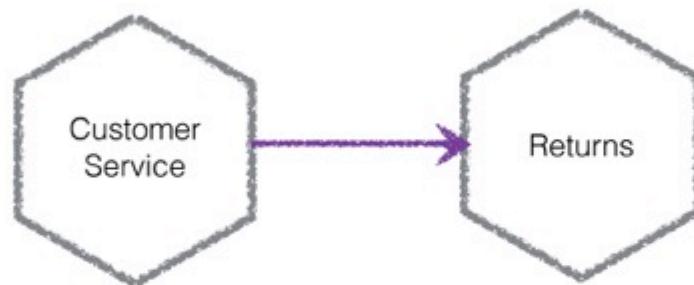


4XX vs 5XX status codes



4XX - you did something wrong!

4XX vs 5XX status codes



4XX - you did something wrong!

5XX - there is something wrong at my end...

https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

4xx Client Error ([edit])	
The class of status codes is intended for the occurrence in which the client makes an invalid request. The server should not be blamed.	
400 Bad Request	The server cannot or will not process the request due to an apparent client error (e.g., malformed request syntax, too large size, invalid request message).
401 Unauthorized ([RFC 7235-1])	Similar to 403 Forbidden, but specifically for use when authentication is required and has either not been provided. The response must include the required authentication information.
402 Payment Required	Note: Some erroneous HTTP 401 when an IP address is banned from the website (usually the website domain) and that specific address is refused point reserved for future use. The original intention was that this code might be used as part of some form of digital cash or micropayment scheme, or that it be reserved for future use.
403 Forbidden	The request was valid, but the server is refusing to respond to it. The user might be logged in but does not have the necessary permissions for the requested resource.
404 Not Found	The requested resource could not be found but may be available in the future. Subsequent requests by the client are permissible.
405 Method Not Allowed	A request method is not supported for the requested resource; for example, a GET request on a form which requires data to be presented via POST, or a HEAD request on a form.
406 Not Acceptable	The requested resource is capable of generating encodings not acceptable according to the Accept headers sent in the request. See Content negotiation.
407 Proxy Authentication Required ([RFC 7235-1])	Proxy authentication is required to access the resource.
408 Request Timeout	The client must authenticate itself with the proxy.
410 Length Required	The server timeout waiting for the request. According to HTTP specifications: "The client did not produce a request within the time that the server was prepared to accept it." This indicates that there was no explicit length of the resource, which is required for the requested resource.
411 Range Not Satisfiable	The request does not contain one of the preconditions that the requester put on the request.
412 Precondition Failed ([RFC 7232])	The request is larger than the server is willing or able to process. Previously called: Preempted Entity Too Large.
413 URI Too Long ([RFC 7231-1])	The URI provided is too long for the server to process. Often the result of too much data being encoded as a query-string of a GET request, mismatch or unsupported media type.
414 Unprocessable Entity	The request entity has a media type which the service resource does not support. For example, the client uploads an image as image/segment but the service does not support image/segment.
415 Unsupported Media Type	The client has used a portion of a file (byte serving), but the service cannot supply that portion. For example, if the client asked for a part of the file #415.
416 Expectation Failed ([RFC 7234])	The client did not meet the requirements of the Expect request header field.
417 I'm a Teapot ([RFC 2324])	This code was defined in 1988 as one of the traditional HTTP Agent Fools pieces, in RFC 2324's Hyper Text Coffee Pot Control Protocol, and is no longer used.
420 Misdirected Request ([RFC 7231-1])	The request was directed at a server that is not able to produce a response (for example because a connection failed).
423 Unprocessable Entity ([RFC 4918])	The request was well-formed but was unable to be followed due to semantic errors.
424 Locked ([RFC 4918])	The resource has a lock being assessed in its state.
425 Too Many Requests ([RFC 4918])	The client has issued too many requests in a given amount of time. Intended for use with rate-limiting schemes.
426 Upgrade Required ([RFC 4918])	The client needs to switch to a different protocol such as TLS 1.3, given in the Upgrade header field.
428 Precondition Failed ([RFC 6585])	The origin server requires the request to be conditional. Intended to prevent the "lost update" problem, where a client GETs a resource's state, modifies it, and then POSTs back to the origin server.
429 Too Many Requests ([RFC 6585])	The user has sent too many requests in a given amount of time. Intended for use with rate-limiting schemes.
430 Request Header Fields Too Large ([RFC 6585])	The server is unwilling to process the request because either an individual header field, or the header field collectively, exceed length.
451 Unavailable For Legal Reasons	A server is unwilling to accept the request because it legal does not allow delivery of the resource or the user does not have permission to access the resource.

28 4XX Error Codes

https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

4xx Client Error [edit]	
The class of status codes is intended for the occurrence in which the client makes an invalid request. The server should not consider the request invalid.	
400 Bad Request	
The server cannot or will not process the request due to an apparent client error (e.g., malformed request syntax, too large size, invalid request message).	
401 Unauthorized [RFC 7235-1]	
Similar to 403 Forbidden, but specifically for use when authentication is required and has failed or has not yet been provided. The response must include the required <code>WWW-Authenticate</code> header.	
Note: Some erroneous HTTP 401 when an IP address is banned from the website (usually the website domain) and that specific address is refused just because it's an IP.	
402 Payment Required	
Reserved for future use. The original intention was that this code might be used as part of some form of digital cash or micropayment scheme, or for the user to pay for the service.	
403 Forbidden	
The request was valid, but the server is refusing to respond to it. The user might be logged in but does not have the necessary permissions for the resource.	
404 Not Found	
The requested resource could not be found but may be available in the future. Subsequent requests by the client are permissible.	
405 Method Not Allowed	
A request method is not supported for the requested resource; for example, a GET request on a form which requires data to be presented via POST, or a PUT request on a file.	
406 Not Acceptable	
The requested resource is capable of generating content not acceptable according to the <code>Accept</code> headers sent in the request.	
407 Proxy Authentication Required [RFC 7235-1]	
The client must authenticate itself with the proxy.	
408 Request Timeout	
The server timed out waiting for the request. According to HTTP specifications, "The client did not produce a request within the time that the server was prepared to accept it." This means that there was no idle connection for the user, so either an <code>idleTimeout</code> value is set in the configuration or the user is updating his/her connection.	
409 Conflict	
Indicates that the resource requested is no longer available and will not be available again. This should be used when a resource has been inadvertently changed or destroyed, and a "404 Not Found" may be used instead.	
410 Length Required	
The request does not specify the length of the content, which is required for the requested resource.	
411 precondition Failed [RFC 7232-1]	
The server does not meet one of the preconditions that the requester put on the request.	
412 Precondition Failed [RFC 7232-1]	
The request is larger than the server is willing or able to process. Previously called <code>Request Entity Too Large</code> .	
413 URI Too Long [RFC 7231-1]	
The URI provided is too long for the server to process. Often the result of too much data being encoded as a query-string on a GET request, mismatch or unsupported media type.	
414 Unsupported Media Type	
The request entity has a media type which the service resource does not support. For example, the client uploads an image as <code>image/segment</code> but the <code>HTTP Range-Content-Format</code> header is <code>text/html</code> .	
The client has asked for a portion of the file (byte serving), but the server cannot supply that portion. For example, if the client asked for a part of the file #1000-10000, the server cannot supply the requirements of the <code>Accept-Range</code> header.	
415 I'm a Teapot [RFC 3261]	
This code was defined in 1996 as one of the traditional HTTP April Fools jokes, in RFC 3261, <code>Hyper Text Coffee Pot Control Protocol</code> , and is no longer supported.	
416 Misdirected Request [RFC 7231-1]	
The request was directed at a server that is not able to produce a response (for example because a connection reused).	
417 Expectation Failed [RFC 7231-1]	
The request failed because it is being assessed in batches.	
418 I'm a Teapot [RFC 6585]	
The request failed due to the failure of a previous request (e.g., a <code>PROPFIND</code>).	
421 Misunderstood [RFC 7231-1]	
The client misunderstood a different protocol such as <code>TLSv1.2</code> , given in the <code>Upgrade</code> header field.	
424 Precondition Required [RFC 6585]	
The origin server requires the request to be conditional. Intended to prevent the "lost update" problem, where a client GETs a resource's state, modifies it, and then sends a PUT.	
429 Too Many Requests [RFC 6585]	
The user has sent too many requests in a given amount of time. Intended for use with rate-limiting schemes.	
431 Request Header Fields Too Large [RFC 6585]	
451 Unreachable For Legal Reasons	
A server received a new request to a legal document for direct access to a resource. So it sent a response that includes the requested resource.	

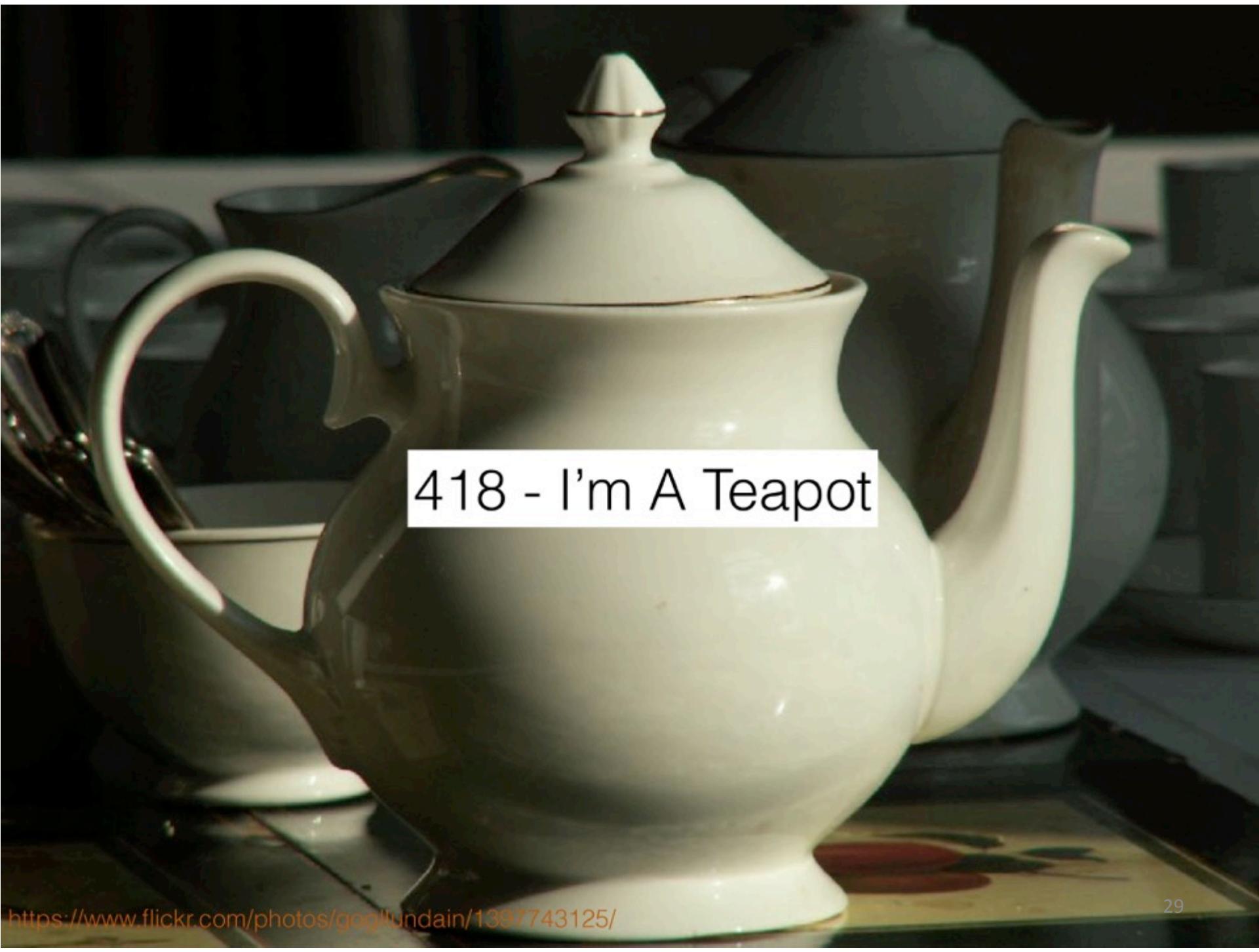
28 4XX Error Codes

5xx Server Error [edit]	
The server failed to fulfill an apparently valid request.	
Response status codes beginning with the digit "5" indicate cases in which the server is aware that it has encountered a user error (e.g., a client that has made a mistake). These response codes are applicable to any request method.	
500 Internal Server Error	
A generic error message, given when an unexpected condition was encountered and no more specific message is suitable.	
501 Not Implemented	
The server either does not recognize the request method, or it lacks the ability to fulfill the request. Usually this implies that the server is acting as a gateway or proxy and received an invalid response from the upstream server.	
502 Bad Gateway	
The server was acting as a gateway or proxy and received an invalid response from the upstream server.	
503 Service Unavailable	
The server is currently unavailable (because it is overloaded or down for maintenance). Generally, this is a temporary condition.	
504 Gateway Timeout	
The server was acting as a gateway or proxy and did not receive a timely response from the upstream server.	
505 HTTP Version Not Supported	
The server does not support the HTTP protocol version used in the request.	
506 Variant Also Negotiates [RFC 2295]	
Transparent content negotiation for the request results in a circular reference.	
507 Insufficient Storage (WebDAV; RFC 4918)	
The server is unable to store the representation needed to fulfill the request.	
508 Loop Detected (WebDAV; RFC 5842)	
The server detected an infinite loop while processing the request (sent in lieu of 208 Already Reported).	
510 Not Extended (RFC 2781)	
Further extensions to the request are required for the server to fulfill it.	
511 Network Authentication Required (RFC 6585)	
The client needs to authenticate to gain network access. Intended for use by interesting proxies used to control access.	

11 5XX Error Codes

https://en.wikipedia.org/wiki/List_of_HTTP_status_codes





418 - I'm A Teapot





410 - Gone

Keep it simple

Simple = Synchronous, Request
Response Communication

Synchronous

Synchronous

Asynchronous

Synchronous

Block and wait

Asynchronous

Synchronous

Block and wait

Asynchronous

Fire and (maybe) forget

Synchronous

Block and wait

Asynchronous

Fire and (maybe) forget

Simple to reason about

Synchronous

Block and wait

Asynchronous

Fire and (maybe) forget

Simple to reason about

Technology more
straight forward

Synchronous

Block and wait

Simple to reason about

Technology more
straight forward

Asynchronous

Fire and (maybe) forget

Great for long-
running jobs

Synchronous

Block and wait

Simple to reason about

Technology more
straight forward

Asynchronous

Fire and (maybe) forget

Great for long-
running jobs

And low latency too!

Synchronous

Block and wait

Simple to reason about

Technology more
straight forward

Asynchronous

Fire and (maybe) forget

Great for long-
running jobs

And low latency too!

More complex

Collaboration Styles

Collaboration Styles

Request/Response

Collaboration Styles

Request/Response

Event-based

Collaboration Styles

Request/Response

Initiate a request, expect
a response

Event-based

Collaboration Styles

Request/Response

Initiate a request, expect
a response

Event-based

Things happen,
things react

Request/Response

Request/Response

Event-based

Request/Response

Event-based

Synchronous

Request/Response

Event-based

Synchronous

Asynchronous

Request/Response

Event-based

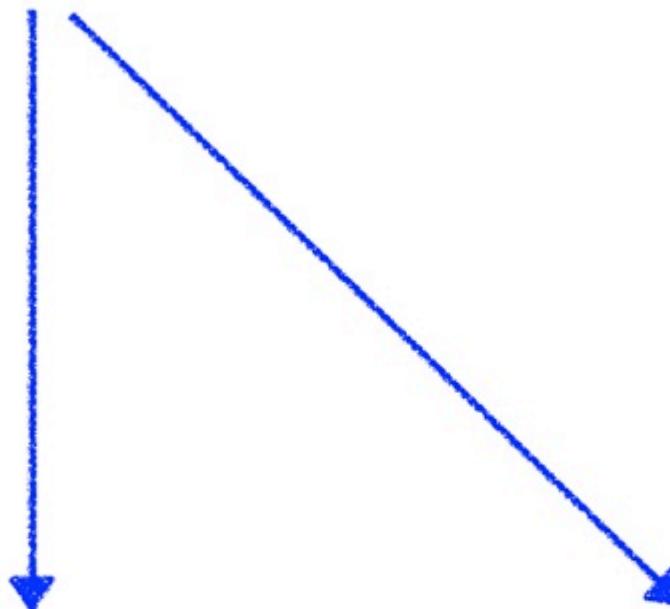


Synchronous

Asynchronous

Request/Response

Event-based



Synchronous

Asynchronous

Request/Response



Synchronous

Event-based



Asynchronous

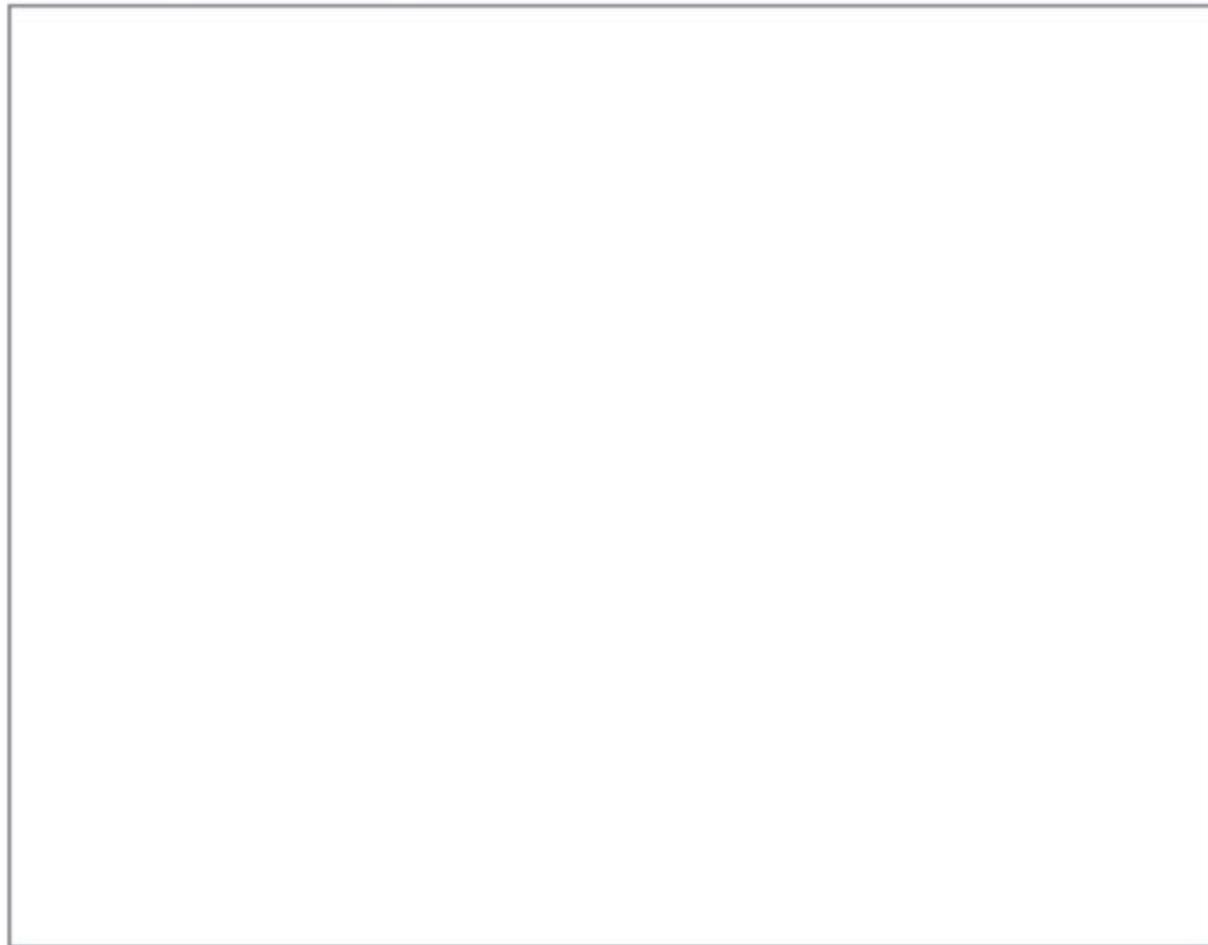


Request/Response



Request/Response

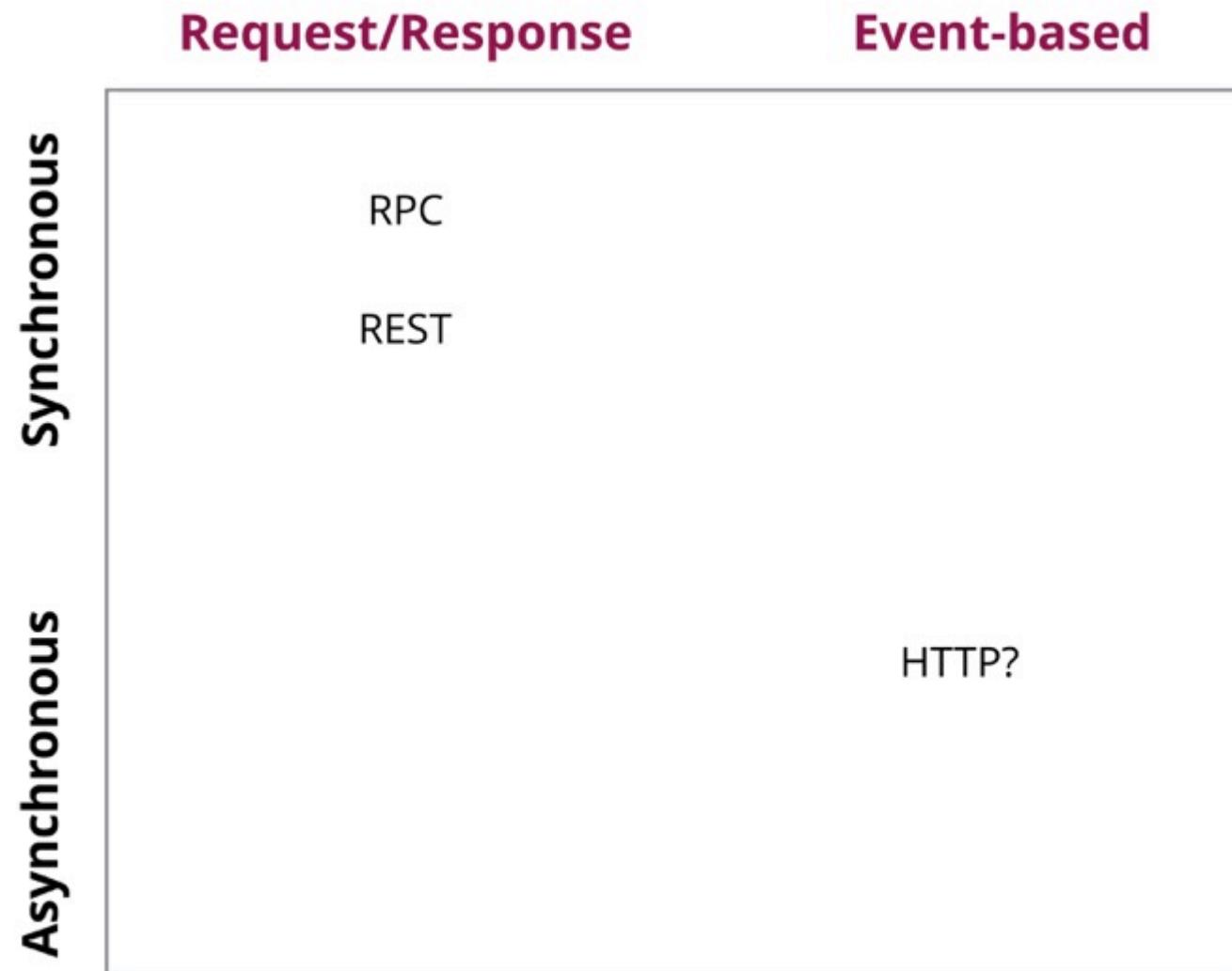
Event-based

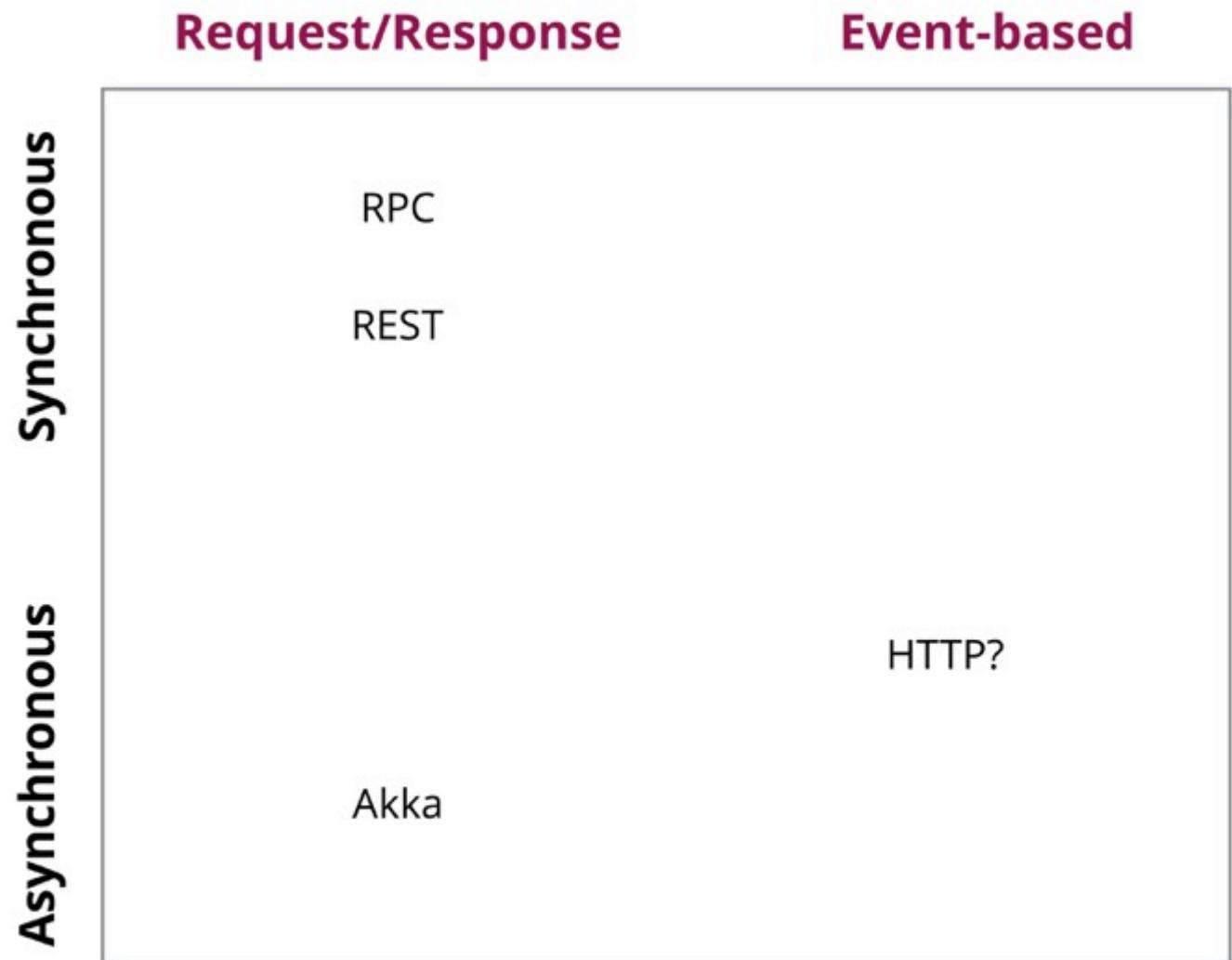


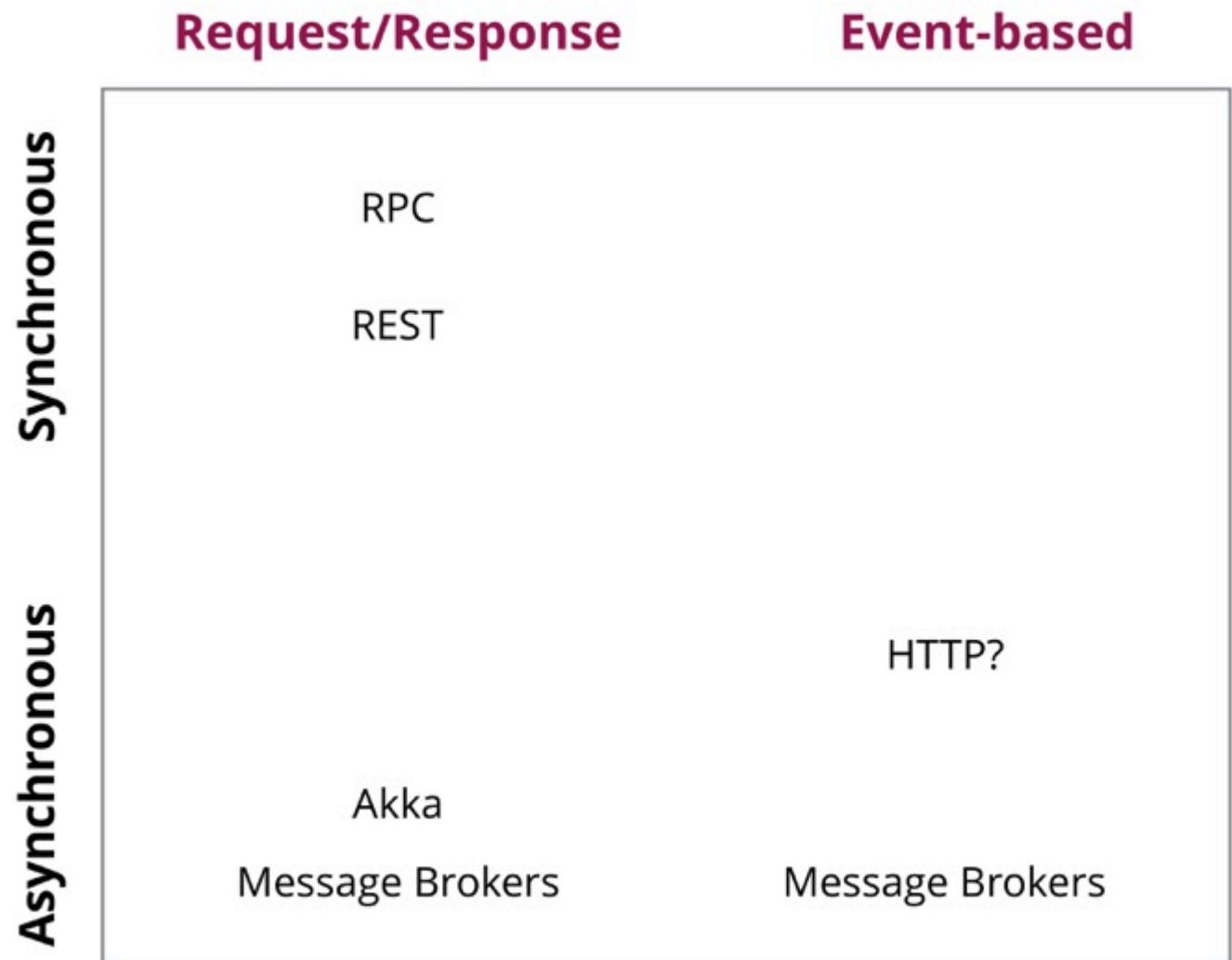
	Request/Response	Event-based
Synchronous		

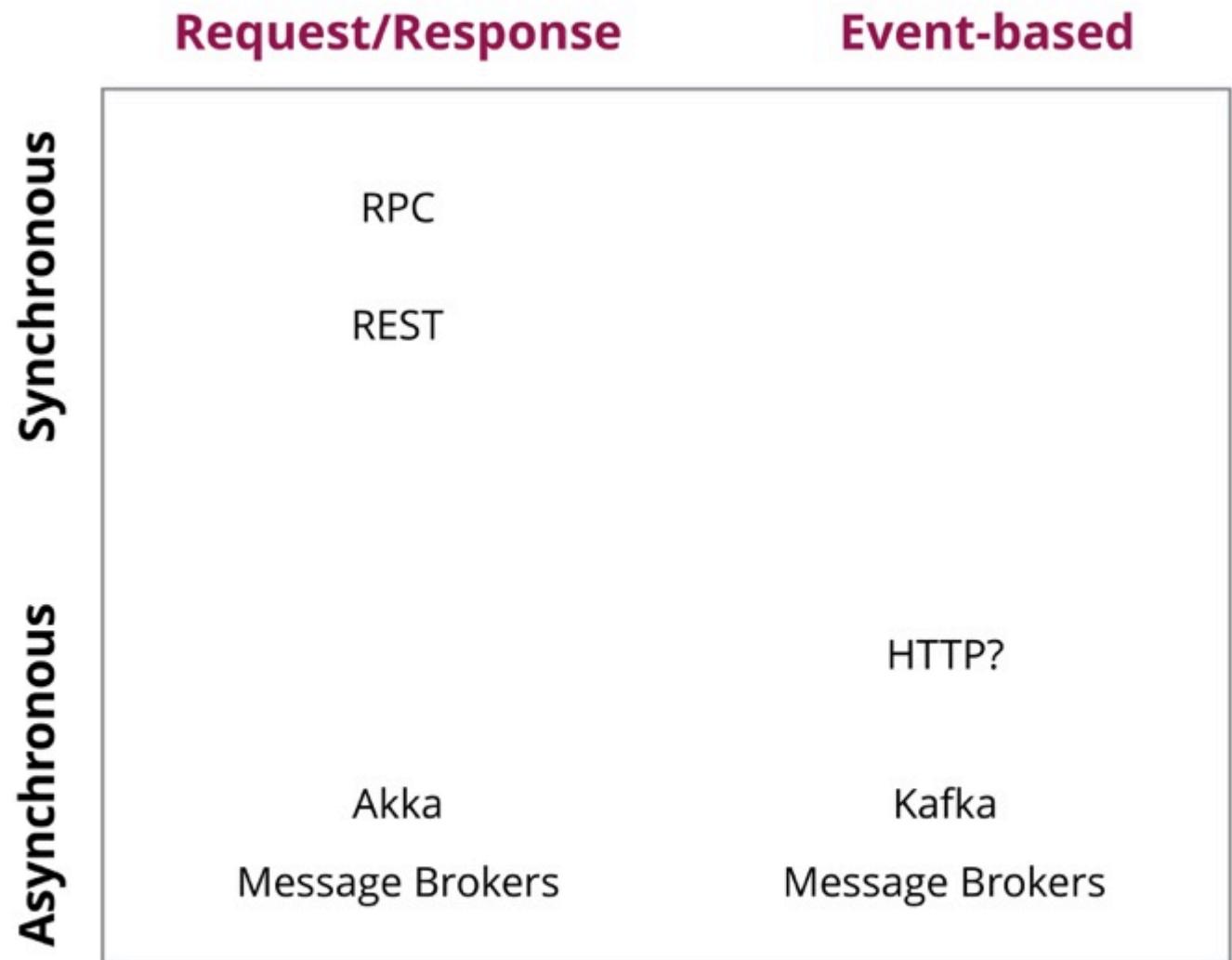
	Request/Response	Event-based
Synchronous		
Asynchronous		











Simple = Synchronous, Request
Response Communication

Simple = Synchronous, Request
Response Communication

HTTP

Very well supported

Very well supported

Good error handling semantics

Very well supported

Good error handling semantics

Cache controls

Very well supported

Good error handling semantics

Cache controls

Easy to scale - good, reliable
(boring?) technology

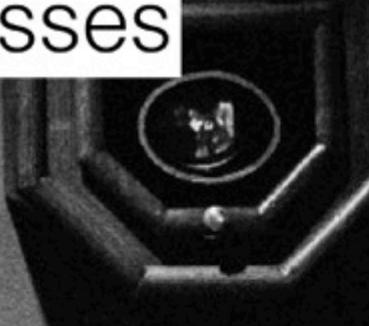
Where does “simple” stop being
good enough?



<https://www.flickr.com/photos/inthe-arena/15501000164/>

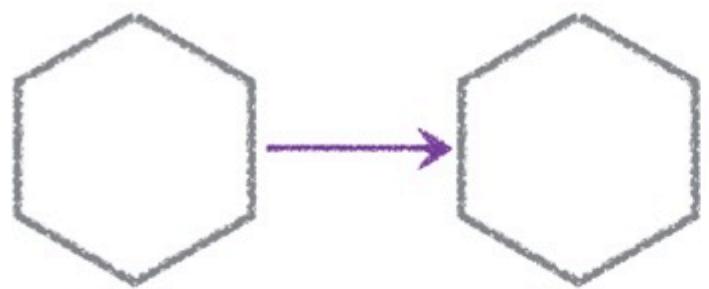


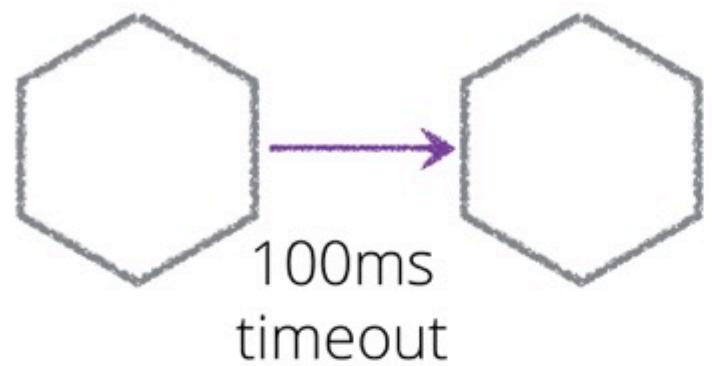
Long-lived processes

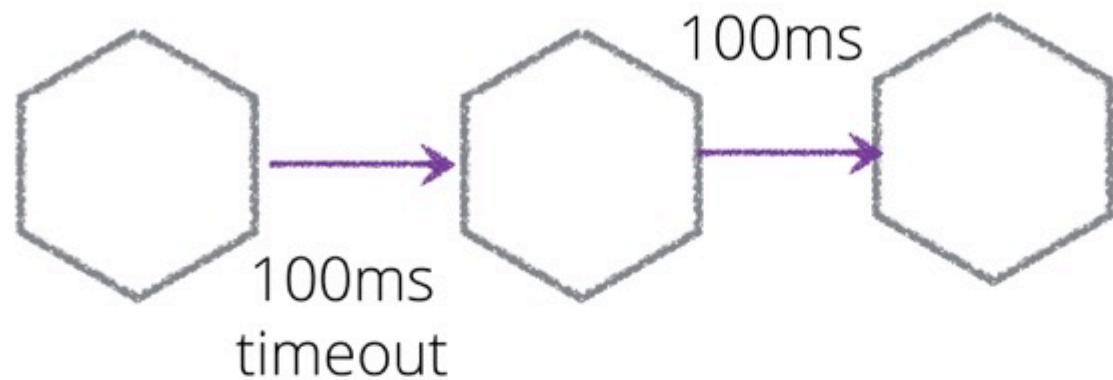


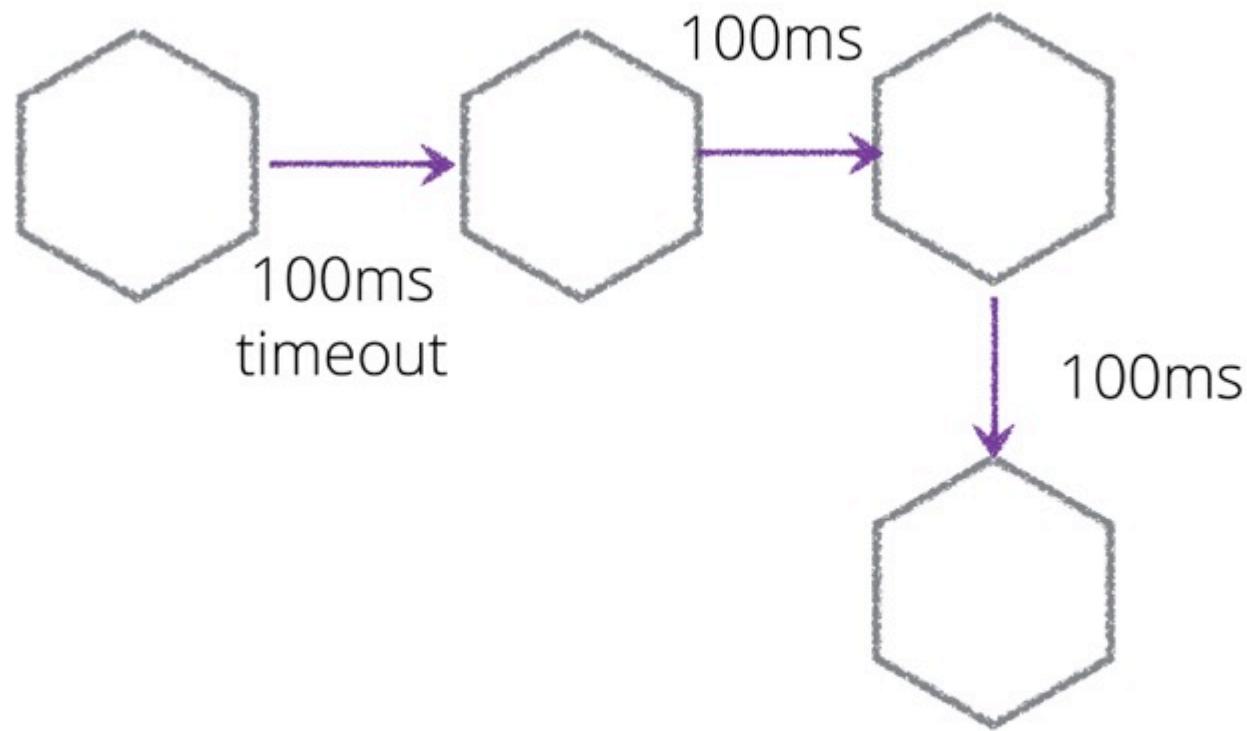


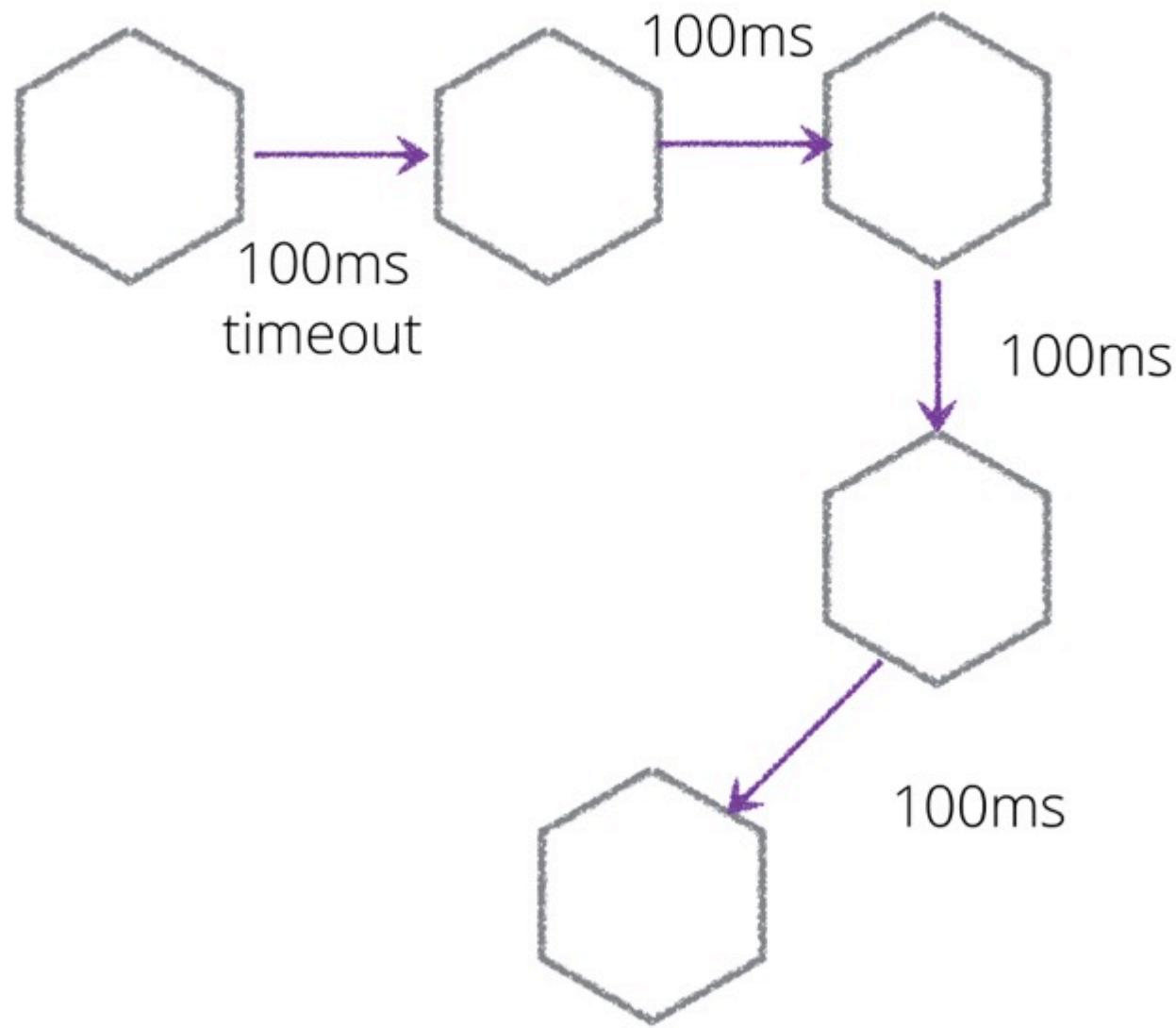
Call overhead

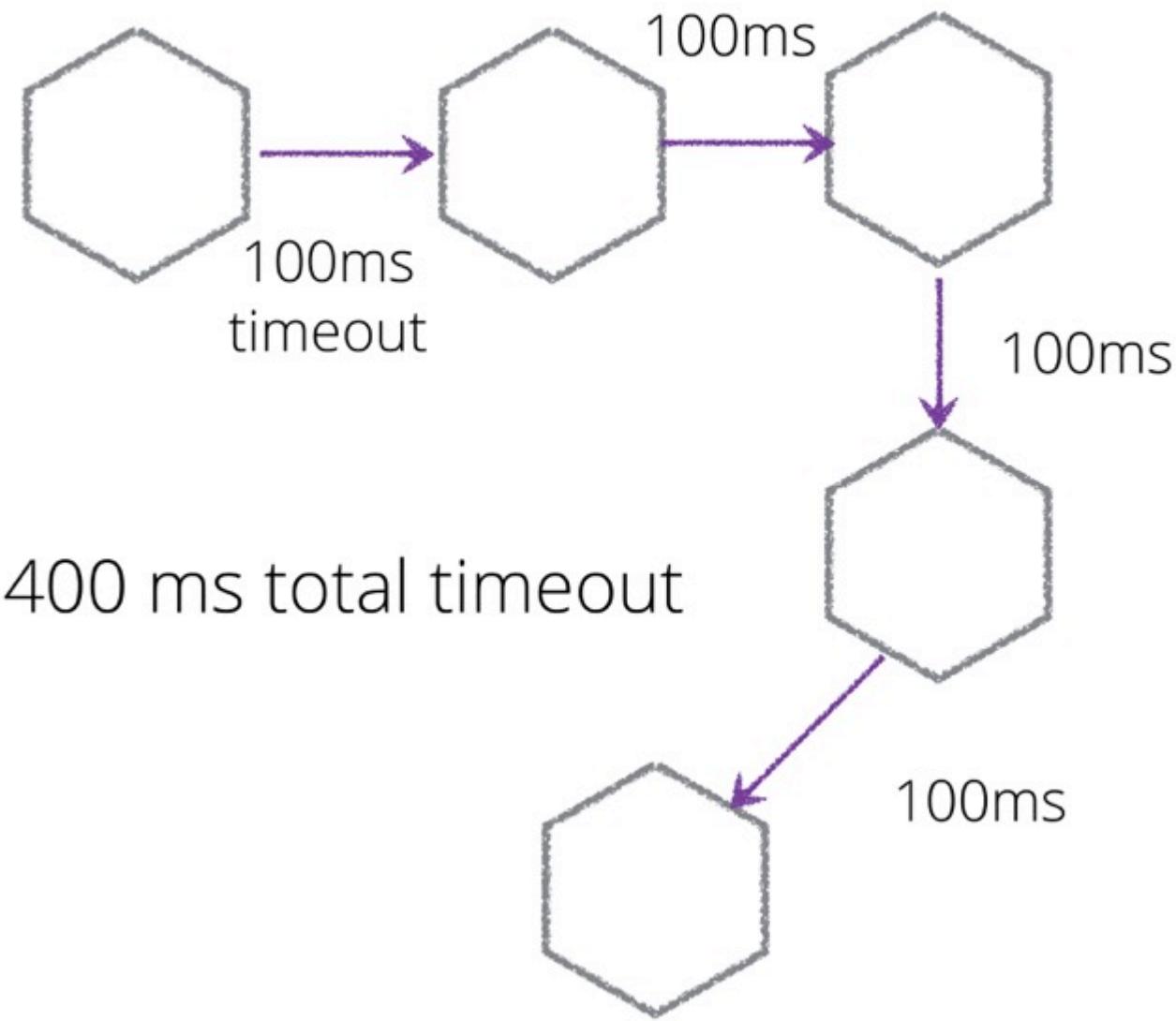




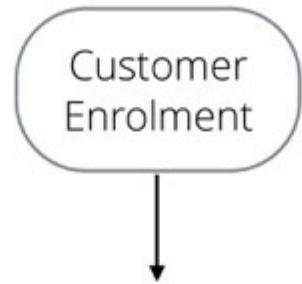


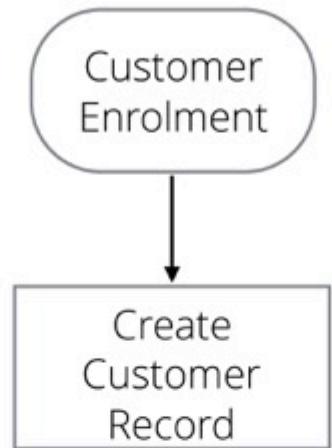


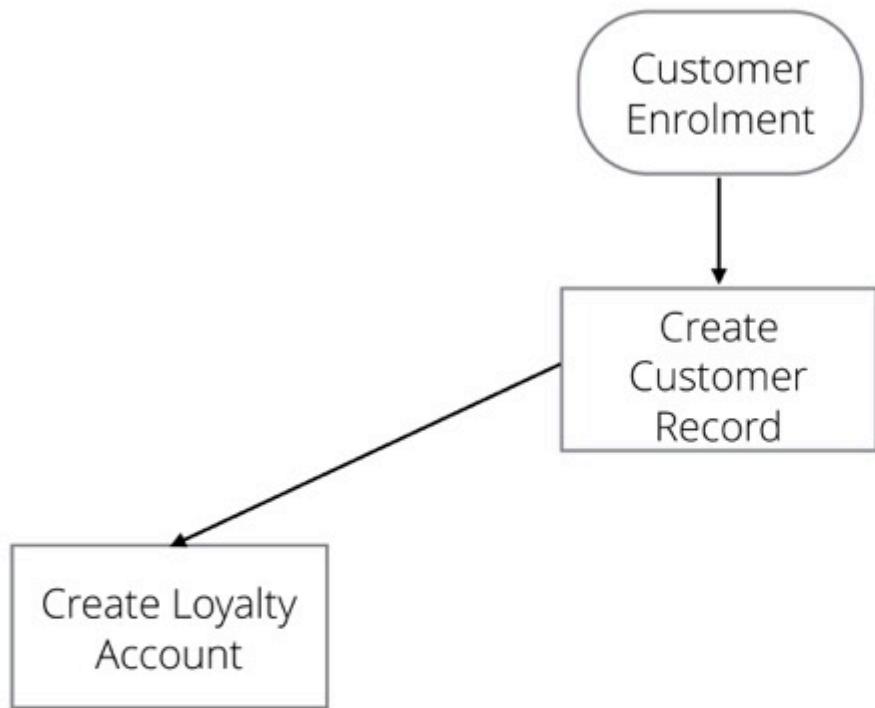


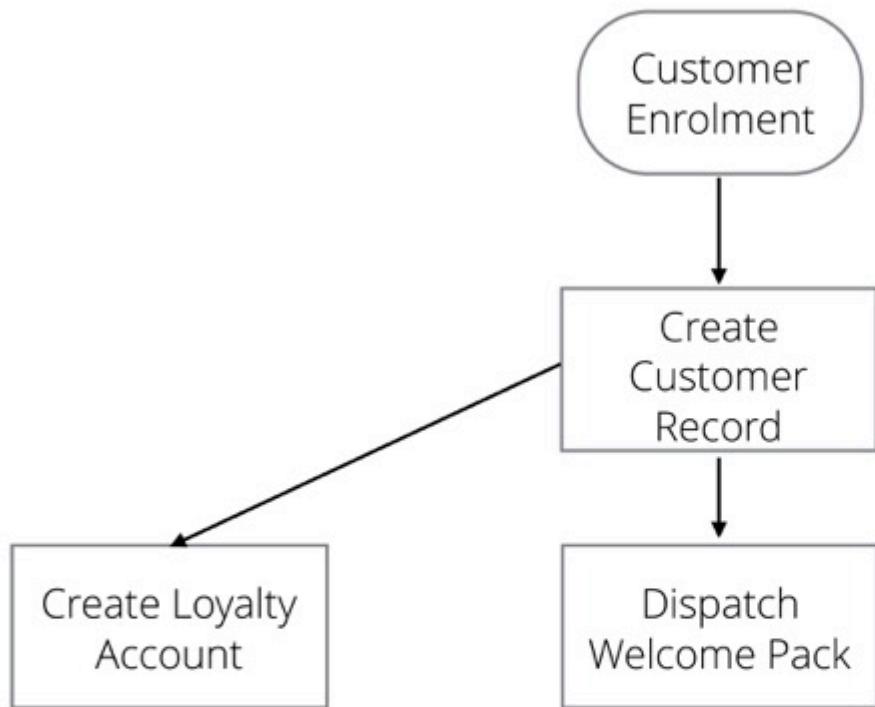


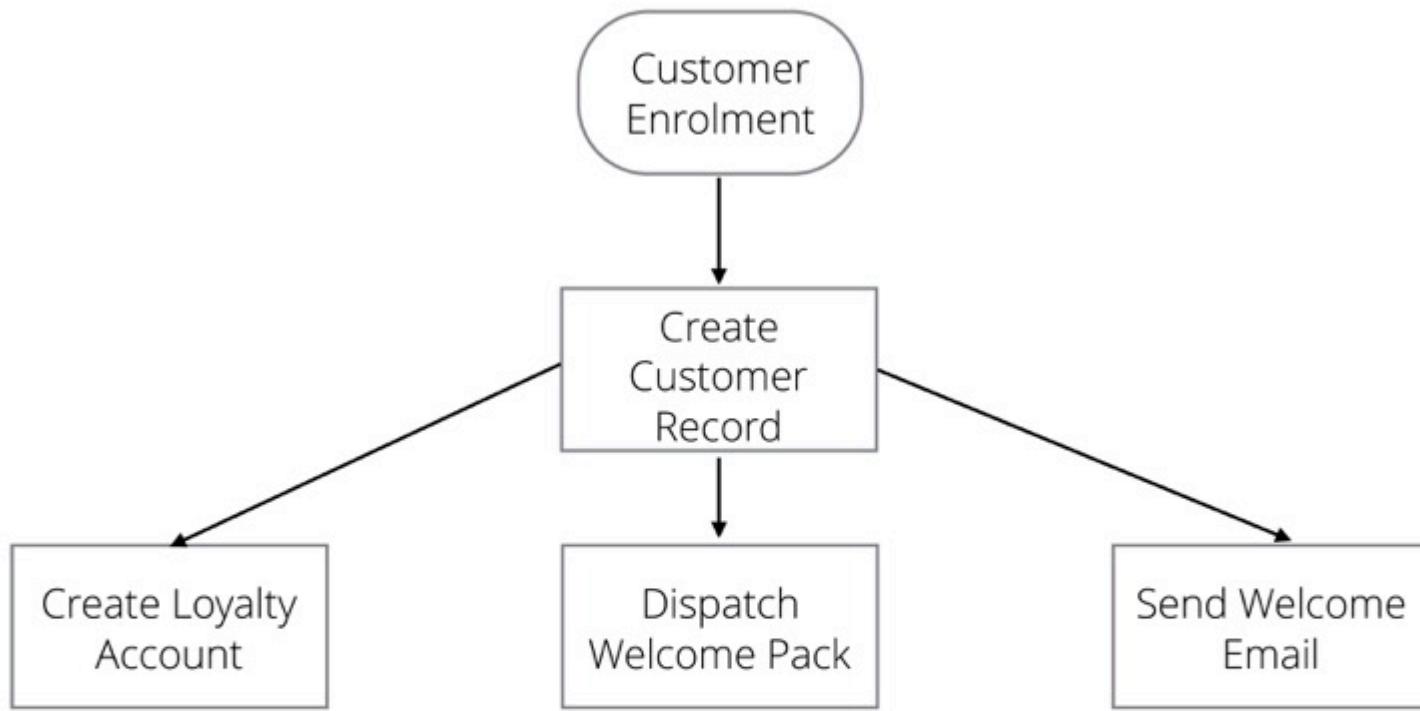
Choreography vs Orchestration

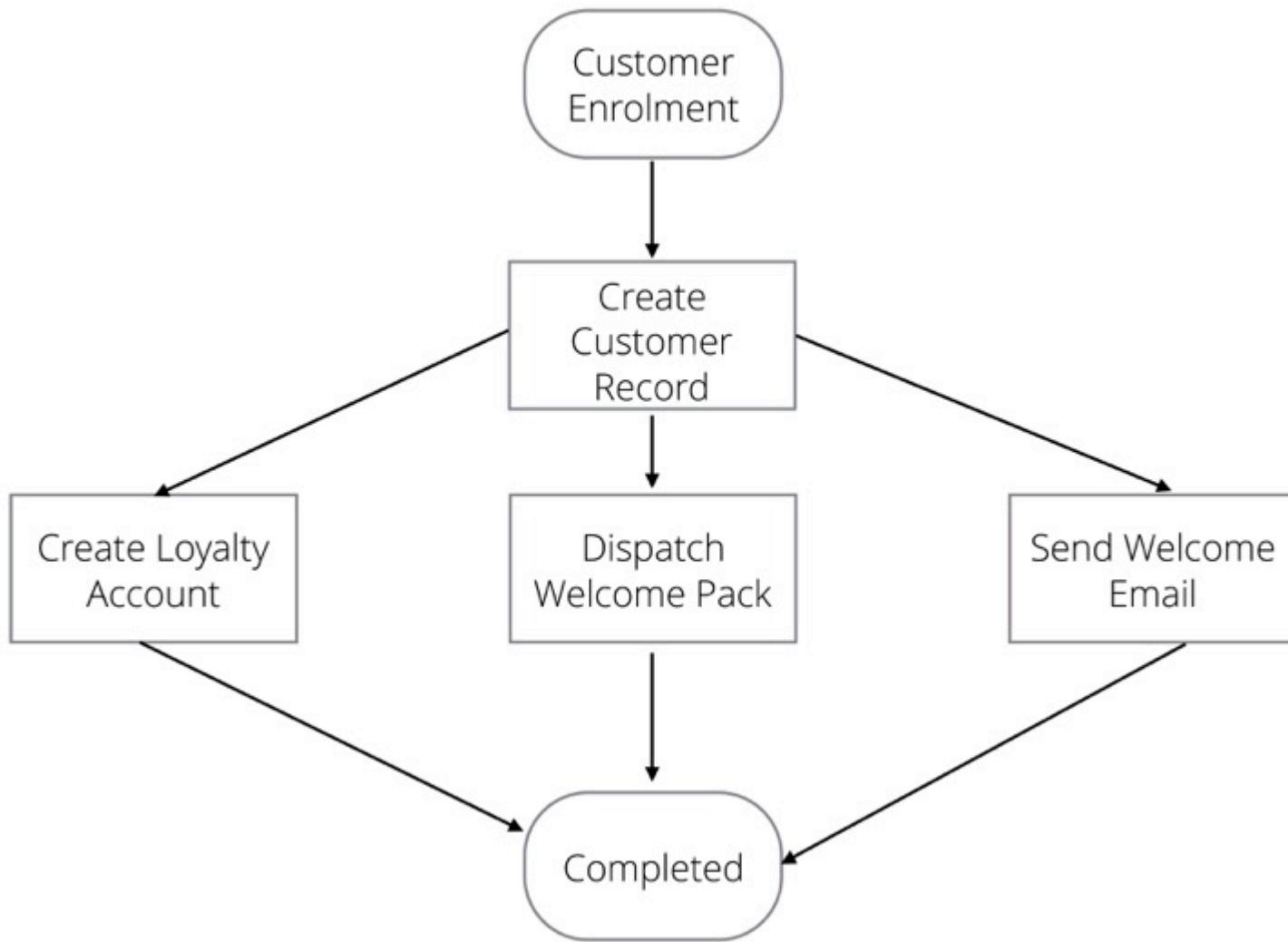








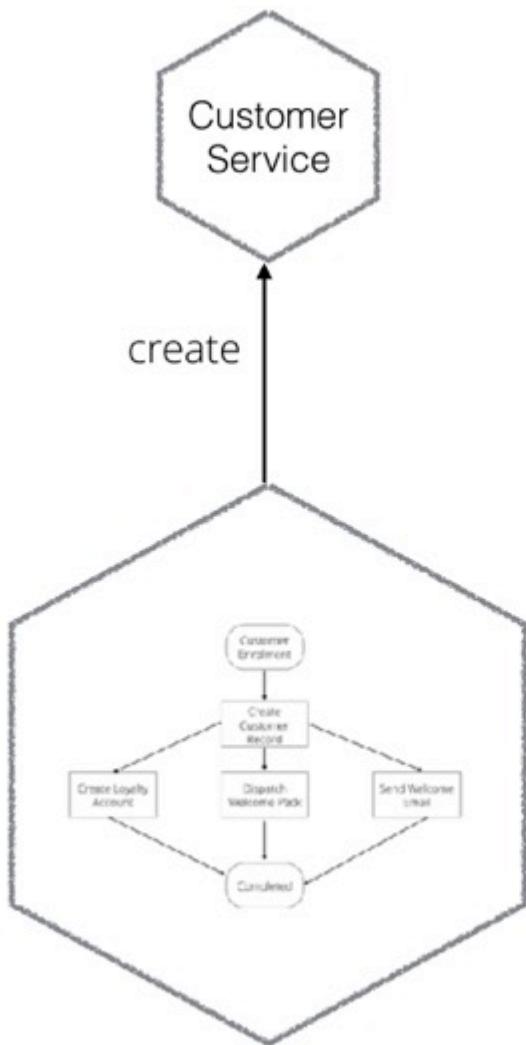




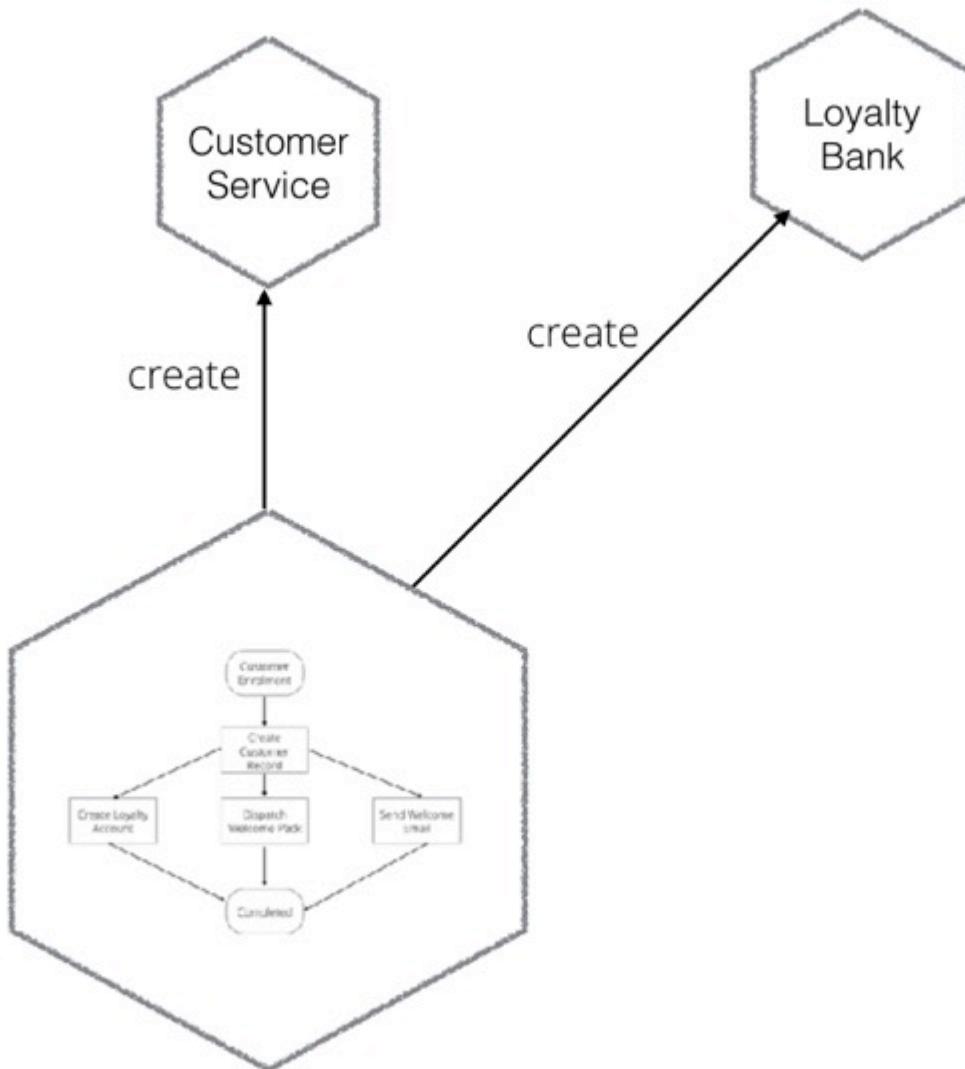
ORCHESTRATION



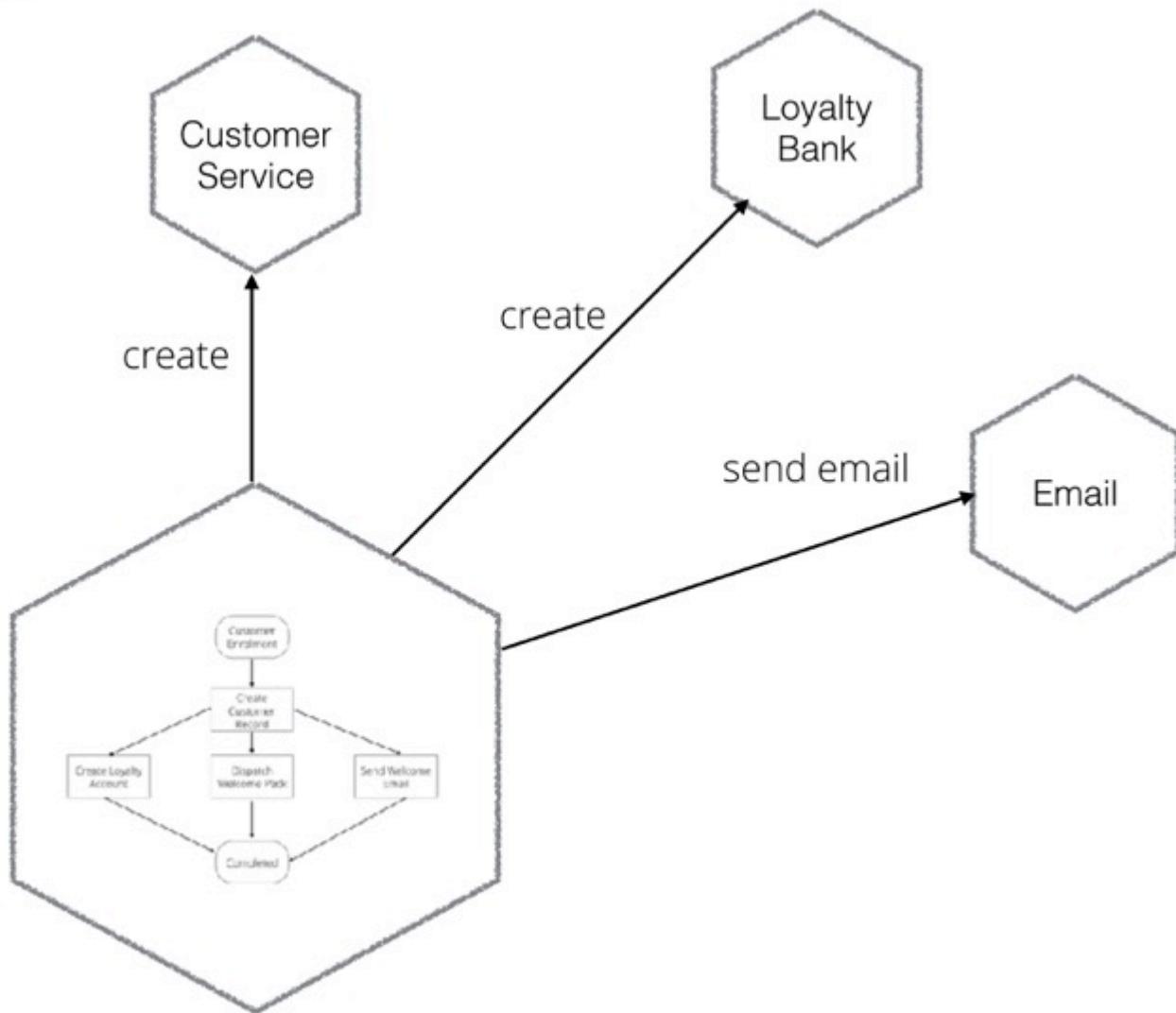
ORCHESTRATION



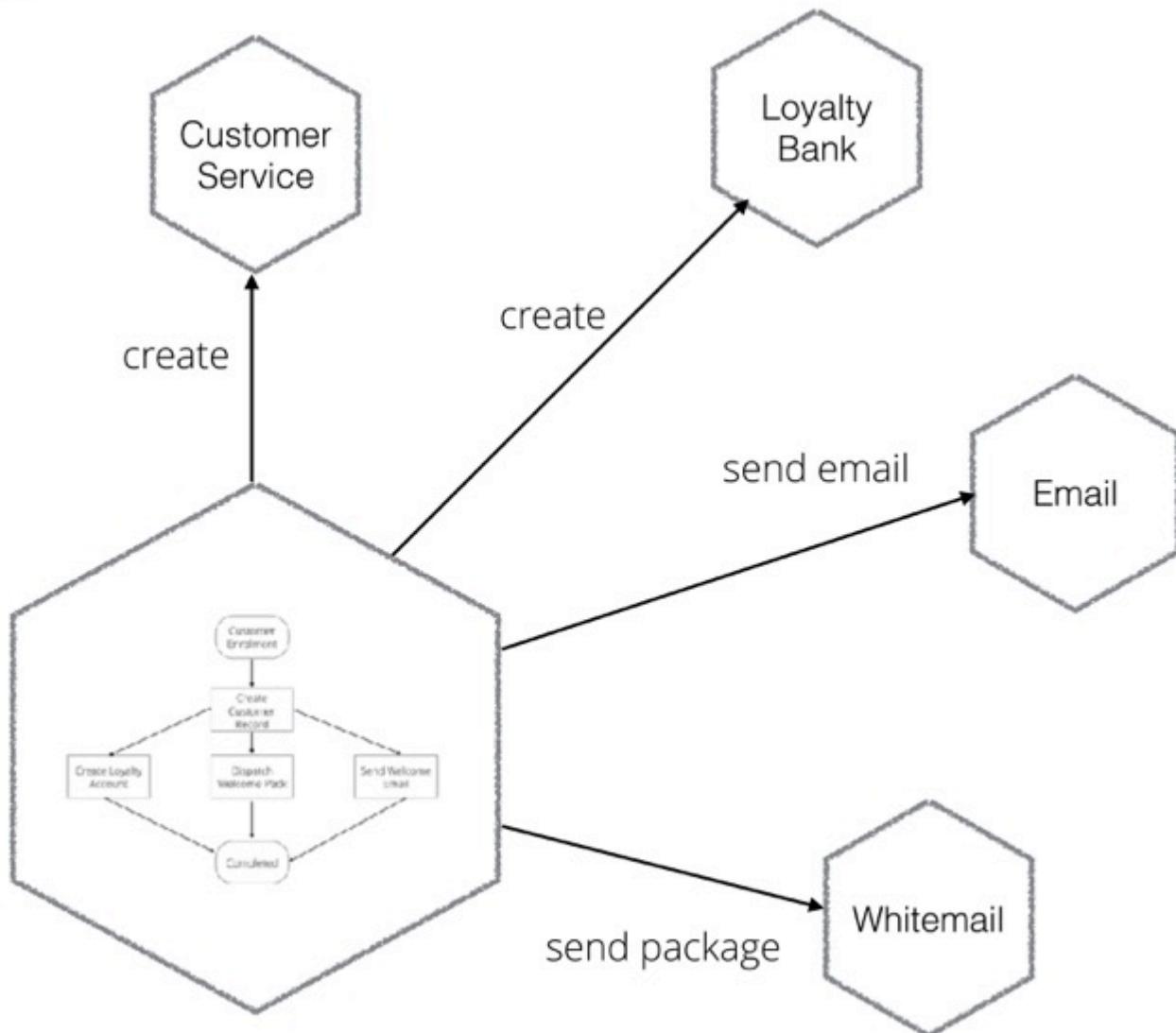
ORCHESTRATION



ORCHESTRATION



ORCHESTRATION



Pros

Pros

Explicit representation of business process

Pros

Explicit representation of business process

Know in-line if there has been a problem

Pros

Explicit representation of business process

Know in-line if there has been a problem

Cons

Pros

Explicit representation of business process

Know in-line if there has been a problem

Cons

Can be fairly coupled

Pros

Explicit representation of business process

Know in-line if there has been a problem

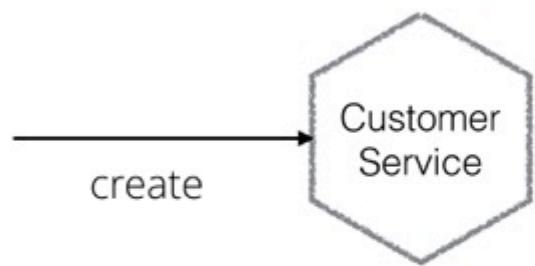
Cons

Can be fairly coupled

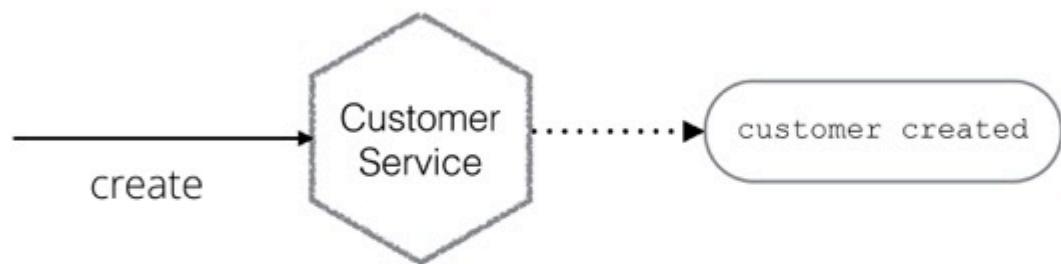
Can lead to overly smart (and dumb
services)

CHOREOGRAPHED

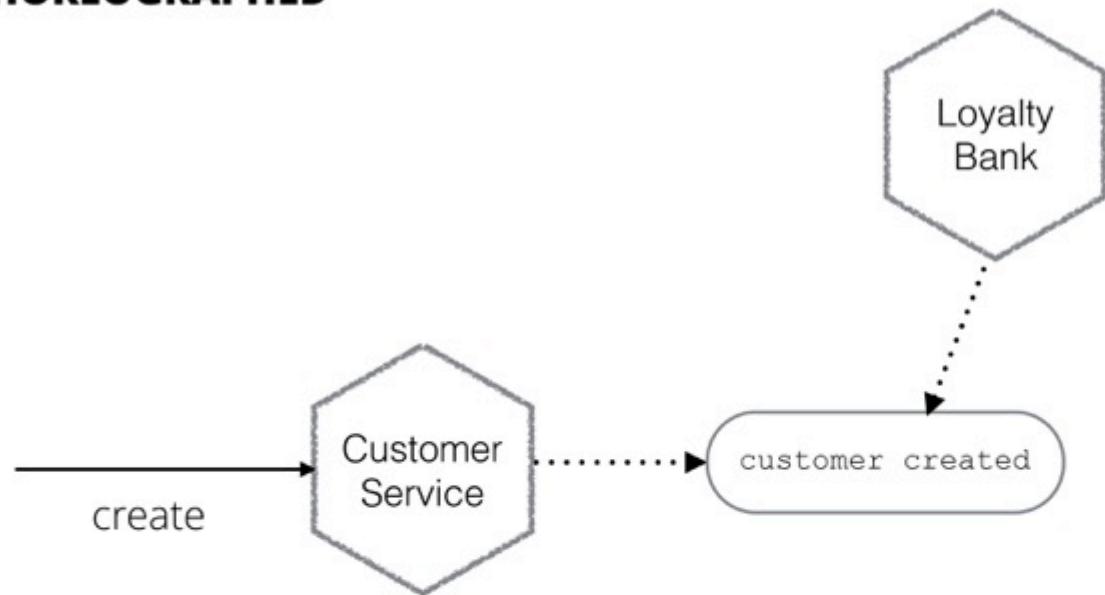
CHOREOGRAPHED



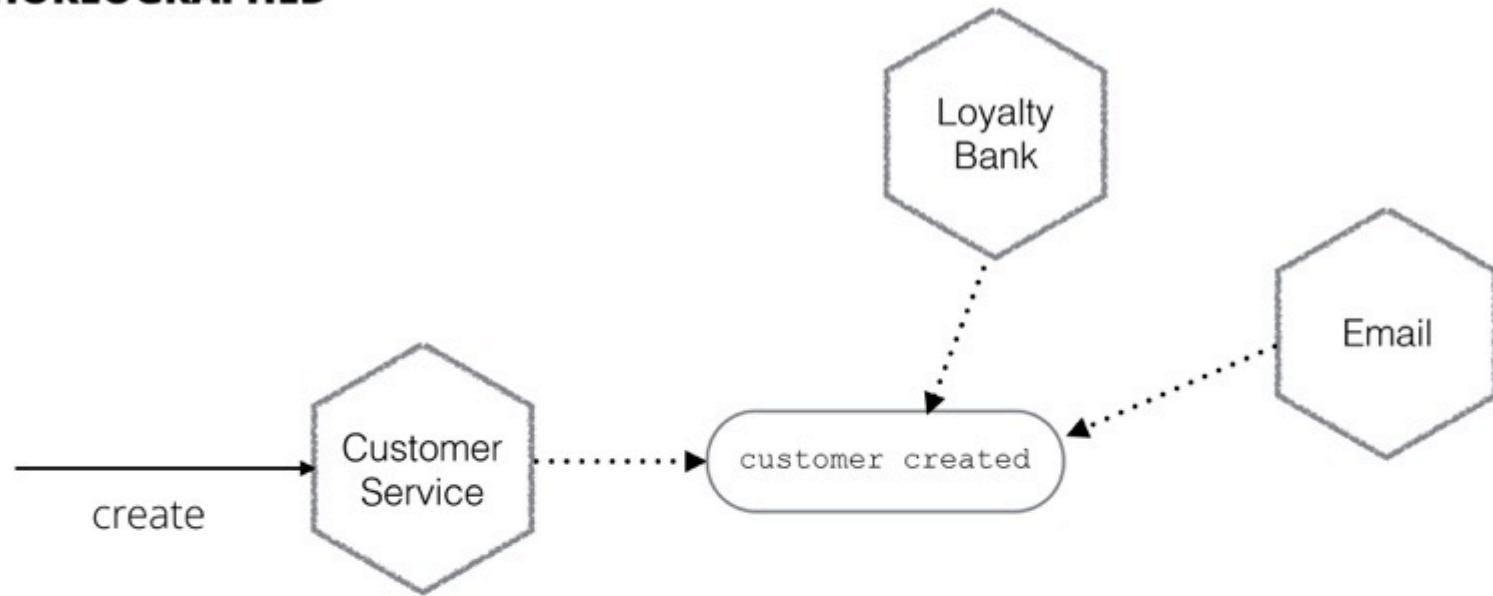
CHOREOGRAPHED



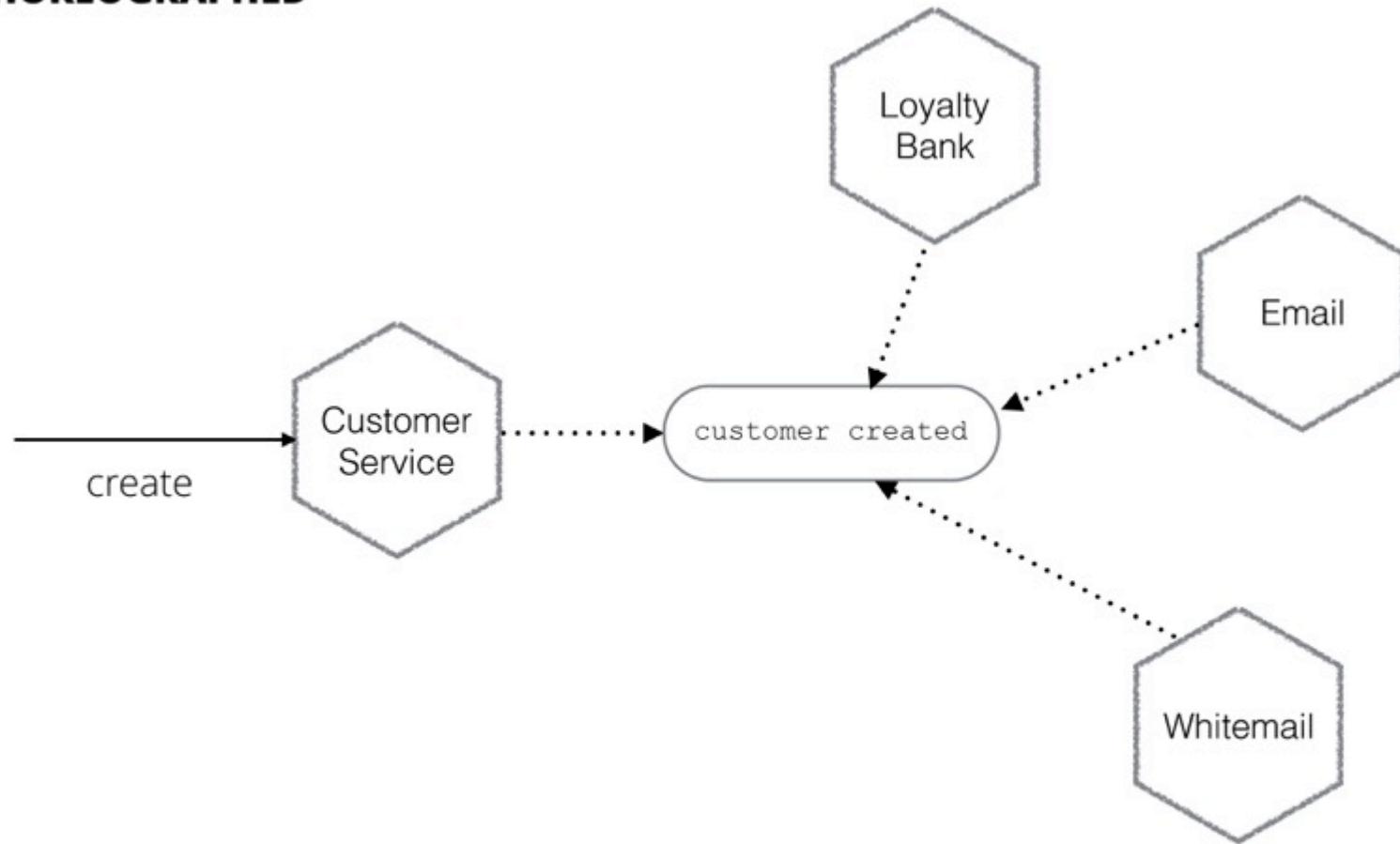
CHOREOGRAPHED



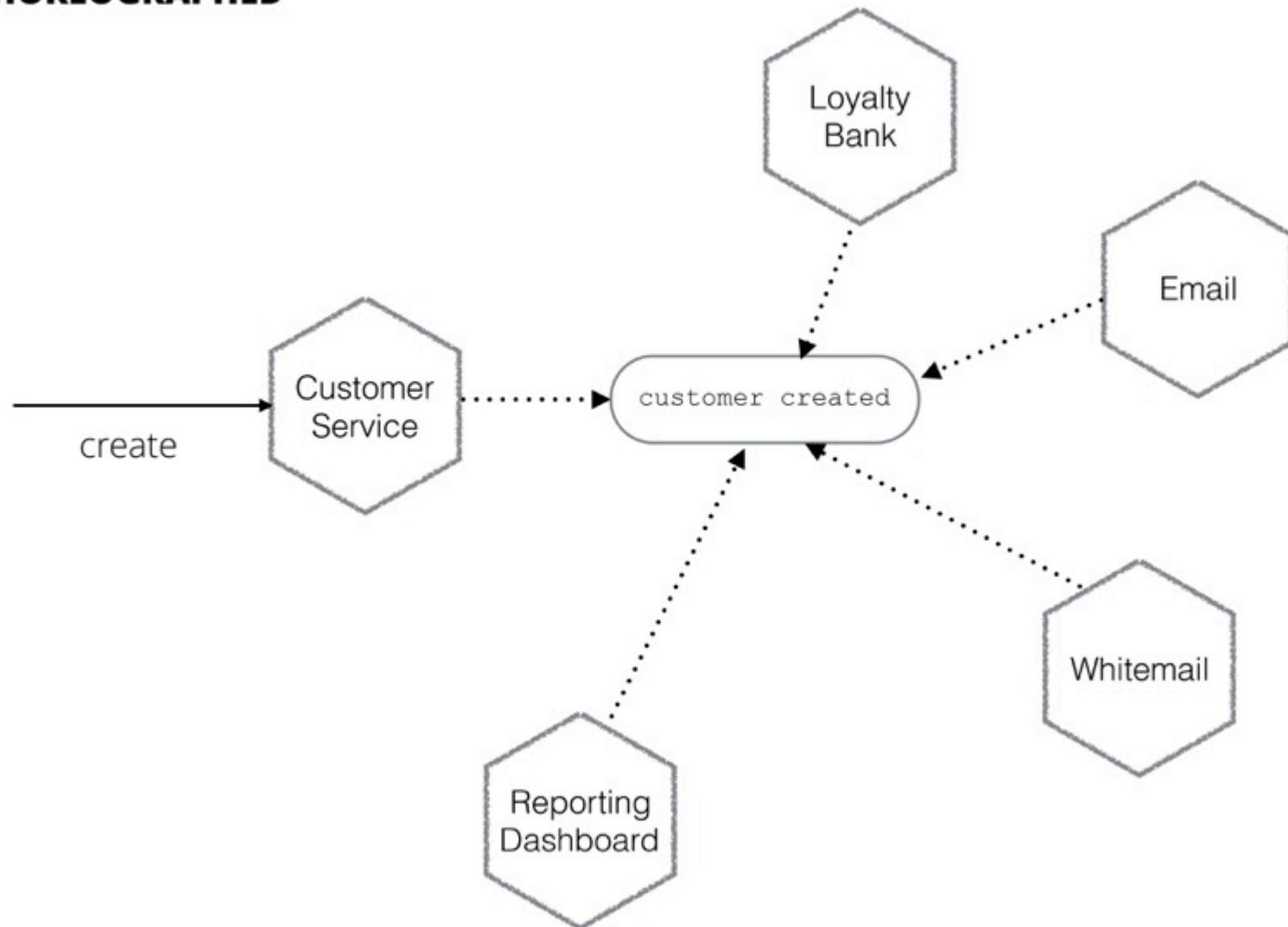
CHOREOGRAPHED



CHOREOGRAPHED



CHOREOGRAPHED



Pros

Pros

Highly decoupled

Pros

Highly decoupled

Evenly distributed smarts

Pros

Highly decoupled

Evenly distributed smarts

Cons

Pros

Highly decoupled

Evenly distributed smarts

Cons

Lost explicit business process mapping

Pros

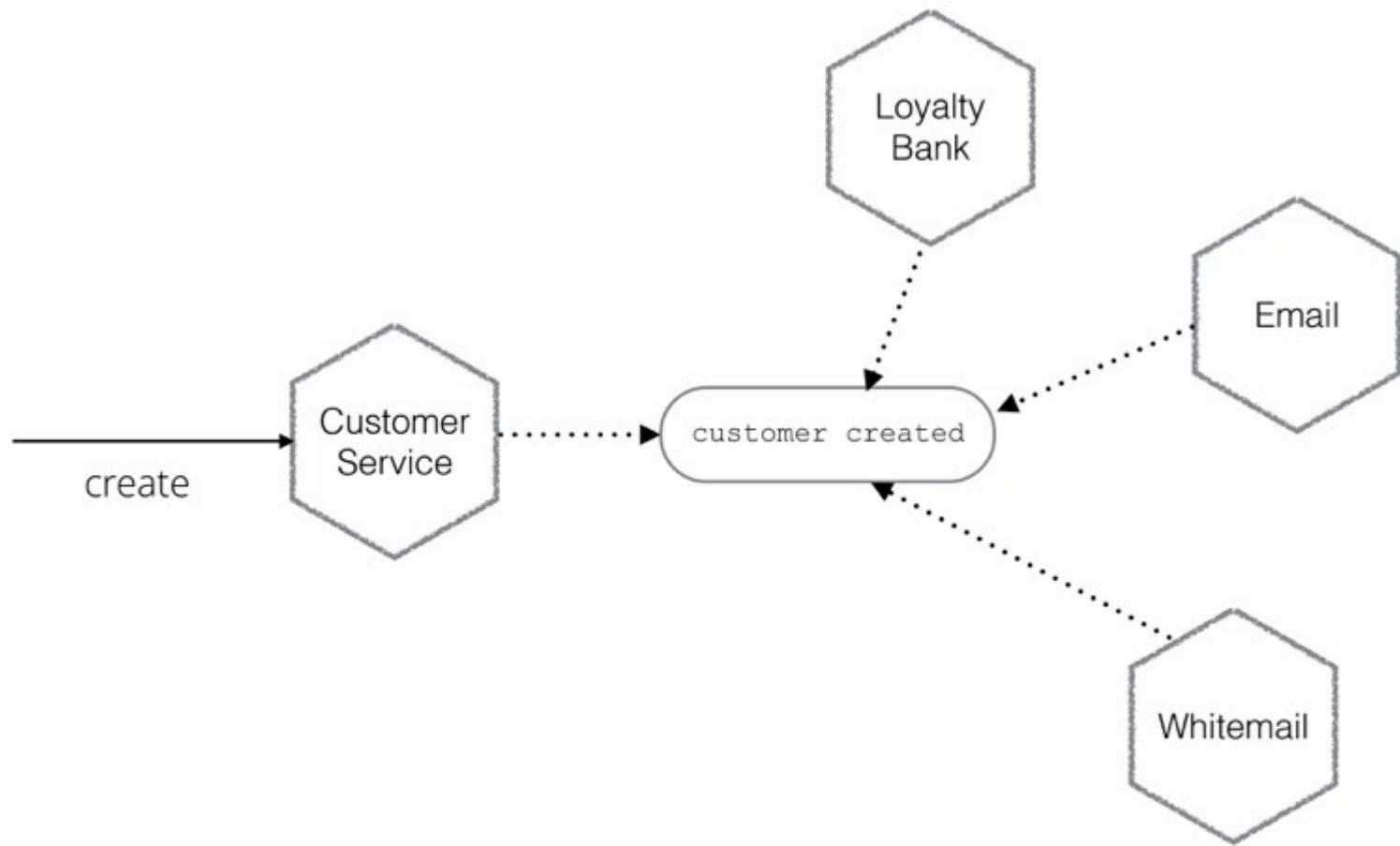
Highly decoupled

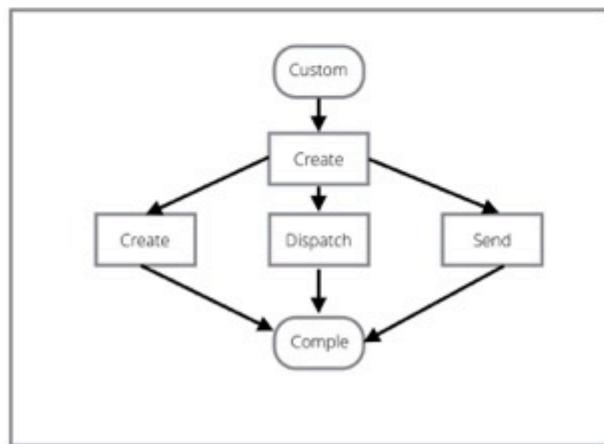
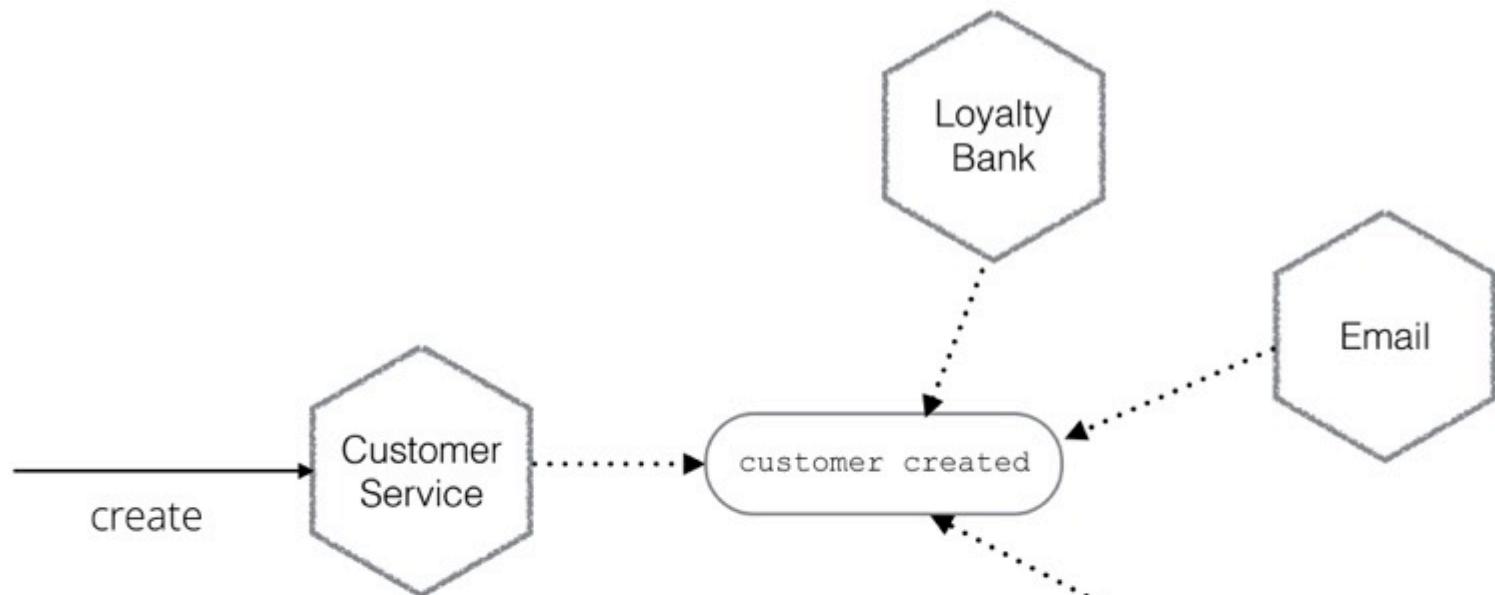
Evenly distributed smarts

Cons

Lost explicit business process mapping

Understanding completion or error states
is complex







Summary

Keep it simple

Summary

Keep it simple

Think about interaction style
first, tech second

Summary

Keep it simple

Think about interaction style
first, tech second

Orchestration vs Choreography

MONOLITHS

TO MICROSERVICES

Sam Newman

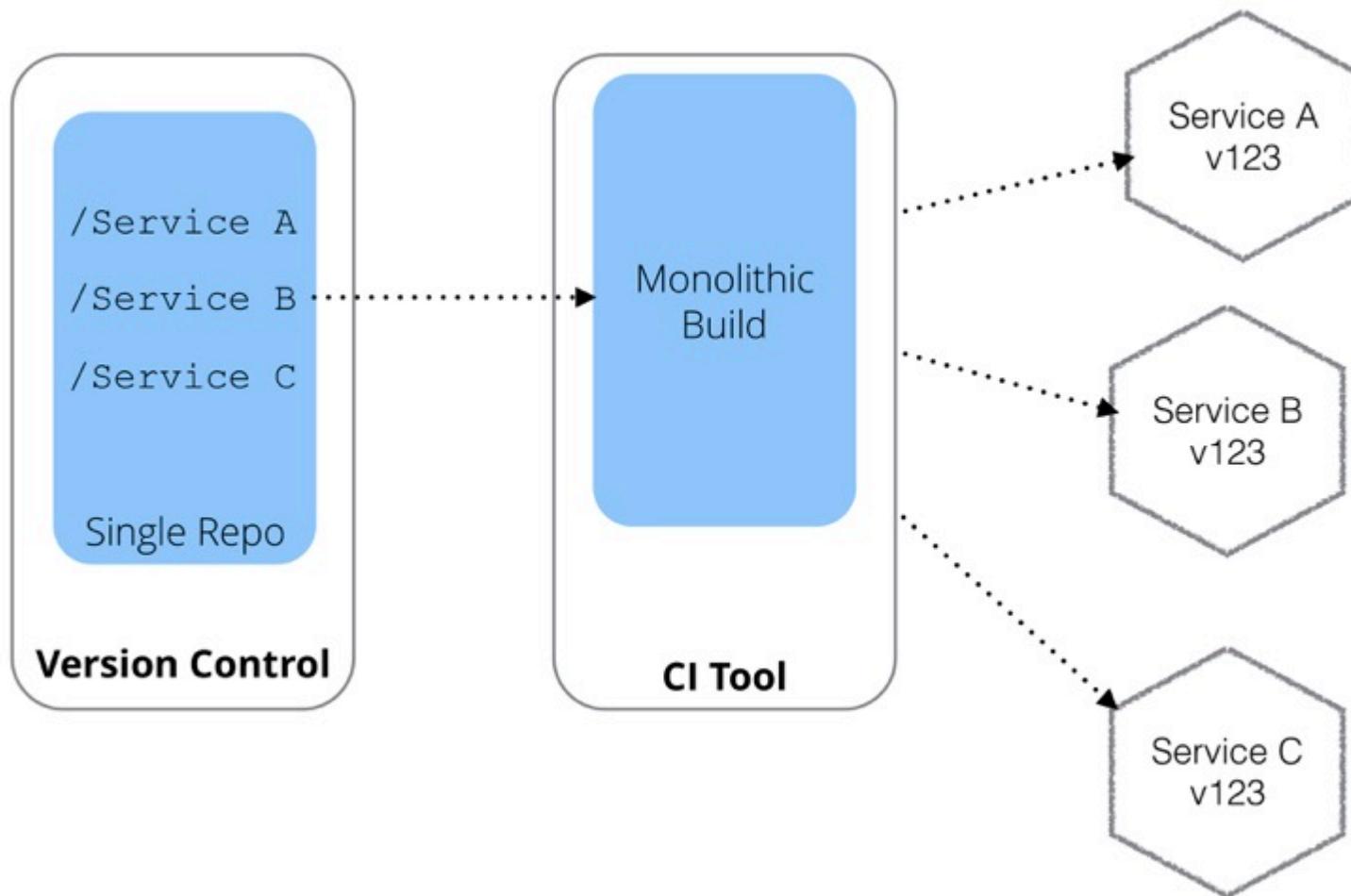
Build & Deployment

Hands up if you're doing Continuous Integration!

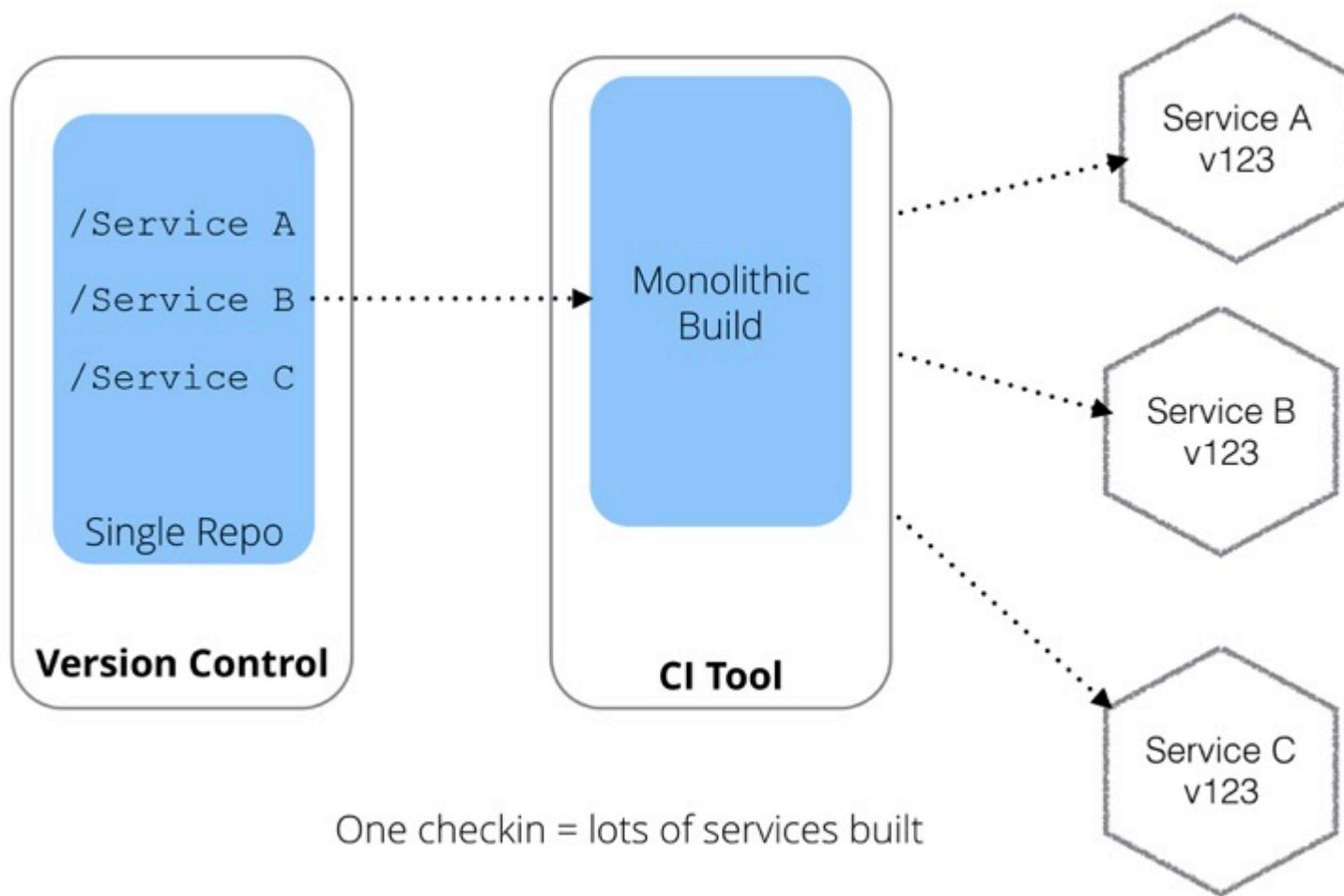
RULES OF CONTINUOUS INTEGRATION

1. Check in to mainline at least once a day
2. Have a suite to validate the integration
3. If the build fails, make it the teams #1 priority to fix it

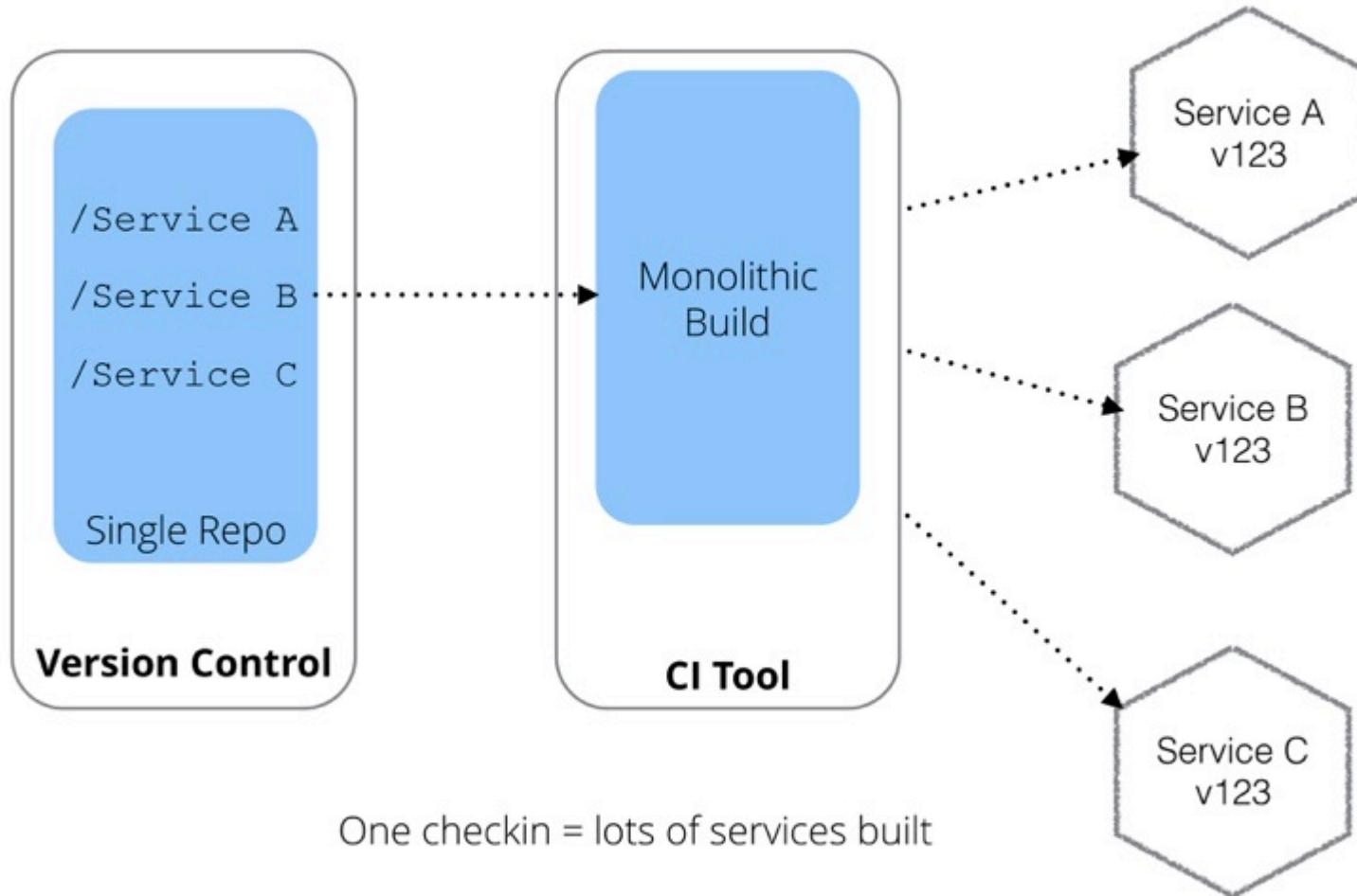
ONE GIANT BUILD



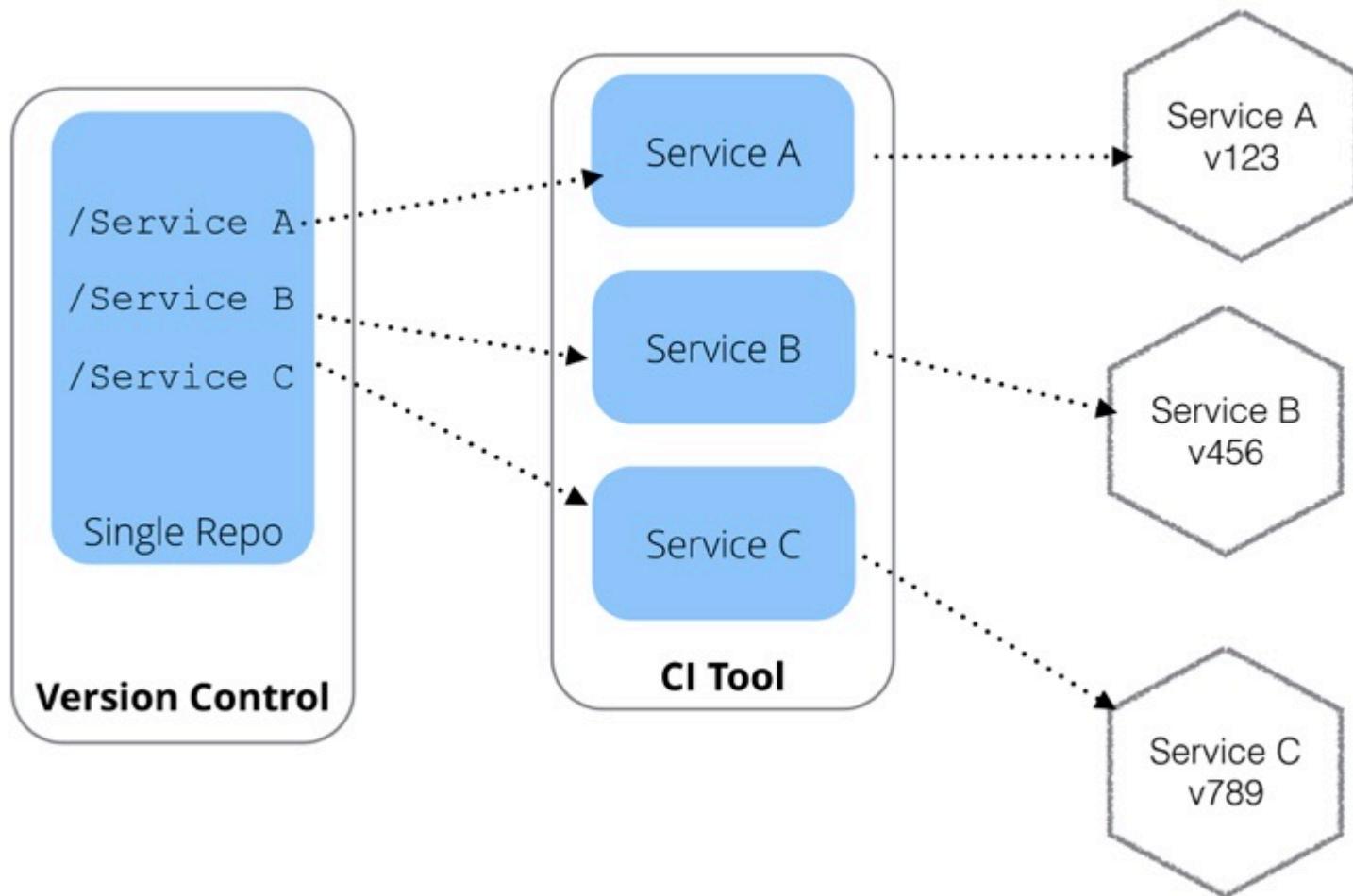
ONE GIANT BUILD



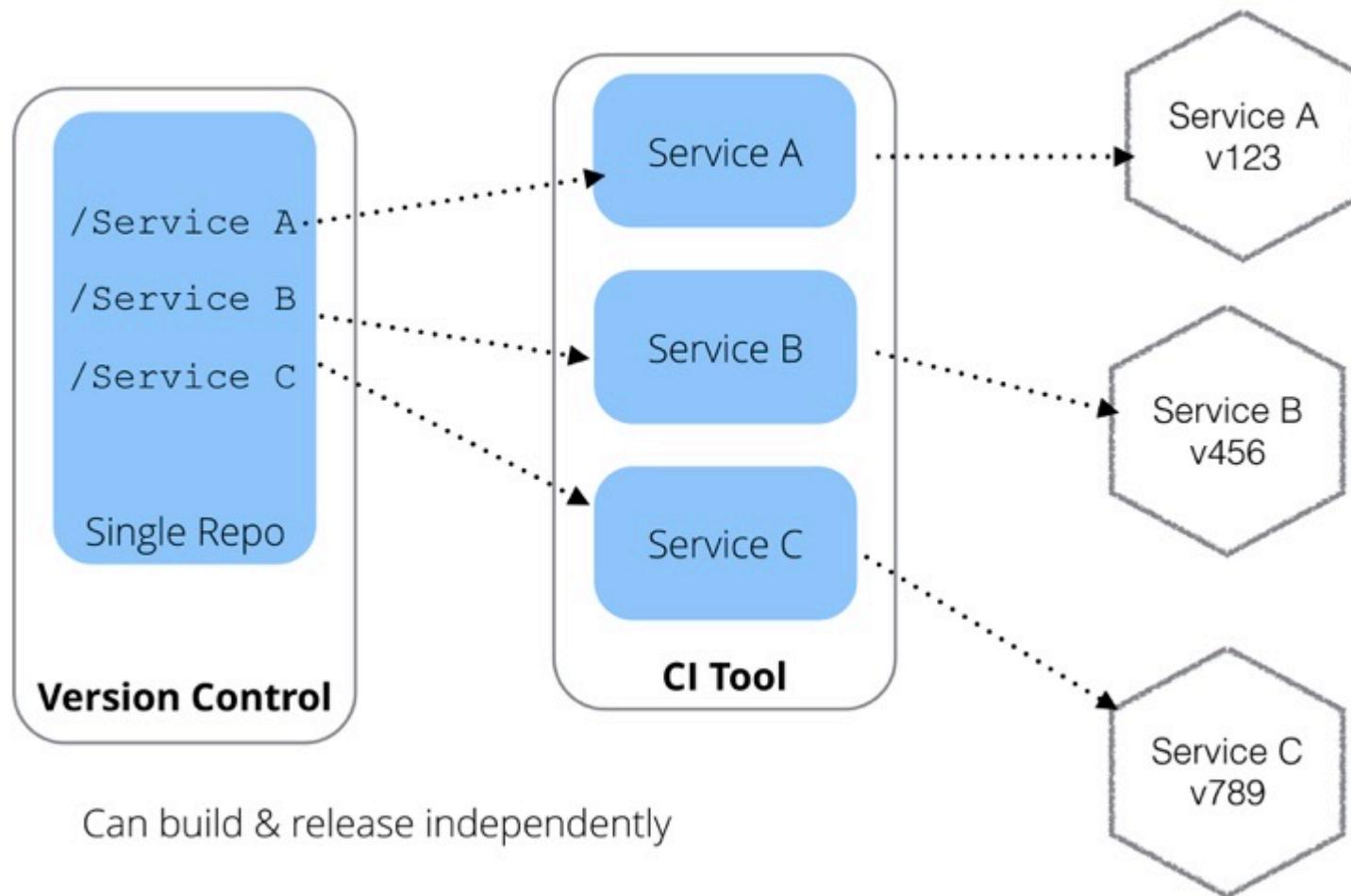
ONE GIANT BUILD



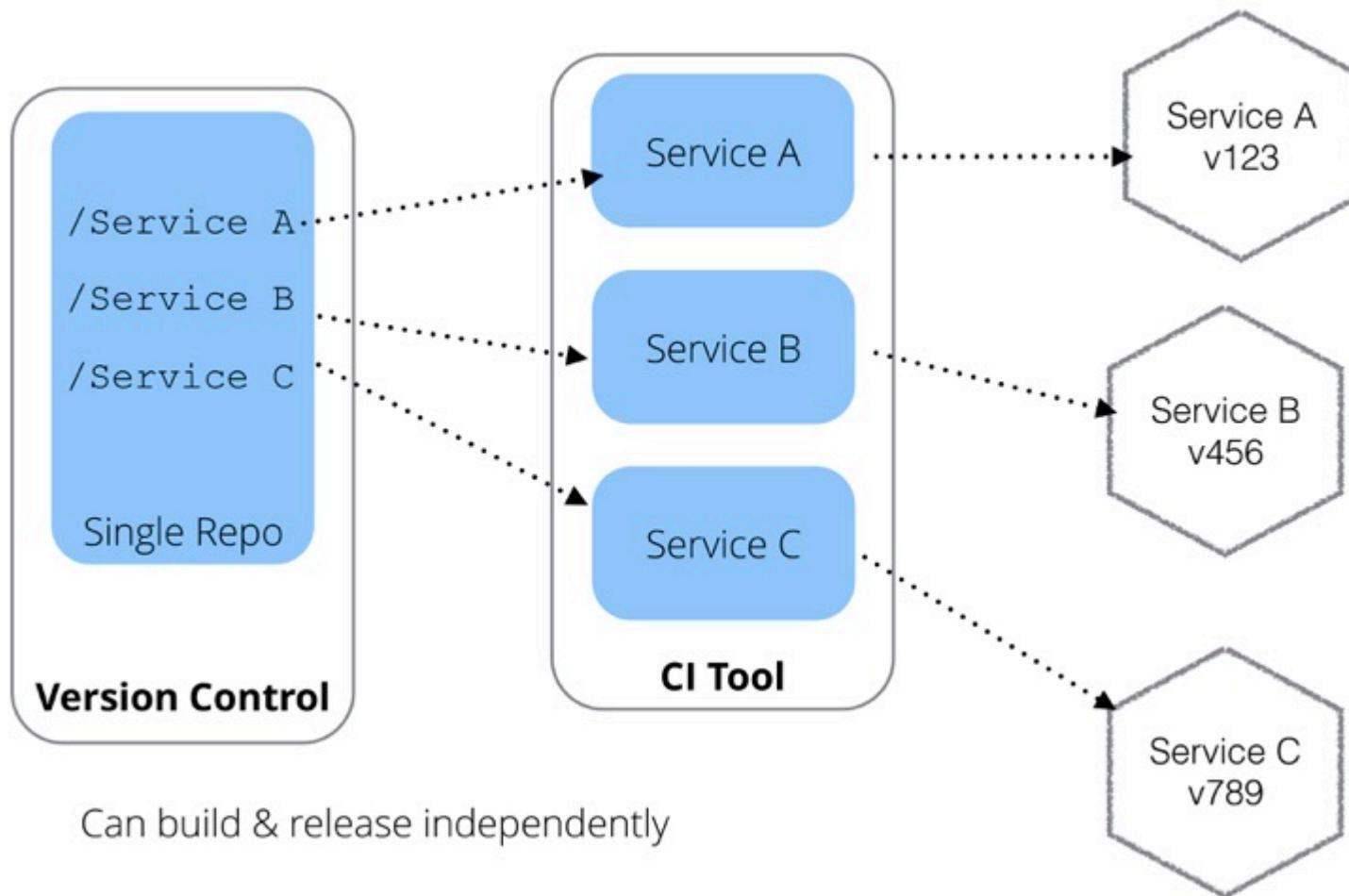
ONE BUILD PER SERVICE, SINGLE REPO



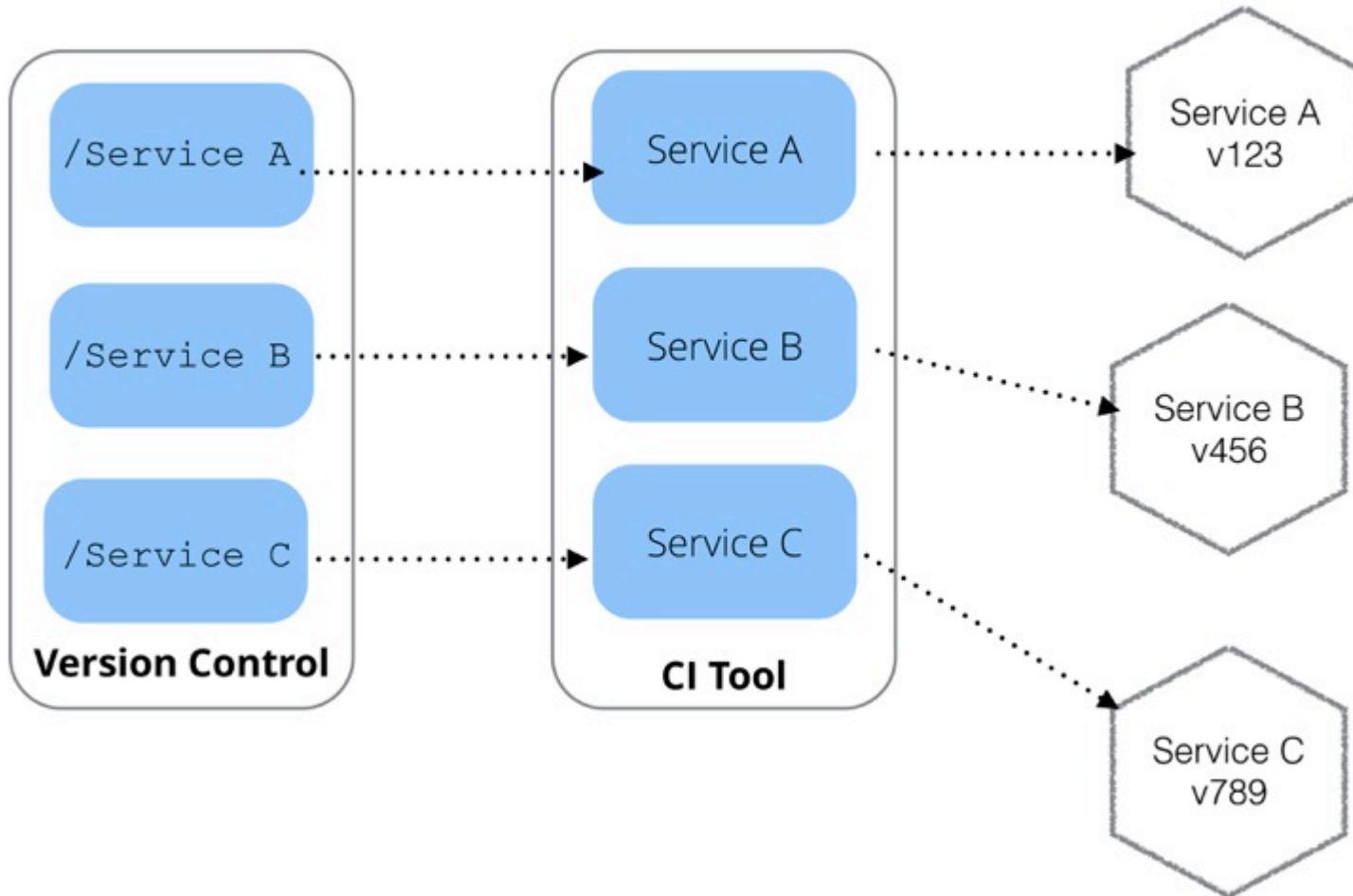
ONE BUILD PER SERVICE, SINGLE REPO



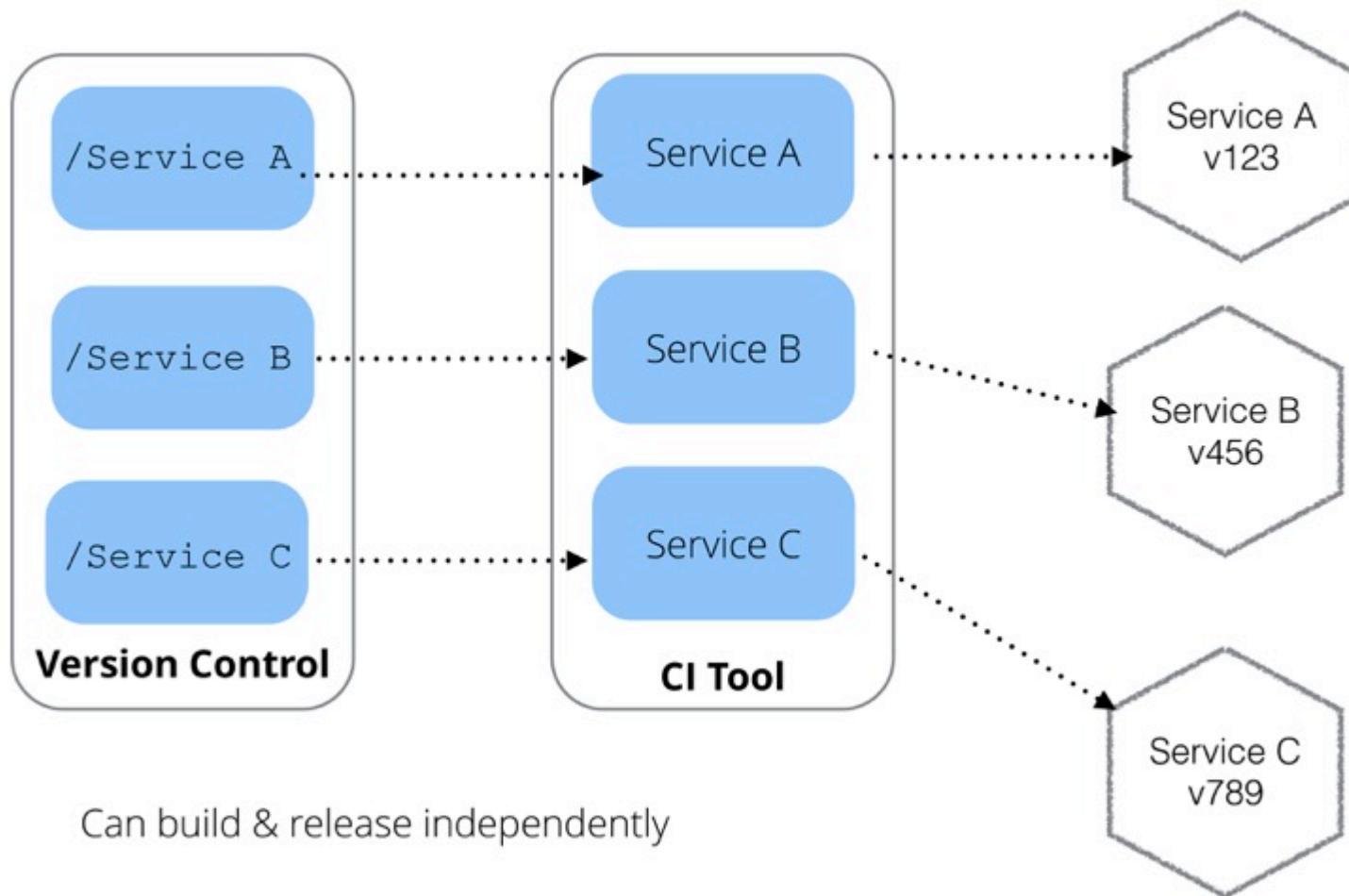
ONE BUILD PER SERVICE, SINGLE REPO



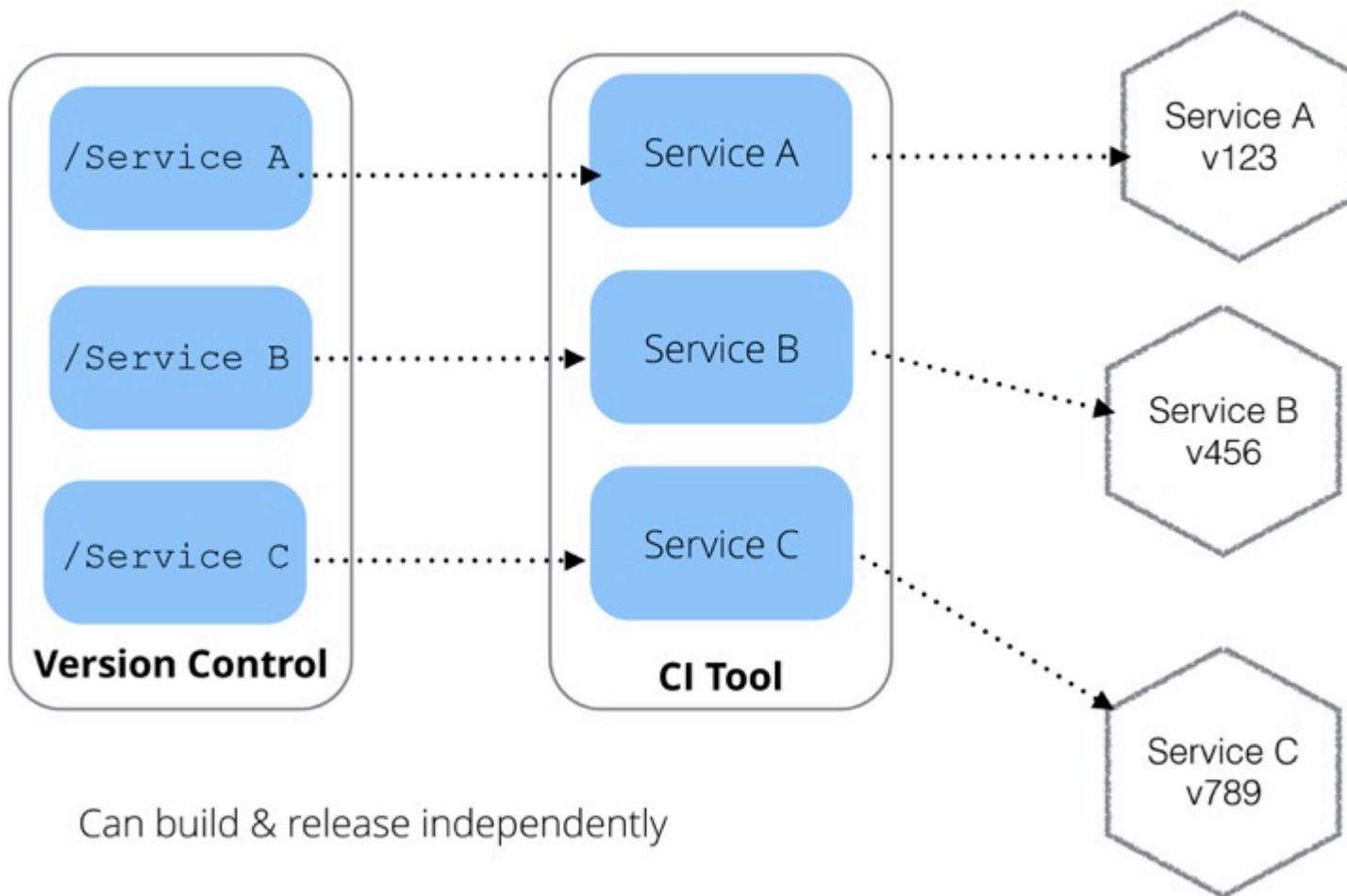
ONE BUILD & REPO PER SERVICE



ONE BUILD & REPO PER SERVICE

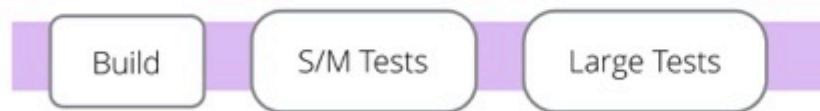


ONE BUILD & REPO PER SERVICE



Firmer separation, but developer workflow can suffer





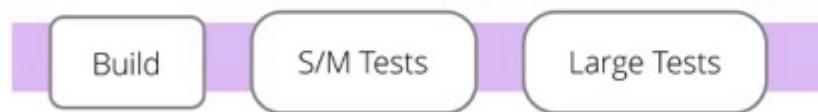


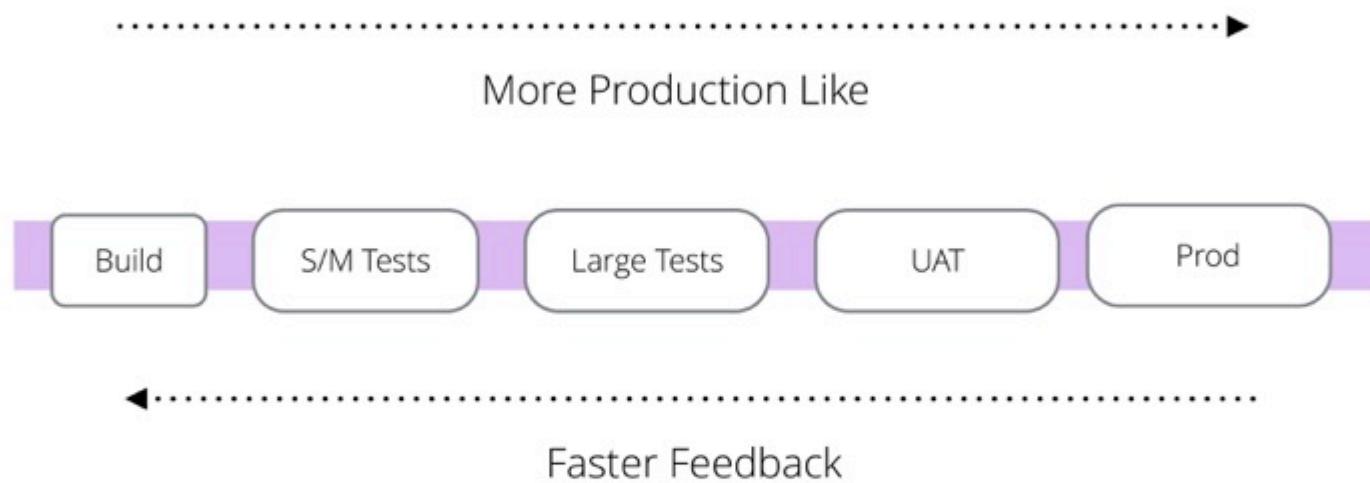
Customer Service



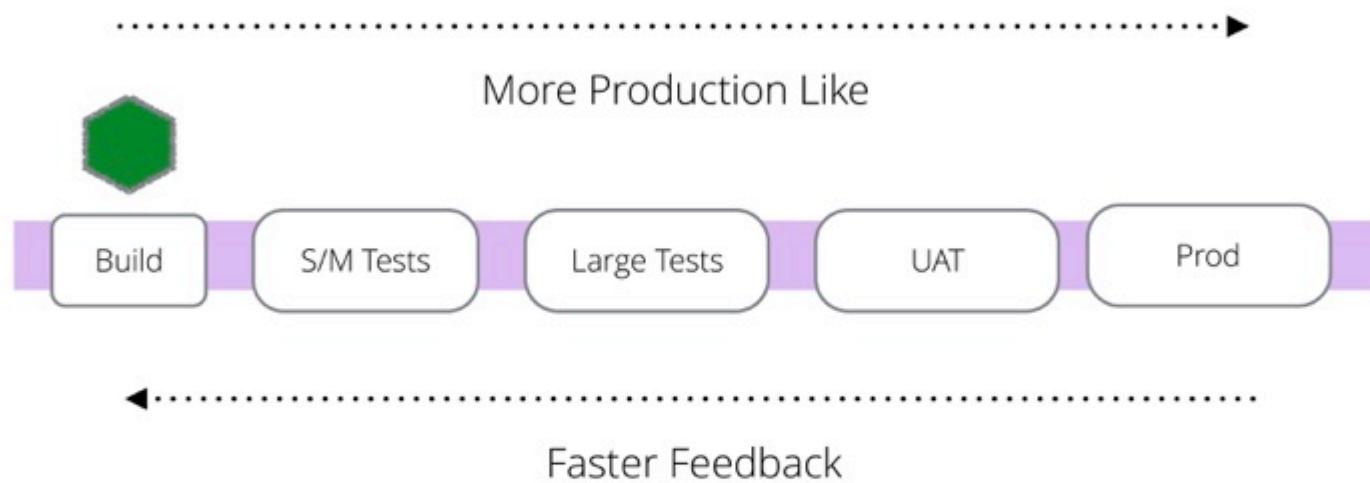


Customer Service

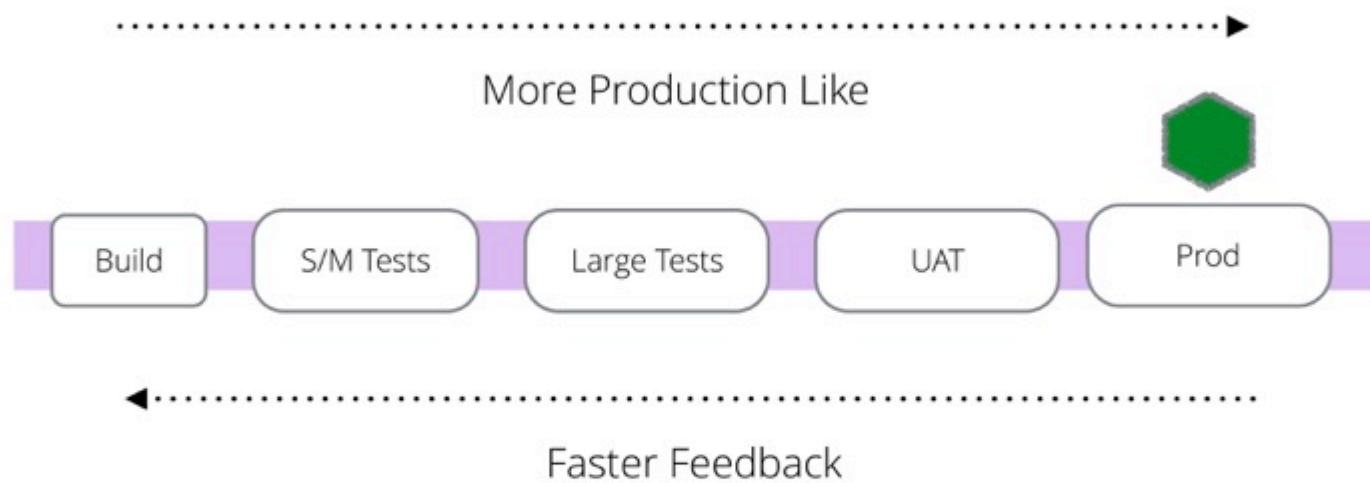




Artifacts should be built once, and moved through environments



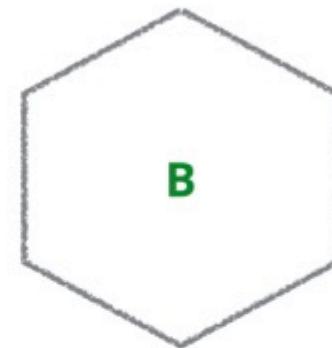
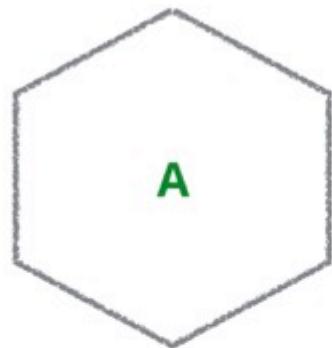
Artifacts should be built once, and moved through environments



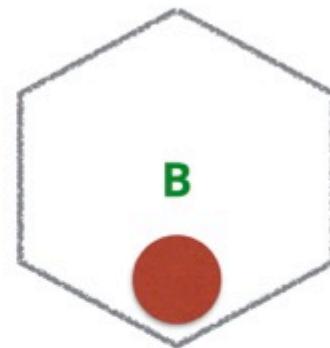
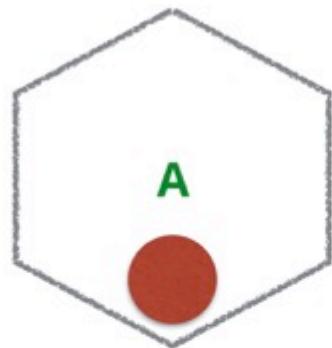
Artifacts should be built once, and moved through environments

Shared Code?

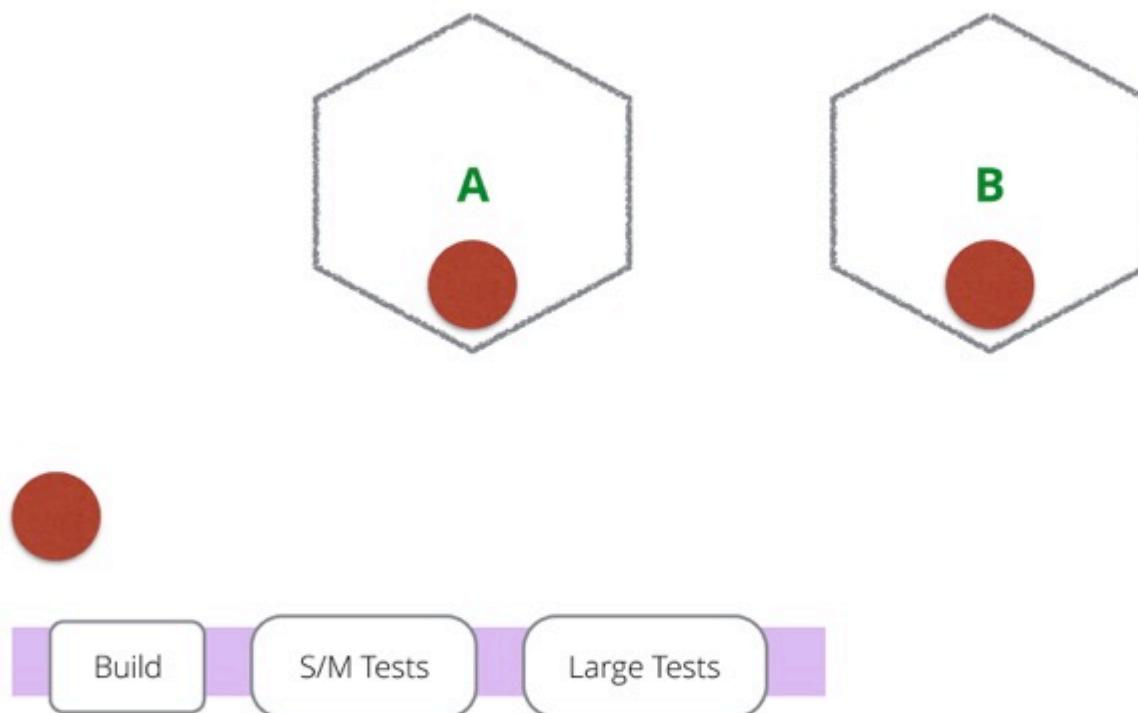
SHARED DEPENDENCIES



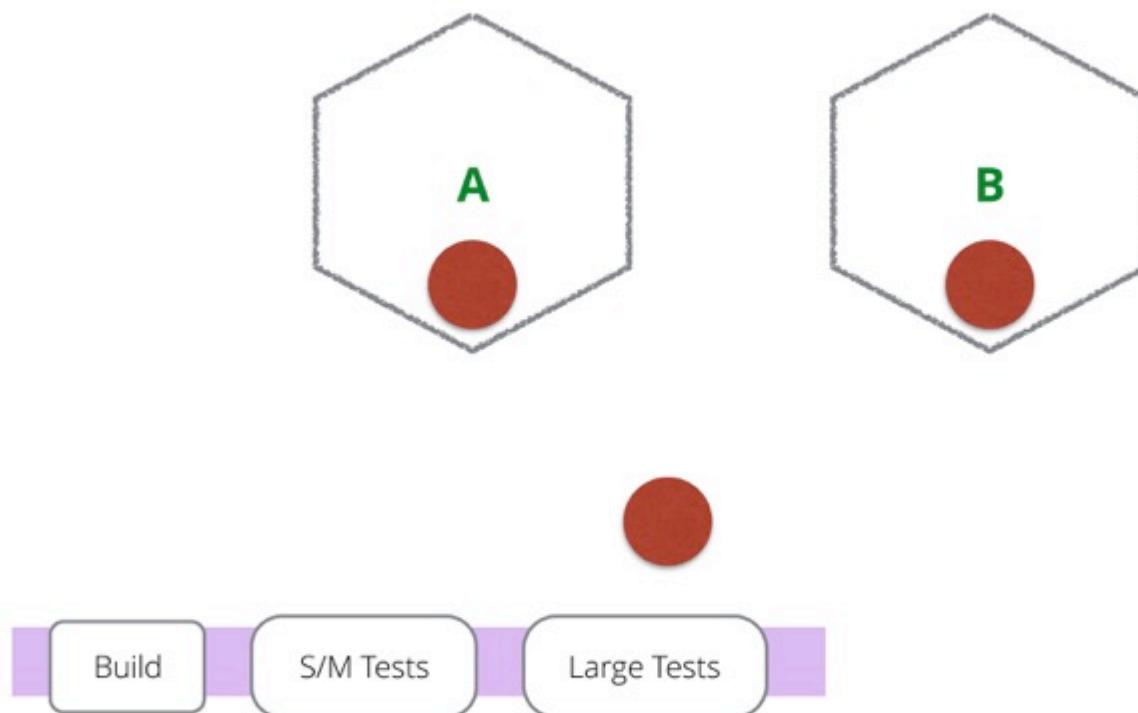
SHARED DEPENDENCIES



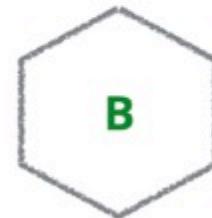
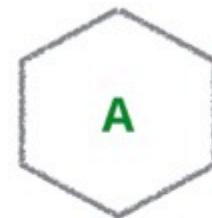
SHARED DEPENDENCIES



SHARED DEPENDENCIES



FIXED DEPENDENCIES



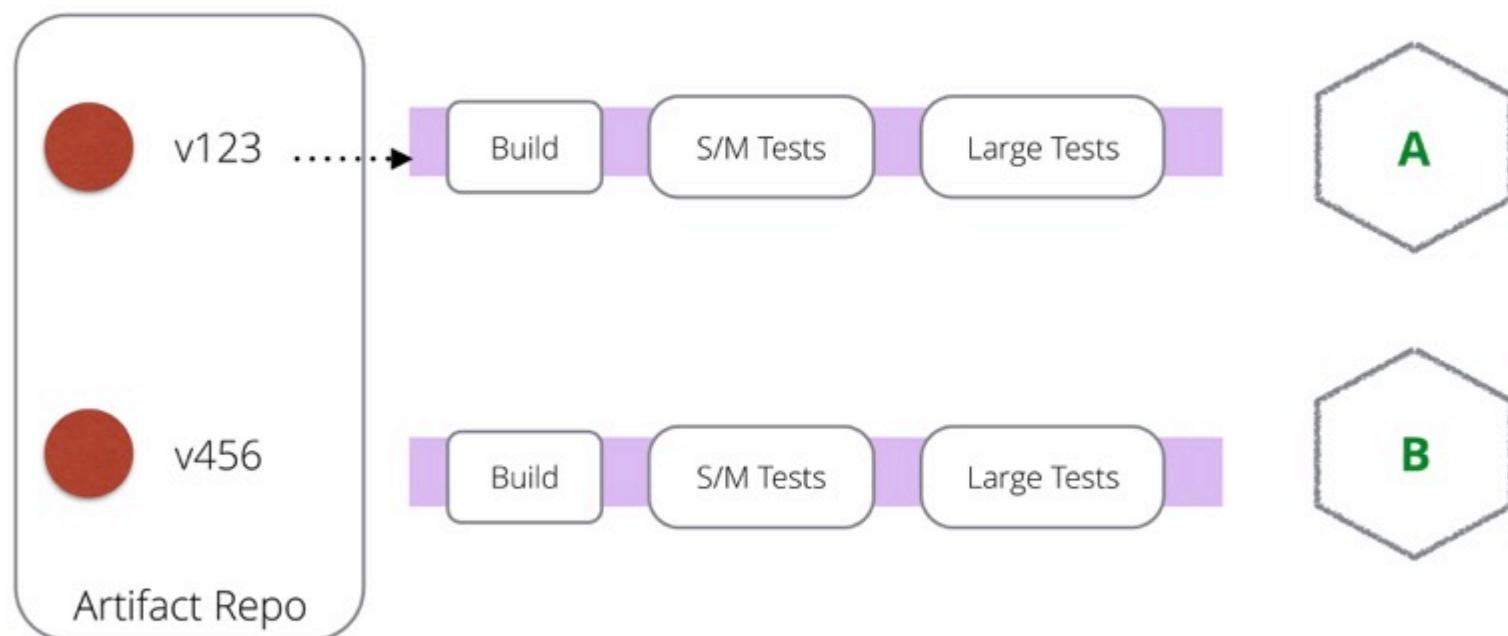
FIXED DEPENDENCIES



FIXED DEPENDENCIES



FIXED DEPENDENCIES



FIXED DEPENDENCIES



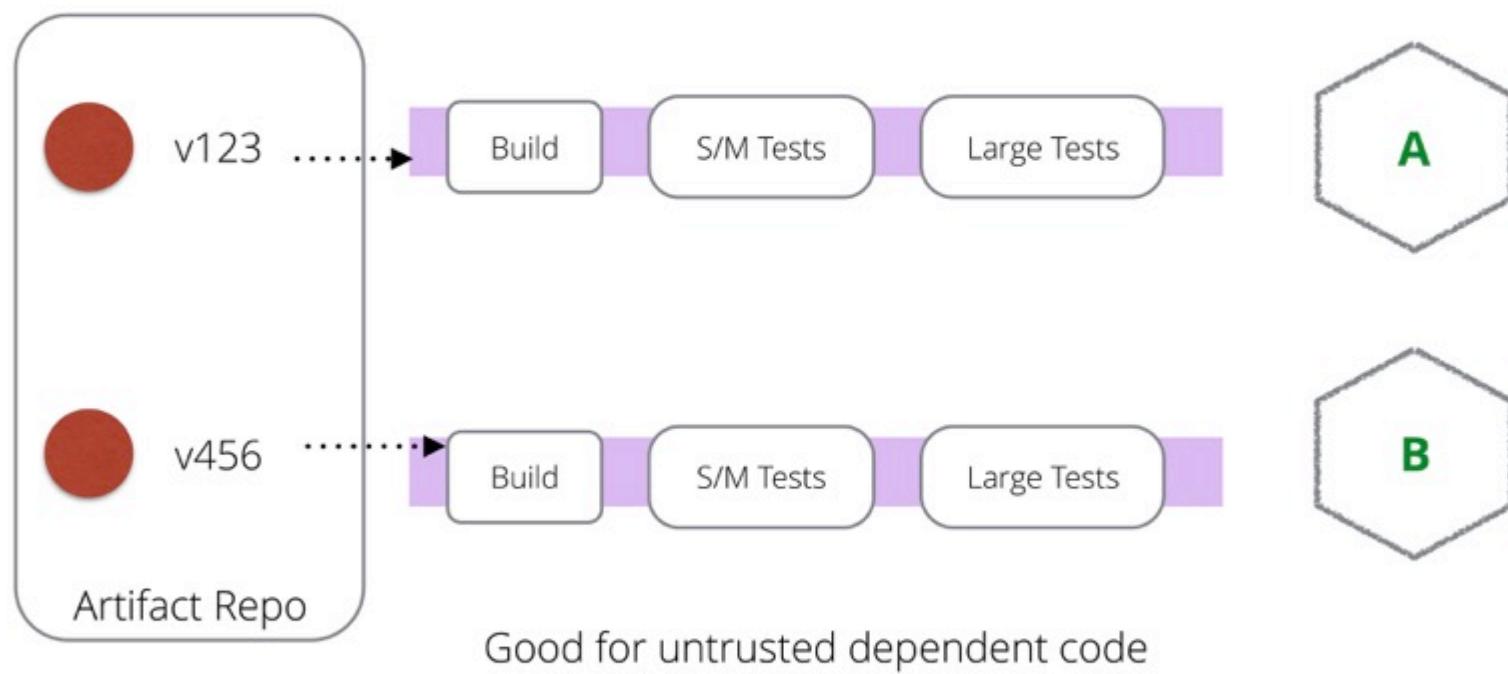
FIXED DEPENDENCIES



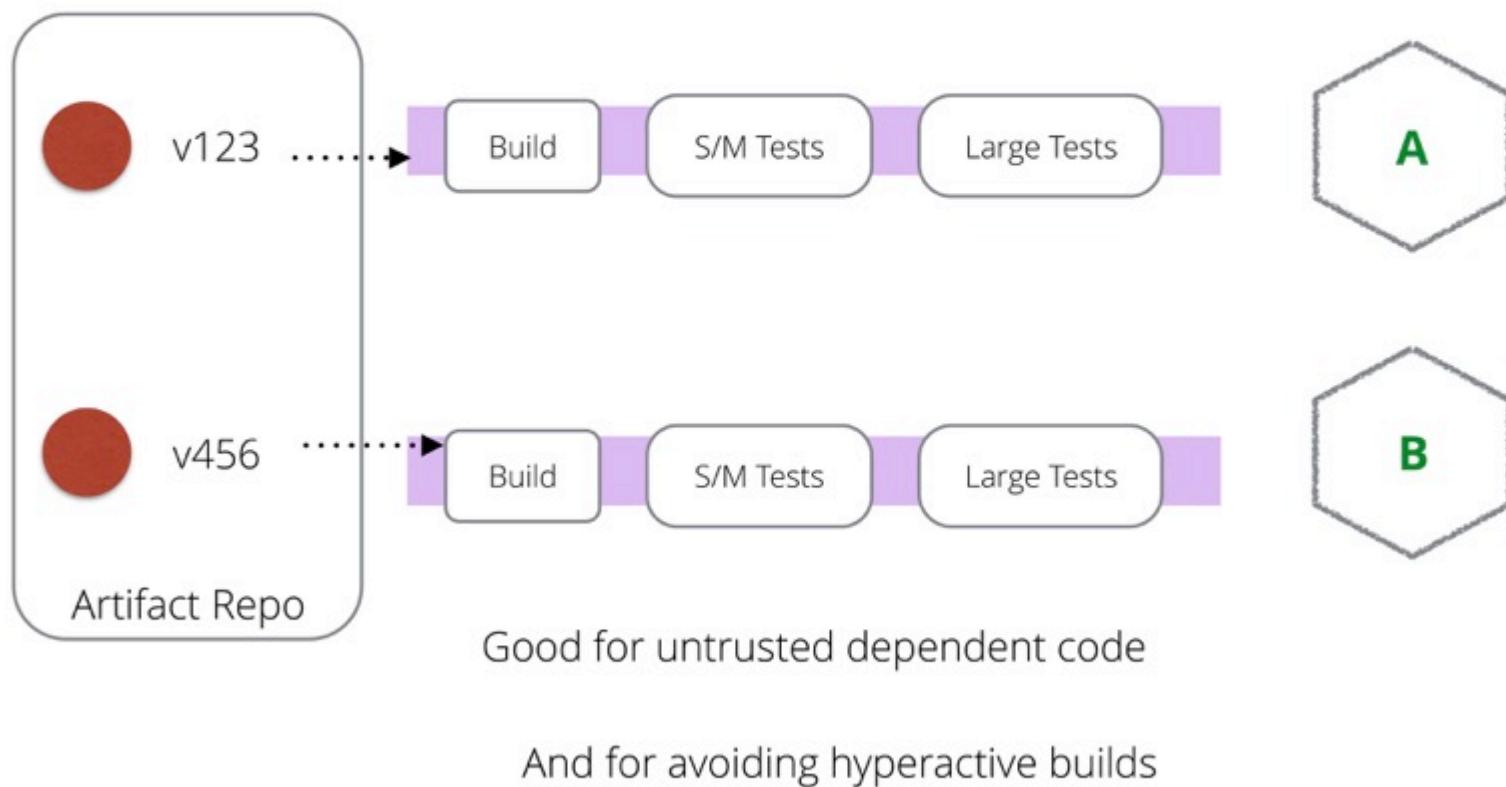
FIXED DEPENDENCIES



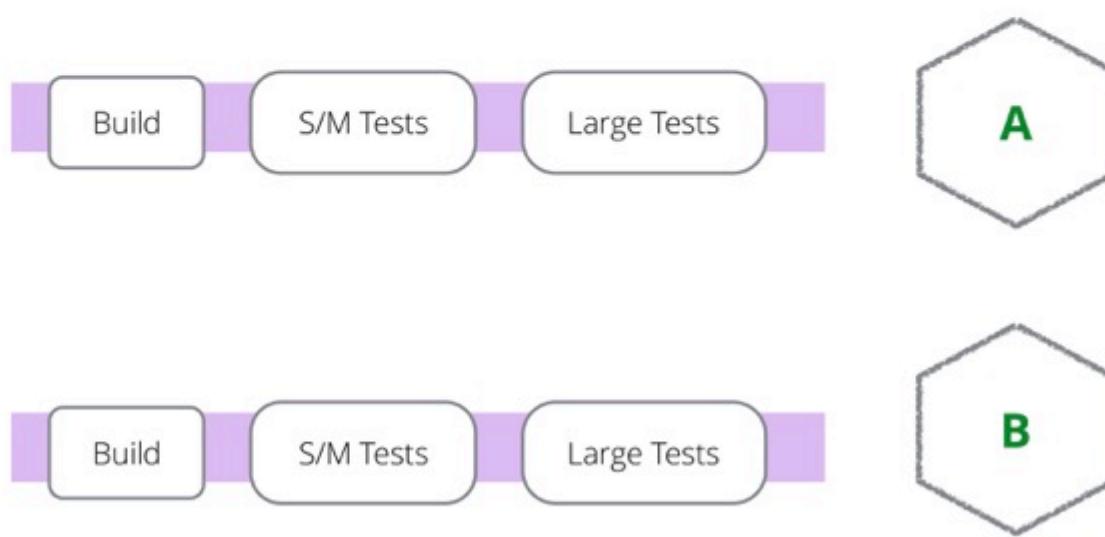
FIXED DEPENDENCIES



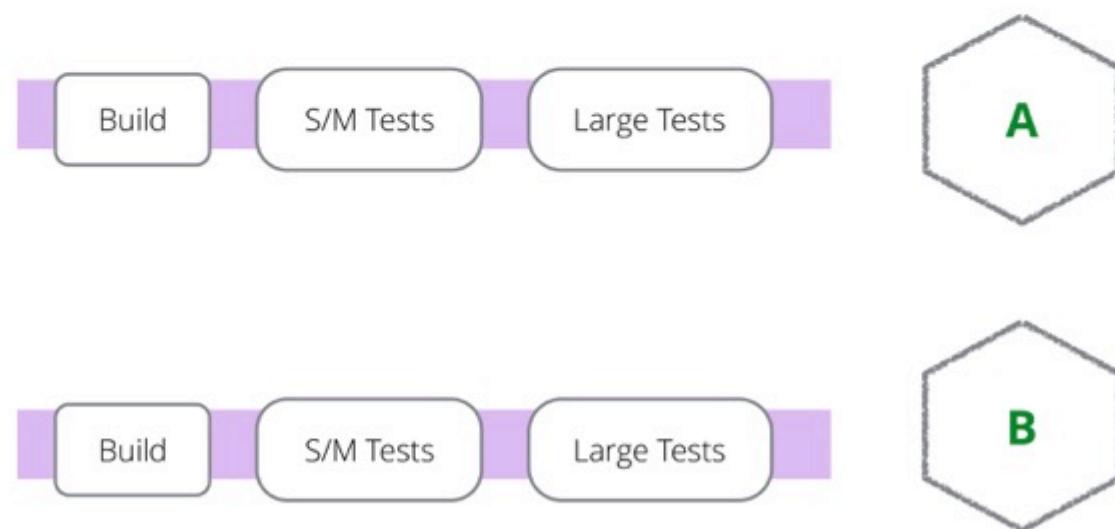
FIXED DEPENDENCIES



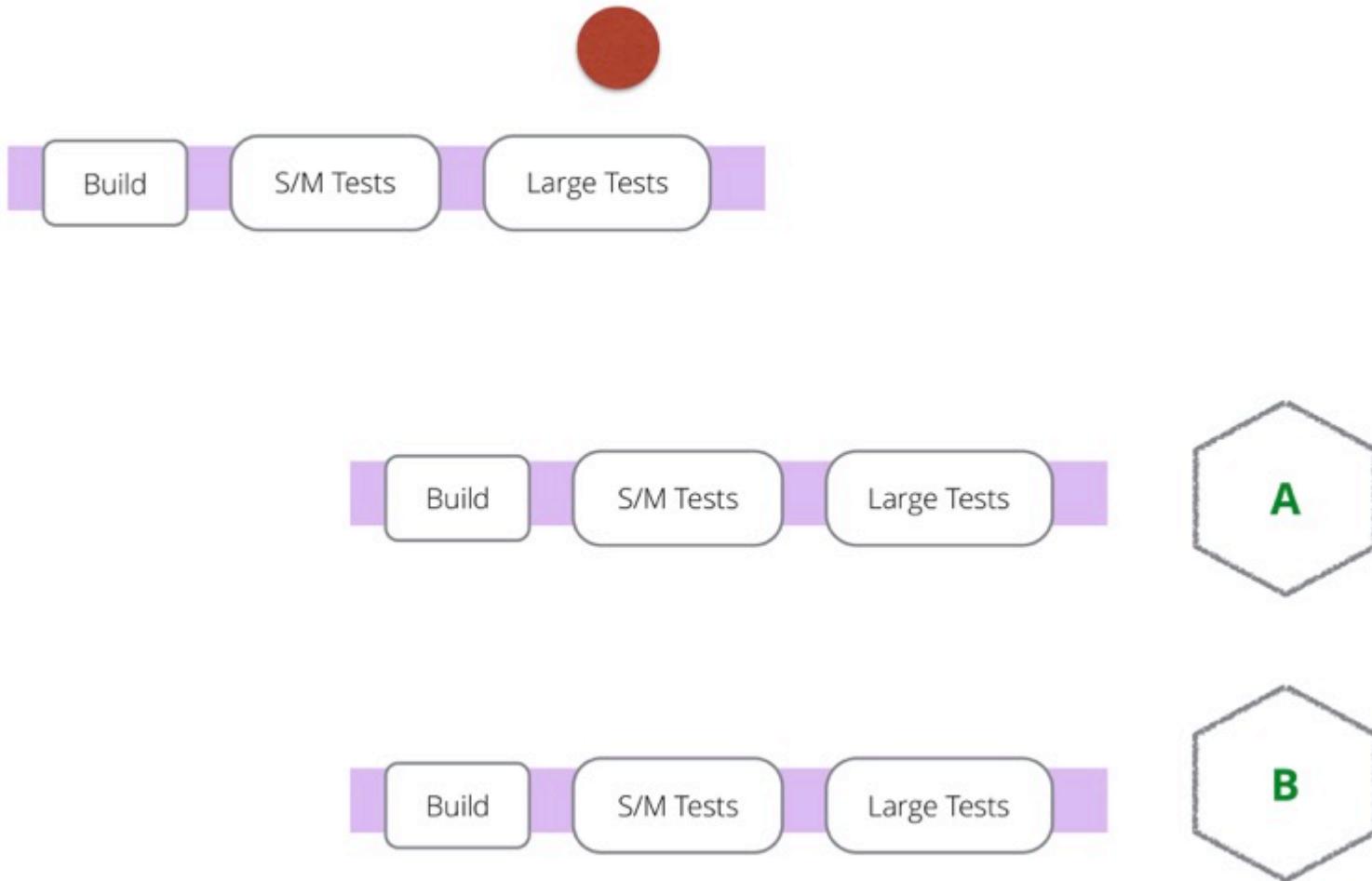
FLUID DEPENDENCIES



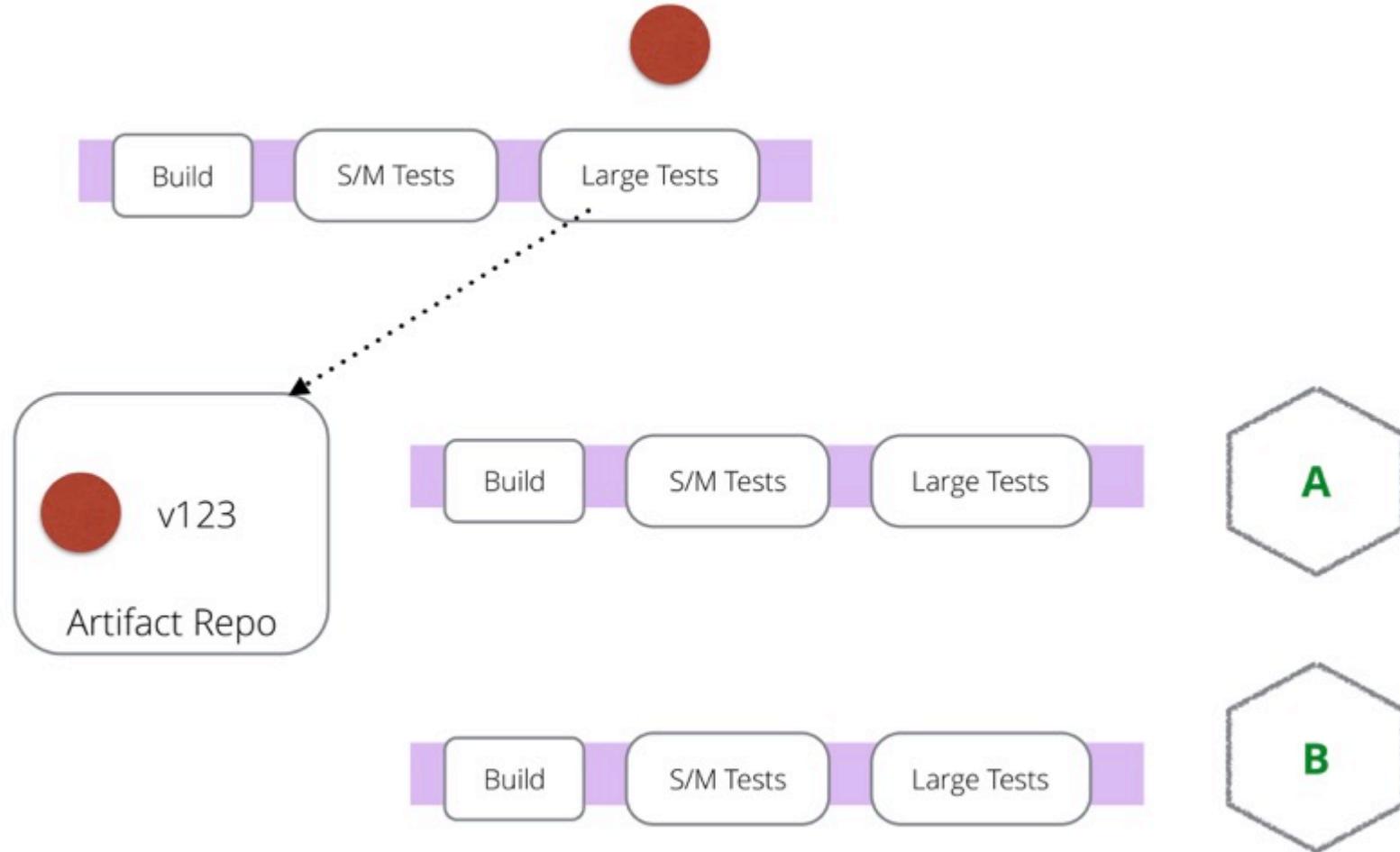
FLUID DEPENDENCIES



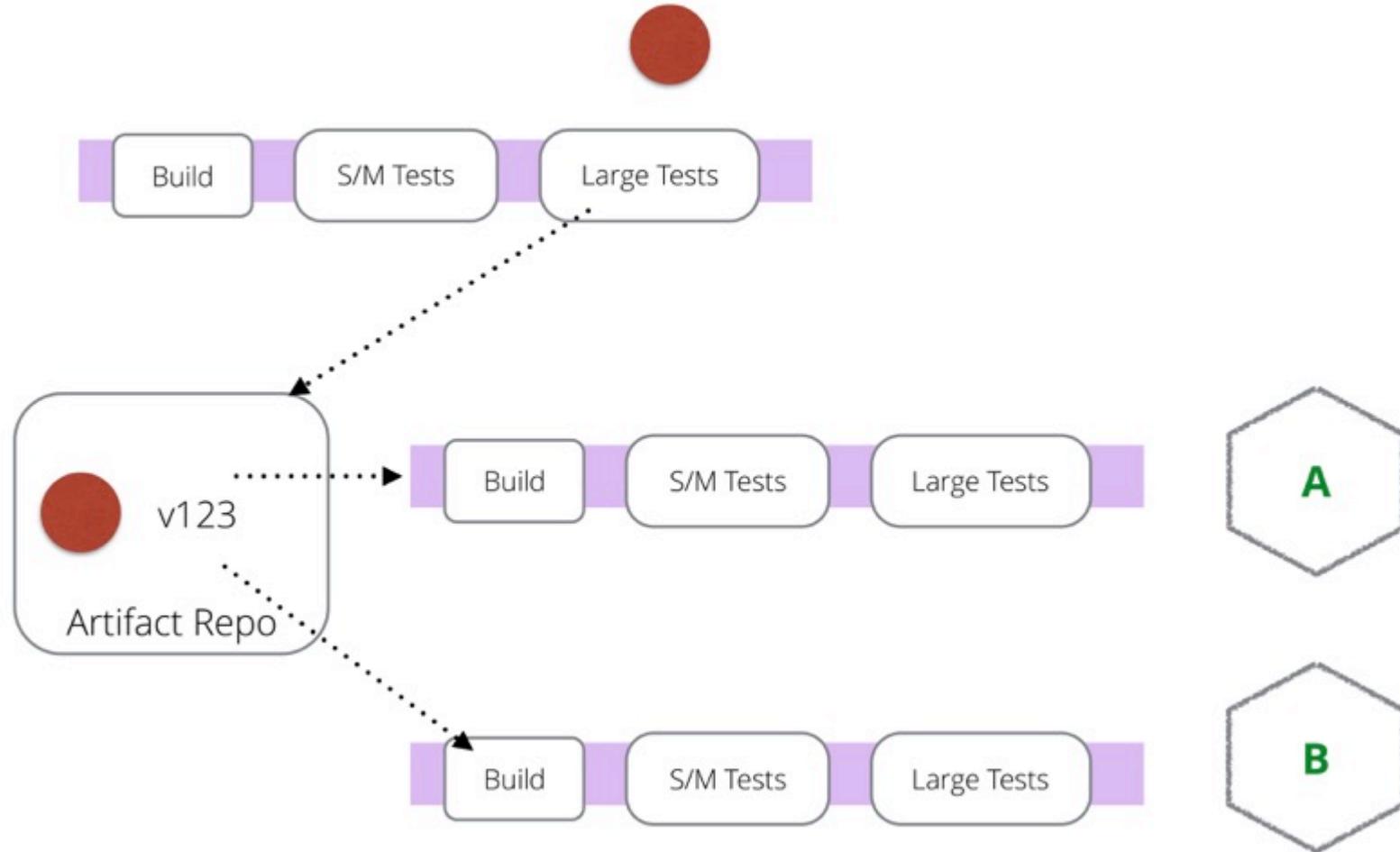
FLUID DEPENDENCIES



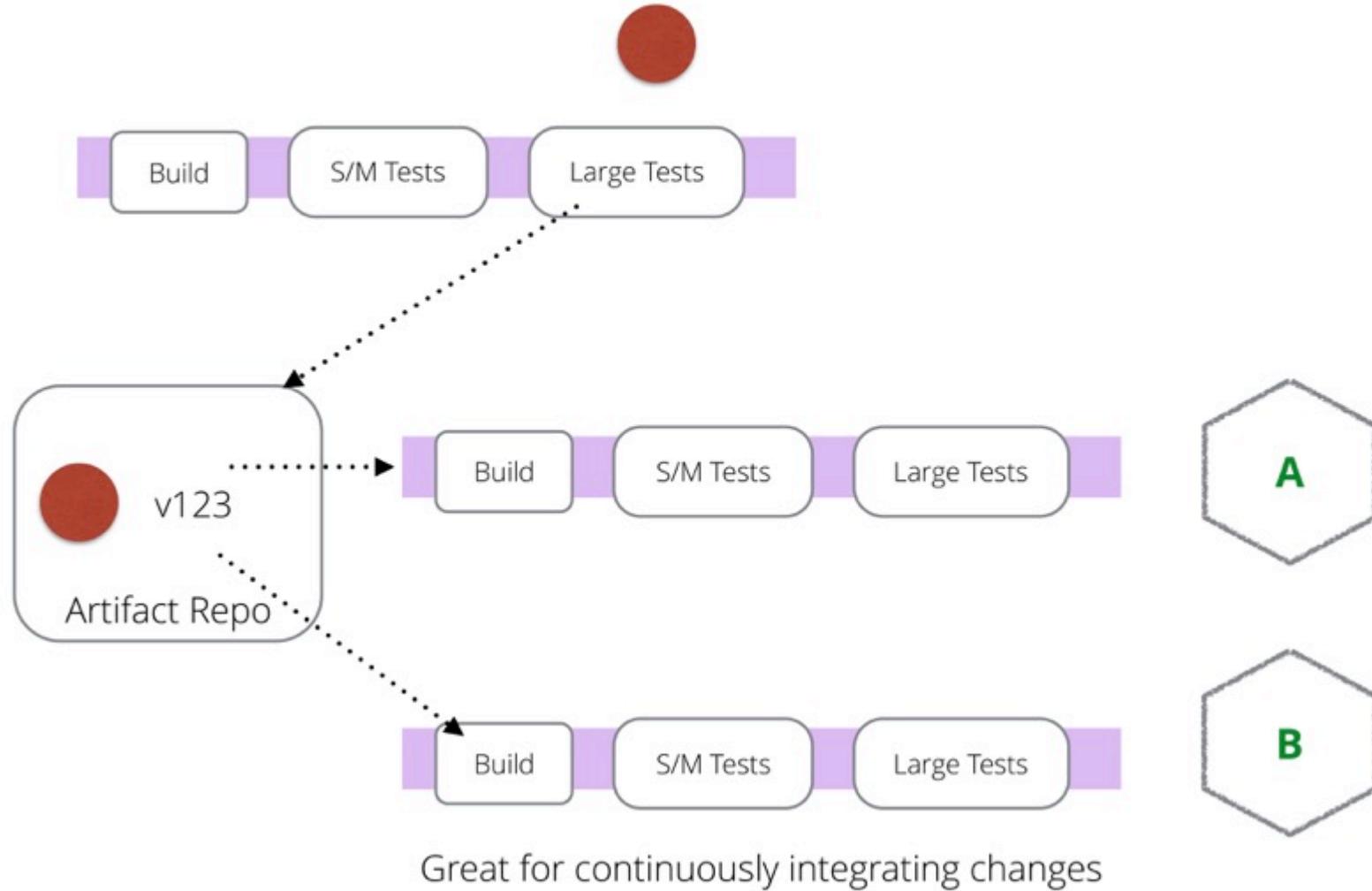
FLUID DEPENDENCIES



FLUID DEPENDENCIES



FLUID DEPENDENCIES













Can we still communicate?

Independent Deployability

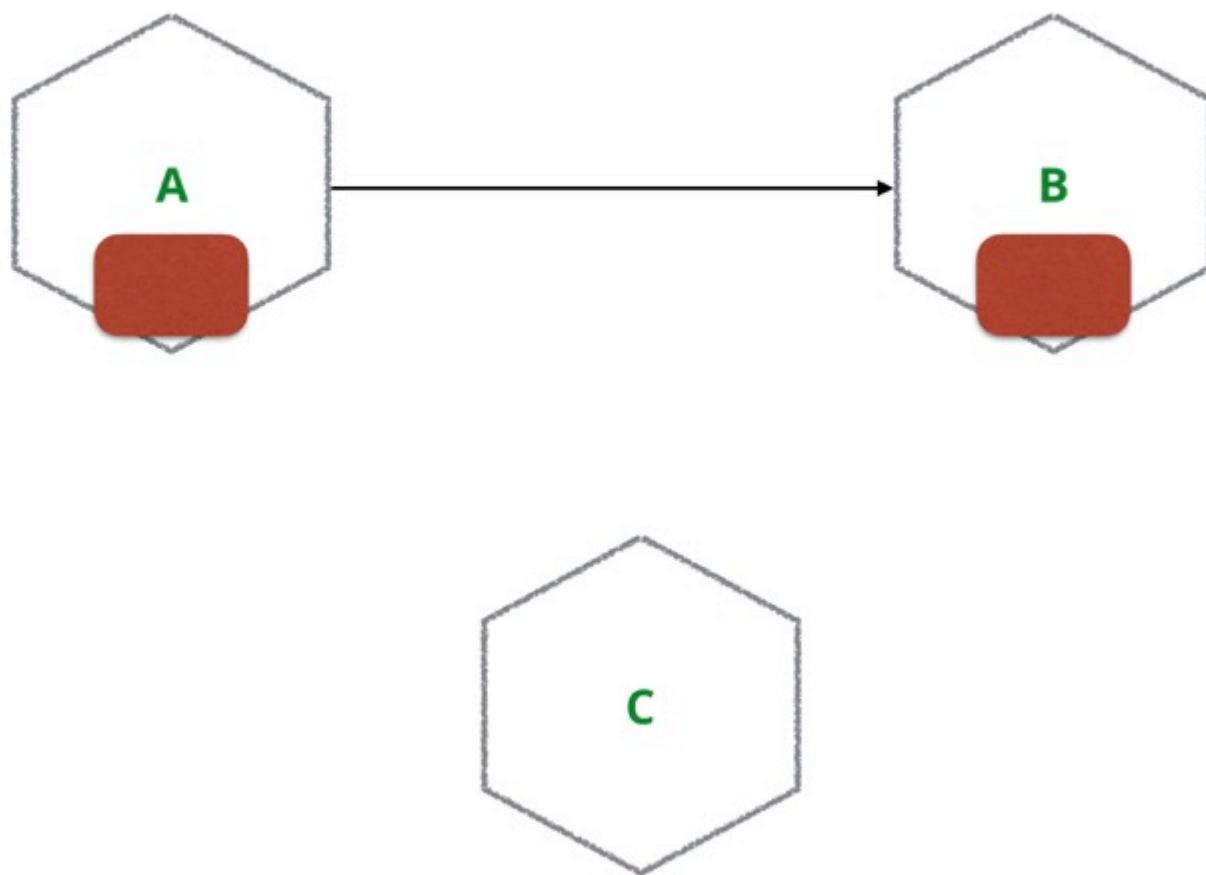
Independent Deployability

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

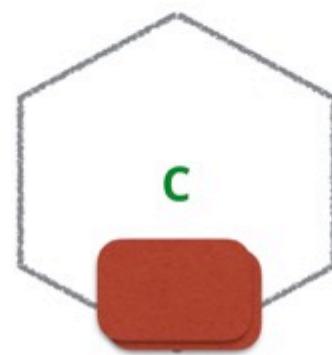
Consider service reuse rather
than library reuse



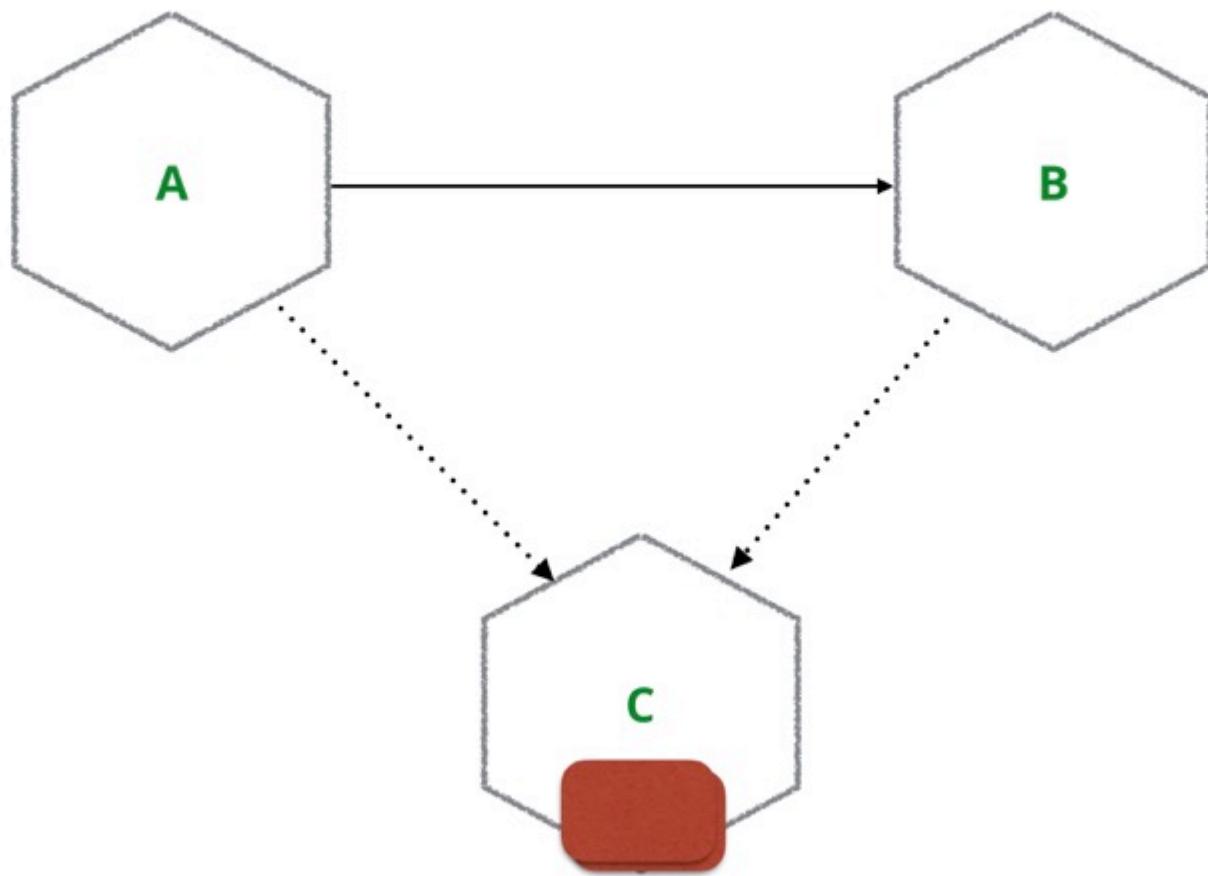
Consider service reuse rather
than library reuse



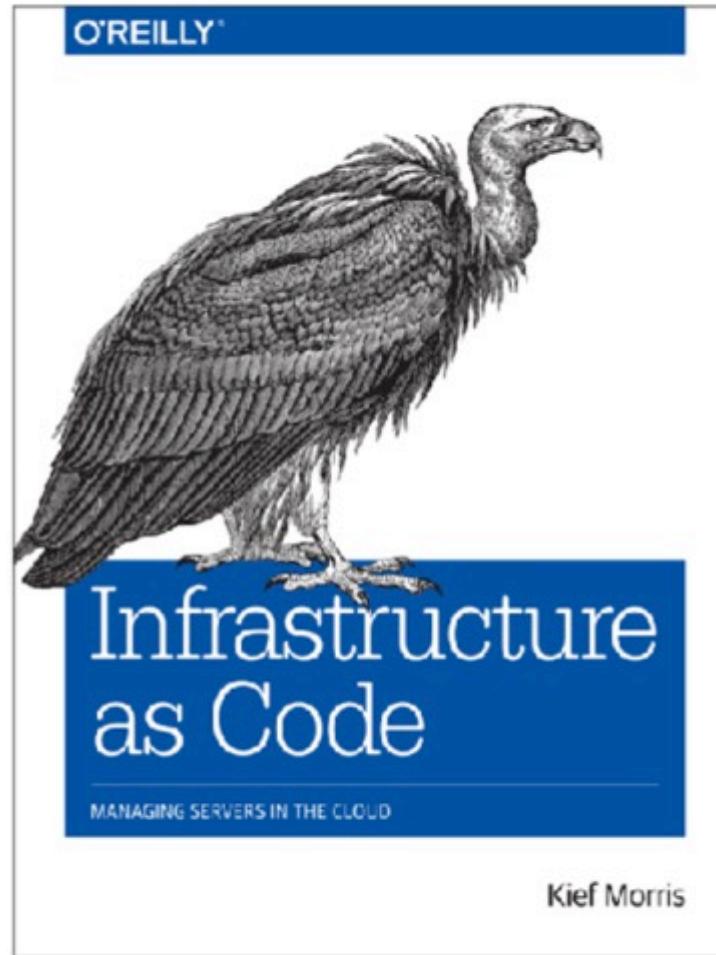
Consider service reuse rather
than library reuse



Consider service reuse rather
than library reuse



INFRASTRUCTURE AS CODE



<http://infrastructure-as-code.com/>

DECLARATIVE HOST CONFIGURATION

```
---
- name: Install Java 1.7
  yum: name=java-1.7.0-openjdk state=present

- name: add group "tomcat"
  group: name=tomcat

- name: add user "tomcat"
  user: name=tomcat group=tomcat home=/usr/share/tomcat createhome=no
  sudo: True

- name: Download Tomcat
  get_url: url=http://archive.apache.org/dist/tomcat/tomcat-7/v7.0.61/bin/apache-tomca
  ...

- name: Extract archive
  command: chdir=/usr/share /bin/tar xvf /opt/apache-tomcat-7.0.61.tar.gz -C /opt/ cre
  ...

- name: Symlink install directory
  file: src=/opt/apache-tomcat-7.0.61 path=/usr/share/tomcat state=link

- name: Change ownership of Tomcat installation
  file: path=/usr/share/tomcat/ owner=tomcat group=tomcat state=directory recurse=yes
```



CHEF™



CHEF™





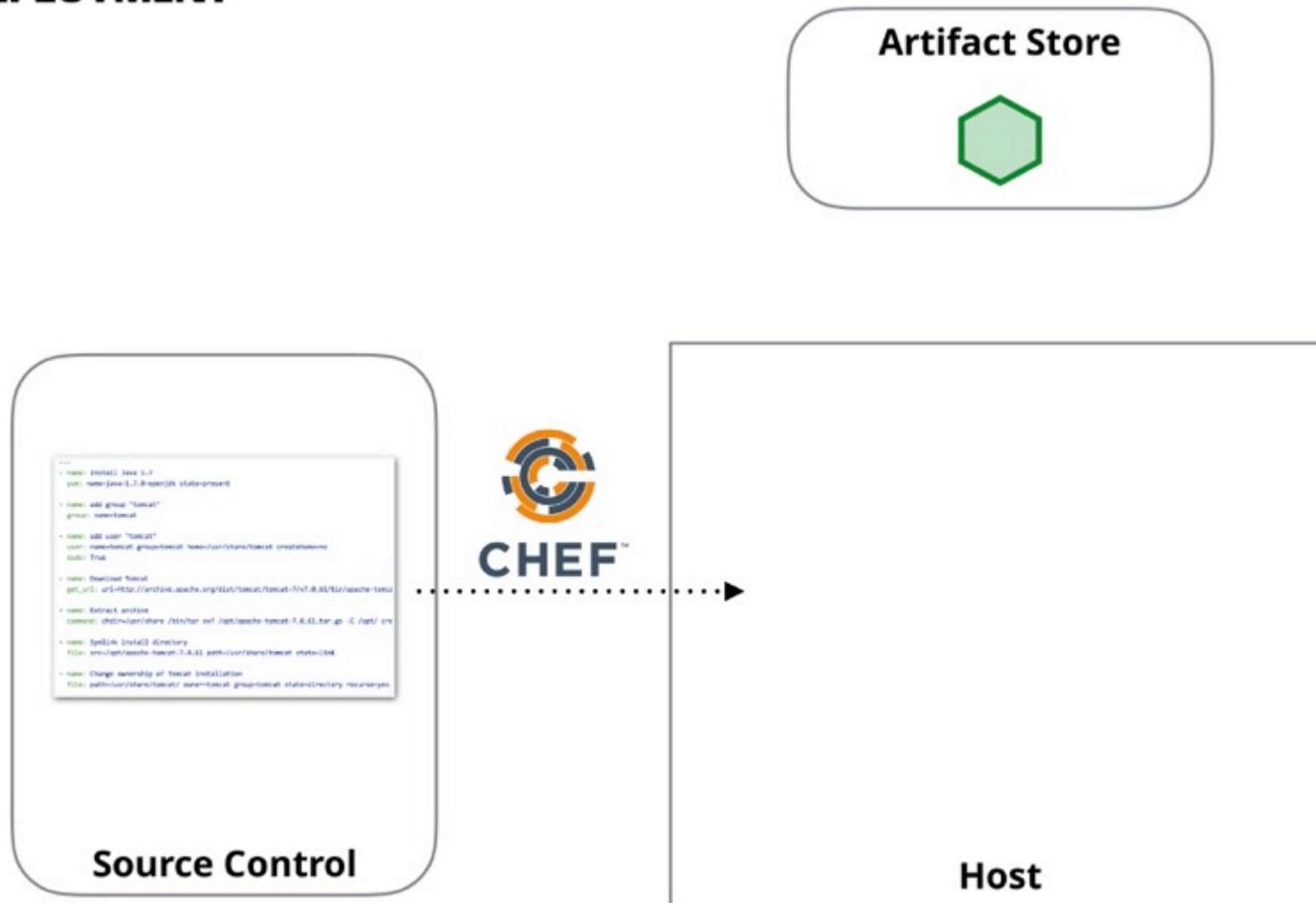
CHEF™



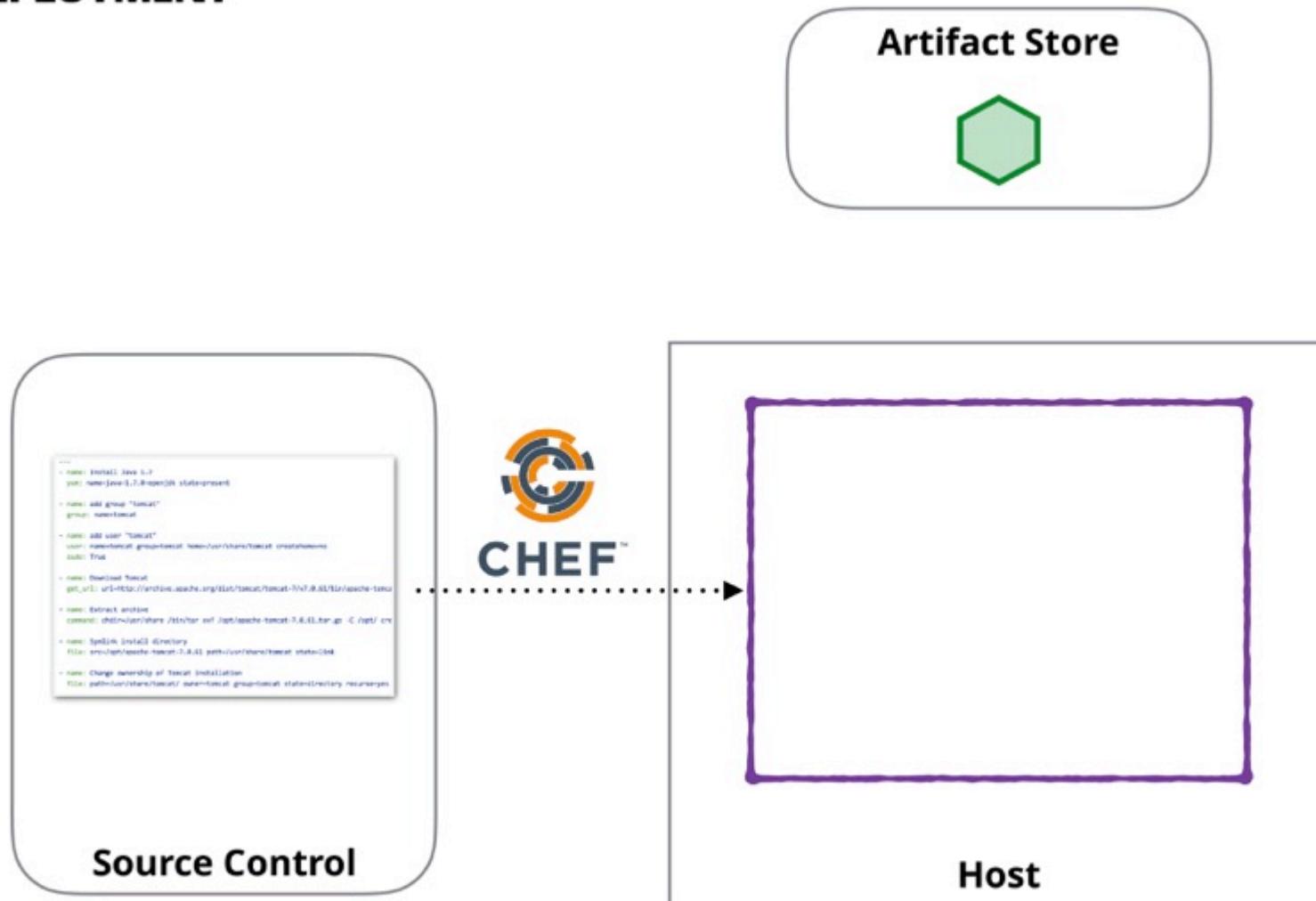
DEPLOYMENT



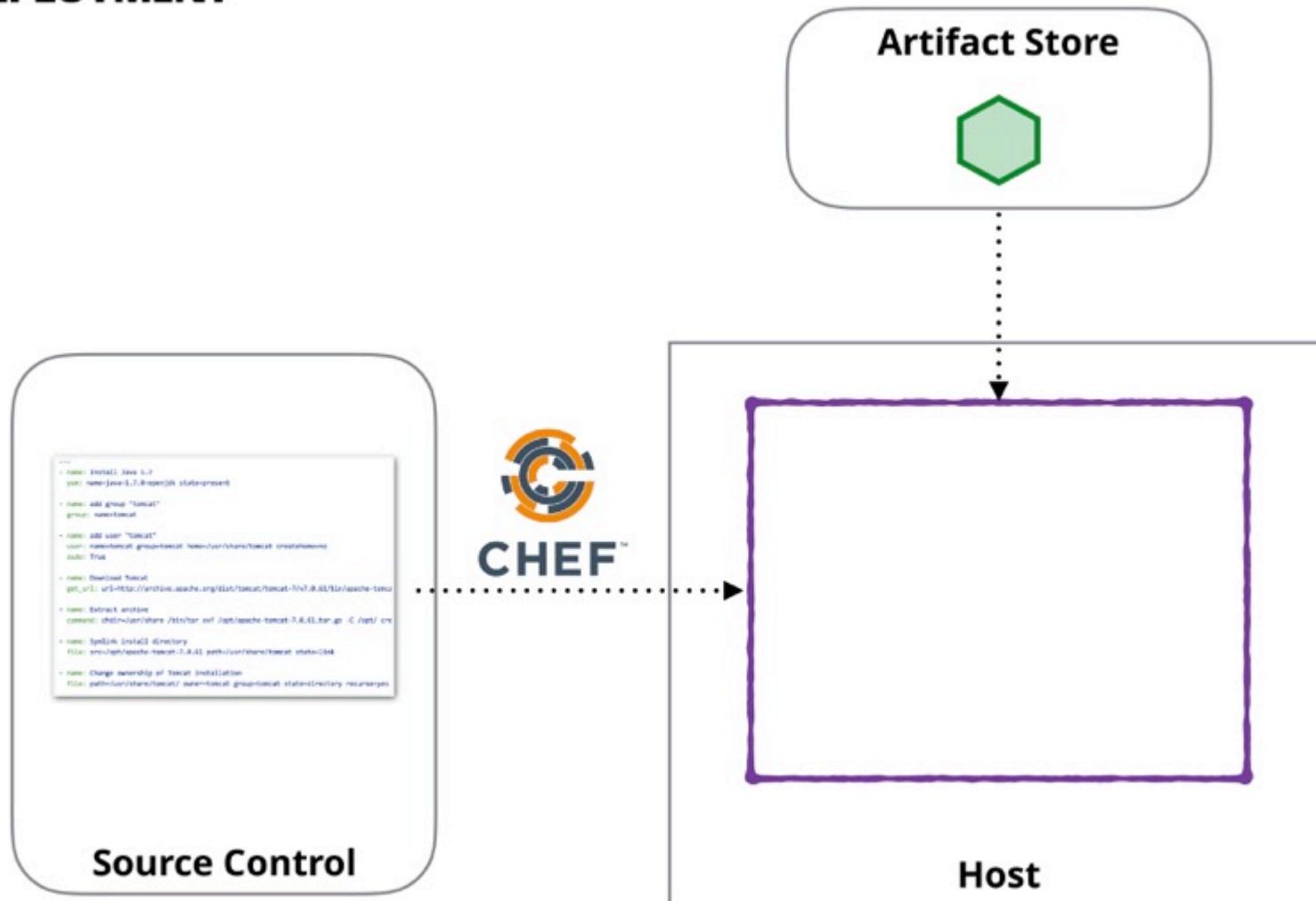
DEPLOYMENT



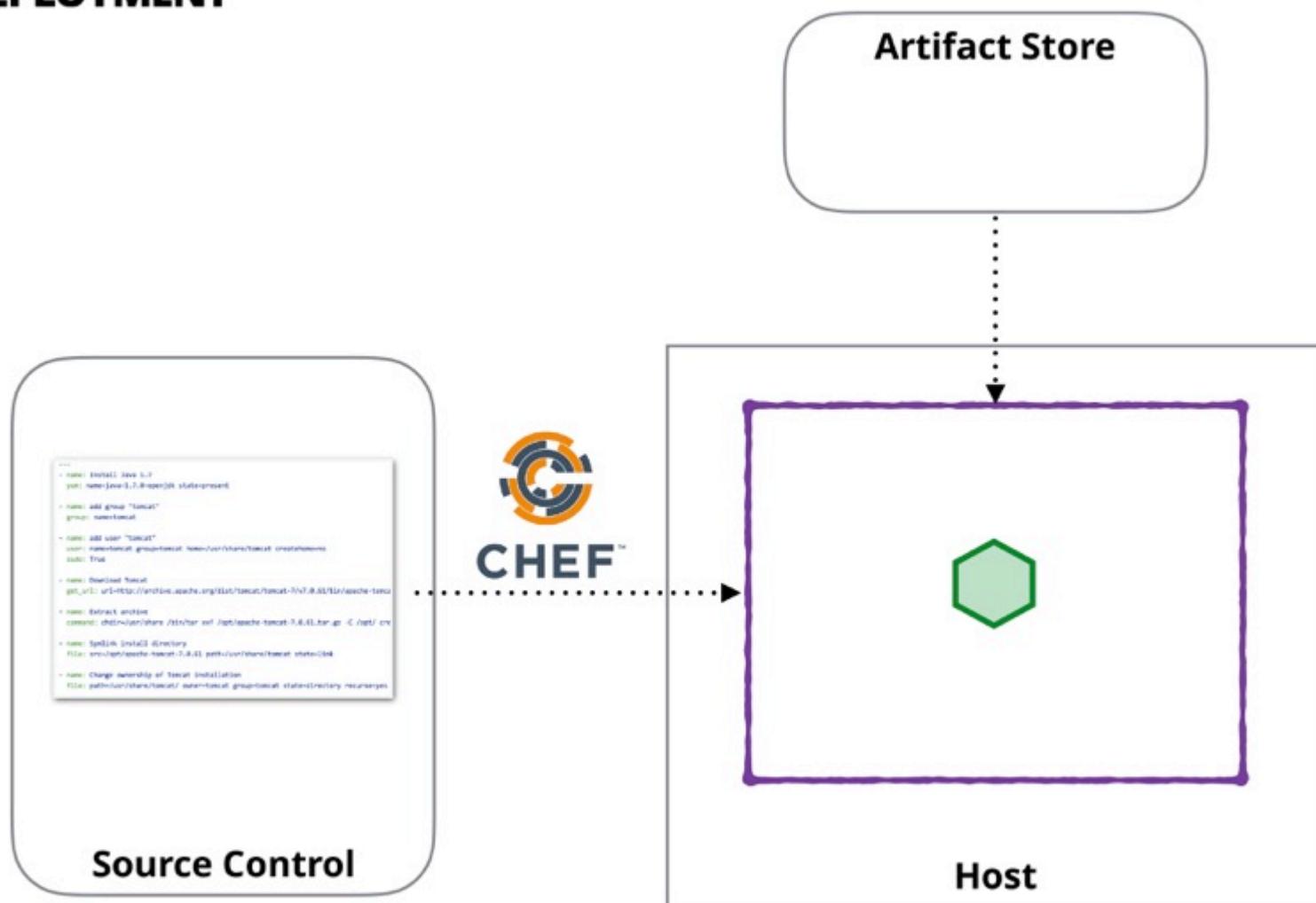
DEPLOYMENT



DEPLOYMENT



DEPLOYMENT



INFRASTRUCTURE AS CODE

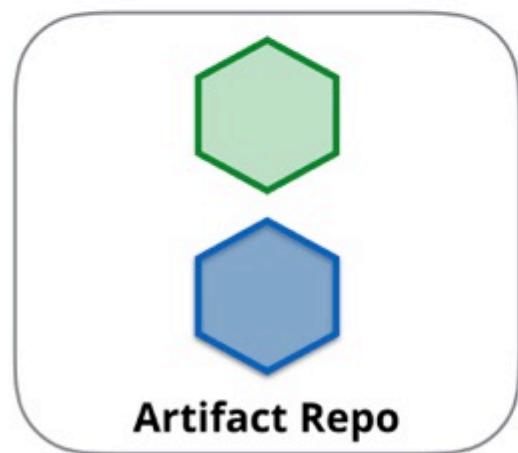
Machine configuration version controlled

Reproduceable

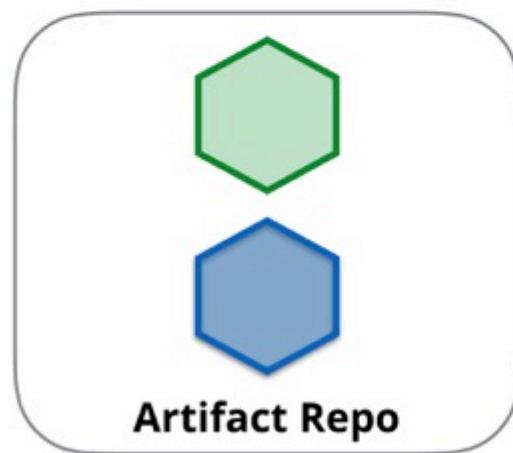
Help rebuild machines

Manage at scale

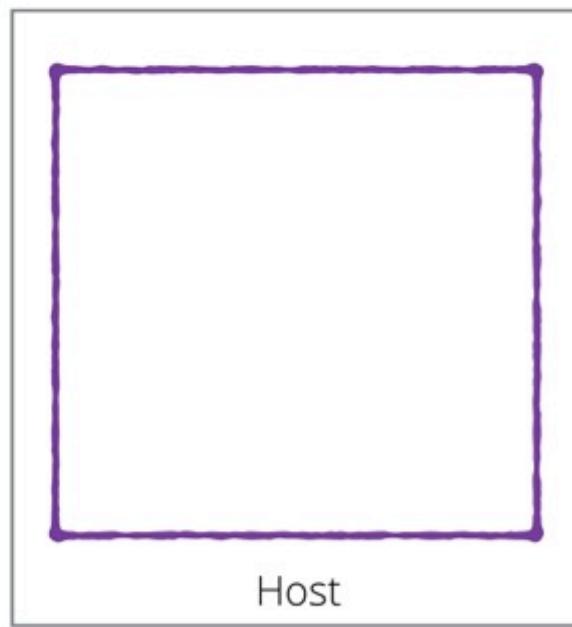
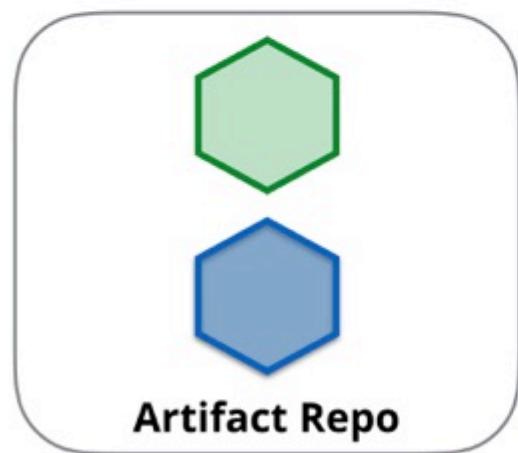
CO-EXISTING SERVICES



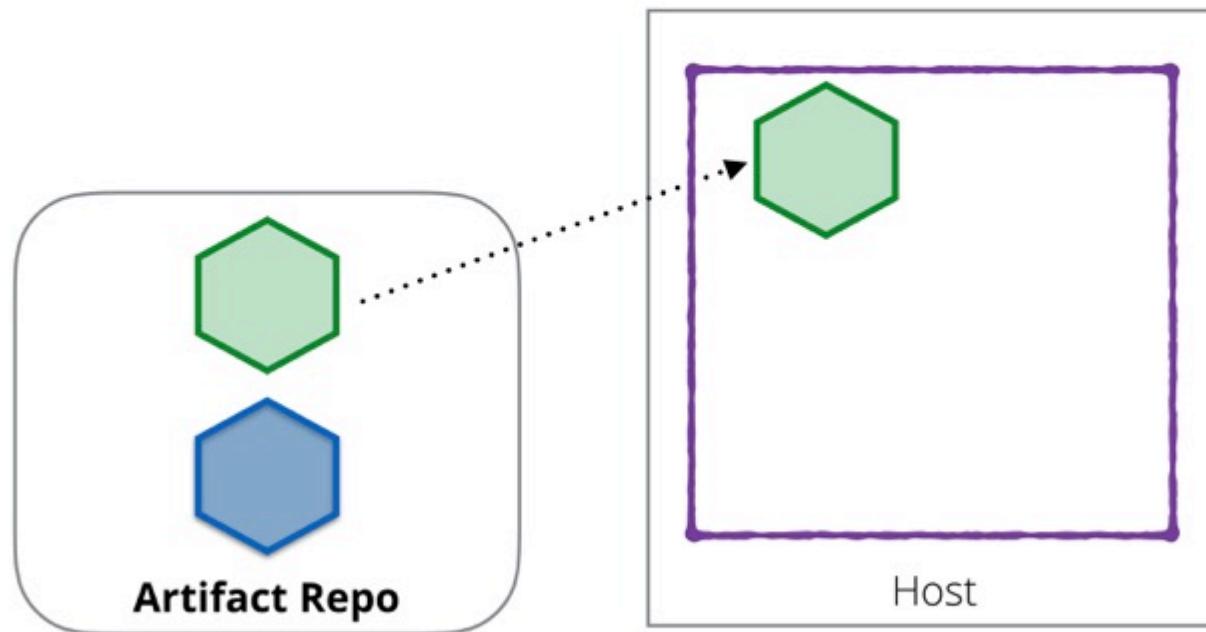
CO-EXISTING SERVICES



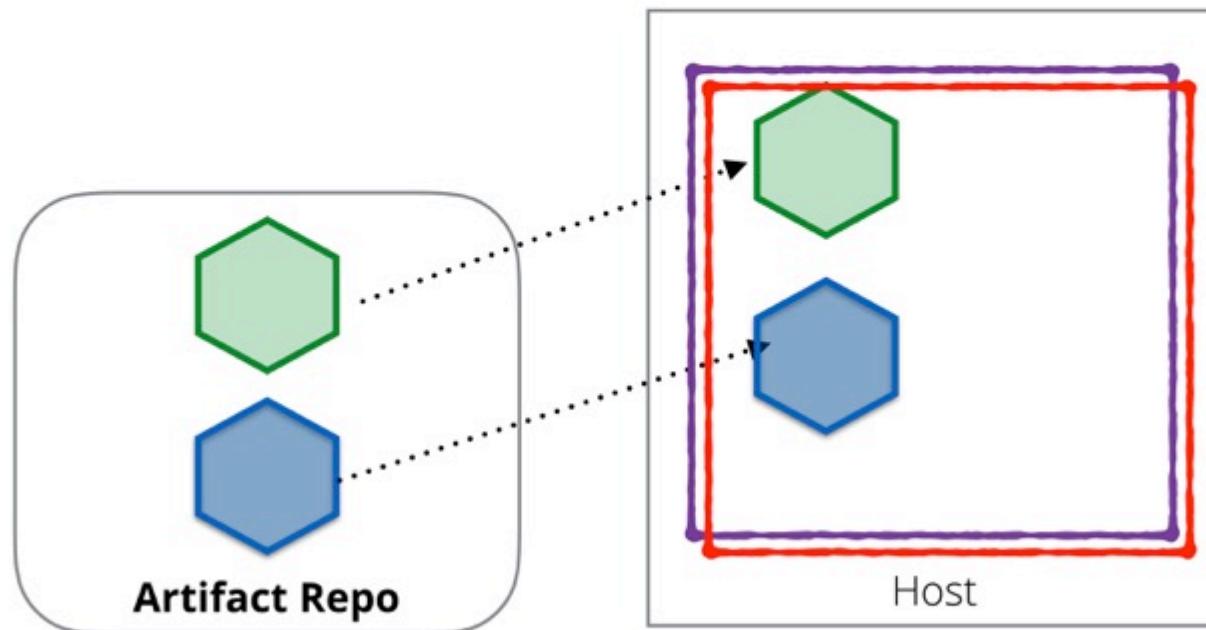
CO-EXISTING SERVICES



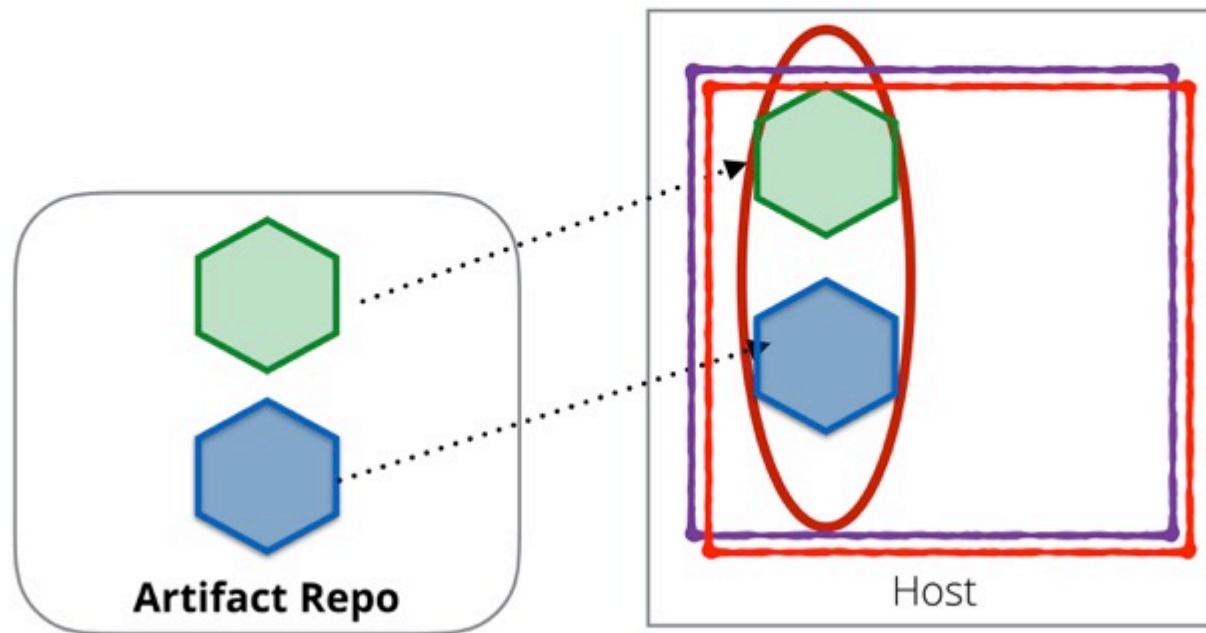
CO-EXISTING SERVICES

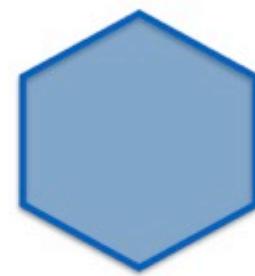
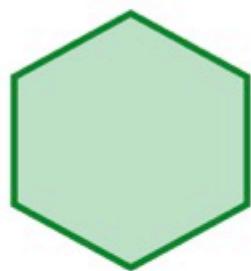


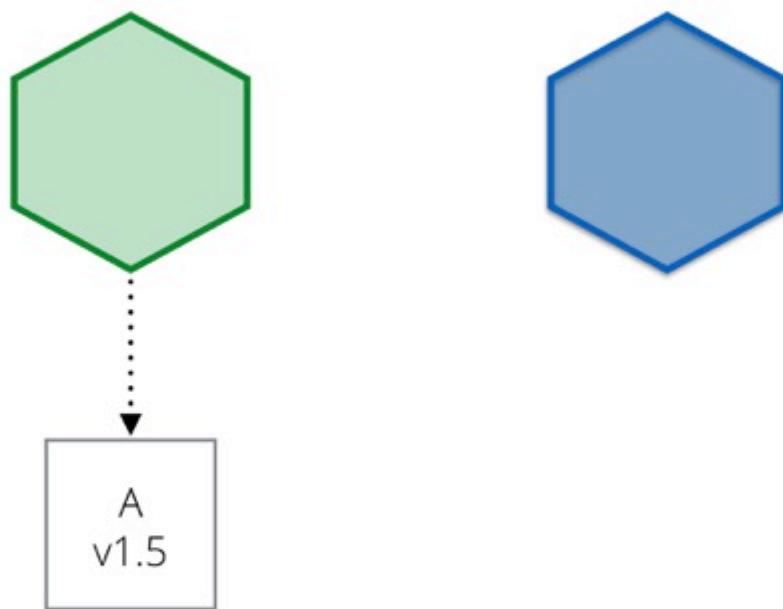
CO-EXISTING SERVICES

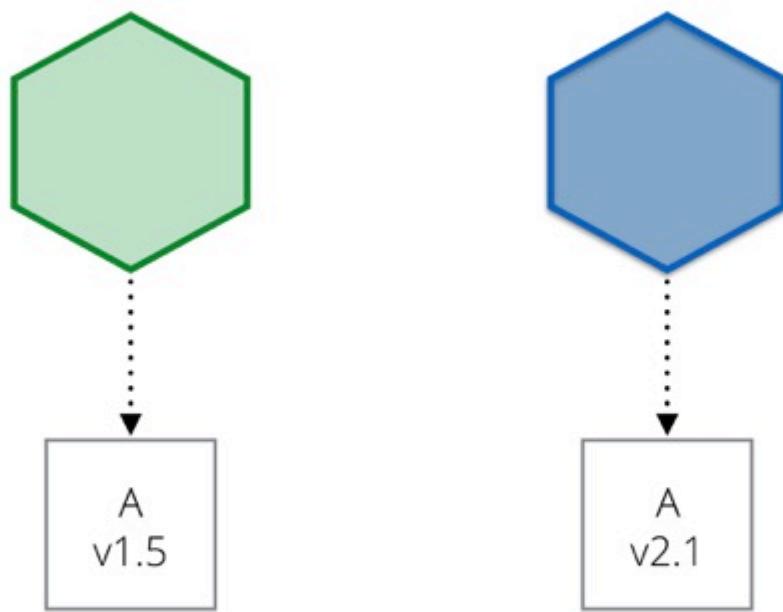


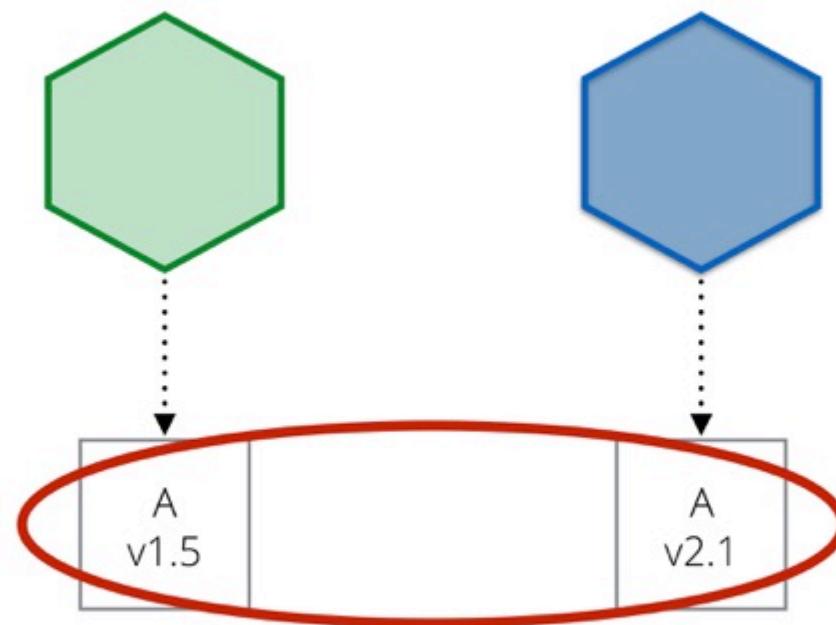
CO-EXISTING SERVICES

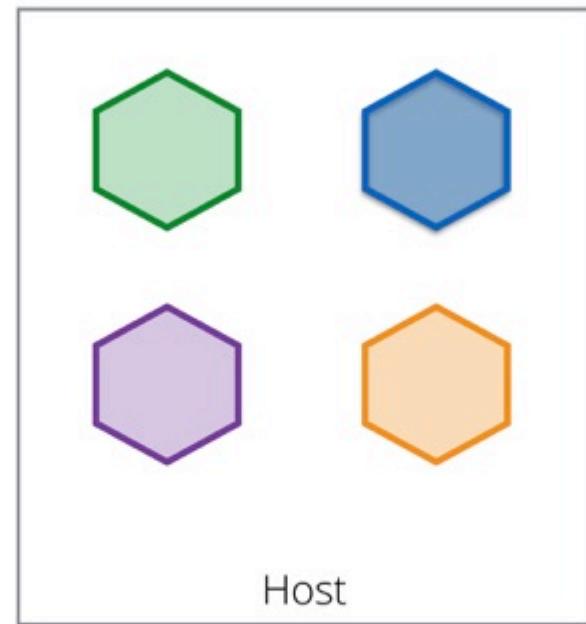


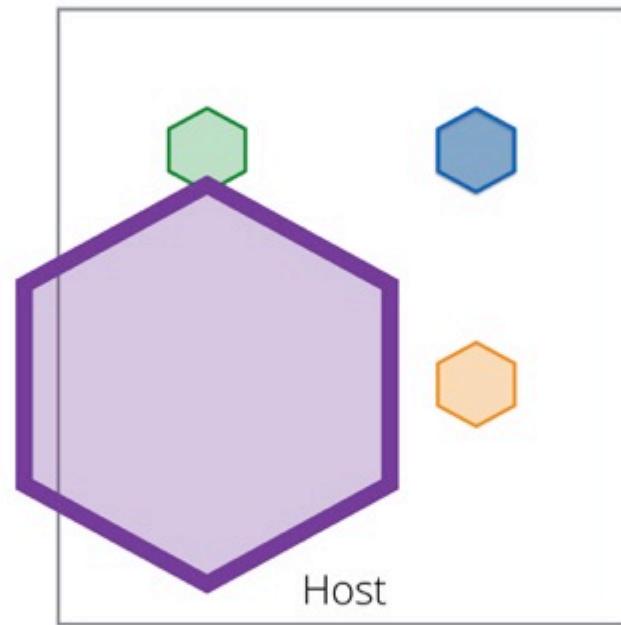


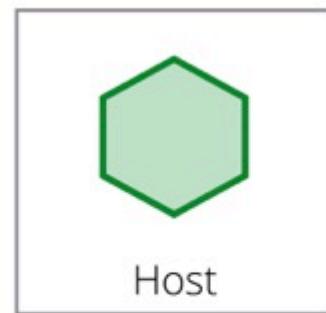




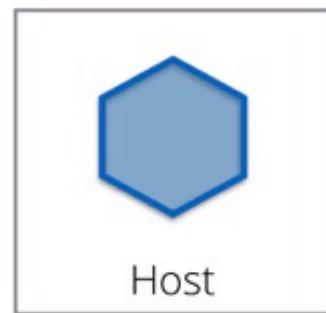




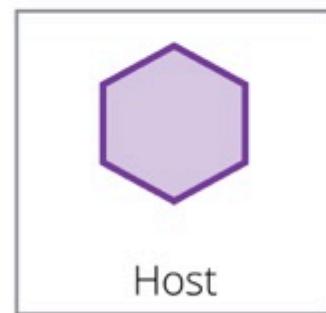




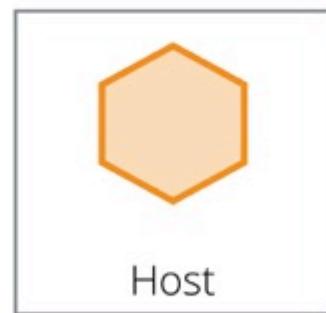
Host



Host



Host



Host

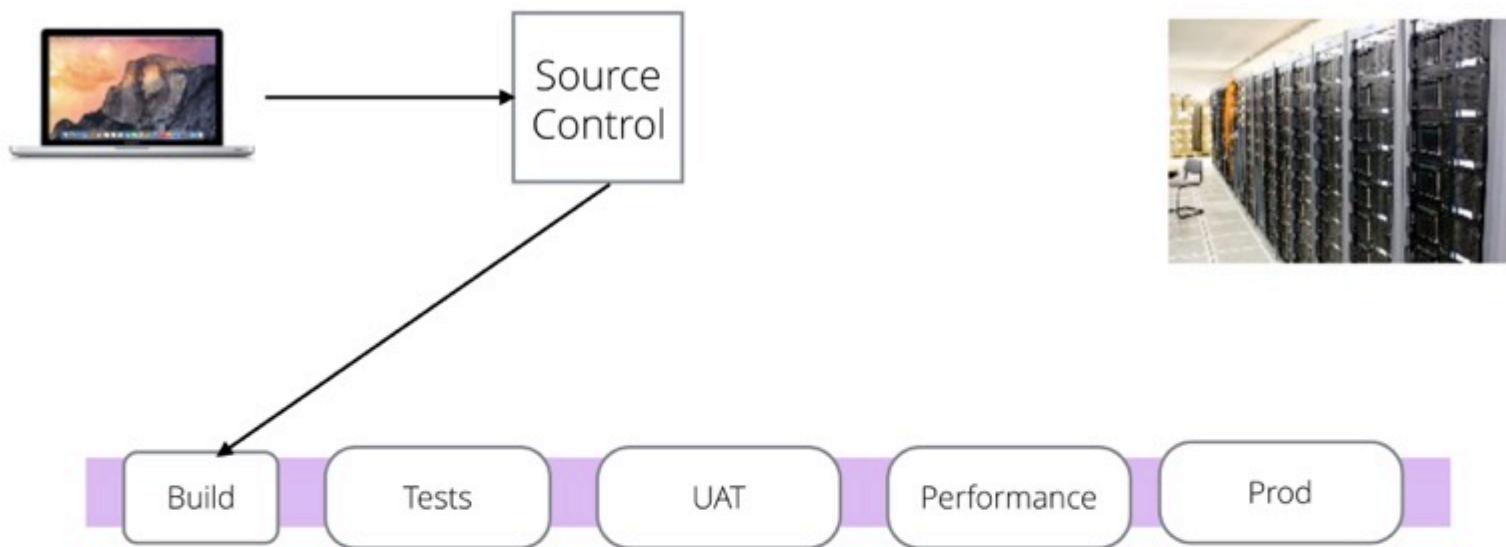


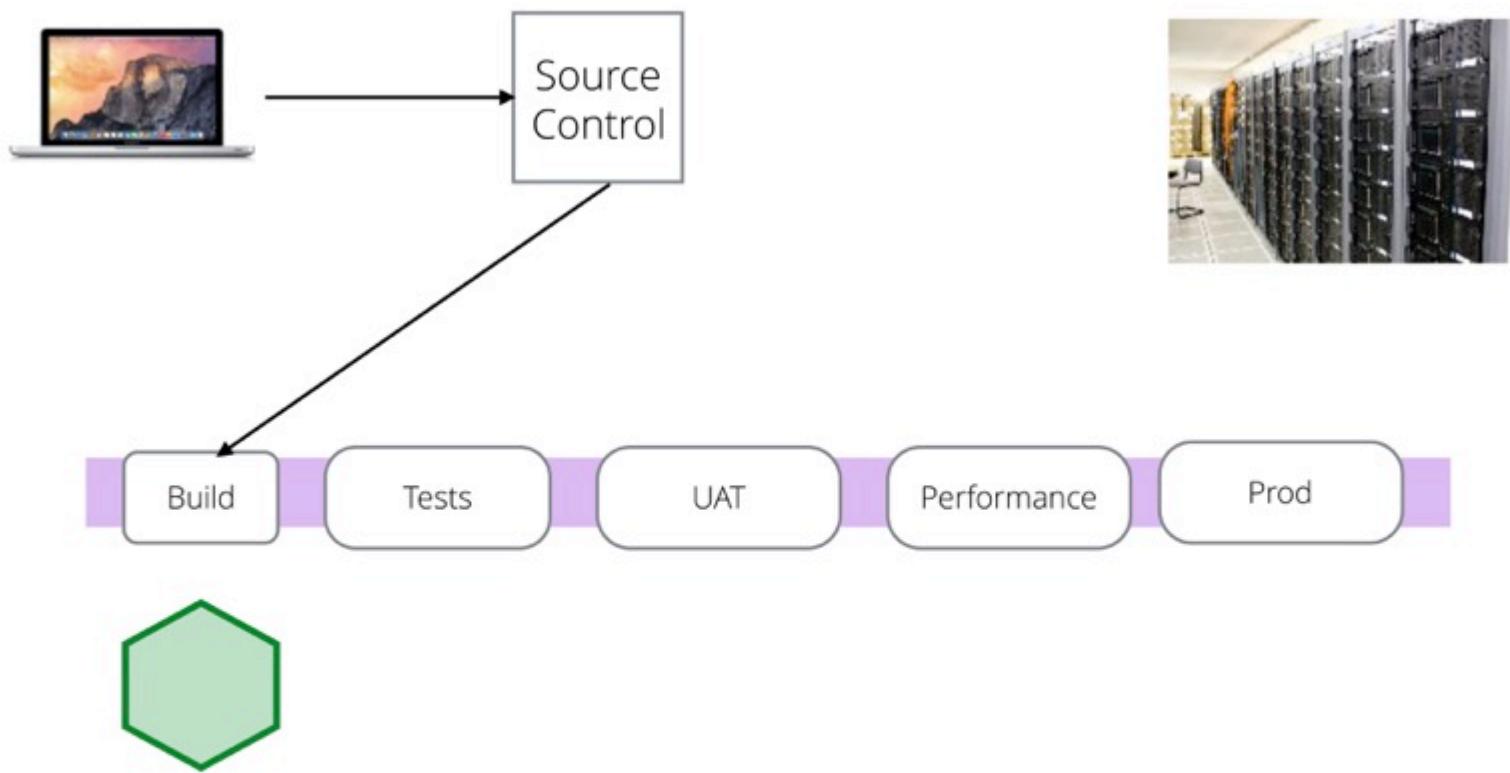
Separate execution environment
makes independent deployment
safer

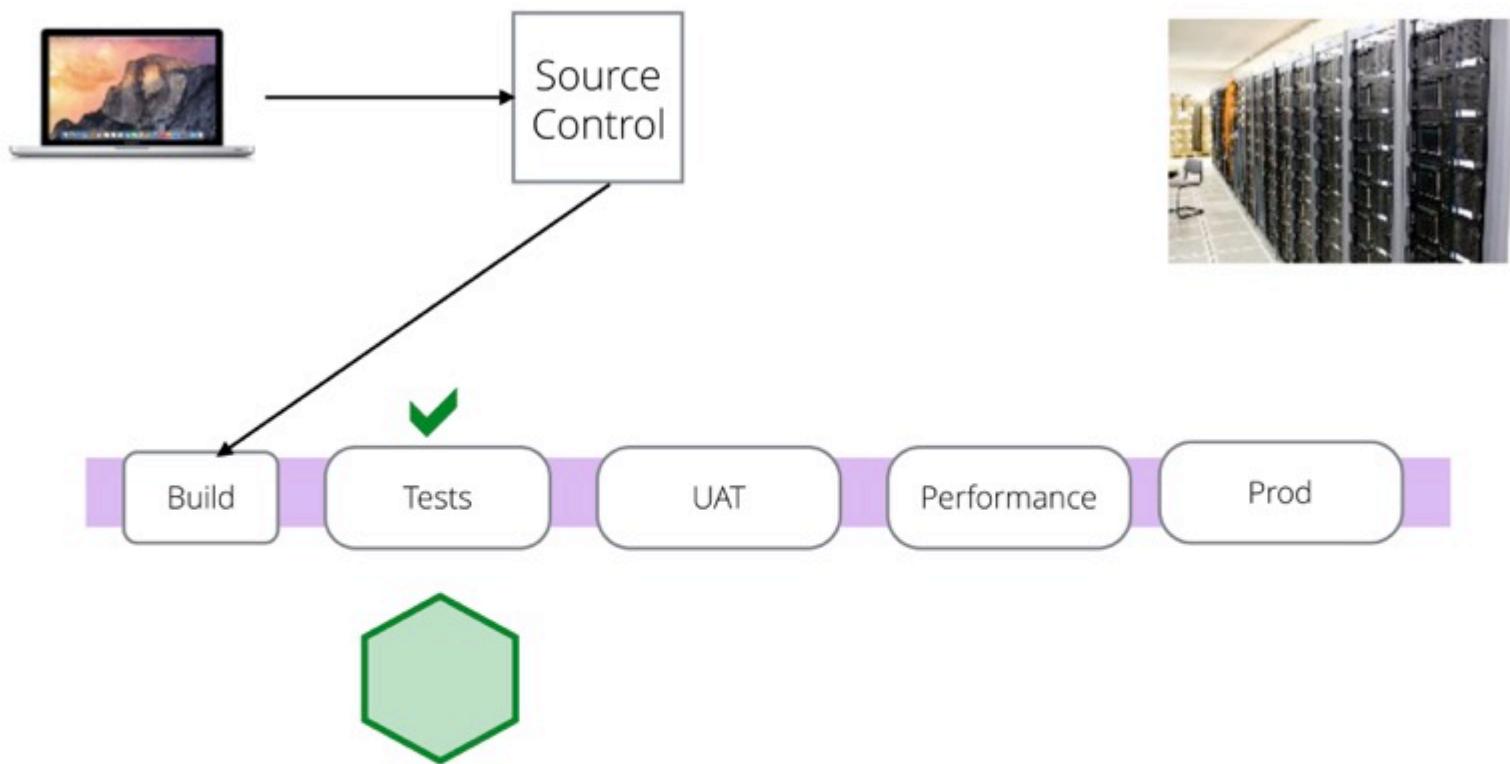


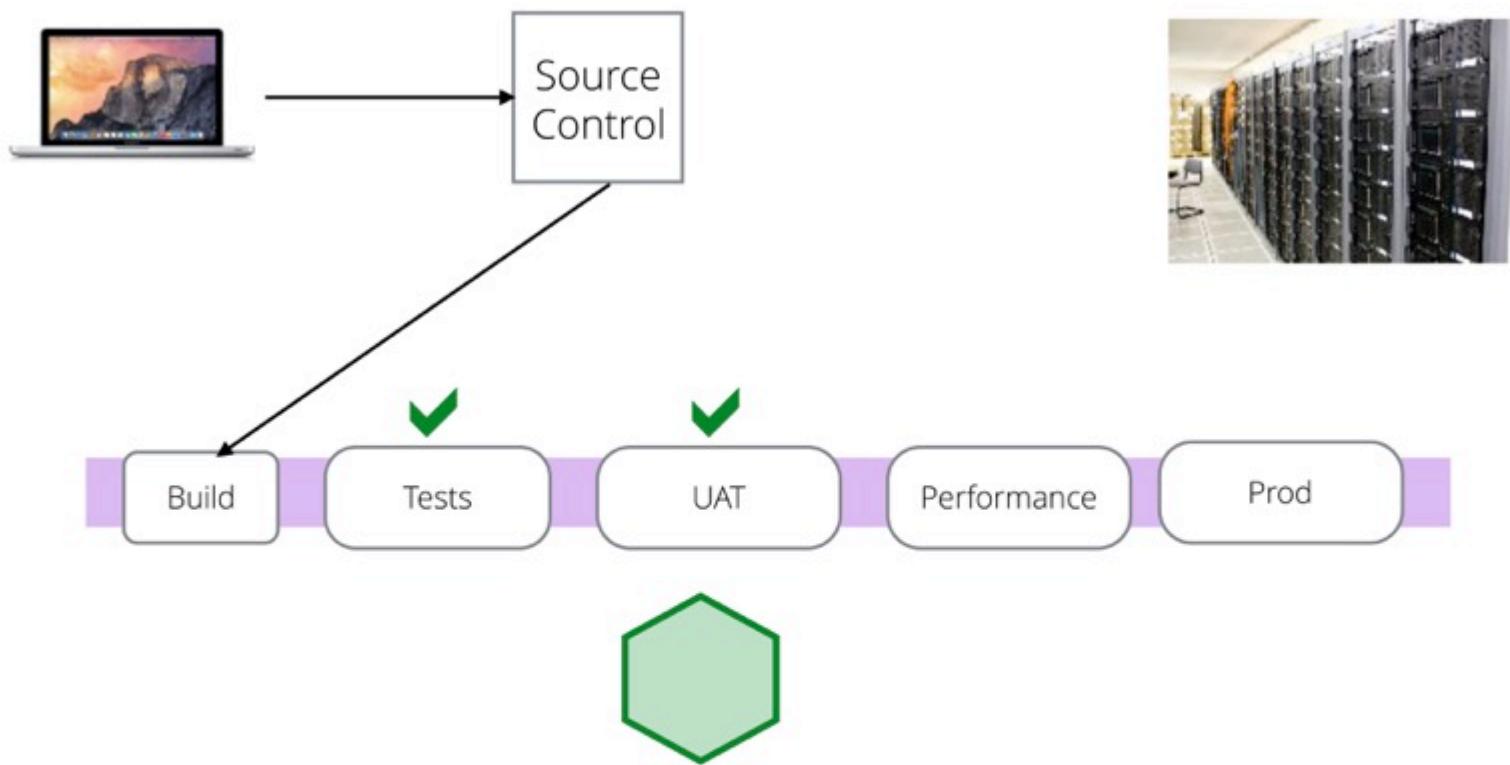


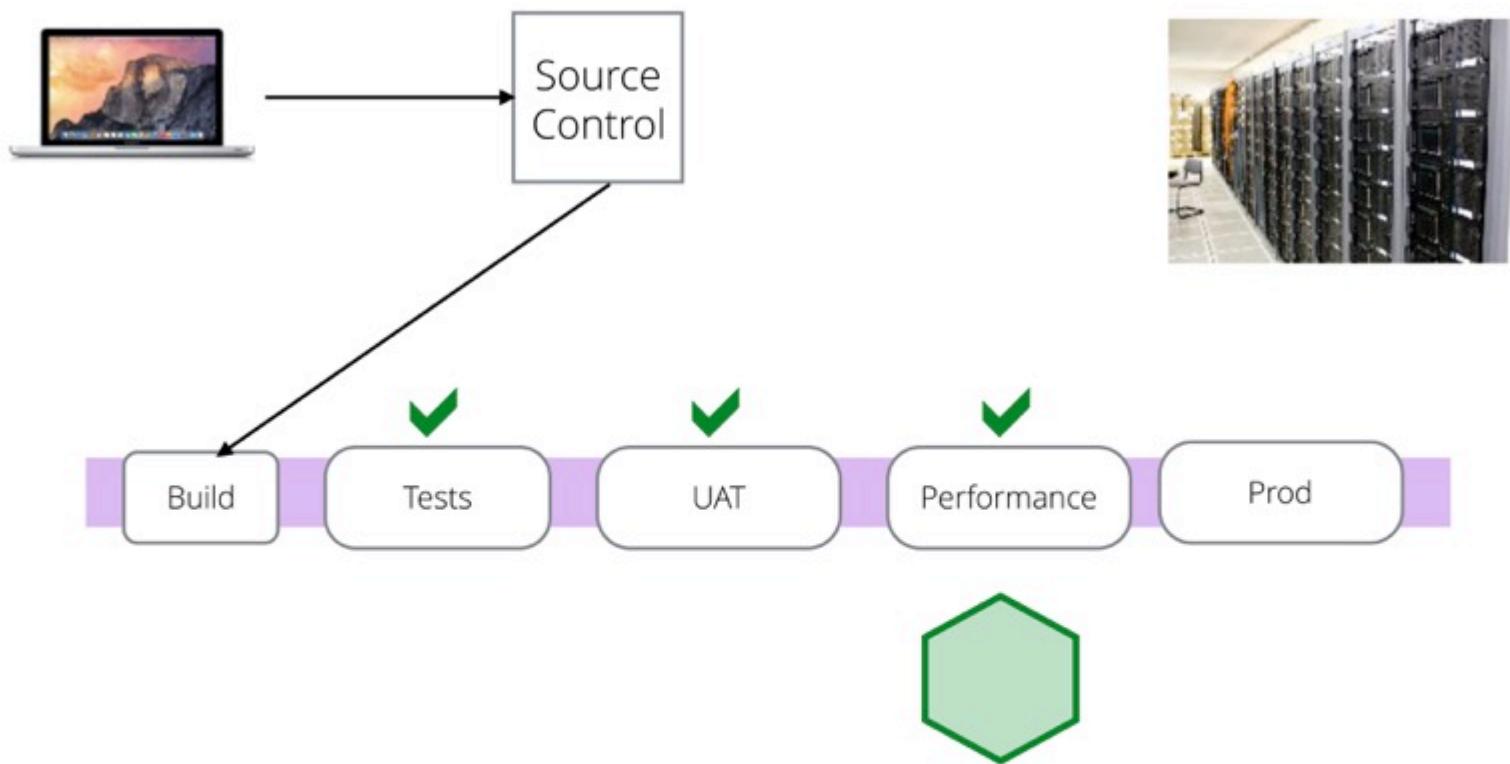


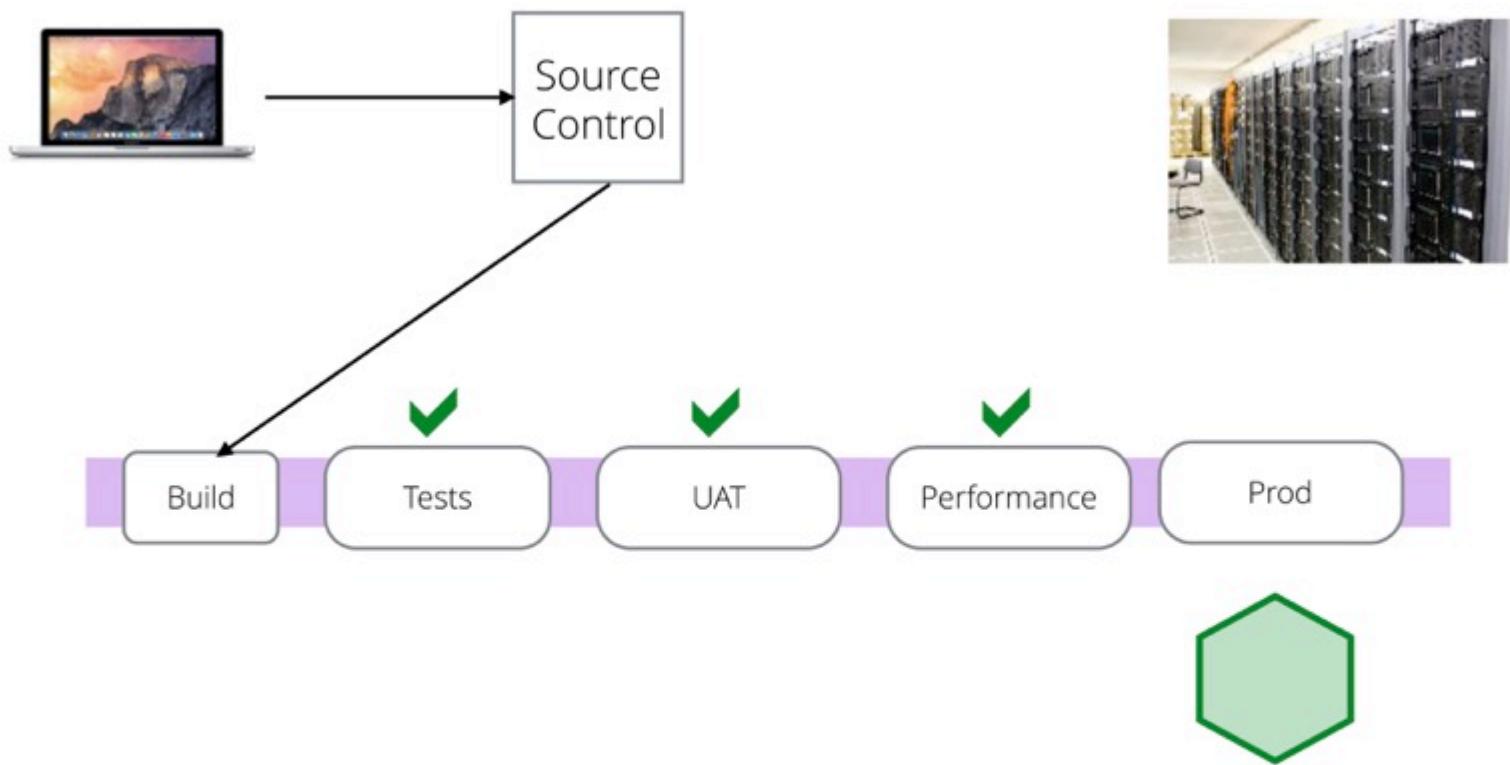


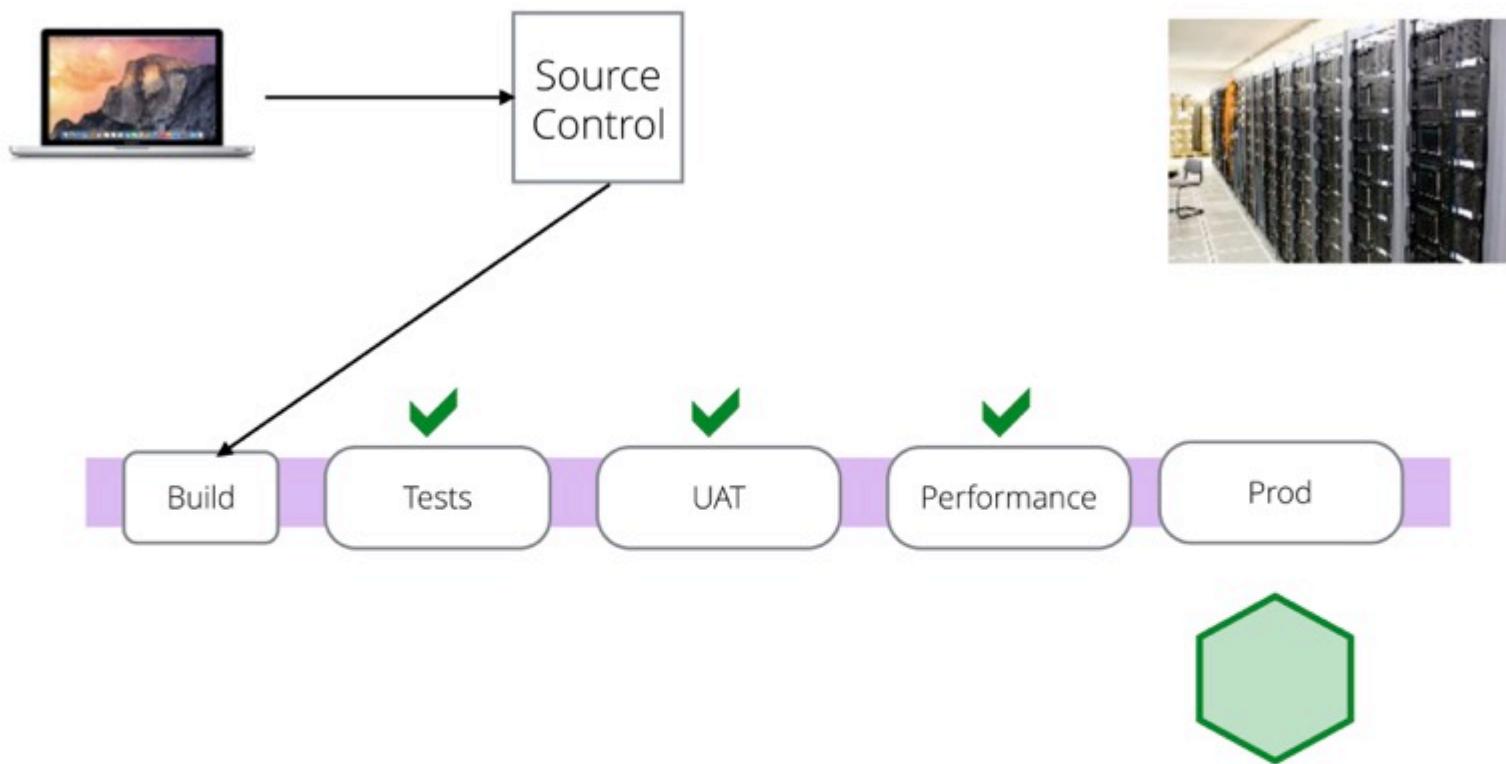




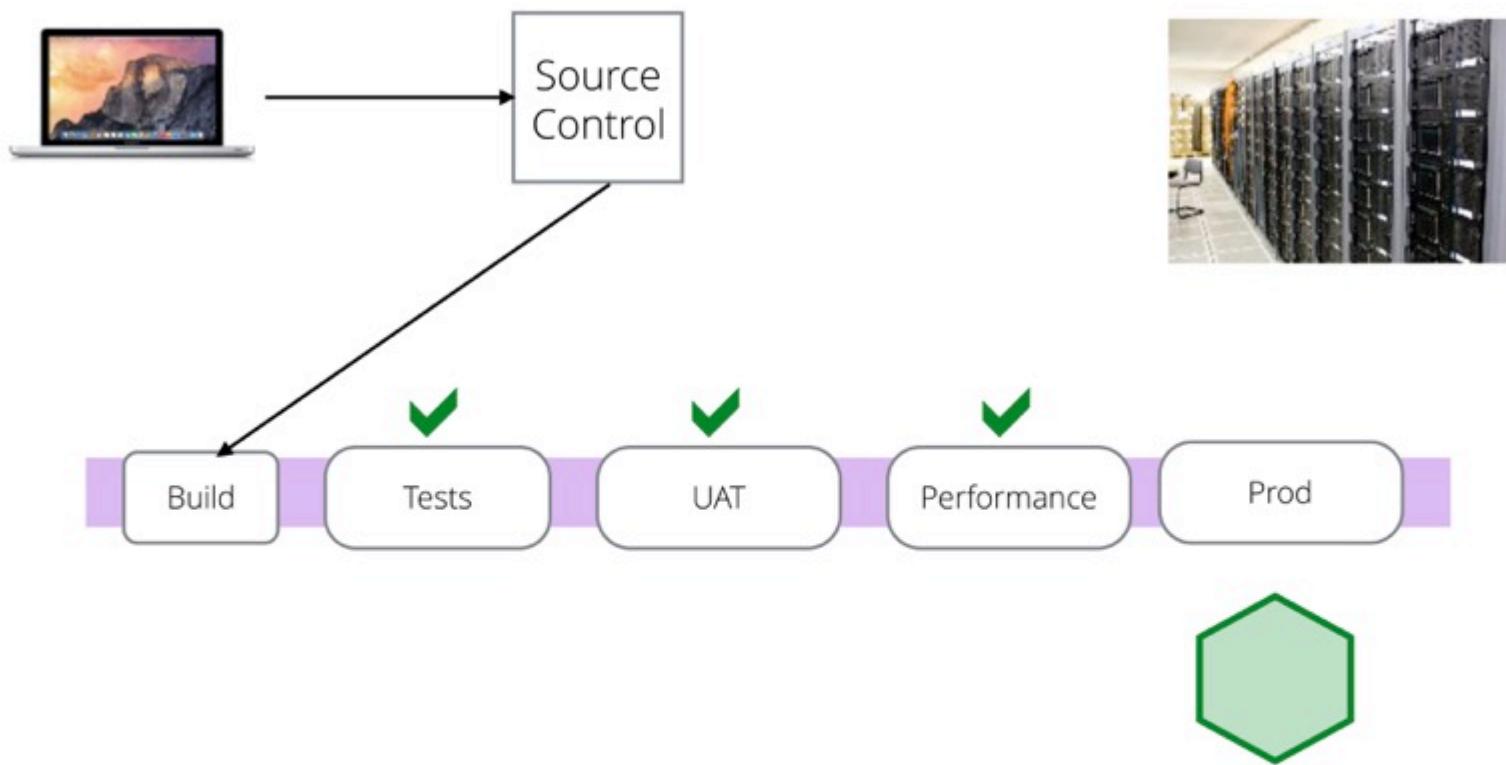








One Artifact For All Environments



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

The diagram illustrates a deployment command with annotations. Two dotted arrows point from the labels "Service Name" and "Version" to the "Returns" and "v456" fields respectively in the command line.

Service Name
Version
local

\$ deploy Returns v456 Production

Service Name
Version
local
latest

\$ deploy Returns v456 Production

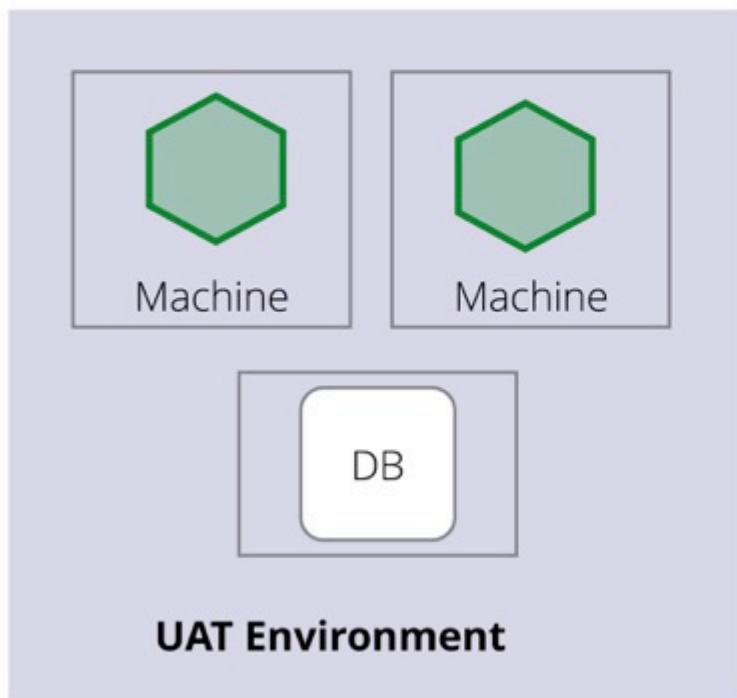
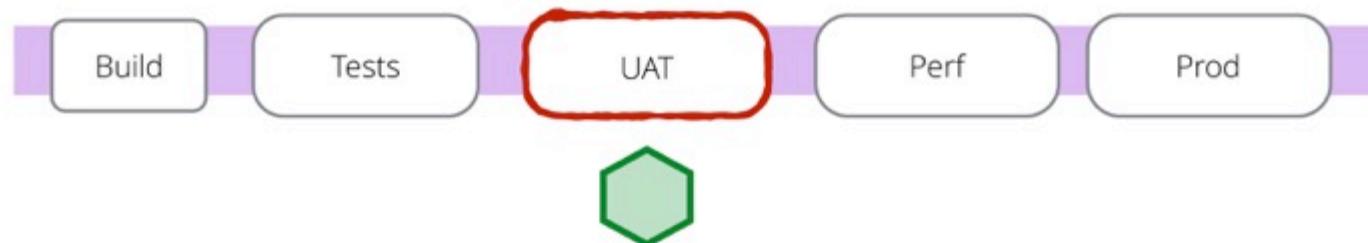
```
$ deploy
```

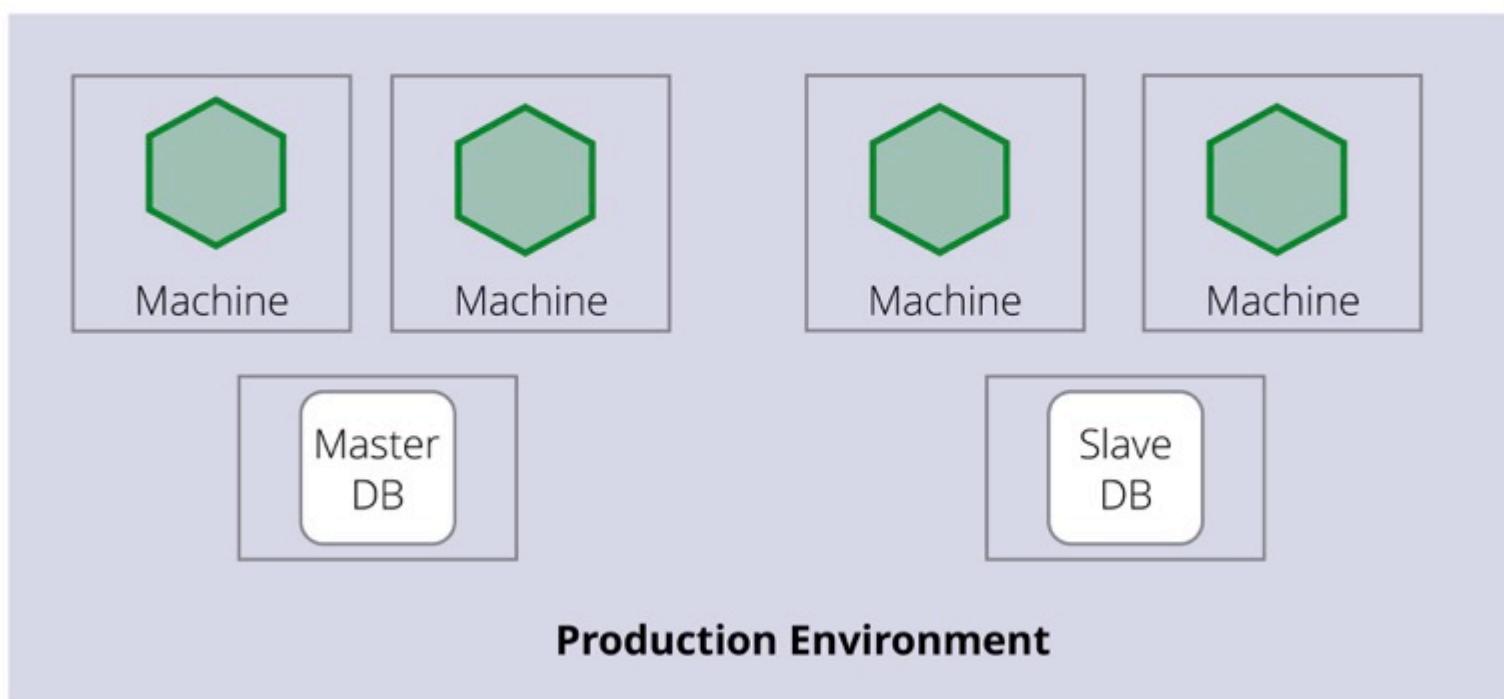
The diagram illustrates the deployment command with three parameters:

- Service Name:** A dotted arrow points from the word "Service" to the placeholder "Name".
- Version:** A dotted arrow points from the word "Version" to the placeholder "local".
- Environment:** A dotted arrow points from the word "Environment" to the placeholder "latest".

The command itself is displayed in a monospaced font, with the parameters highlighted by orange rectangular boxes.

```
Returns v456 Production
```





Same Artifact

Same Artifact

Different Topology

```
# create virtual machine
resource "azurerm_virtual_machine" "helloterraformvm" {
    name = "terraformvm"
    location = "West US"
    resource_group_name = "${azurerm_resource_group.helloterraform.name}"
    network_interface_ids = ["${azurerm_network_interface.helloterraformnic.id}"]
    vm_size = "Standard_A0"

    storage_image_reference {
        publisher = "Canonical"
        offer = "UbuntuServer"
        sku = "14.04.2-LTS"
        version = "latest"
    }

    storage_os_disk {
        name = "myosdisk"
        vhd_uri = "${azurerm_storage_account.helloterraformstorage.primary_blob_endpoints[0].url}/myosdisk.vhd"
        caching = "ReadWrite"
        create_option = "FromImage"
    }

    os_profile {
        computer_name =
        admin_username =
        admin_password =
    }

    os_profile_linux_config {
        disable_password_authentication =
    }

    tags {
        environment =
    }
}
```



```
# create virtual machine
resource "azurerm_virtual_machine" "helloterraformvm" {
    name = "terraformvm"
    location = "West US"
    resource_group_name = "${azurerm_resource_group.helloterraform.name}"
    network_interface_ids = ["${azurerm_network_interface.helloterraformnic.id}"]
    vm_size = "Standard_A0"

    storage_image_reference {
        publisher = "Canonical"
        offer = "UbuntuServer"
        sku = "14.04.2-LTS"
        version = "latest"
    }

    storage_os_disk {
        name = "myosdisk"
        vhd_uri = "${azurerm_storage_account.helloterraformstorage.primary_blob_endpoint}${storagecontainer.name}/myosdisk.vhd"
        caching = "ReadWrite"
        create_option = "FromImage"
    }

    os_profile {
        computer_name =
        admin_username =
        admin_password =
    }

    os_profile_linux_config {
        disable_password_authentication =
    }

    tags {
        environment =
    }
}
```

Version controlled



```
# create virtual machine
resource "azurerm_virtual_machine" "helloterraformvm" {
    name = "terraformvm"
    location = "West US"
    resource_group_name = "${azurerm_resource_group.helloterraform.name}"
    network_interface_ids = ["${azurerm_network_interface.helloterraformnic.id}"]
    vm_size = "Standard_A0"

    storage_image_reference {
        publisher = "Canonical"
        offer = "UbuntuServer"
        sku = "14.04.2-LTS"
        version = "latest"
    }

    storage_os_disk {
        name = "myosdisk"
        vhd_uri = "${azurerm_storage_account.helloterraformstorage.primary_blob_endpoint}${storagecontainer.name}/myosdisk.vhd"
        caching = "ReadWrite"
        create_option = "FromImage"
    }

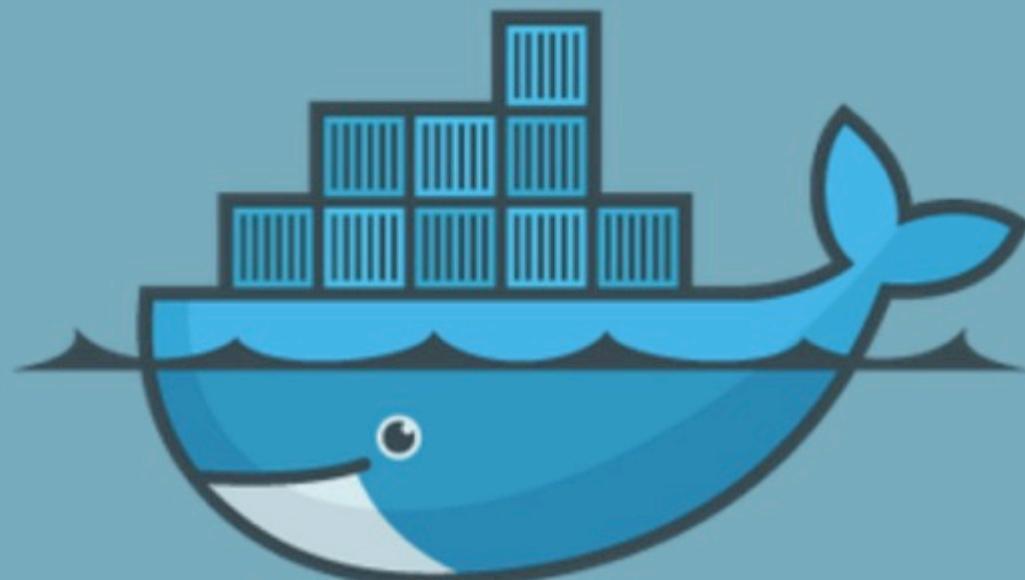
    os_profile {
        computer_name =
        admin_username =
        admin_password =
    }

    os_profile_linux_config {
        disable_password_authentication =
    }

    tags {
        environment =
    }
}
```

Version controlled
Parameterisable



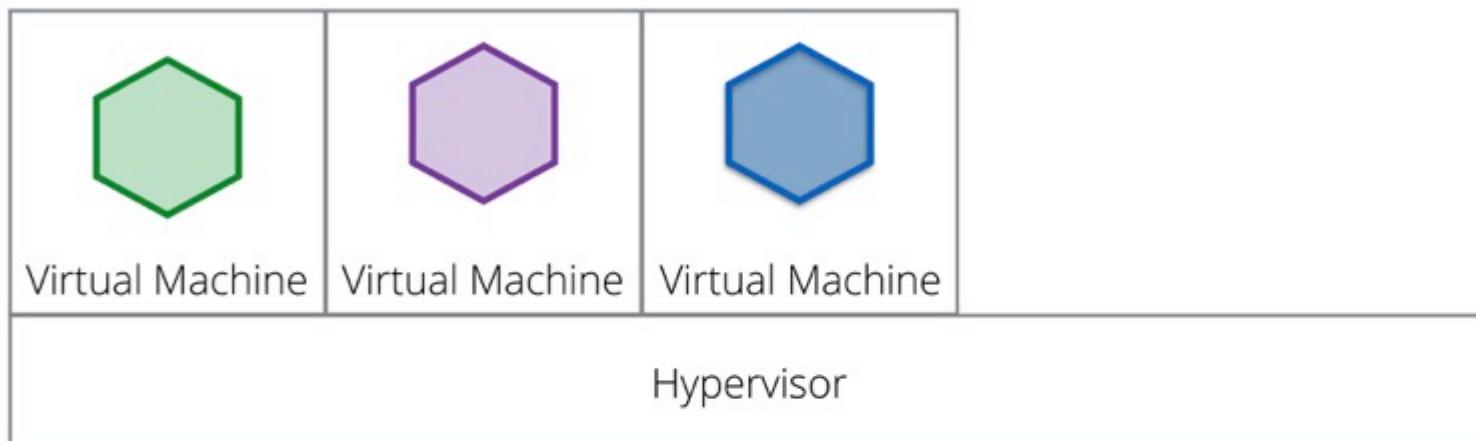


docker

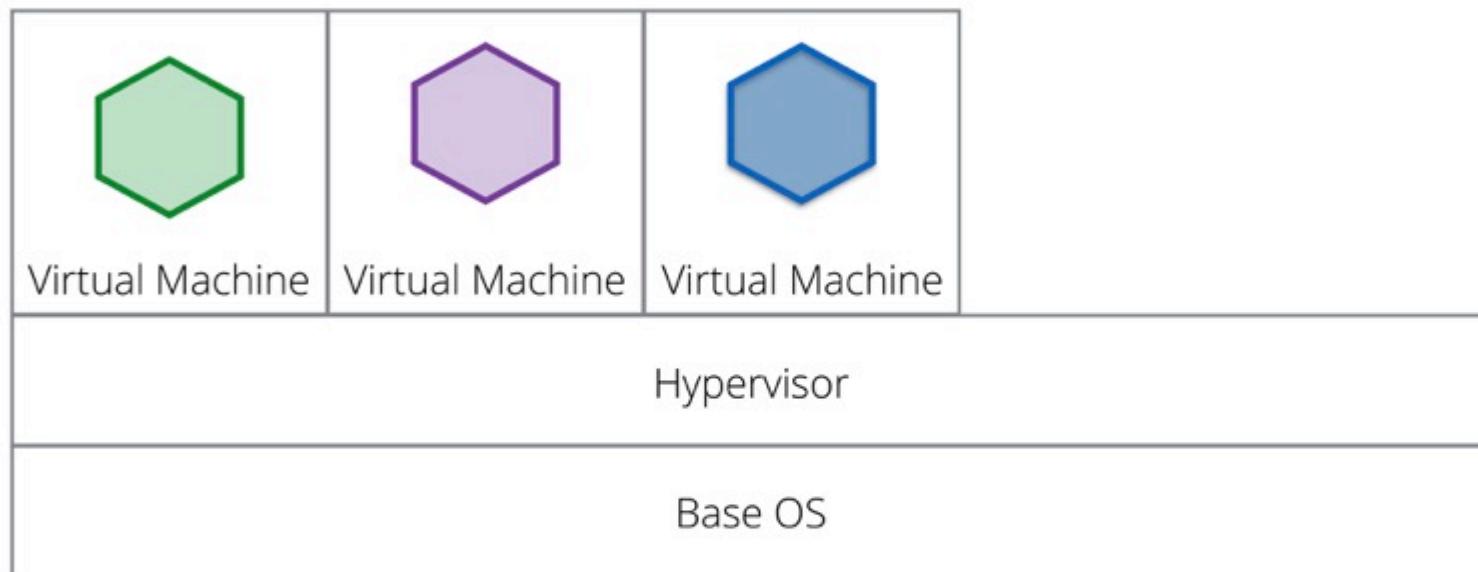
TYPE 2 VIRTUALISATION



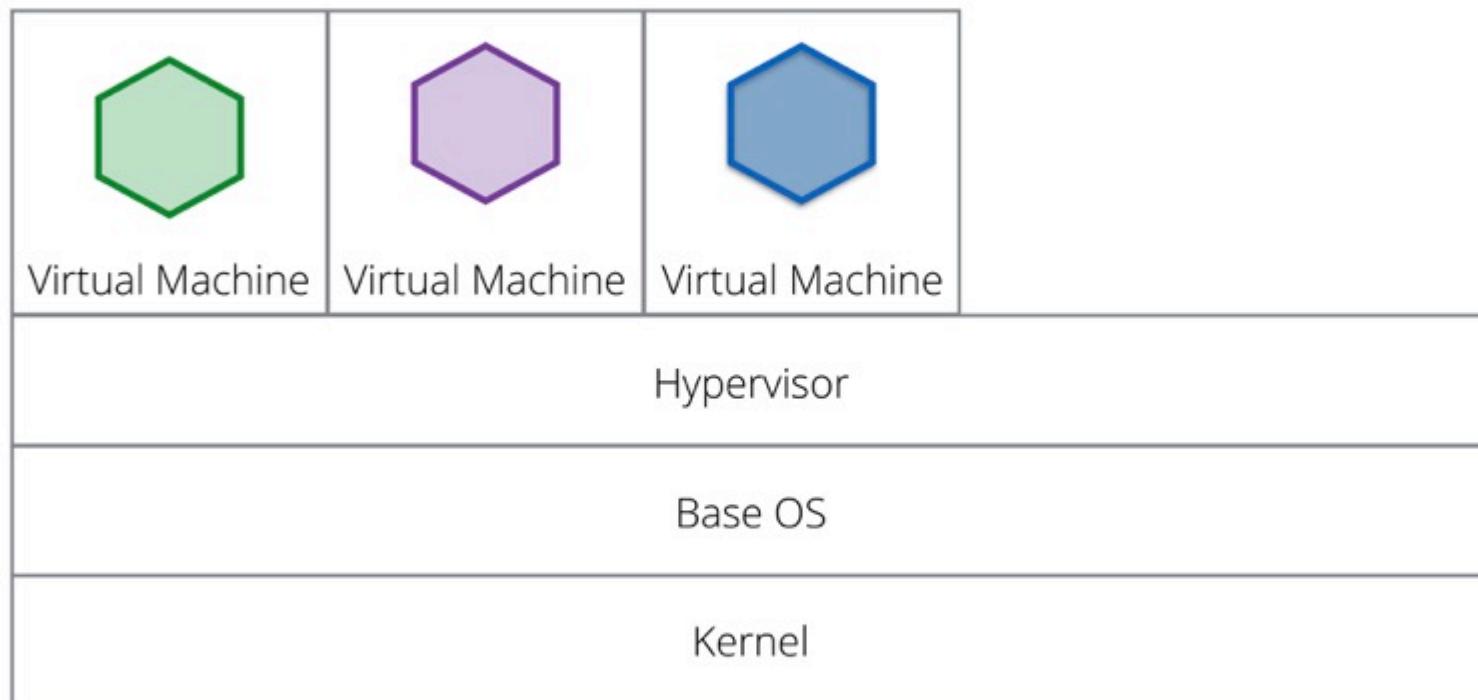
TYPE 2 VIRTUALISATION



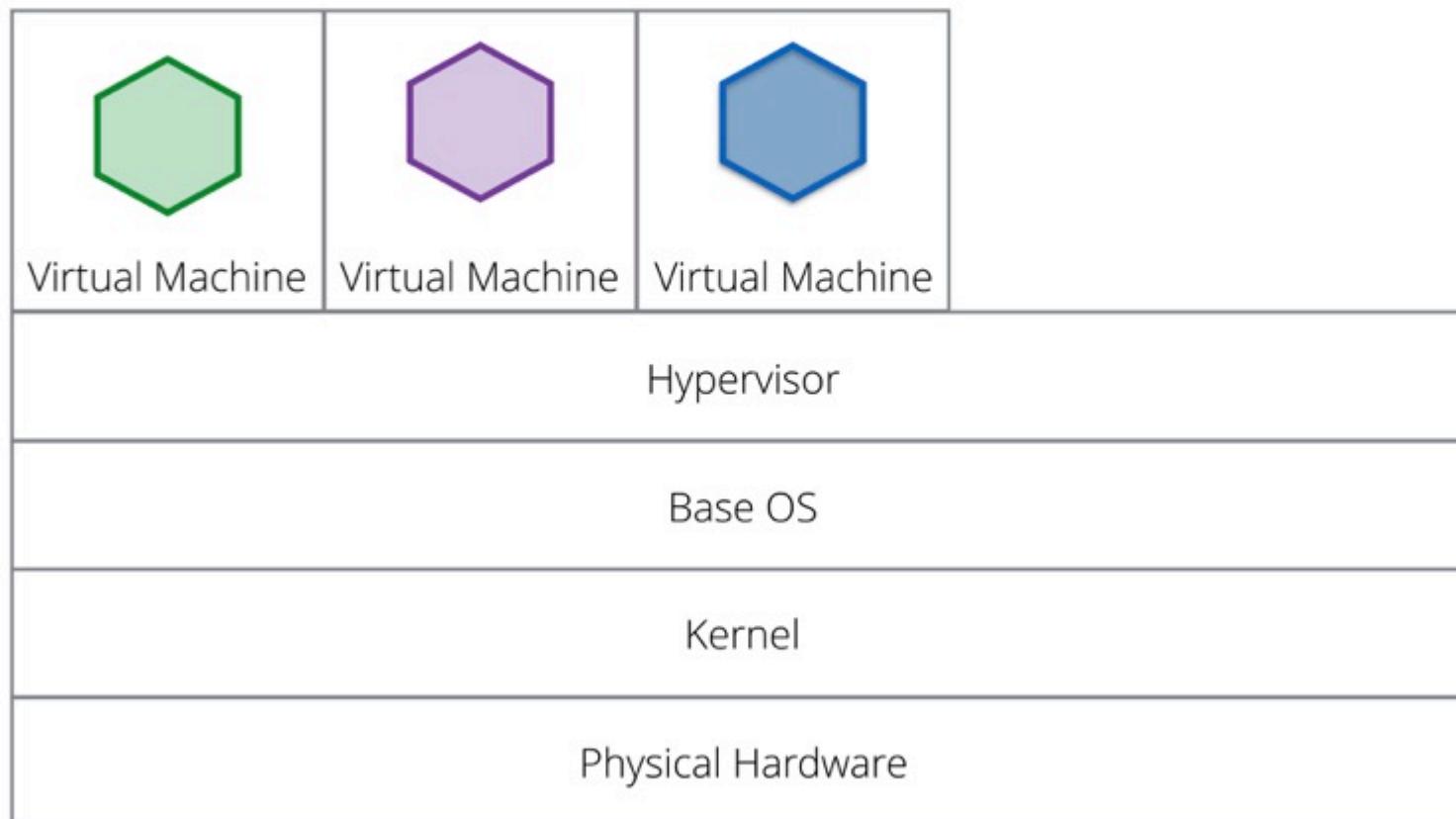
TYPE 2 VIRTUALISATION



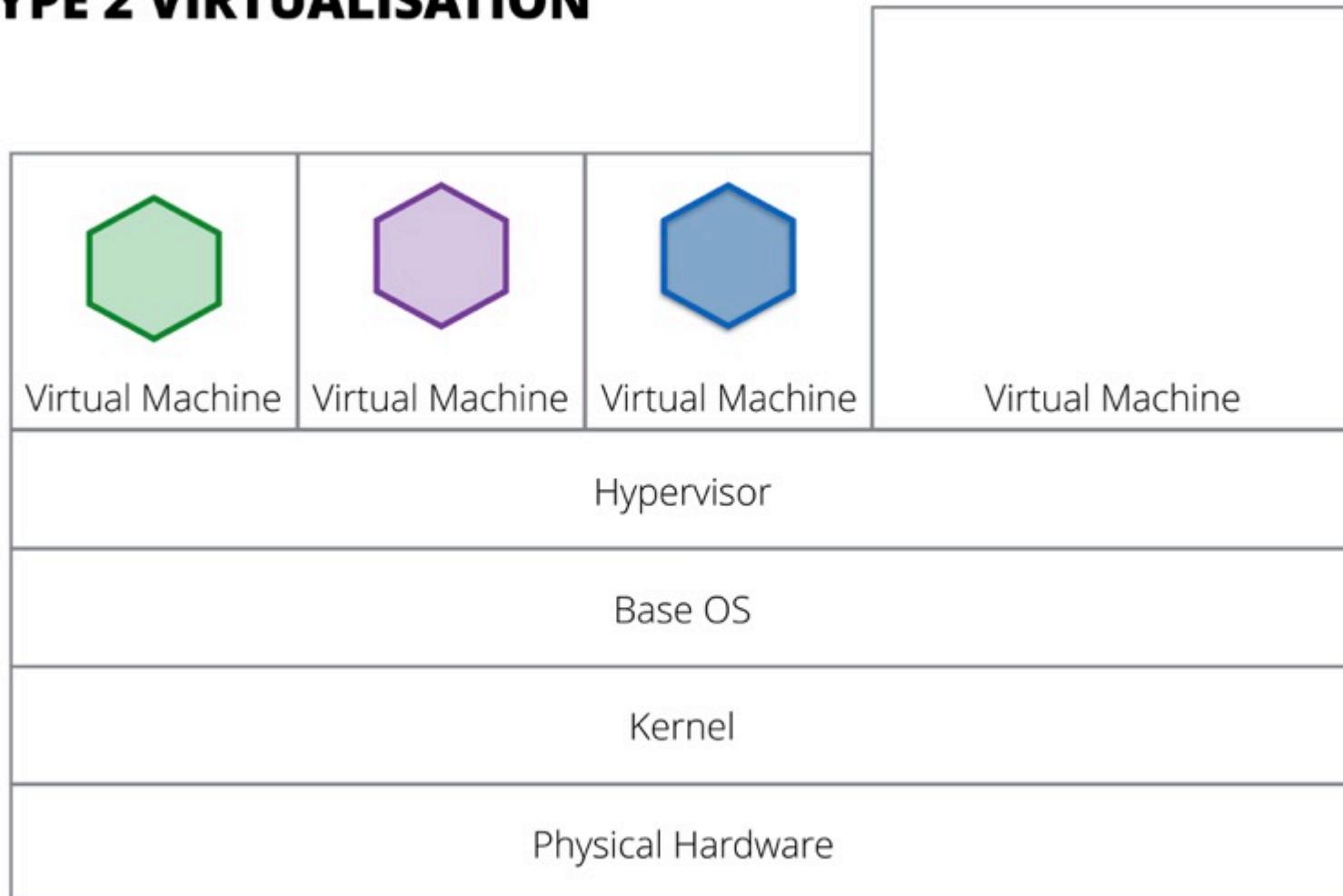
TYPE 2 VIRTUALISATION



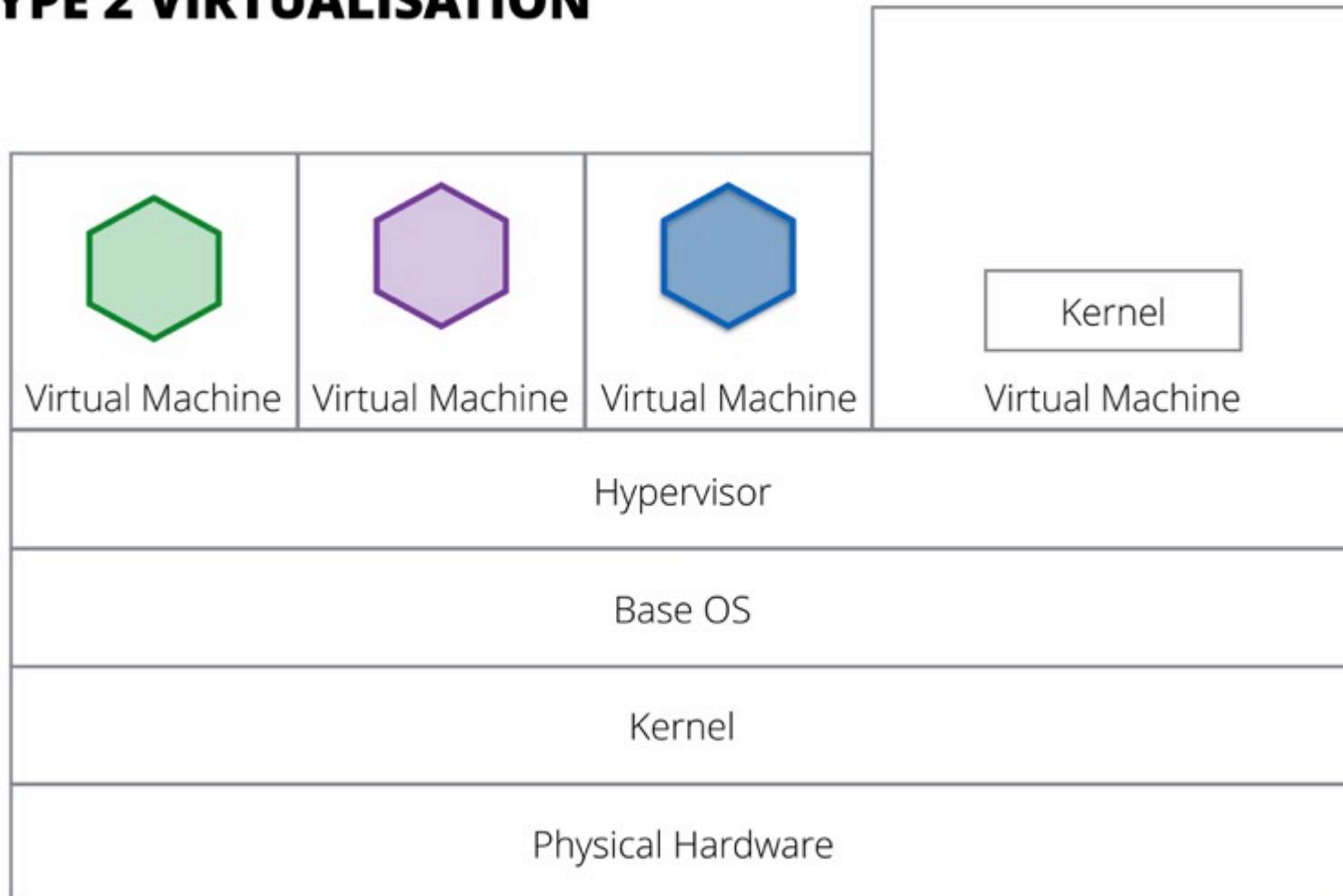
TYPE 2 VIRTUALISATION



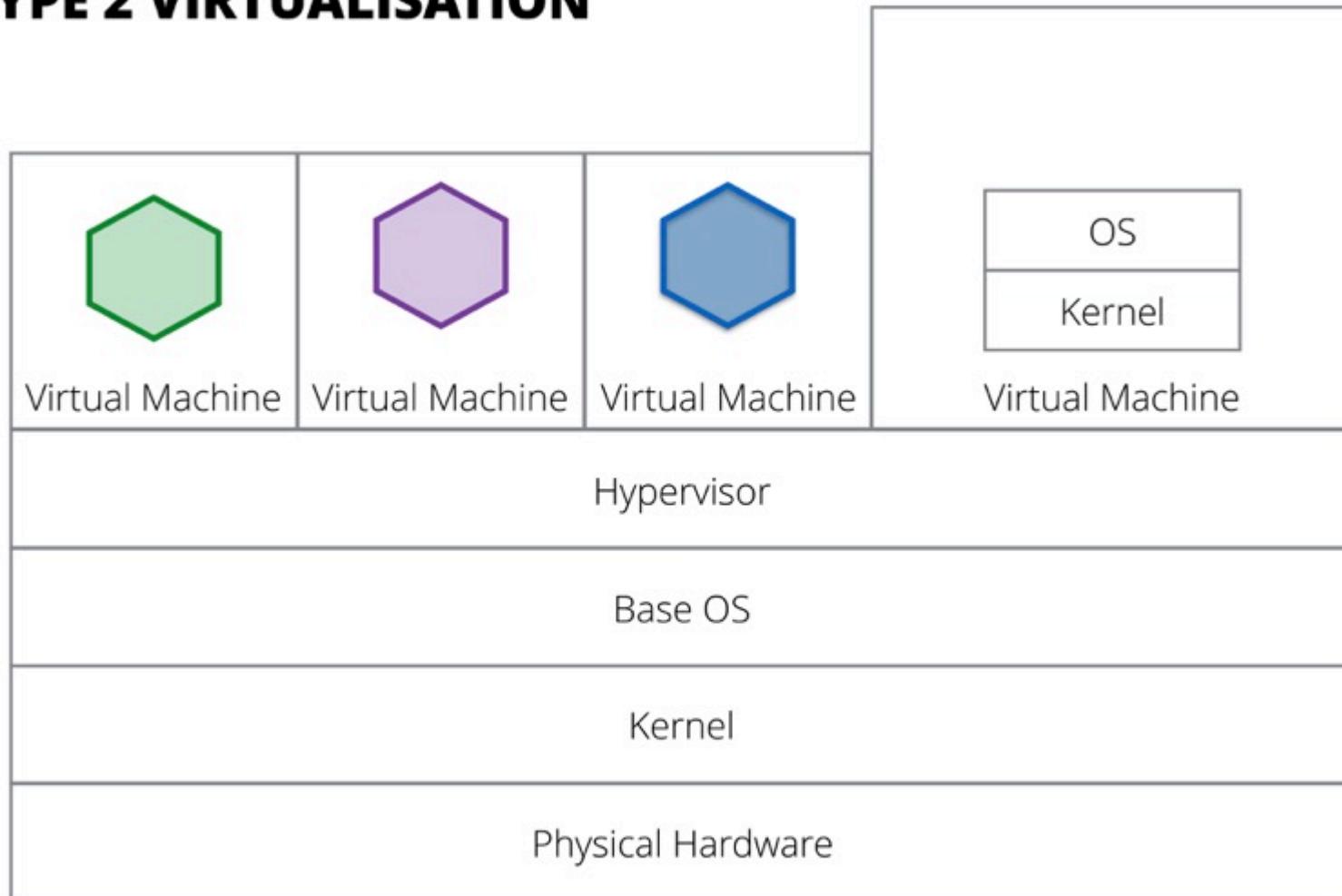
TYPE 2 VIRTUALISATION



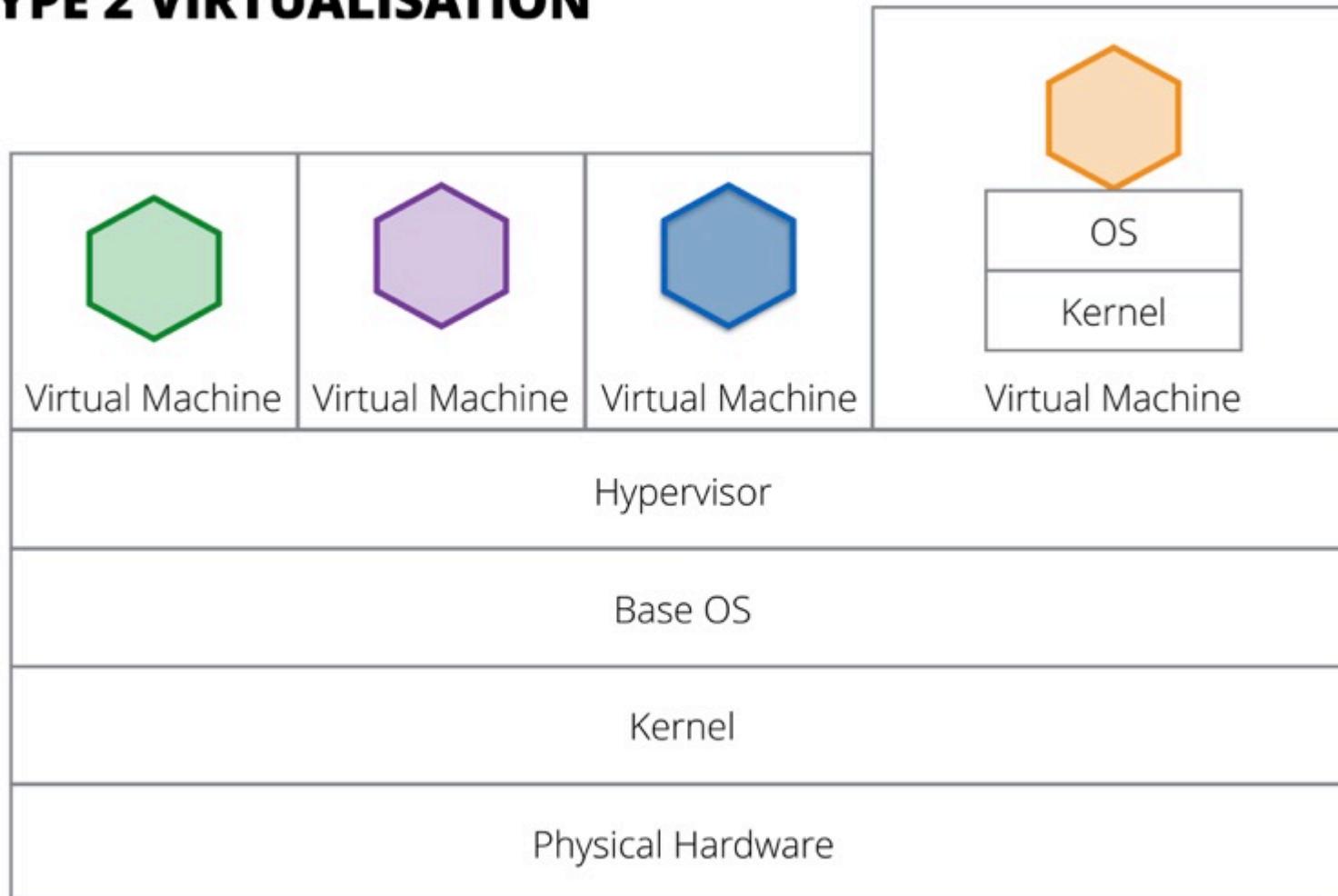
TYPE 2 VIRTUALISATION



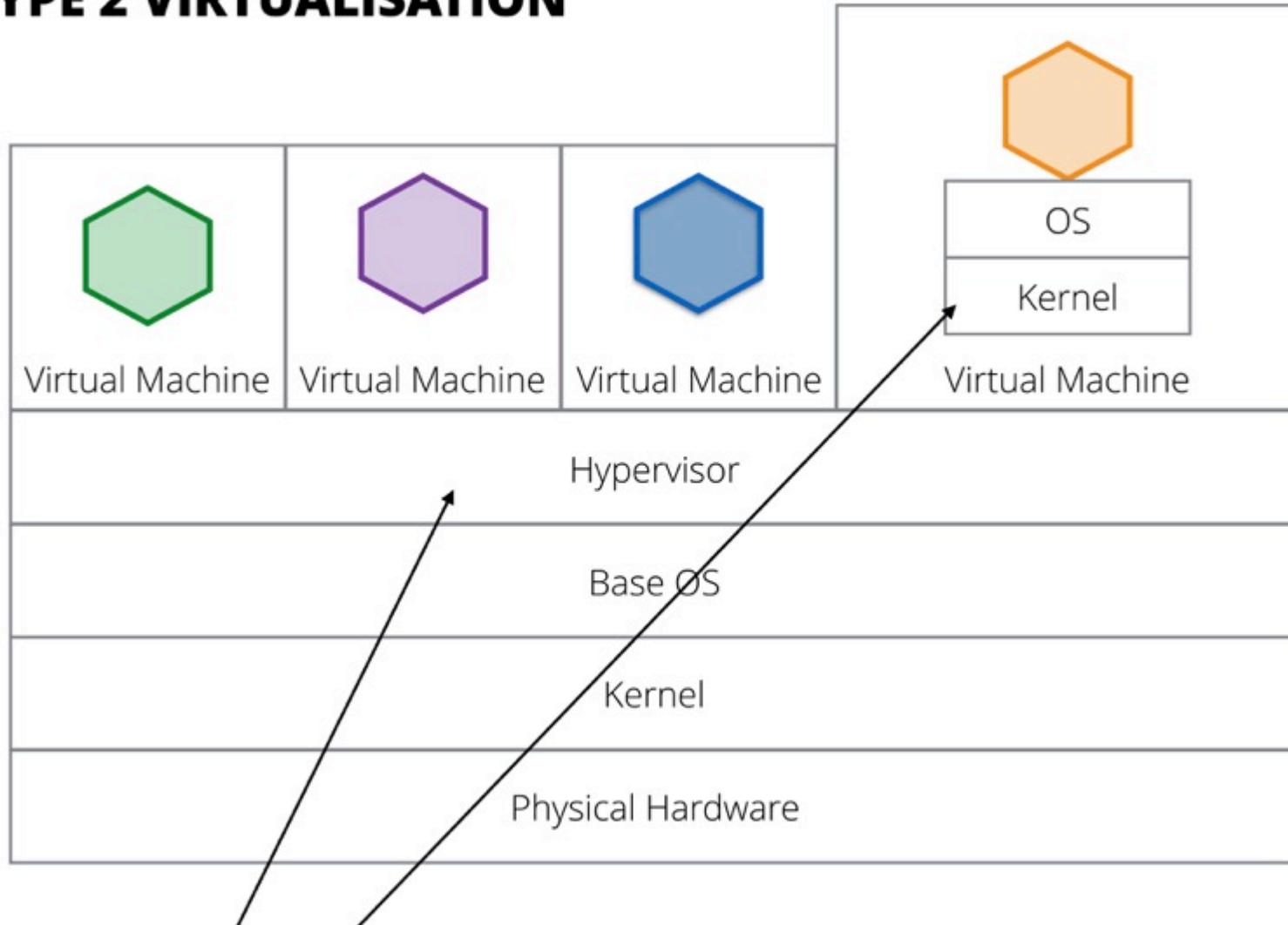
TYPE 2 VIRTUALISATION



TYPE 2 VIRTUALISATION

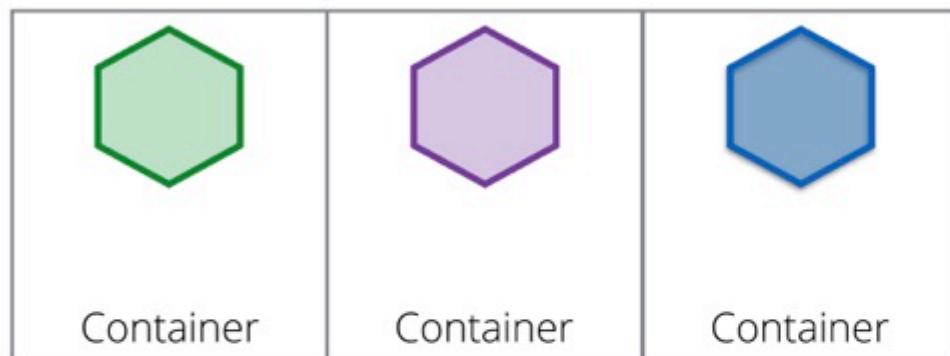


TYPE 2 VIRTUALISATION

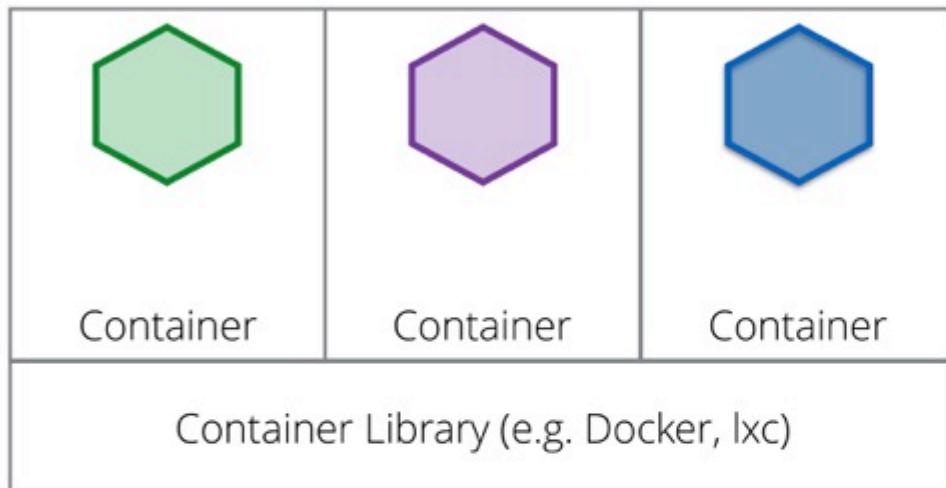


Expensive!

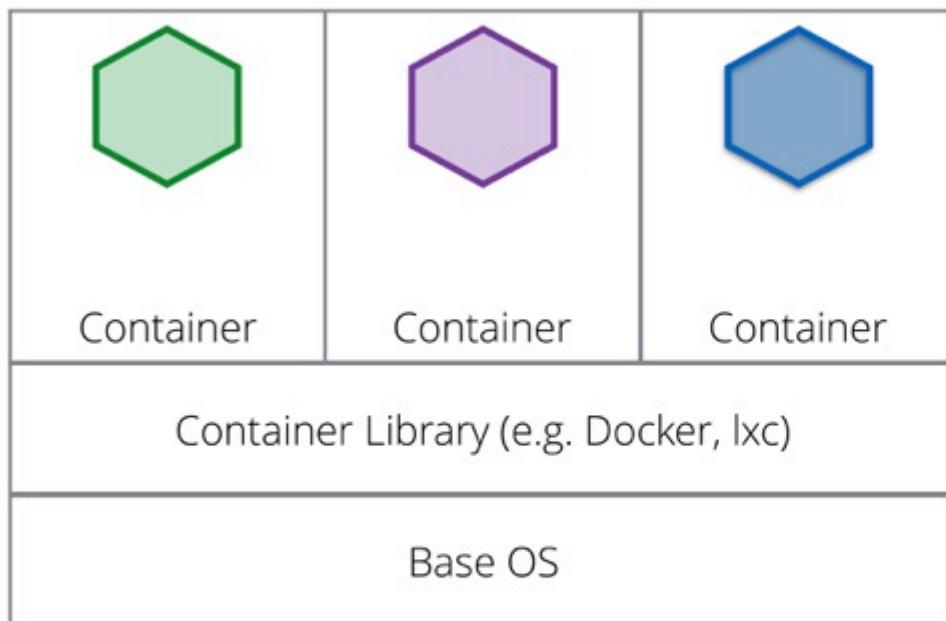
CONTAINERISATION



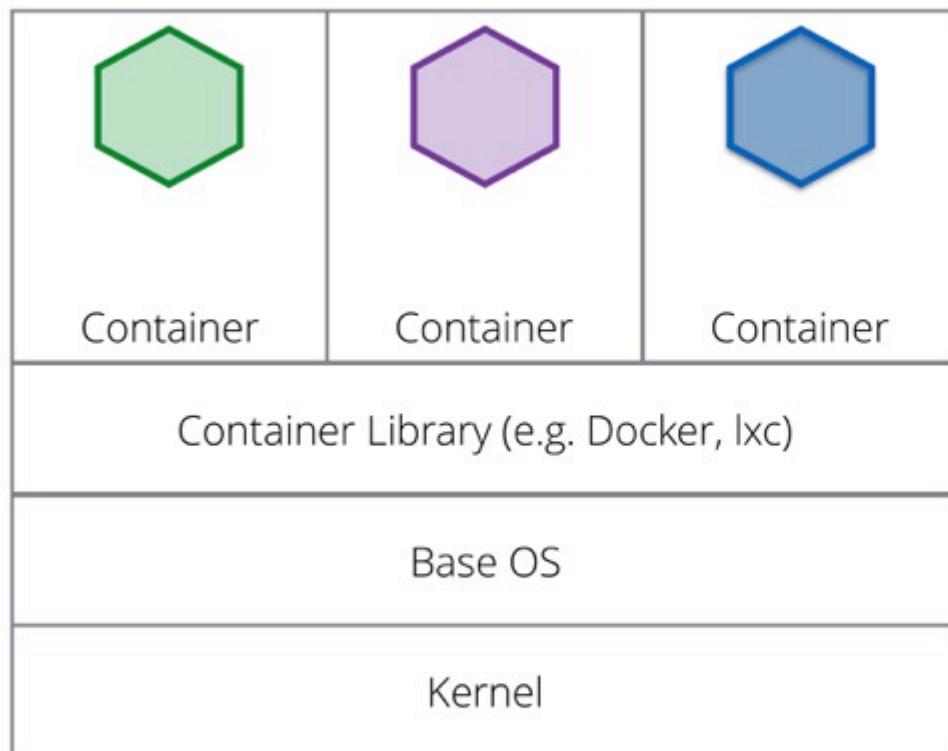
CONTAINERISATION



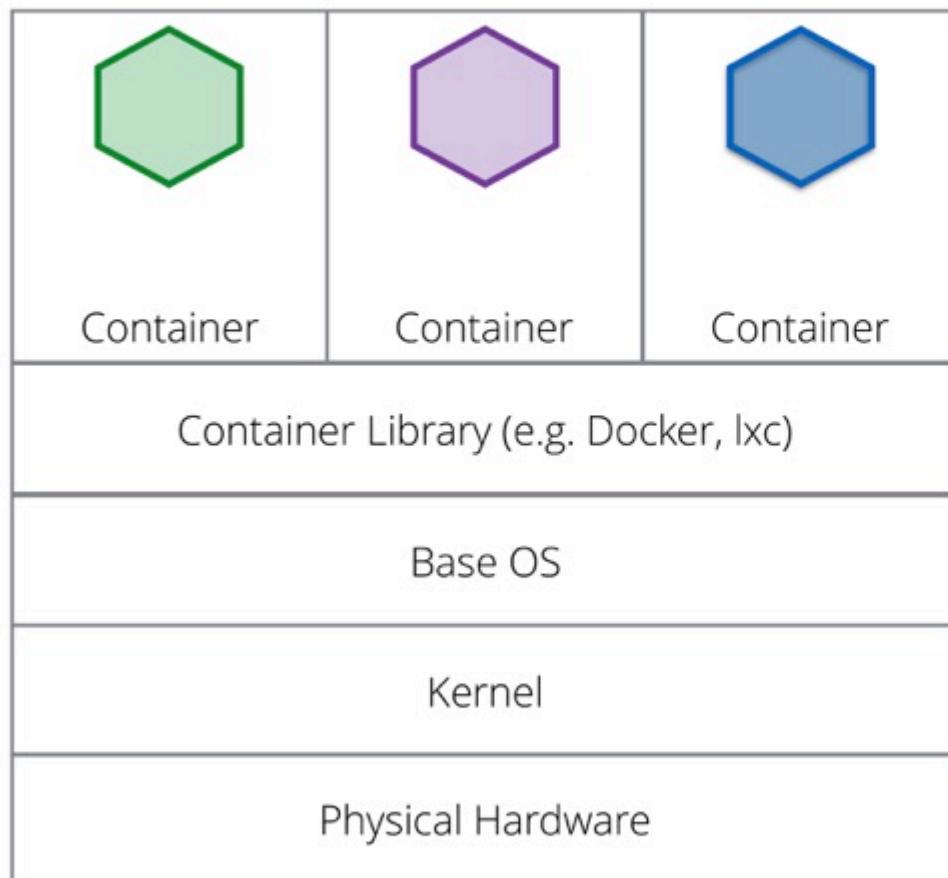
CONTAINERISATION



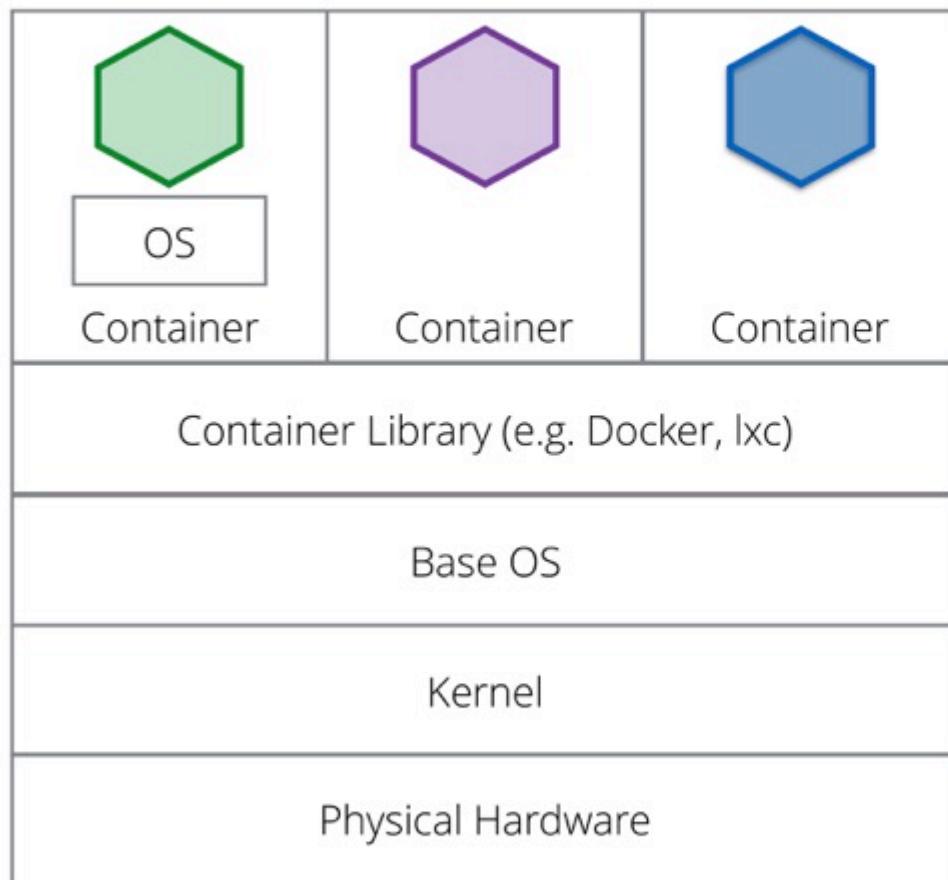
CONTAINERISATION



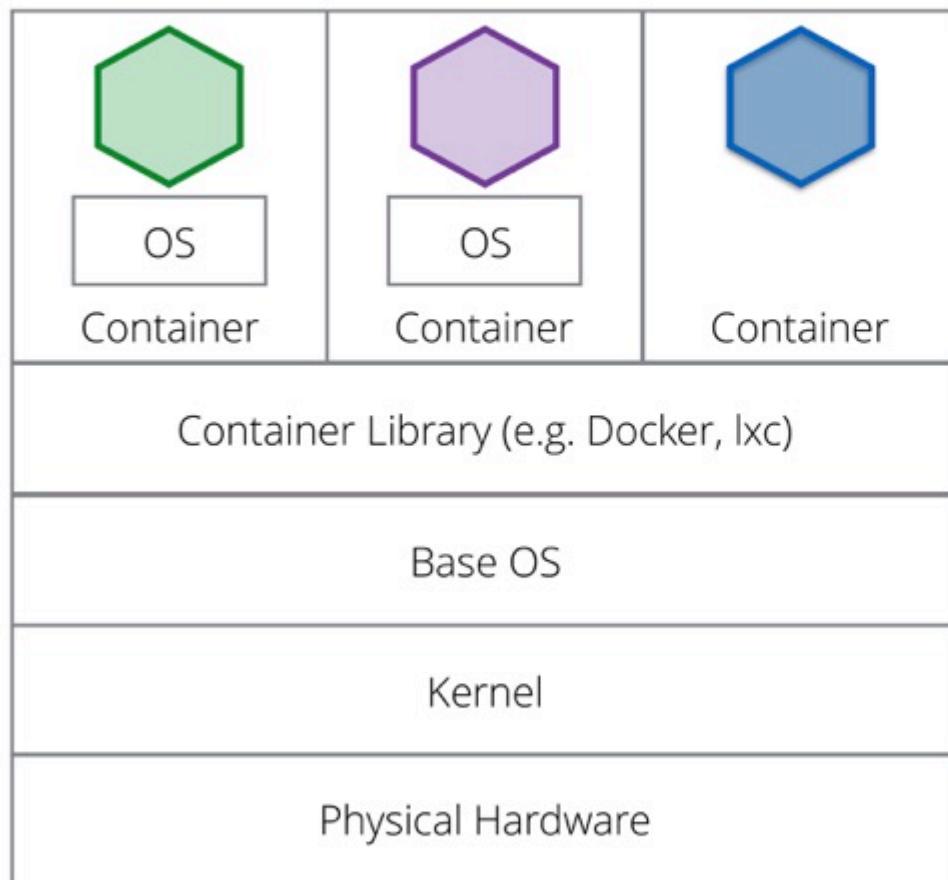
CONTAINERISATION



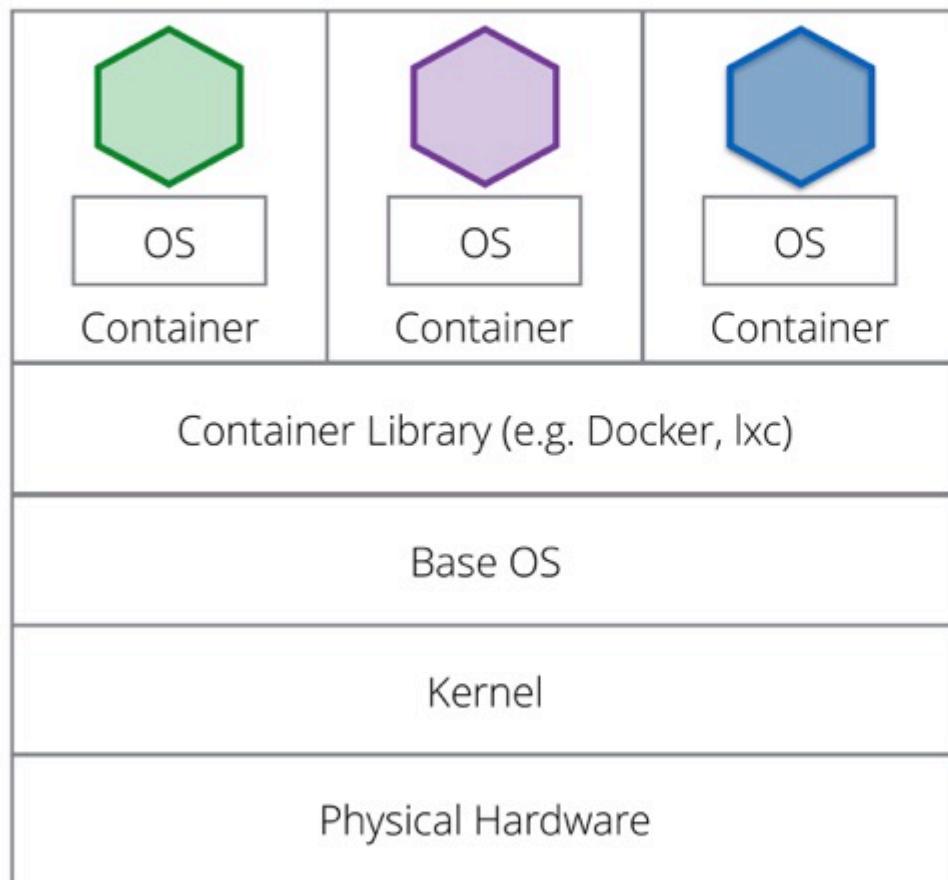
CONTAINERISATION



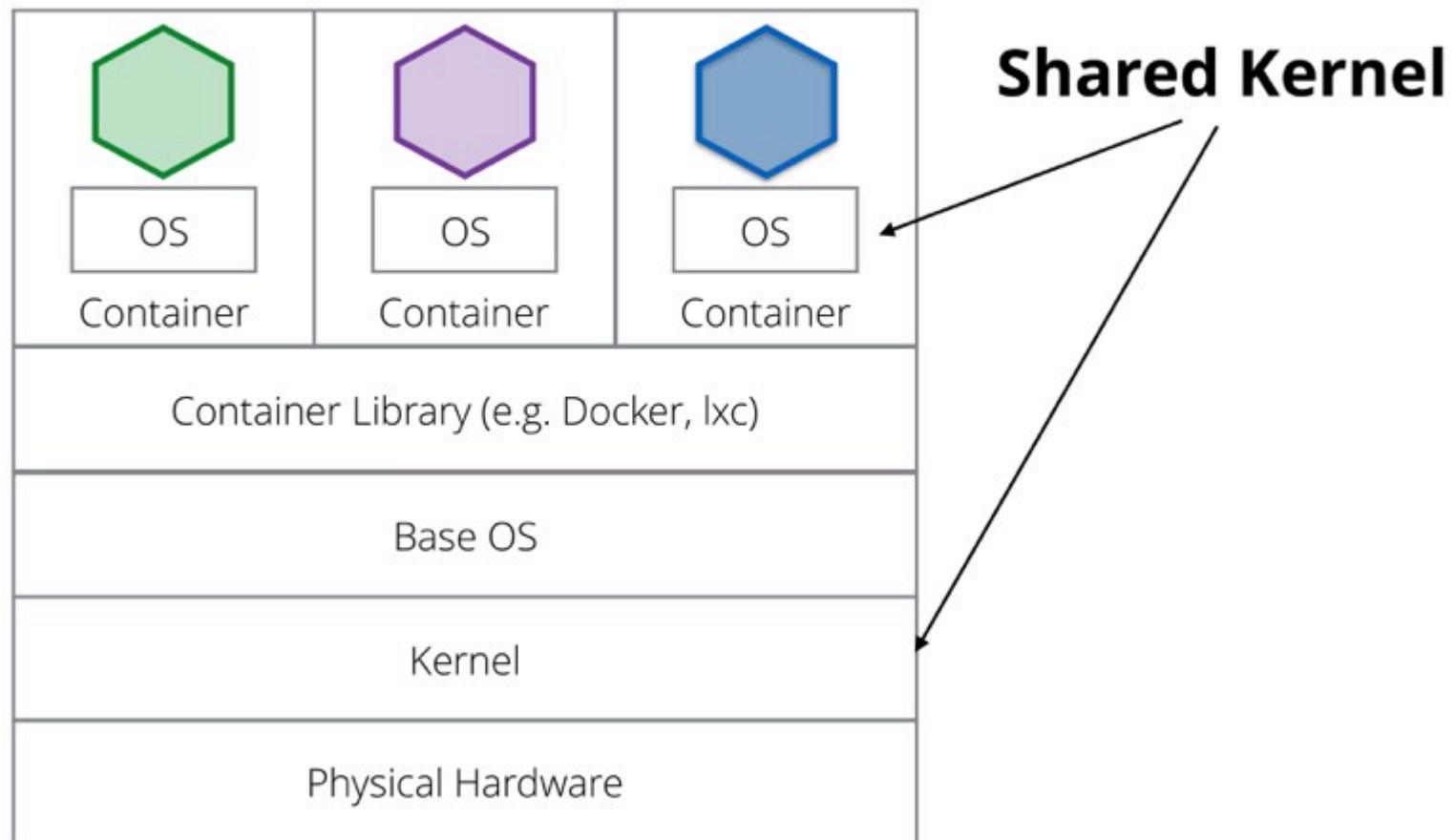
CONTAINERISATION



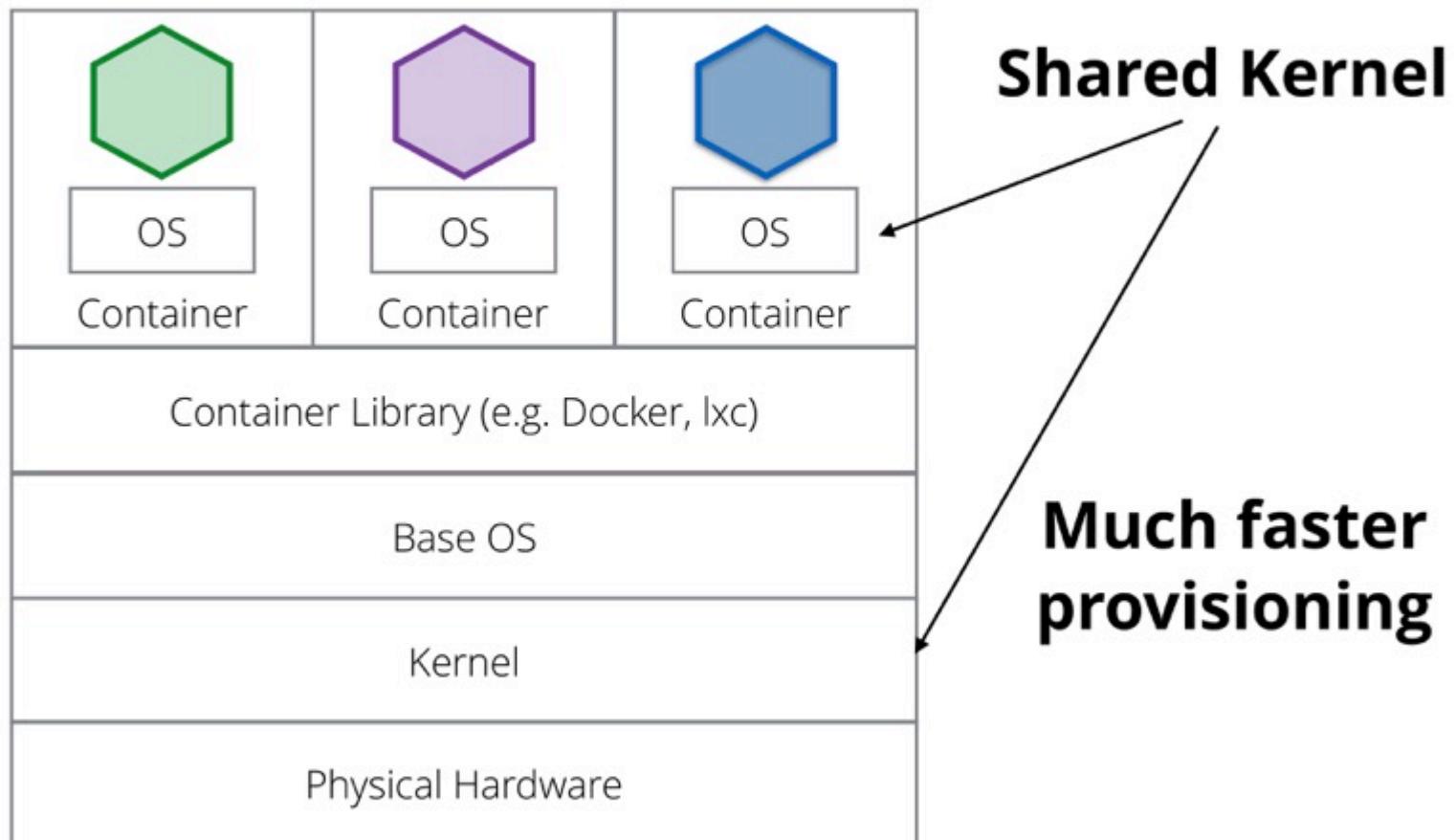
CONTAINERISATION



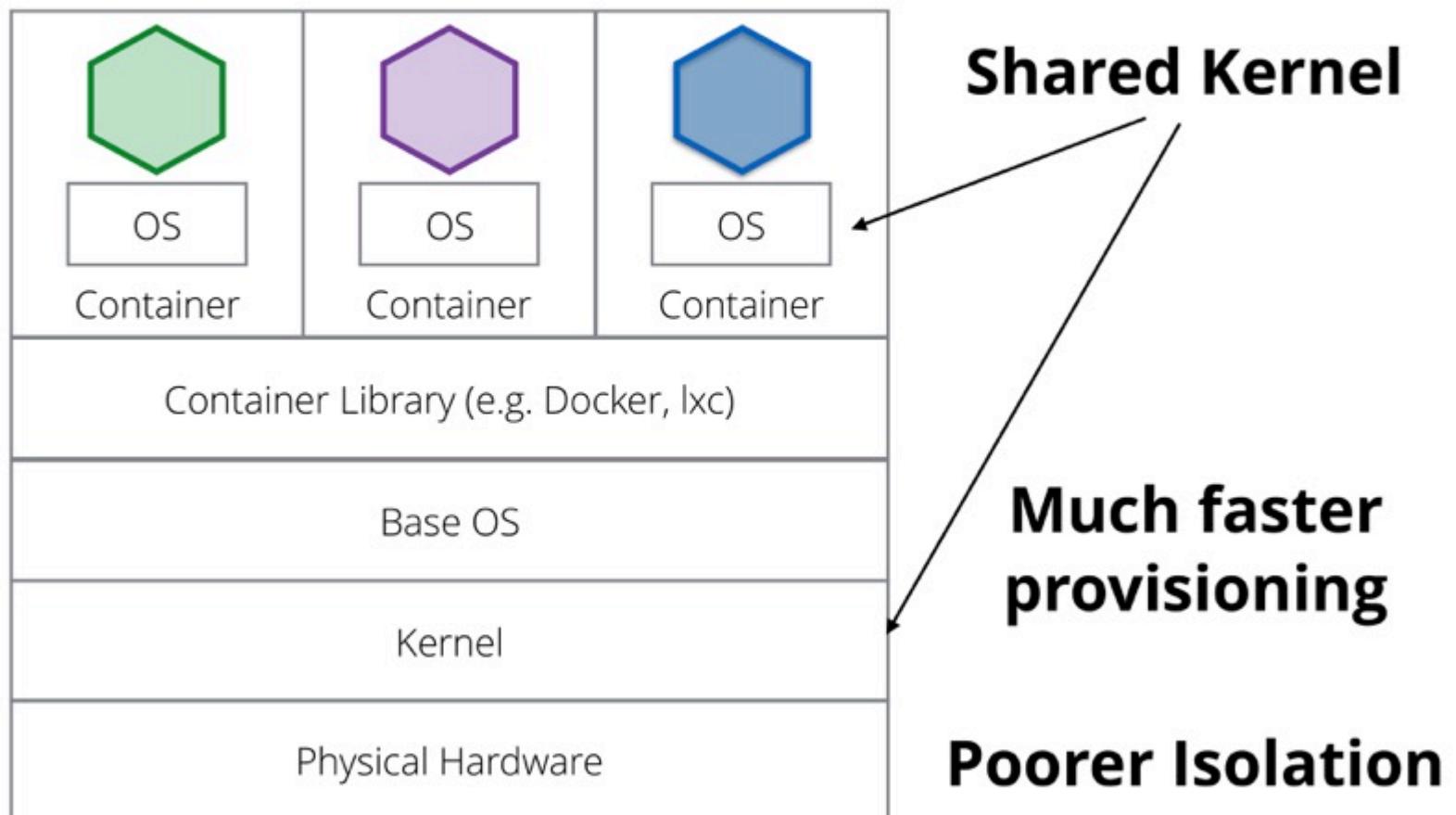
CONTAINERISATION



CONTAINERISATION



CONTAINERISATION



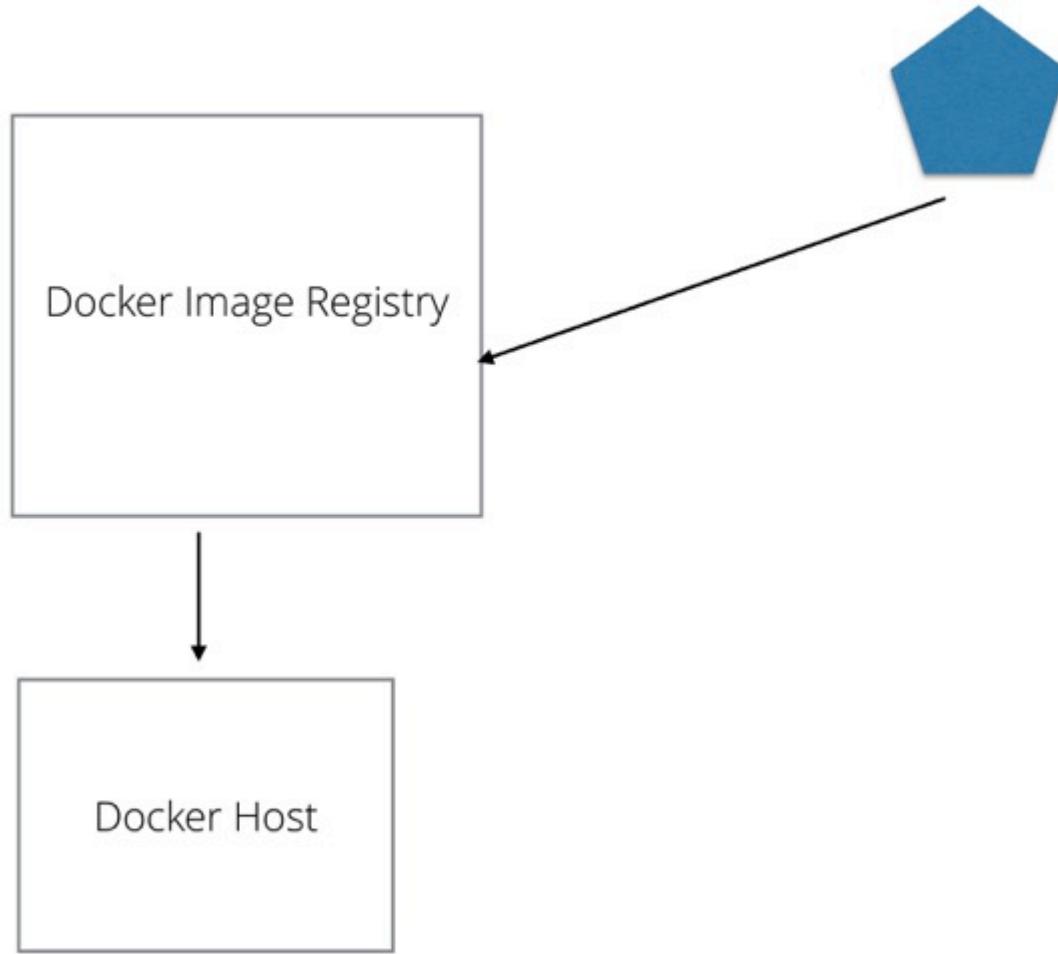
DOCKER



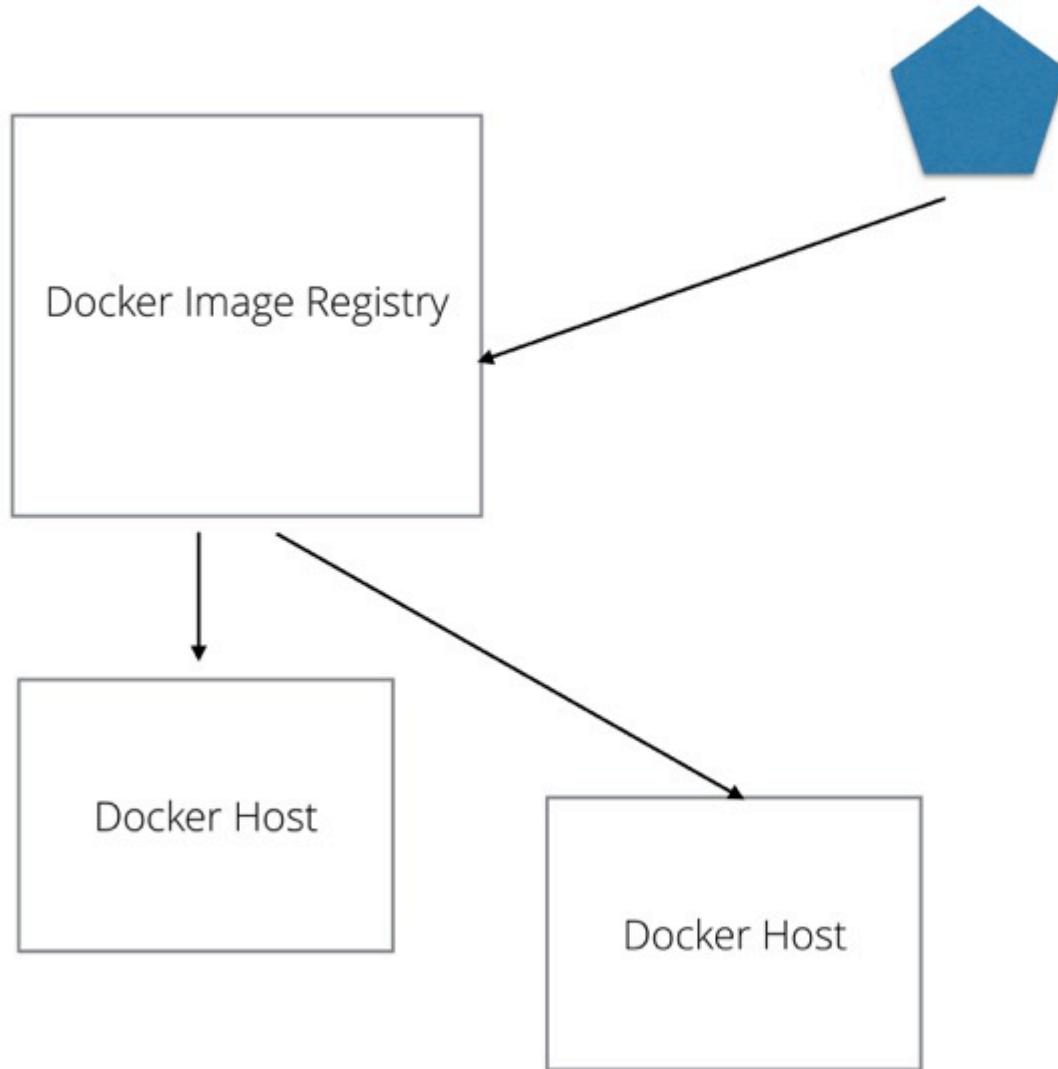
DOCKER



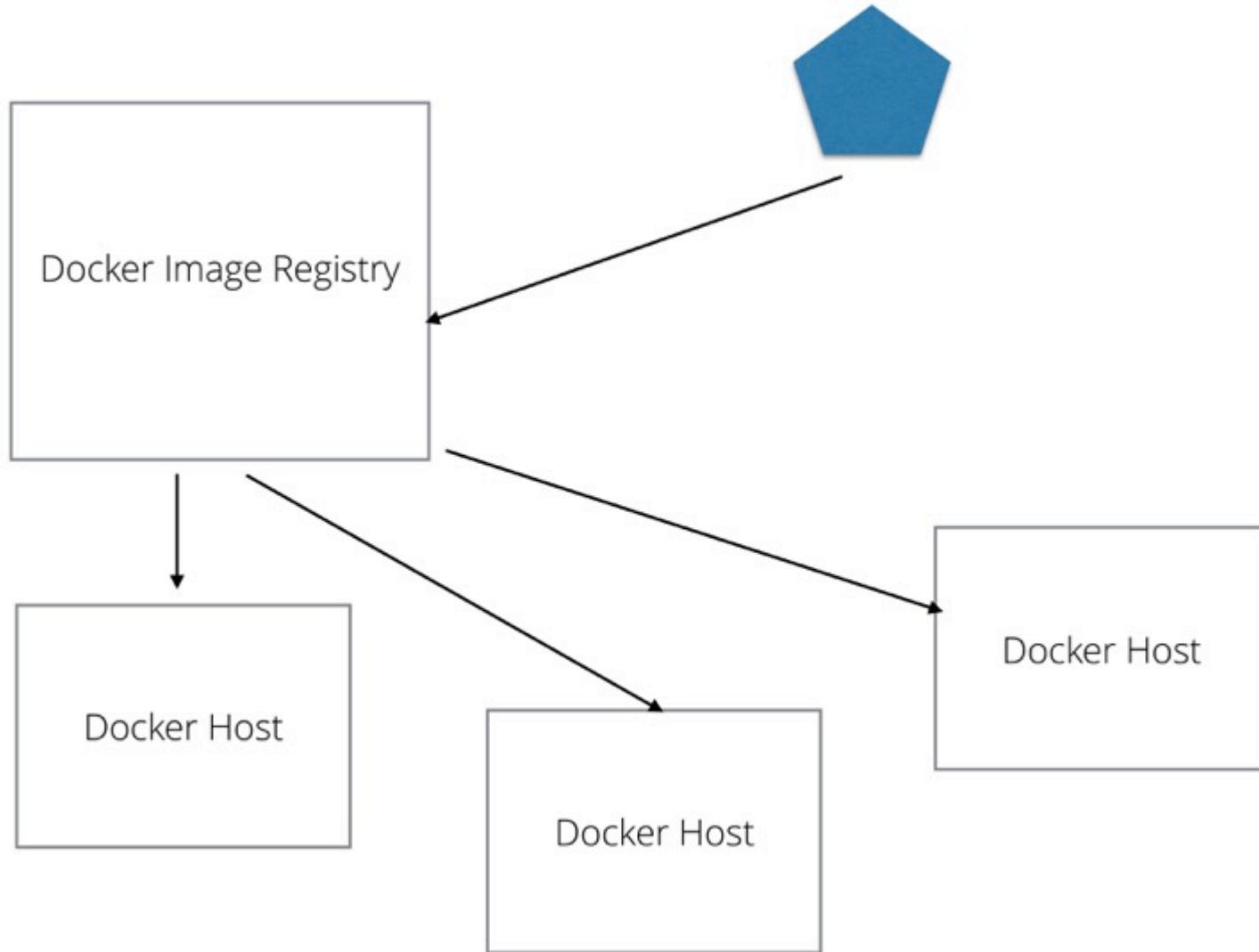
DOCKER

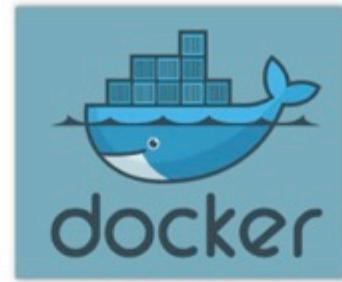


DOCKER



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



kubernetes
by Google™

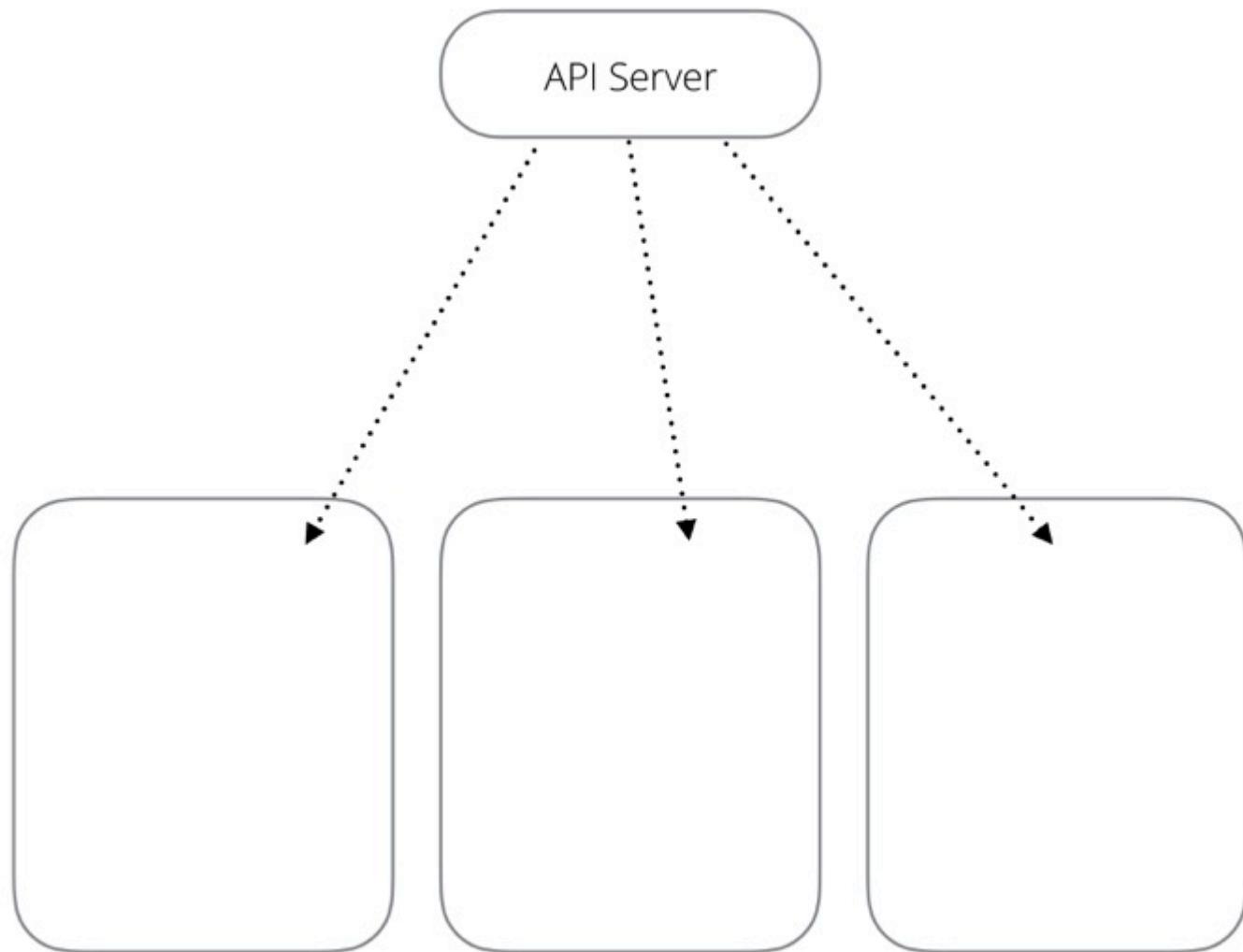
KUBERNETES ARCHITECTURE...SORT OF



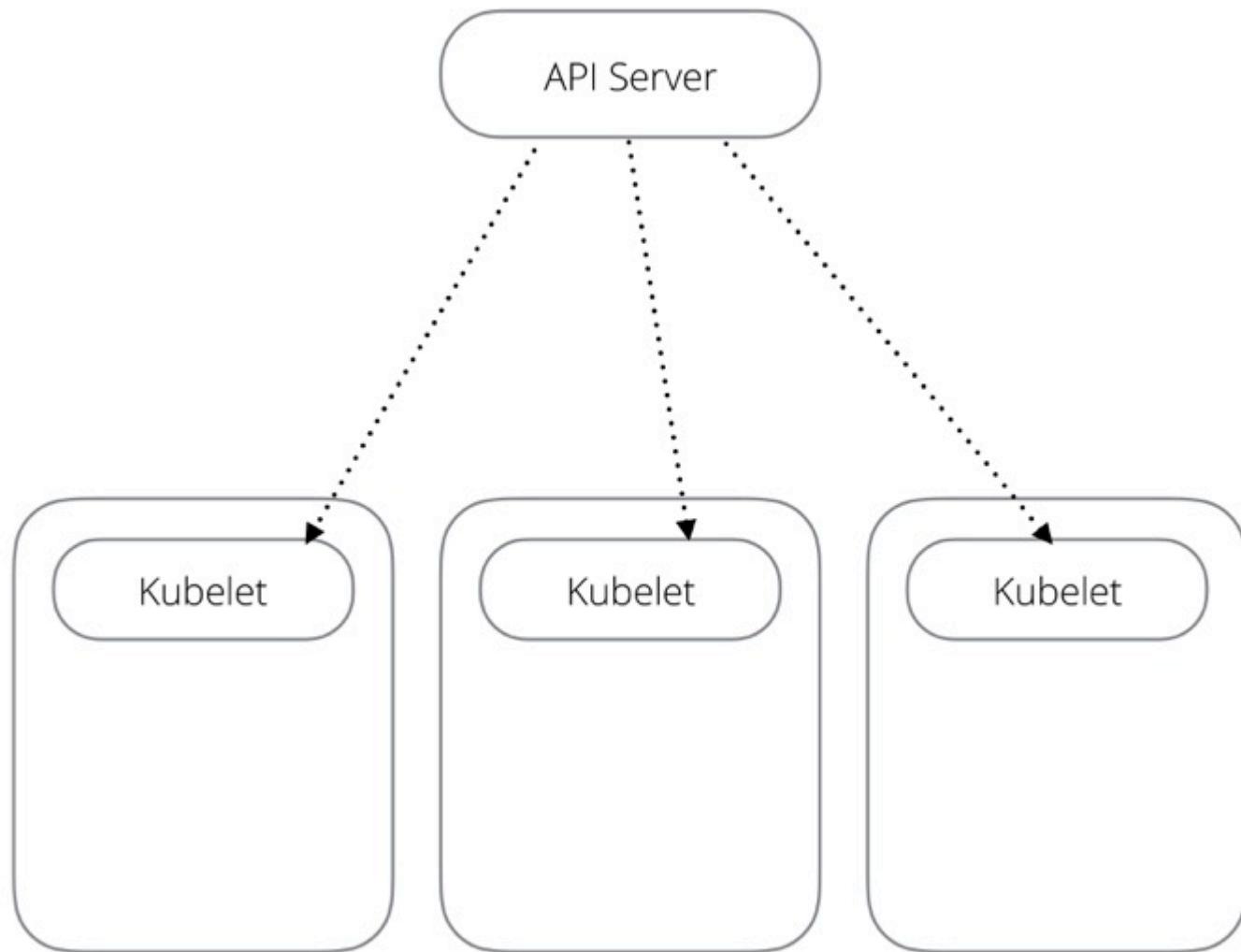
KUBERNETES ARCHITECTURE...SORT OF



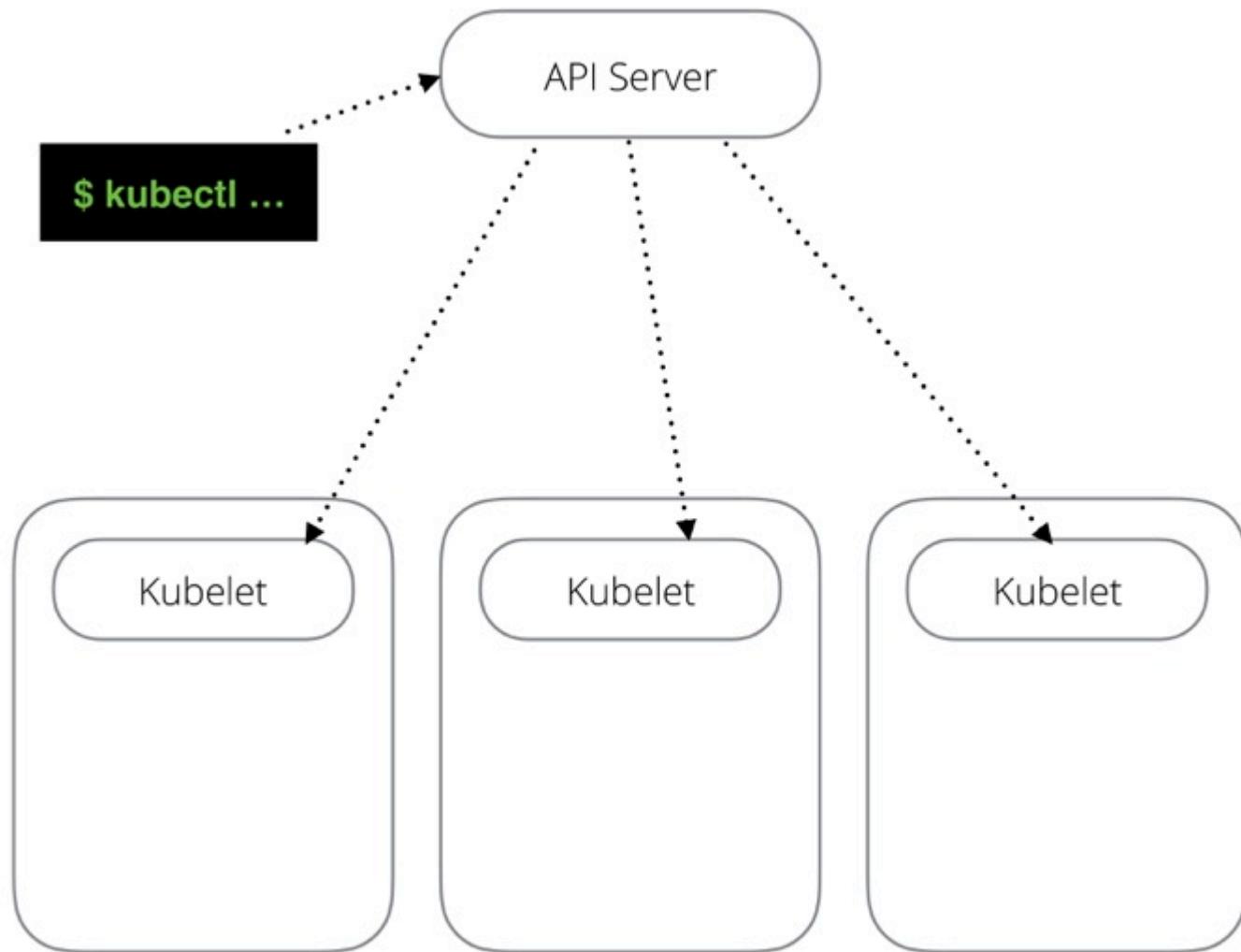
KUBERNETES ARCHITECTURE...SORT OF



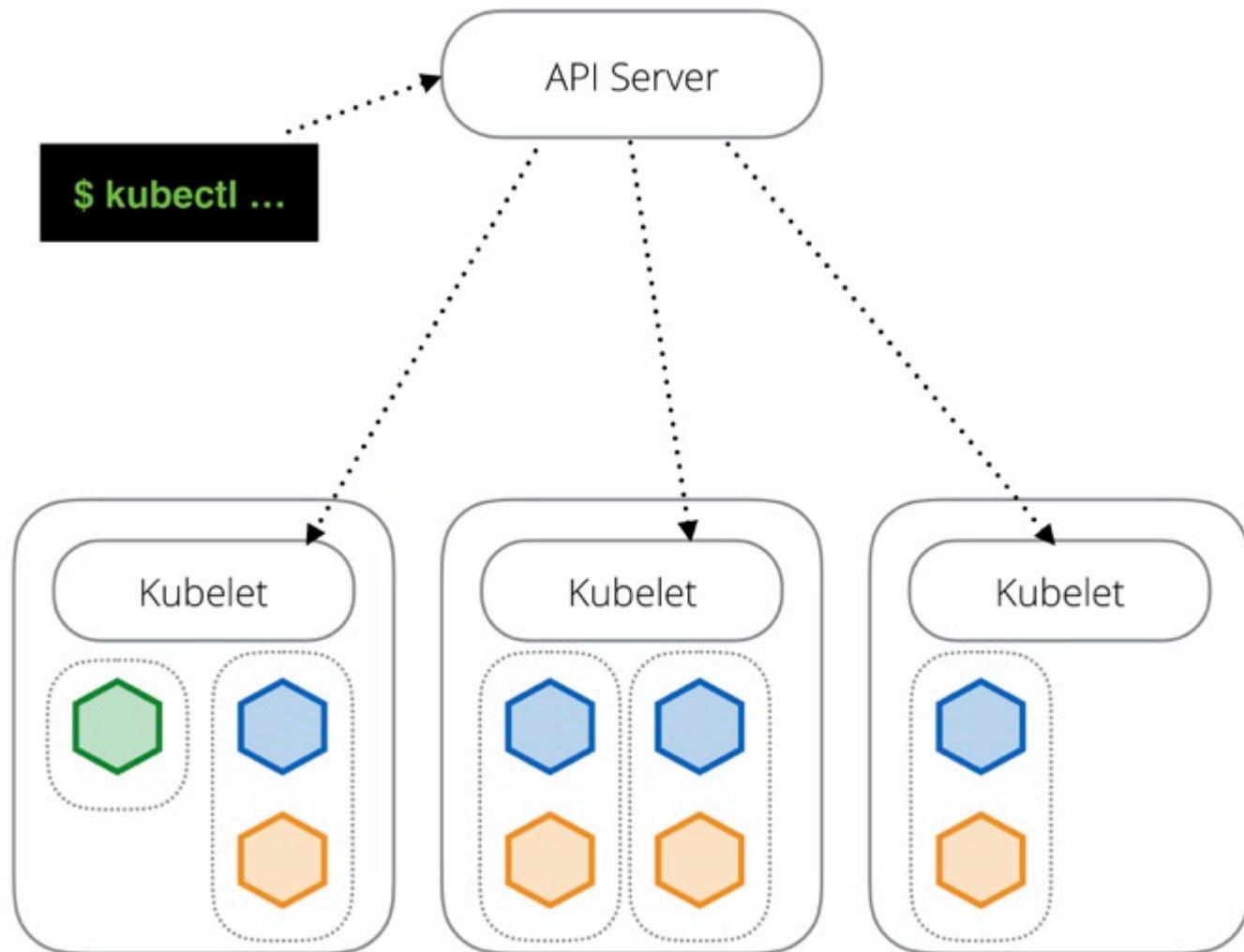
KUBERNETES ARCHITECTURE...SORT OF



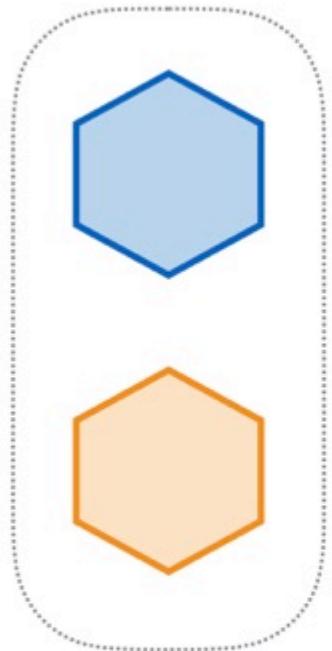
KUBERNETES ARCHITECTURE...SORT OF



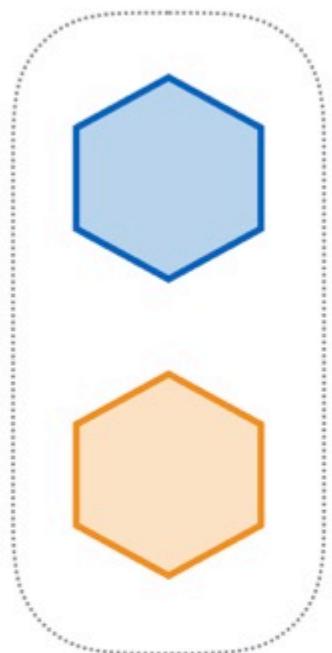
KUBERNETES ARCHITECTURE...SORT OF



PODS?

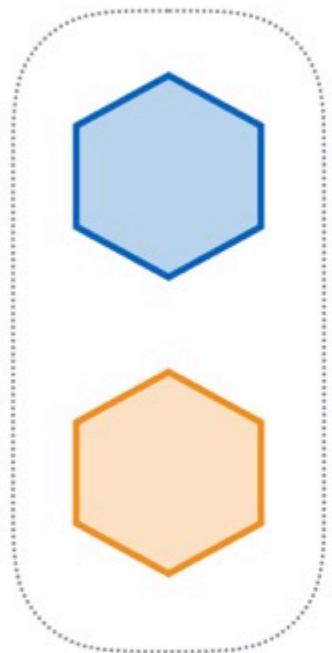


PODS?



A collection of tightly coupled
containers, running on one node

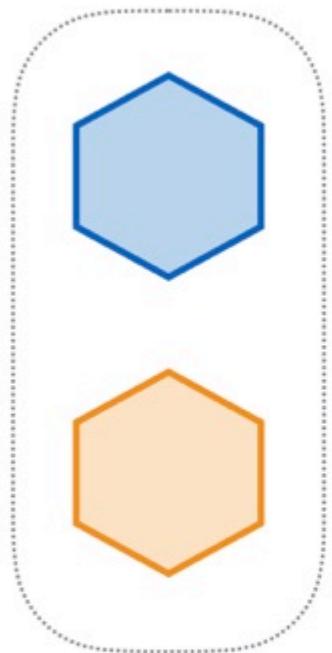
PODS?



A collection of tightly coupled containers, running on one node

Can have metadata, volumes too

PODS?

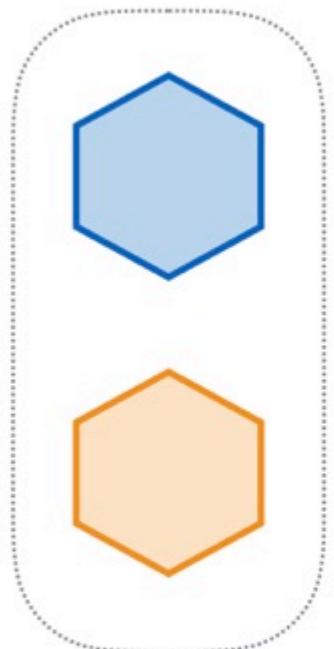


A collection of tightly coupled containers, running on one node

Can have metadata, volumes too

Pods are mortal - not long running!

PODS?



A collection of tightly coupled containers, running on one node

Can have metadata, volumes too

Pods are mortal - not long running!

A pod = a unit of scheduling

SERVICES!

SERVICES!

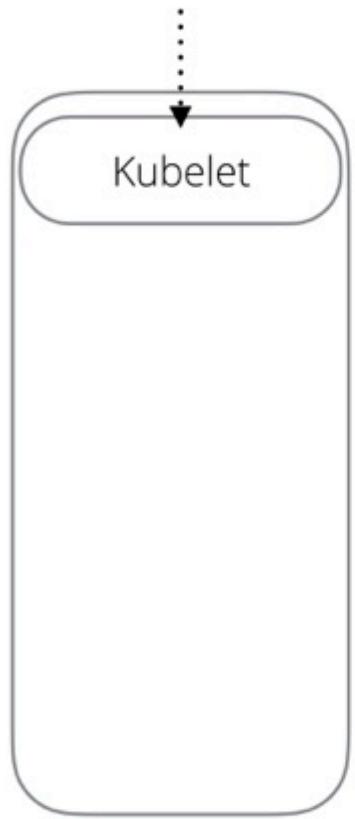
```
{  
  "kind": "Service",  
  "apiVersion": "v1",  
  "metadata": {  
    "name": "my-service"  
  },  
  "spec": {  
    "selector": {  
      "app": "MyApp"  
    },  
    "ports": [  
      {  
        "protocol": "TCP",  
        "port": 80,  
        "targetPort": 9376  
      }  
    ]  
  }  
}
```

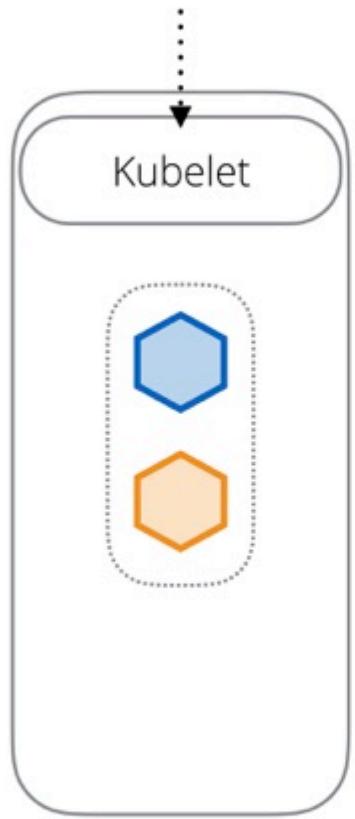
SERVICES!

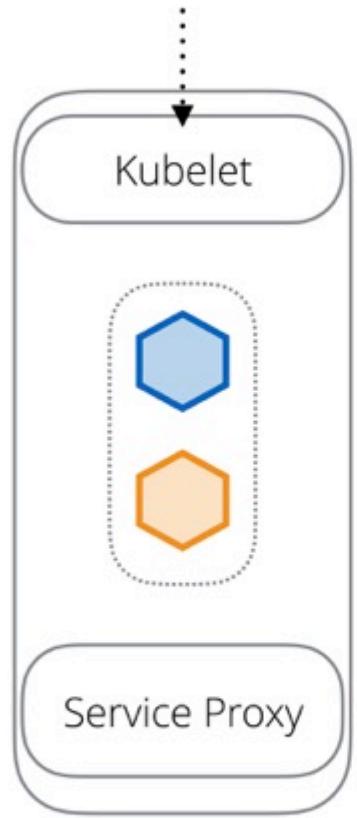
```
{  
  "kind": "Service",  
  "apiVersion": "v1",  
  "metadata": {  
    "name": "my-service"  
  },  
  "spec": {  
    "selector": {  
      "app": "MyApp"  
    },  
    "ports": [  
      {  
        "protocol": "TCP",  
        "port": 80,  
        "targetPort": 9376  
      }  
    ]  
  }  
}
```

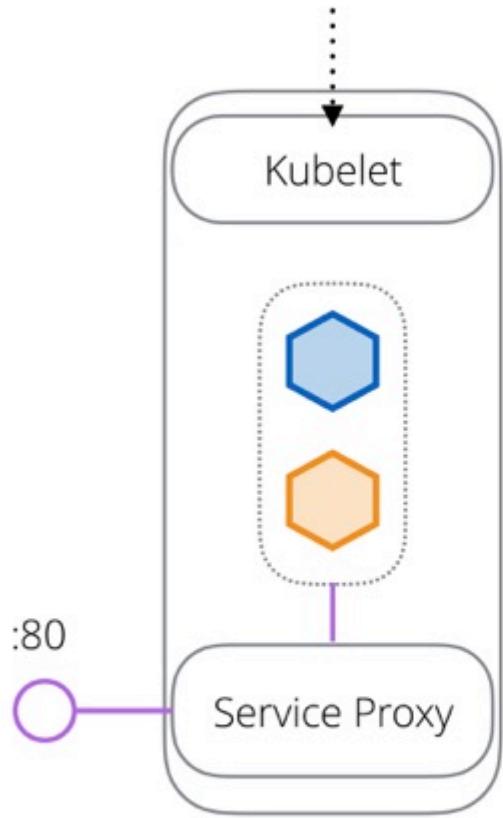
A mapping of metadata and ports to a set of pods

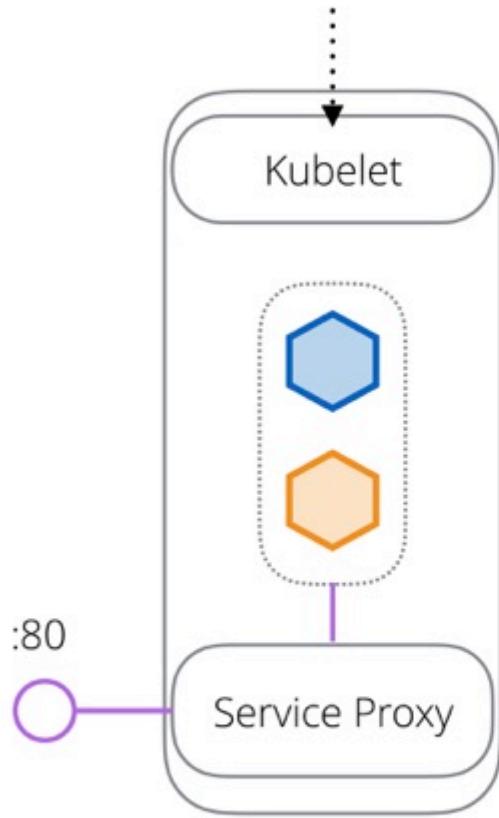




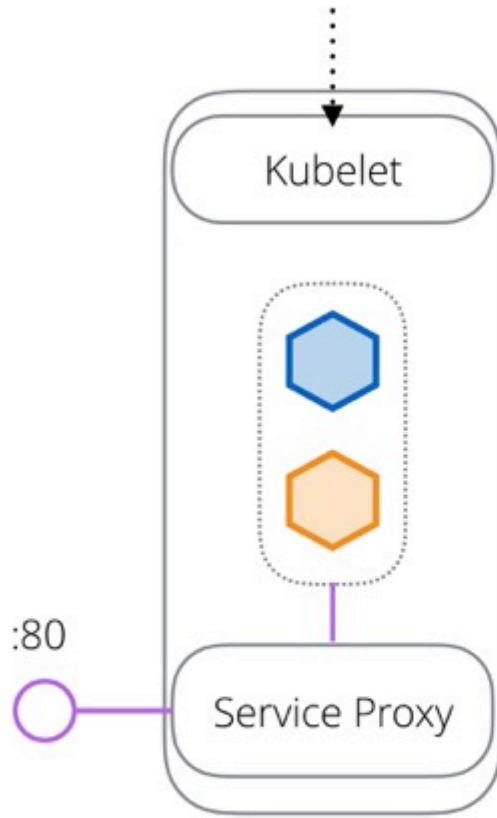








You don't scale a service...



You don't scale a service...

...you scale the pods!

KUBERNETES ON WINDOWS - IN ALPHA

Windows Server Containers



Kubernetes version 1.5 introduces support for Windows Server Containers. In version 1.5, the Kubernetes control plane (API Server, Scheduler, Controller Manager, etc) continue to run on Linux, while the kubelet and kube-proxy can be run on Windows Server.

Note: Windows Server Containers on Kubernetes is an Alpha feature in Kubernetes 1.5.

Prerequisites

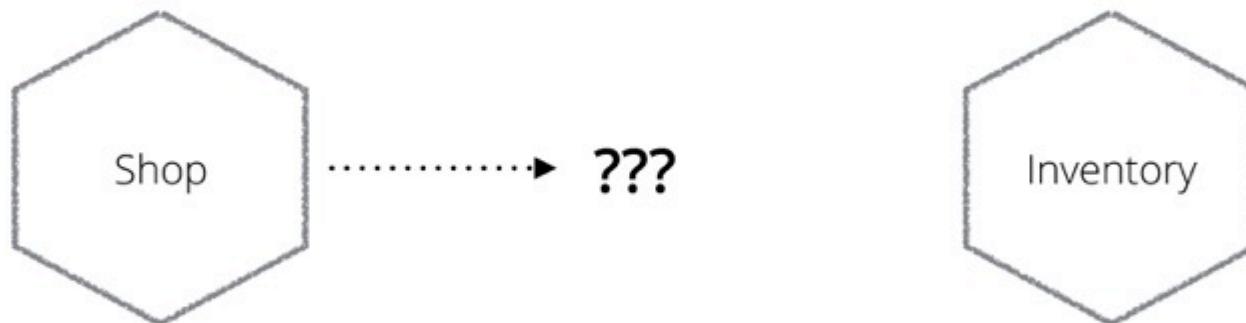
In Kubernetes version 1.5, Windows Server Containers for Kubernetes is supported using the following:

1. Kubernetes control plane running on existing Linux infrastructure (version 1.5 or later)
2. Kubenet network plugin setup on the Linux nodes
3. Windows Server 2016 (RTM version 10.0.14393 or later)
4. Docker Version 1.12.2-cs2-ws-beta or later for Windows Server nodes (Linux nodes and Kubernetes control plane can run any Kubernetes supported Docker Version)

<https://kubernetes.io/docs/getting-started-guides/windows/>

But What About Configuration?

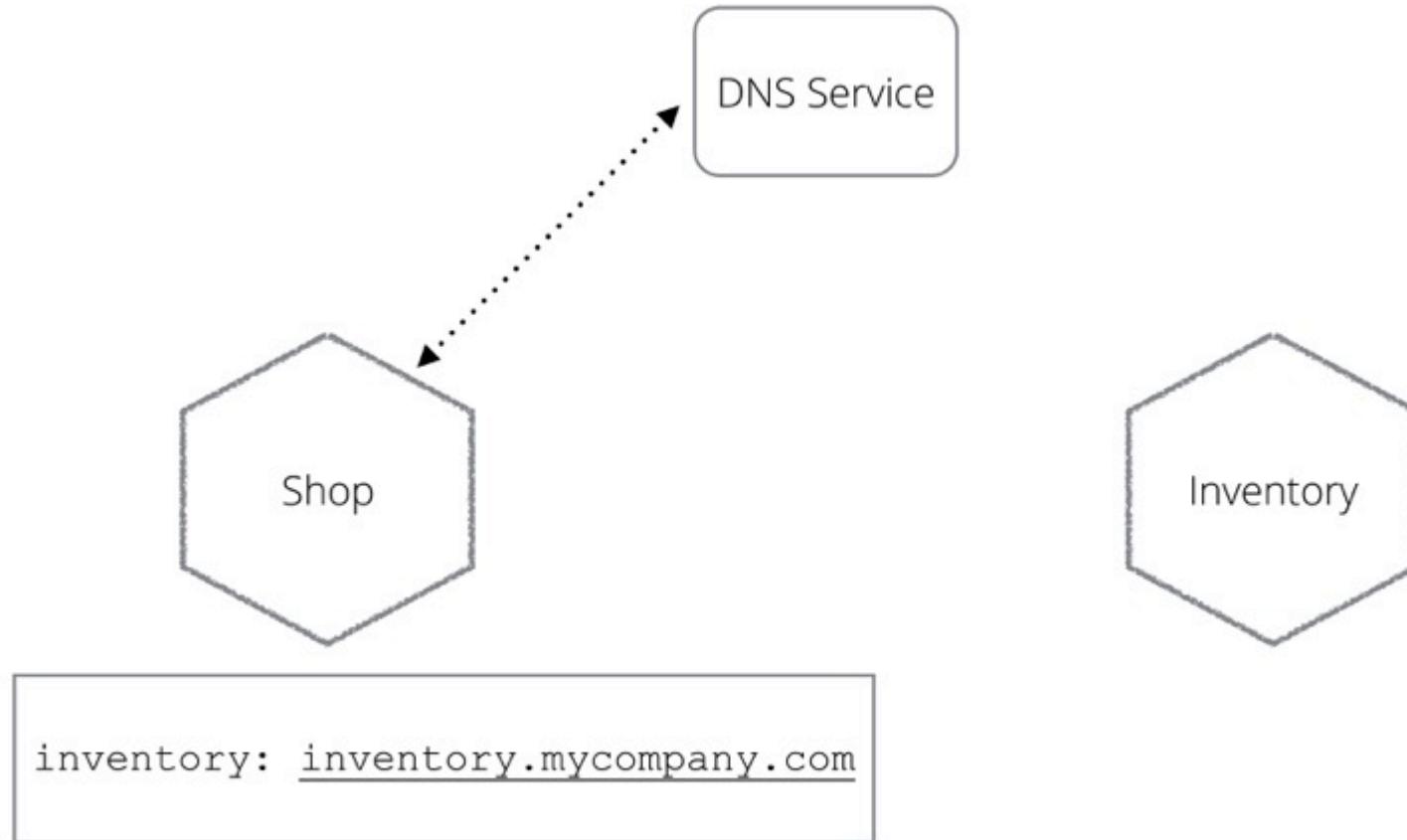
How Do We Find Things?



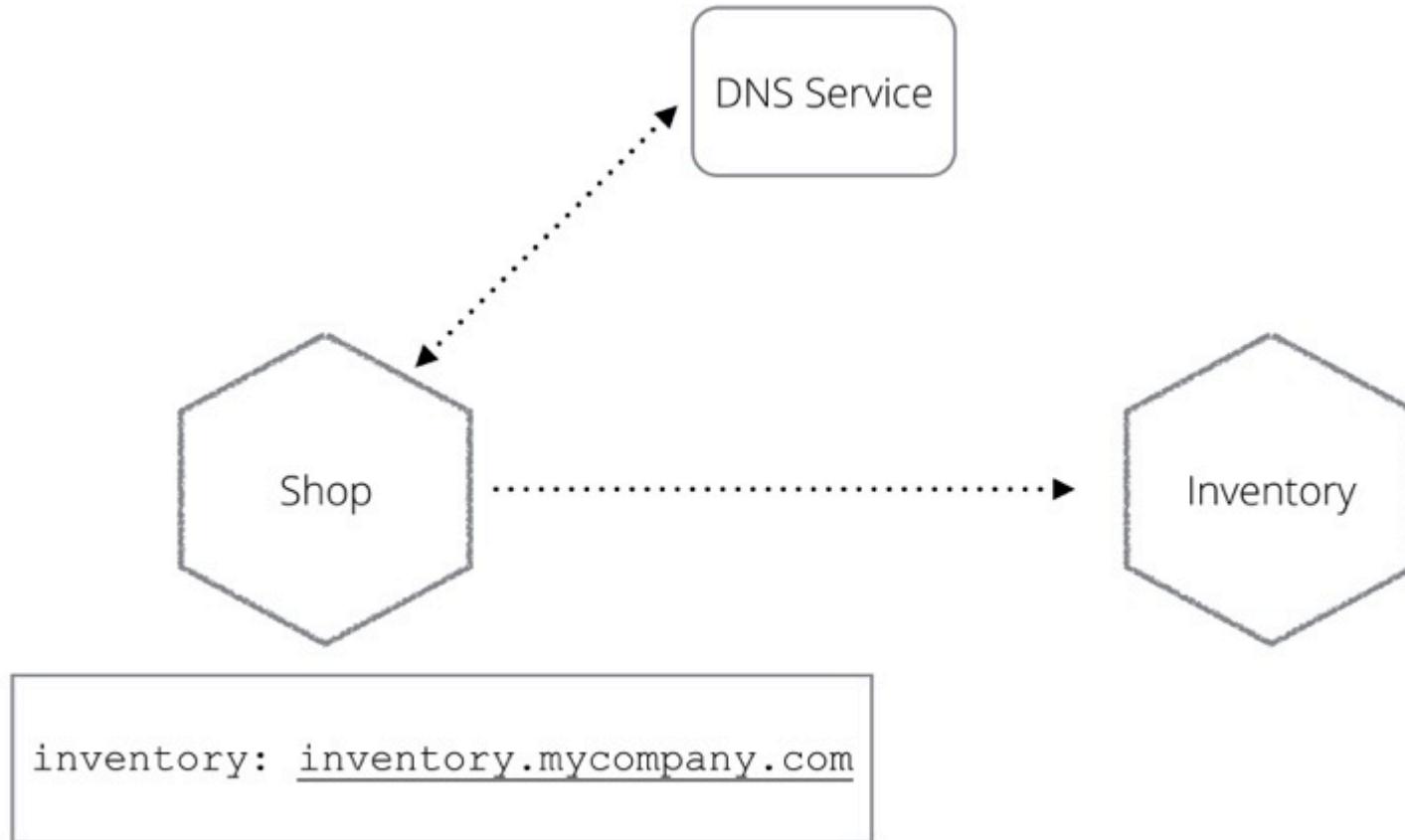
HARDCODE IP!



DNS

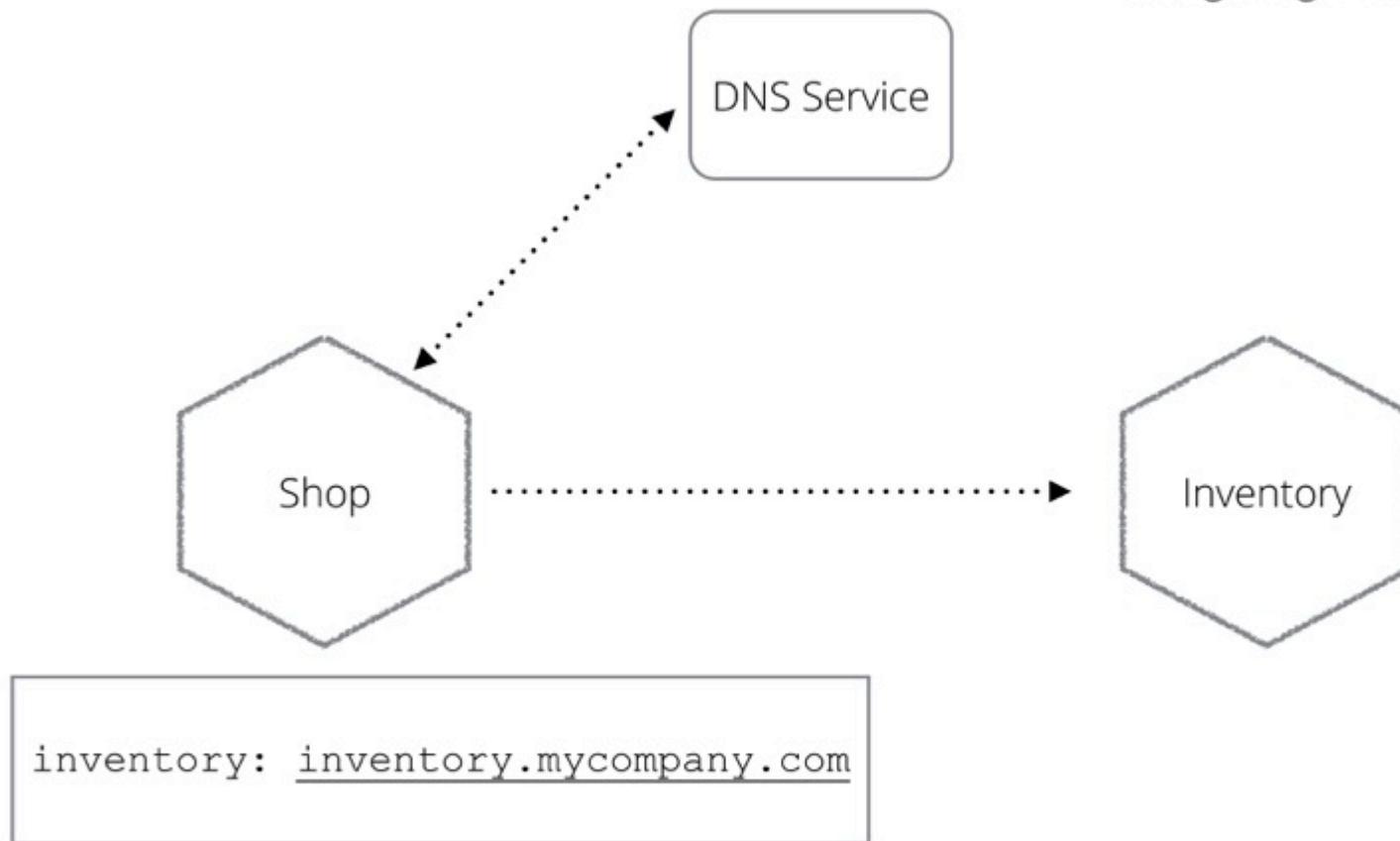


DNS

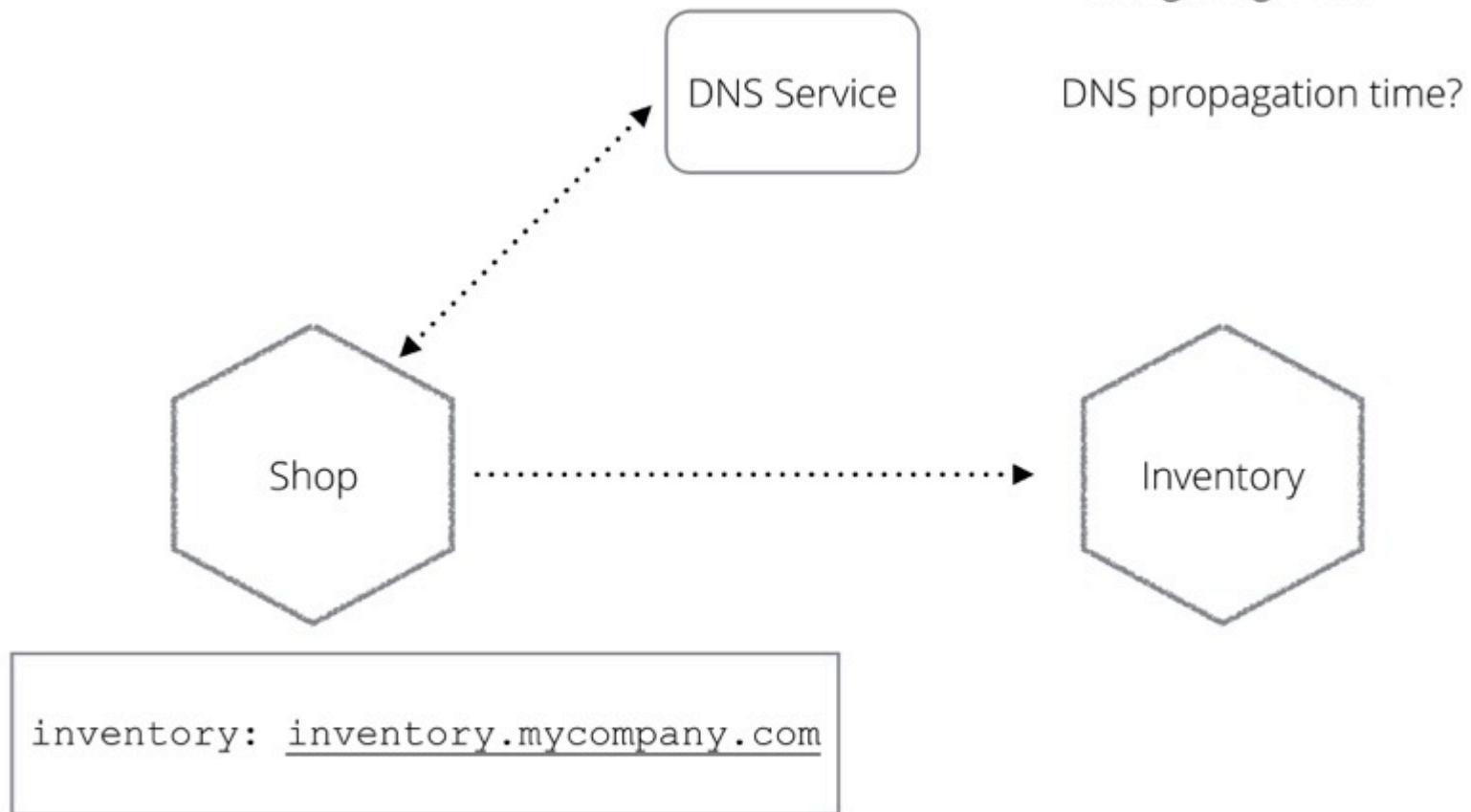


DNS

Configuring DNS?



DNS



Configuring DNS?

DNS propagation time?

SERVICE DISCOVERY TOOL

Service
Discovery

Shop

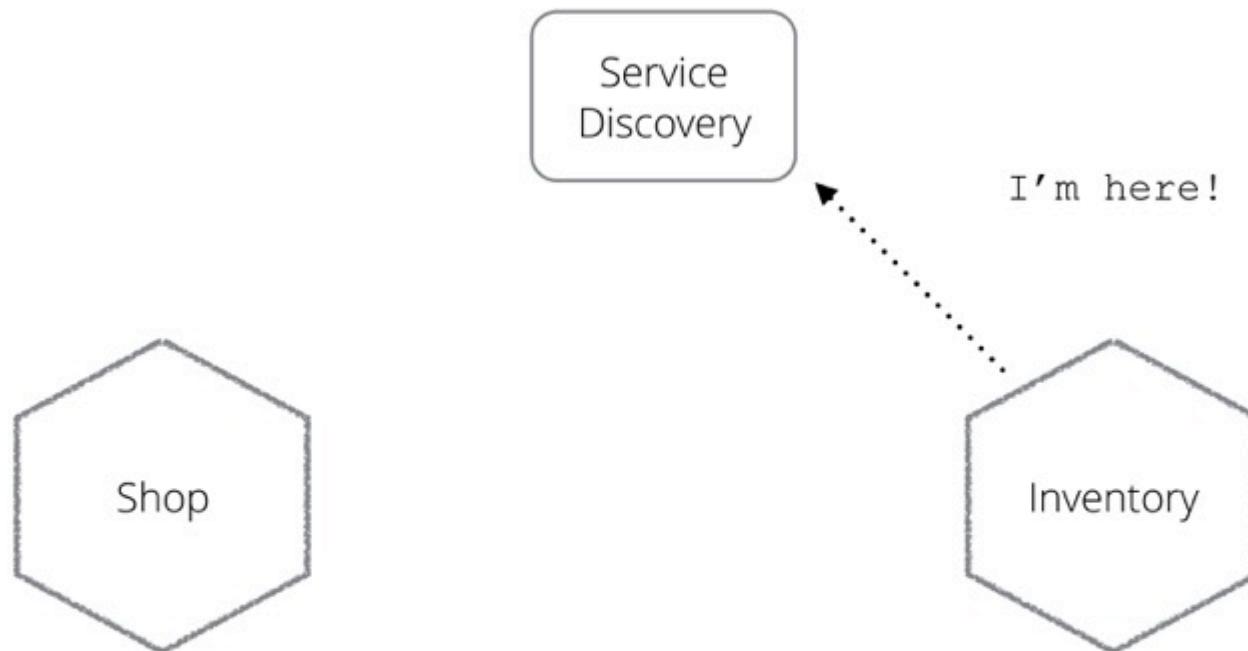
SERVICE DISCOVERY TOOL

Service
Discovery

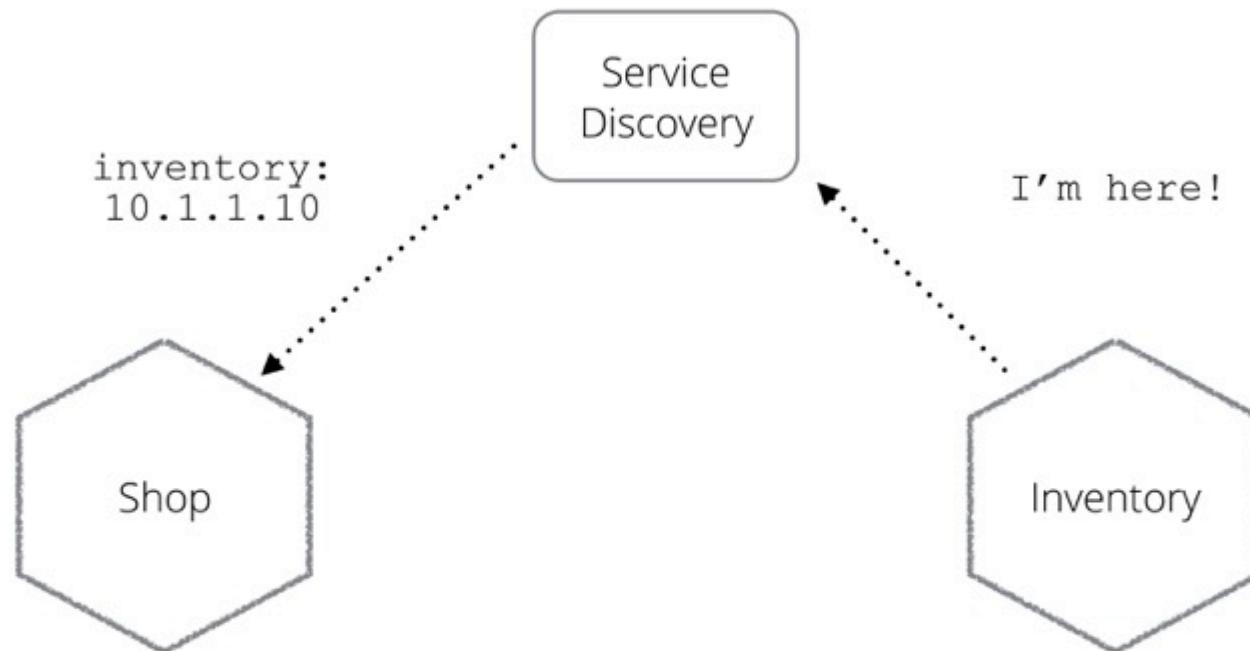
Shop

Inventory

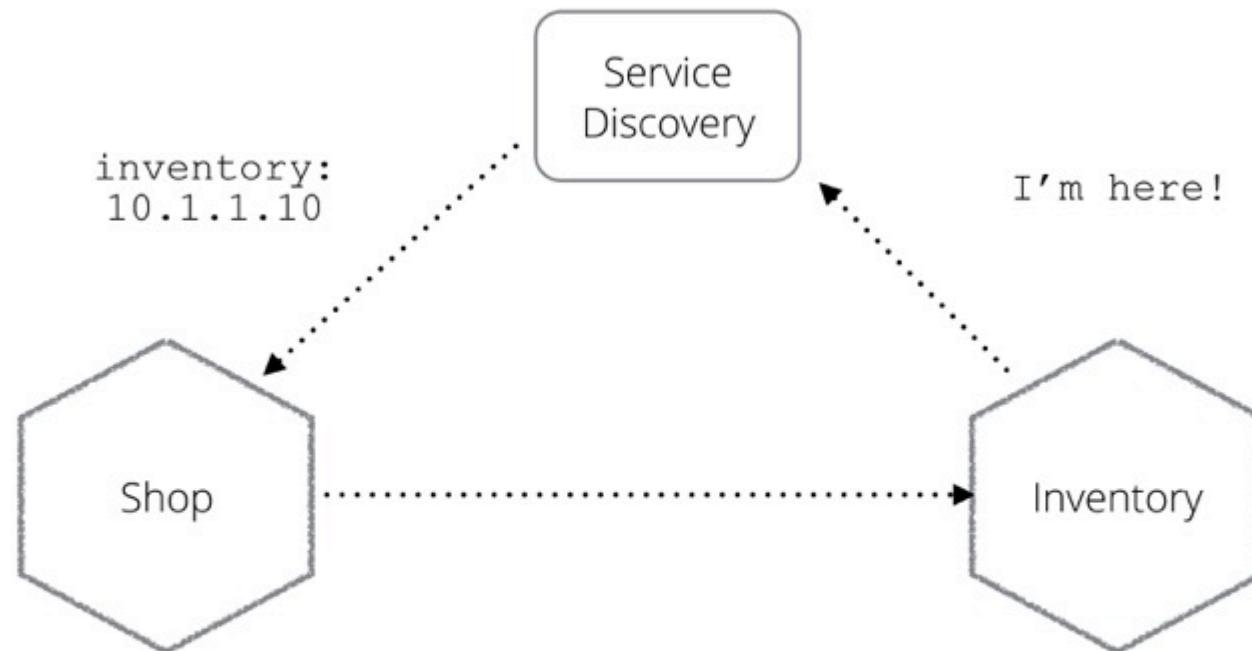
SERVICE DISCOVERY TOOL



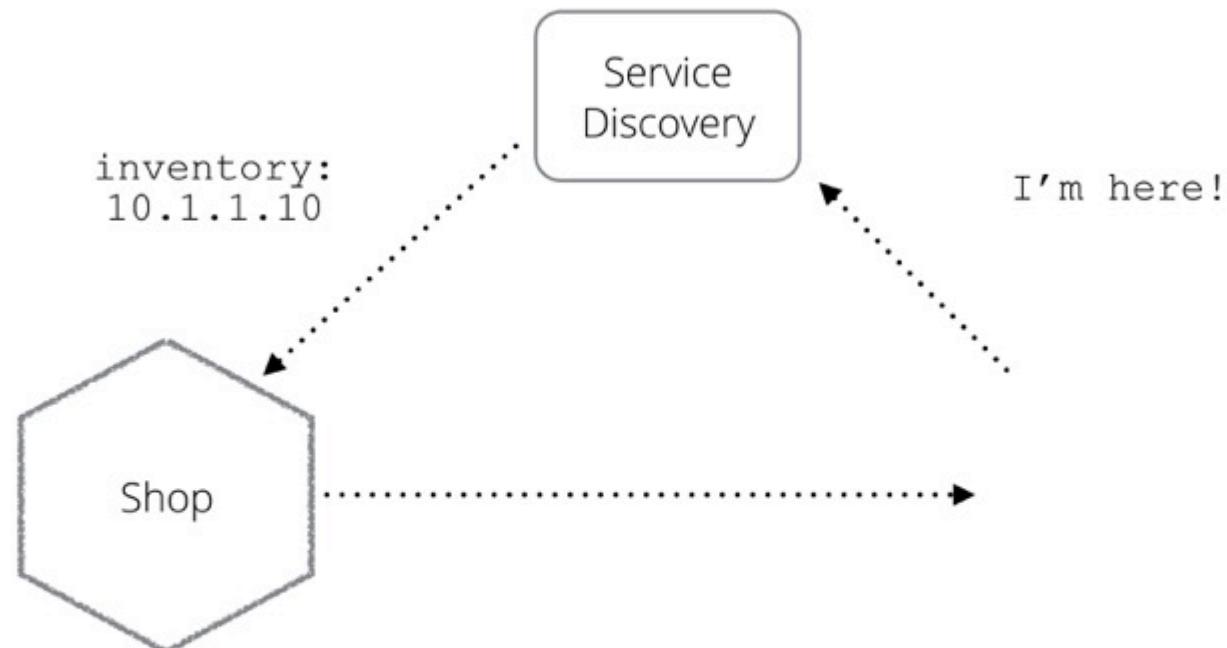
SERVICE DISCOVERY TOOL



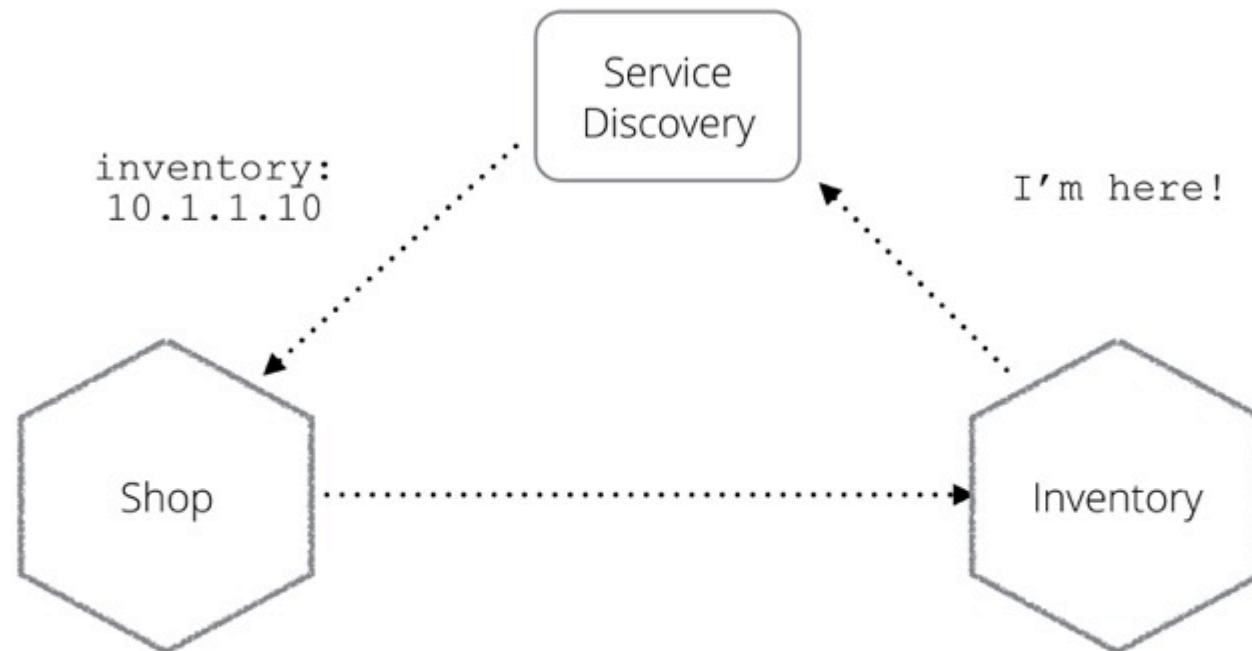
SERVICE DISCOVERY TOOL



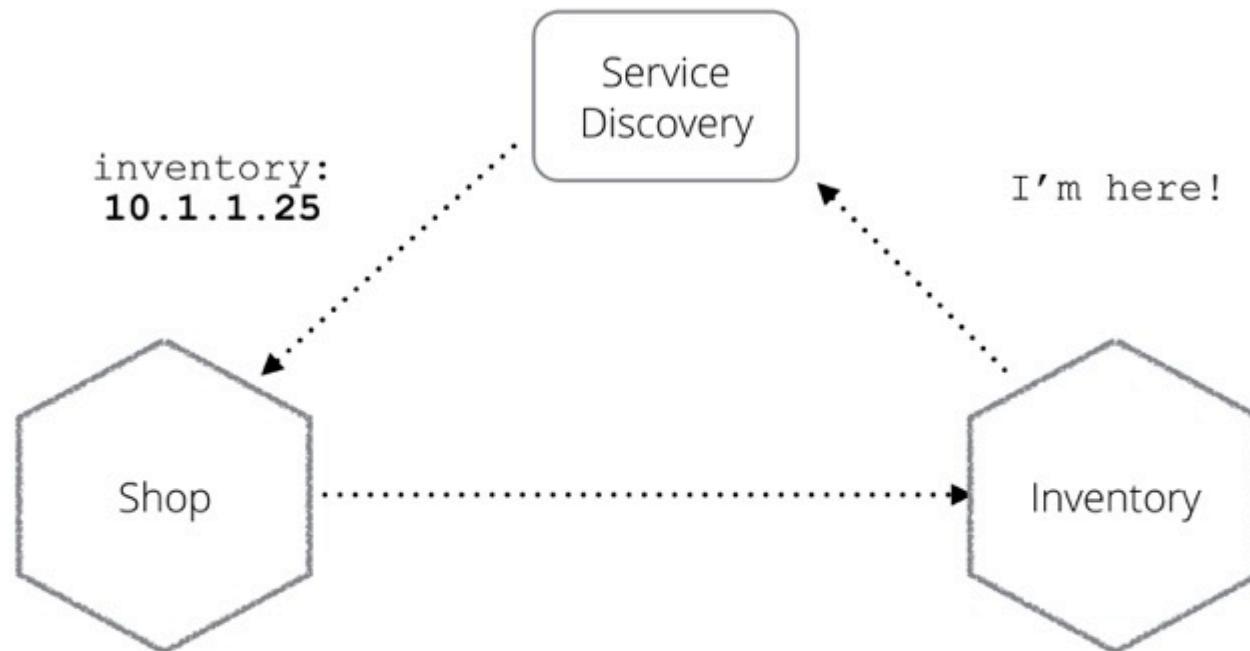
SERVICE DISCOVERY TOOL



SERVICE DISCOVERY TOOL



SERVICE DISCOVERY TOOL

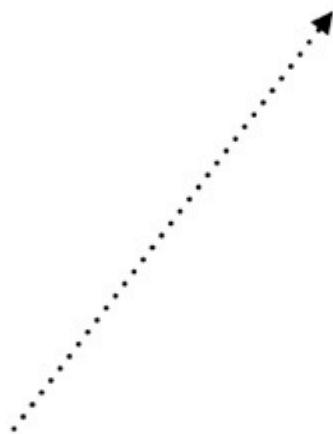




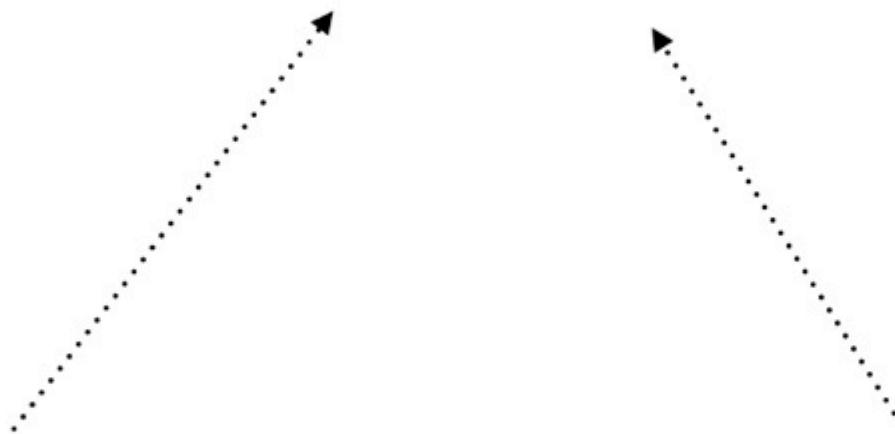








DNS SRV



DNS SRV

REST API





Service Discovery



Service Discovery

Configuration



Service Discovery Configuration

Watches



Service Discovery

Configuration

Watches

Consul-template

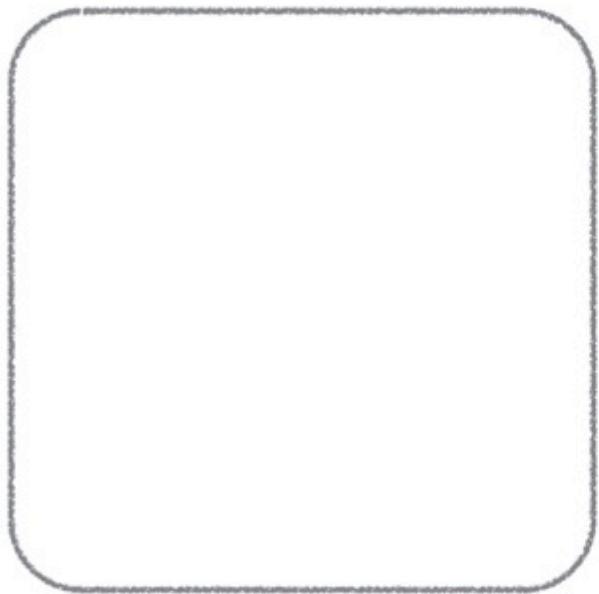
Beware of writing your own service discovery system

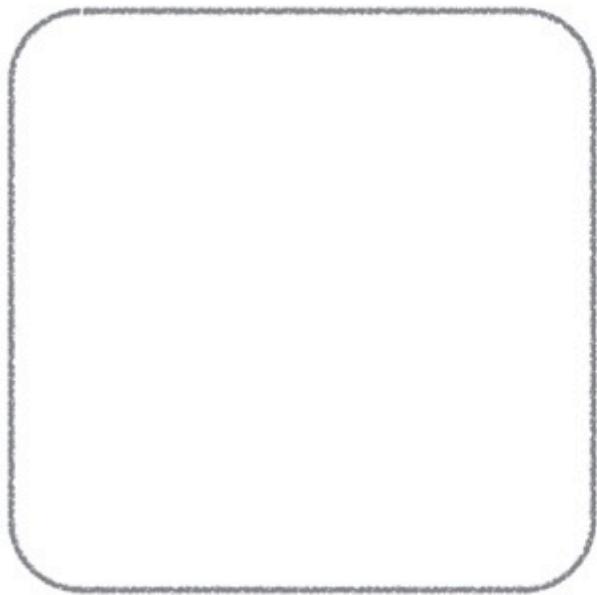
MONOLITHS

TO MICROSERVICES

Sam Newman

Monitoring & Alerting

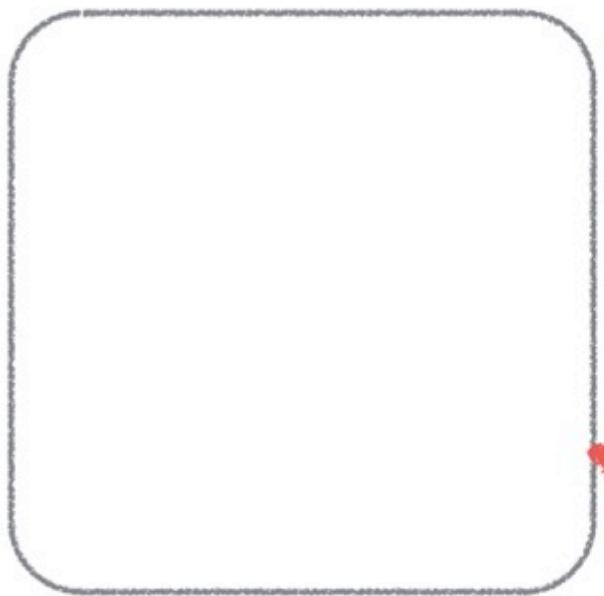




<http://www.flickr.com/photos/kalexanderson/5421517469/>



<http://www.flickr.com/photos/kalexanderson/5421517469/>





Honest Status Page

@honest_update

 Follow

We replaced our monolith with micro services so that every outage could be more like a murder mystery.

RETWEETS

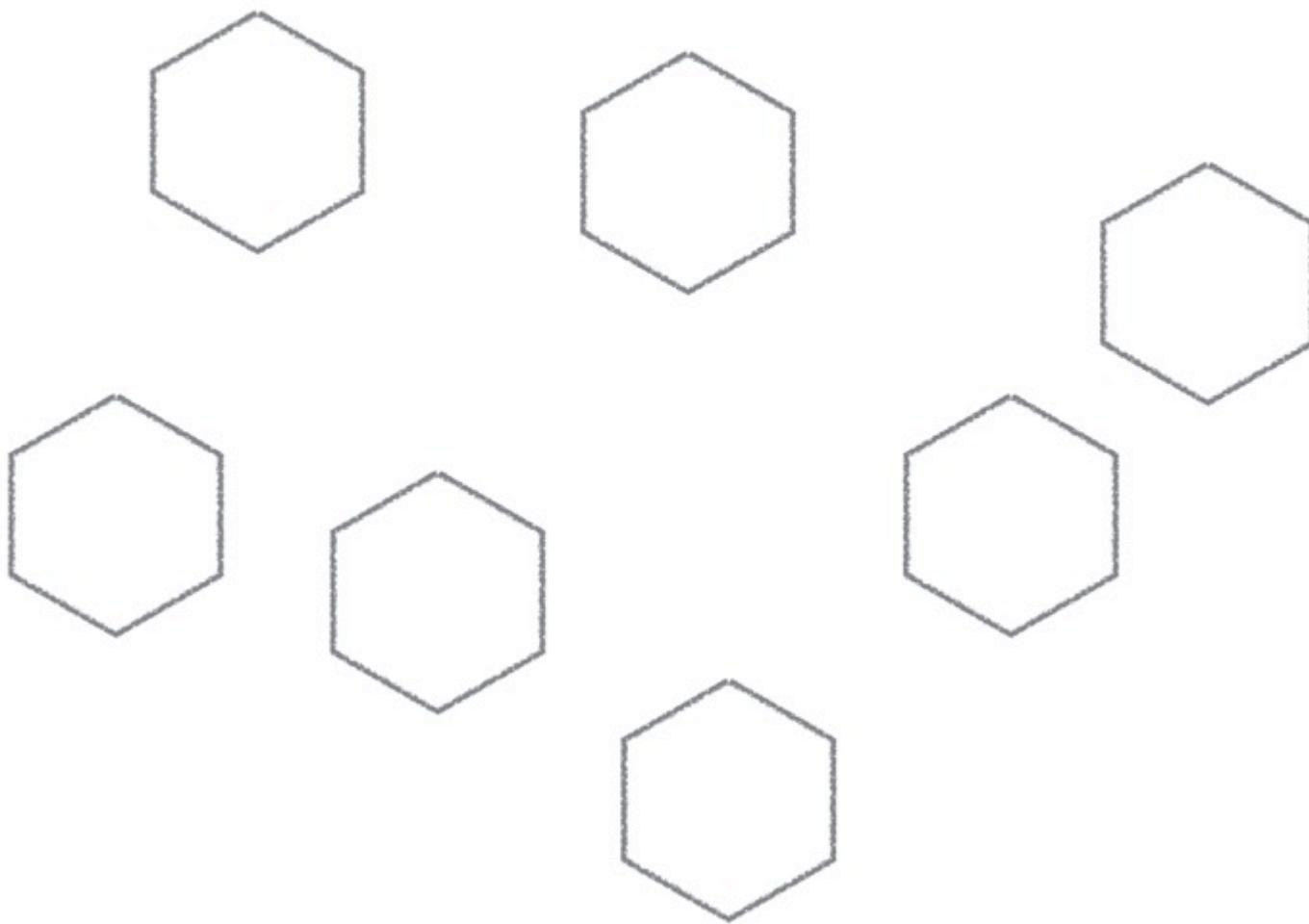
2,880

LIKES

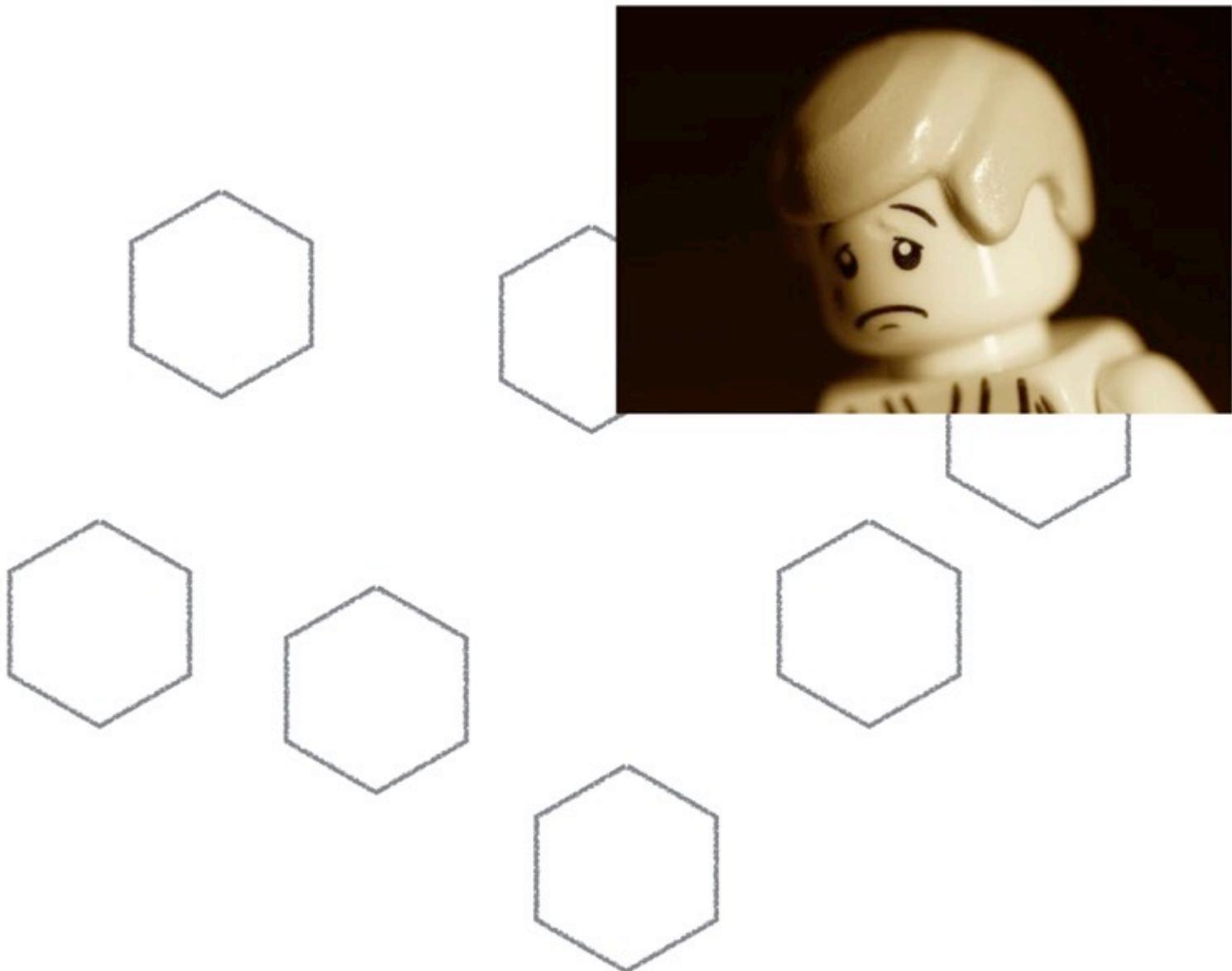
2,261



https://twitter.com/honest_update/status/651897353889259520

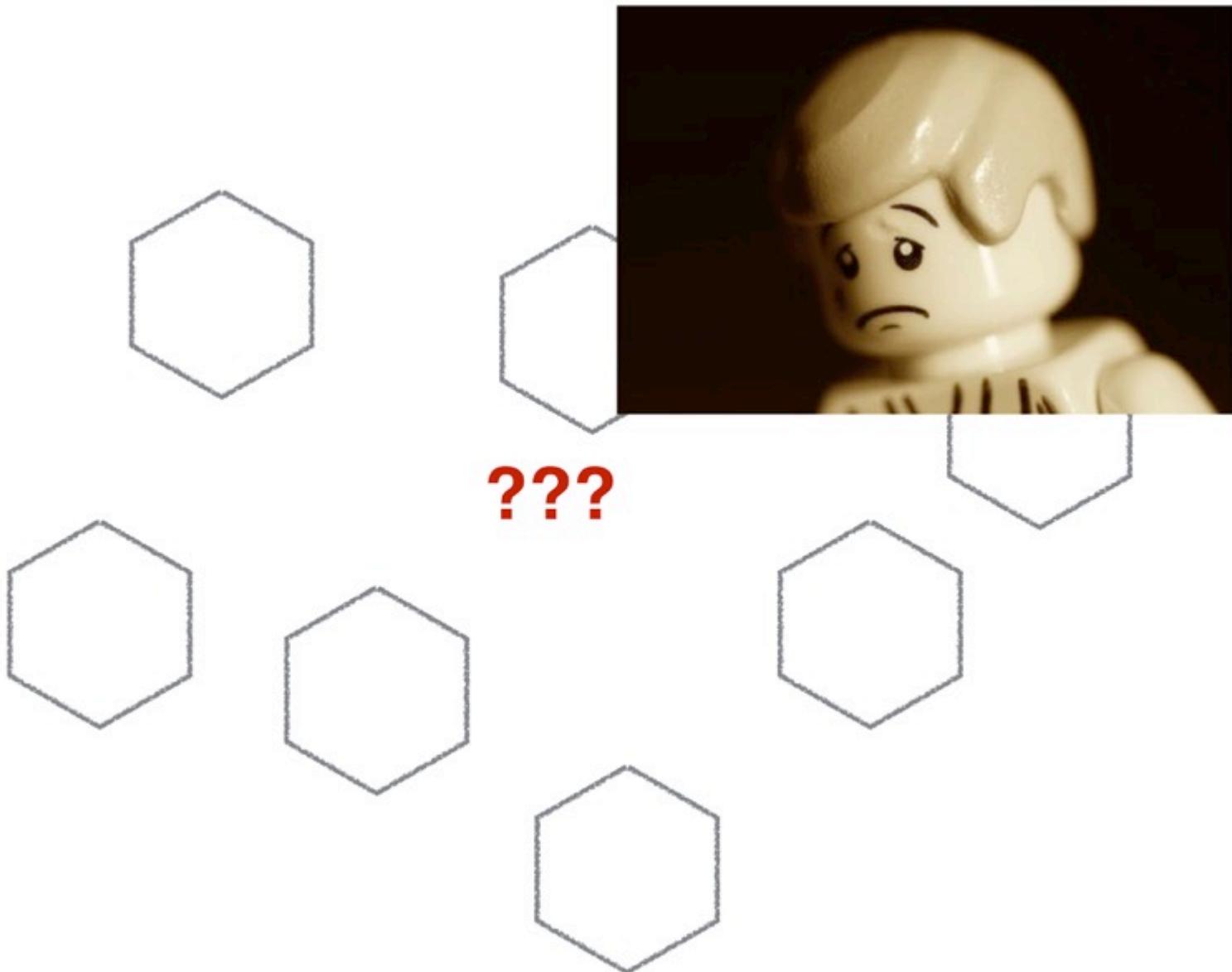


4

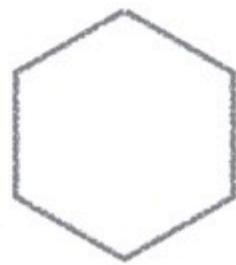
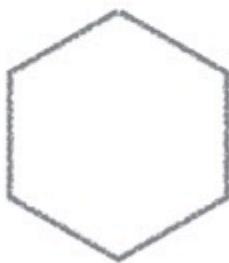
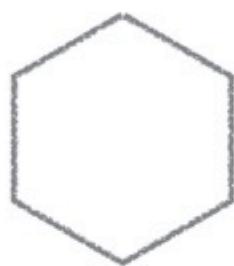
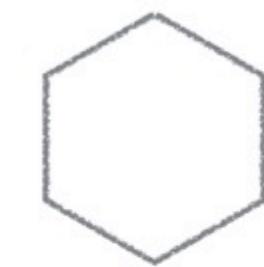


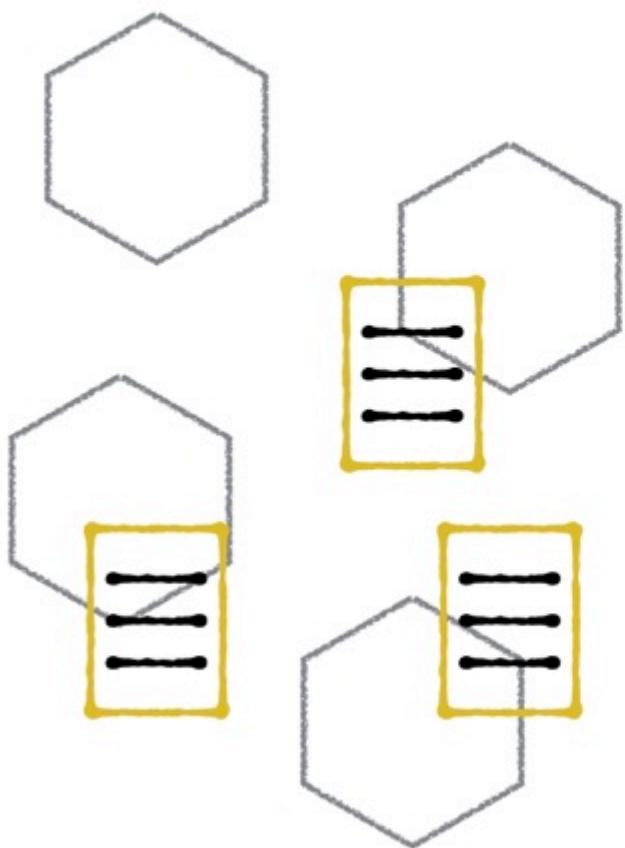
4

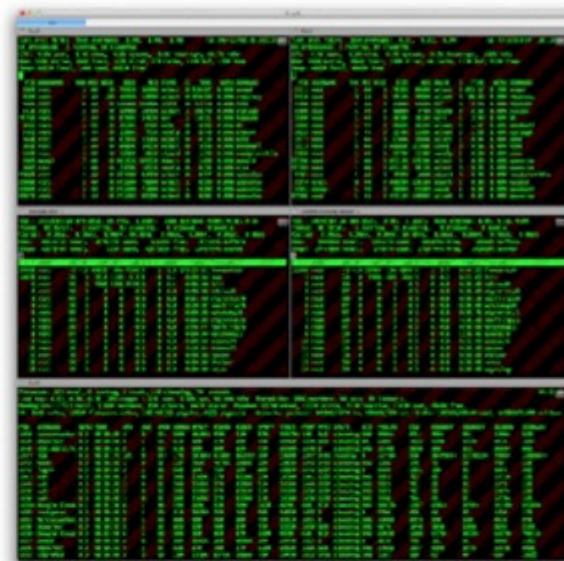
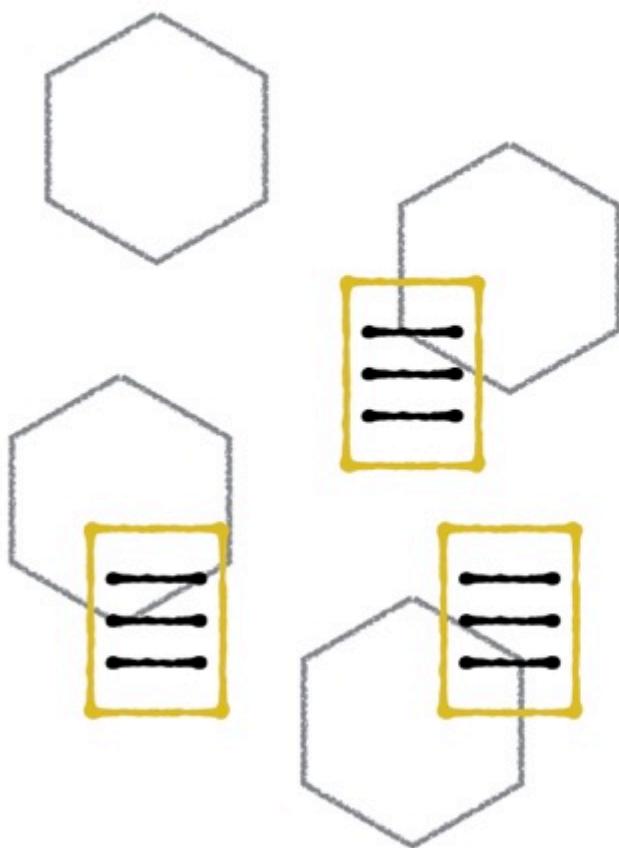
303

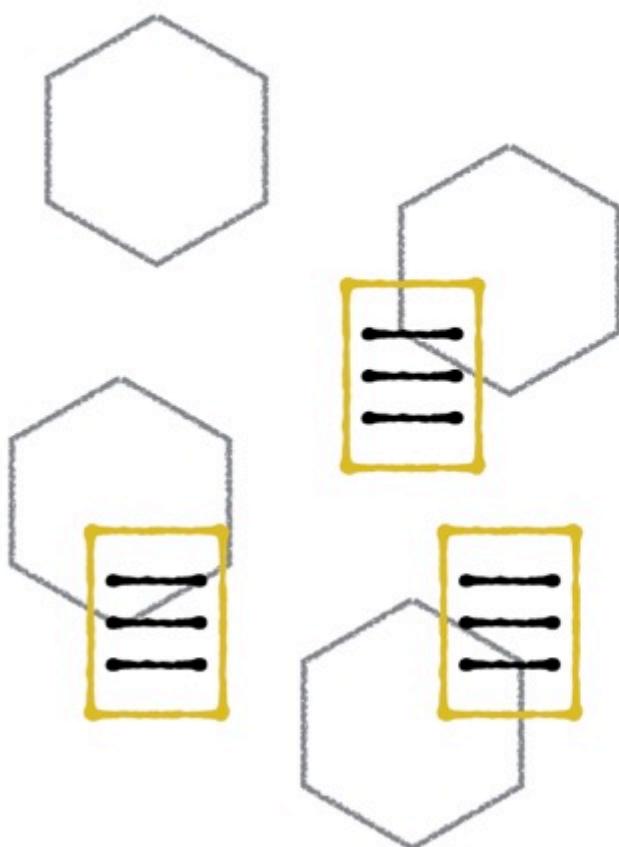


You have to get *much*
better at monitoring









Too many processes

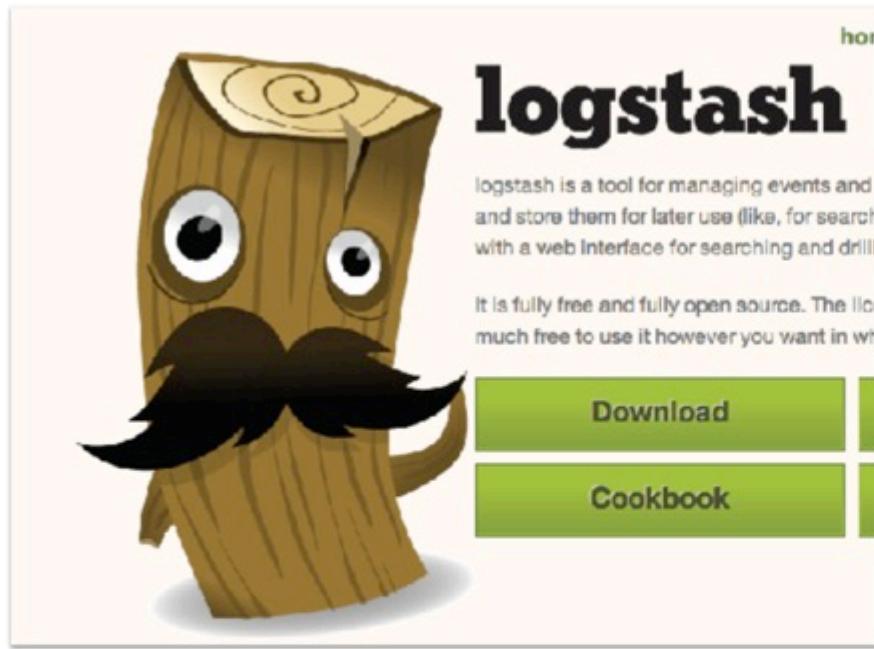
Too many processes

Machines are ephemeral

Extract information from hosts and services, and aggregate it centrally

The first thing you should do:

Log Aggregation



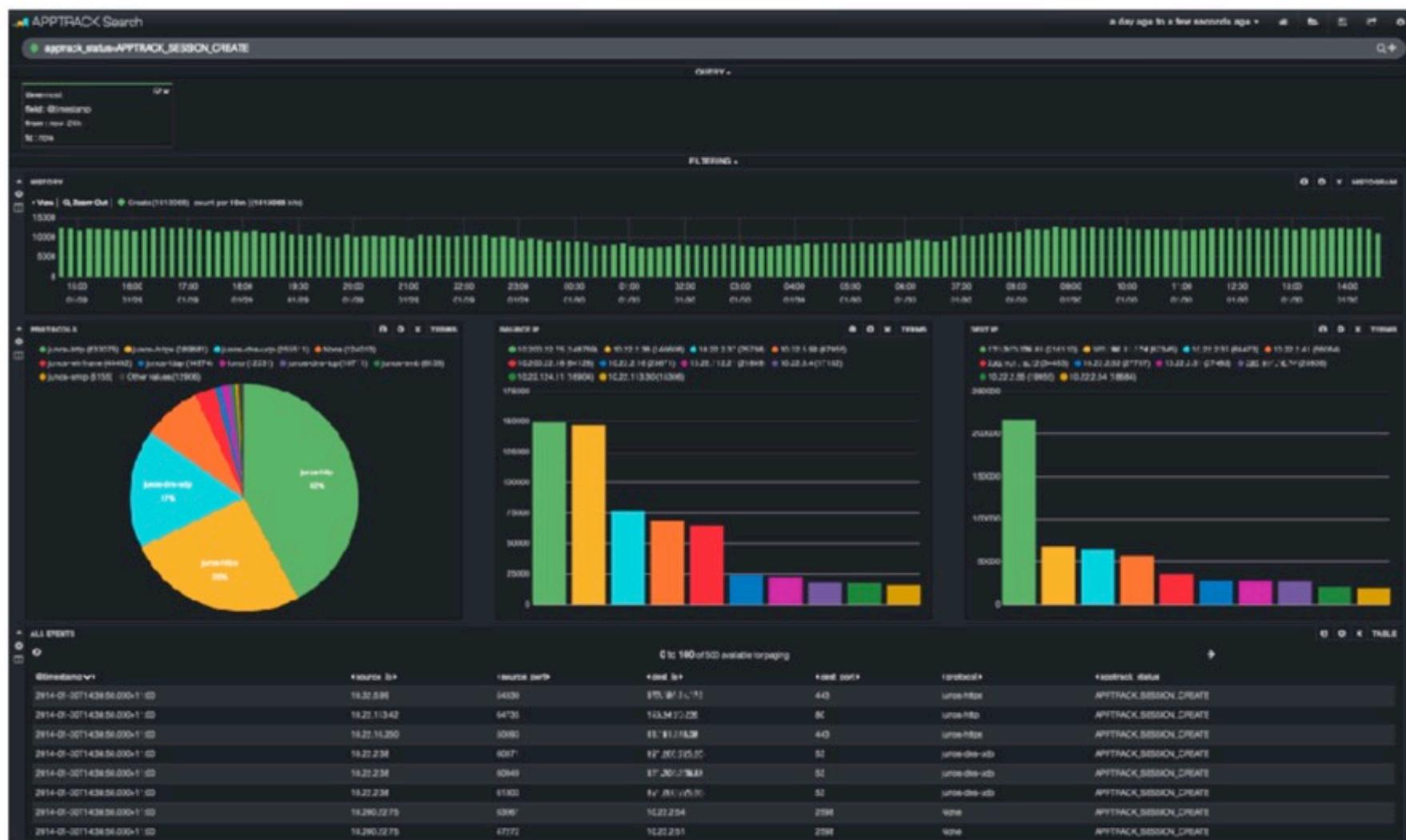
The image shows a screenshot of the Kibana GitHub repository page. At the top left is the Logstash logo, featuring a cartoon tree trunk with a mustache and large eyes. To the right of the logo is the word "logstash". Below this, there is a brief description of Logstash's purpose: "logstash is a tool for managing events and I and store them for later use (like, for search with a web interface for searching and drill...". A note below states: "It is fully free and fully open source. The lic much free to use it however you want in wh".

The main title "Kibana" is prominently displayed in large white letters against a background of a forest scene. Below the title, the tagline "Make sense of a mountain of logs" is followed by a red button that says "Now in Ruby!".

Below the title, there is a green button labeled "Get Started »". Underneath the button, there are links to "GitHub project", "Logstash", and "ElasticSearch". At the bottom of the page, there are two buttons: "Star" with the number 1,198 and "Fork" with the number 287.

A large, bold text at the bottom of the page reads "Every event under one roof".

KIBANA DASHBOARD





splunk®>



<https://humio.com>

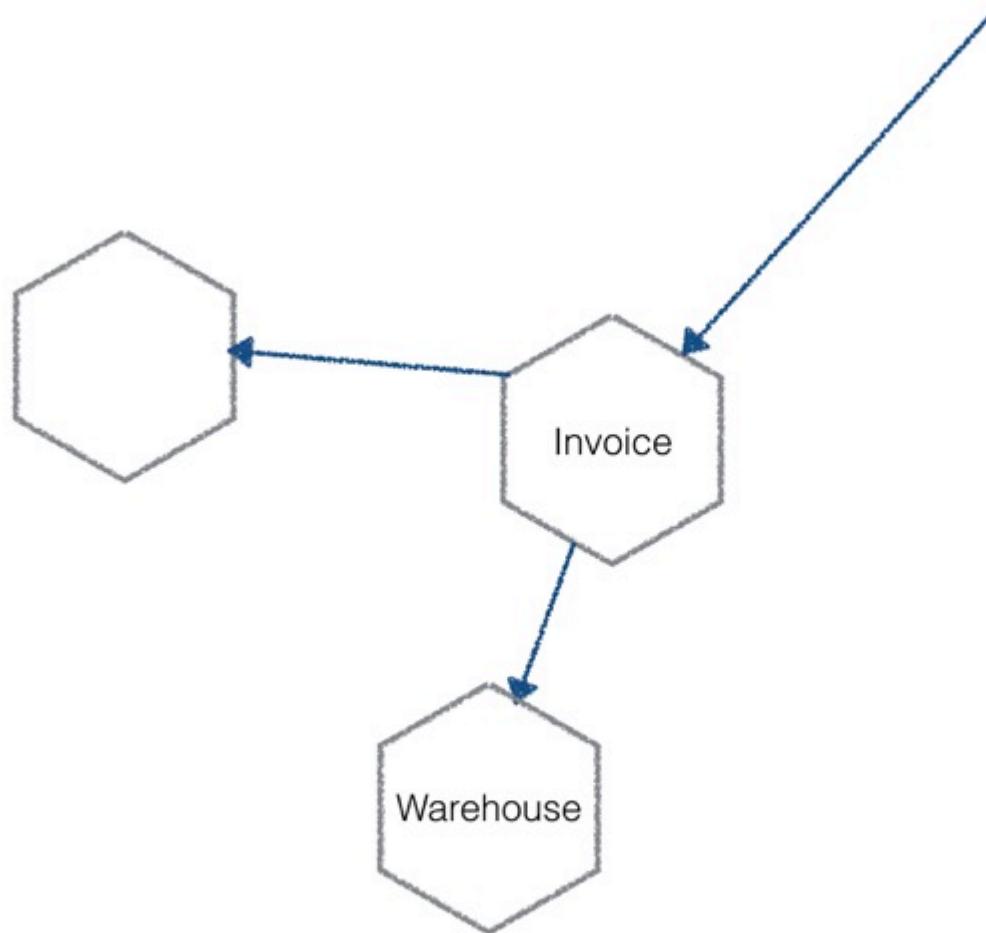
Consistent Format

Are your logs important?

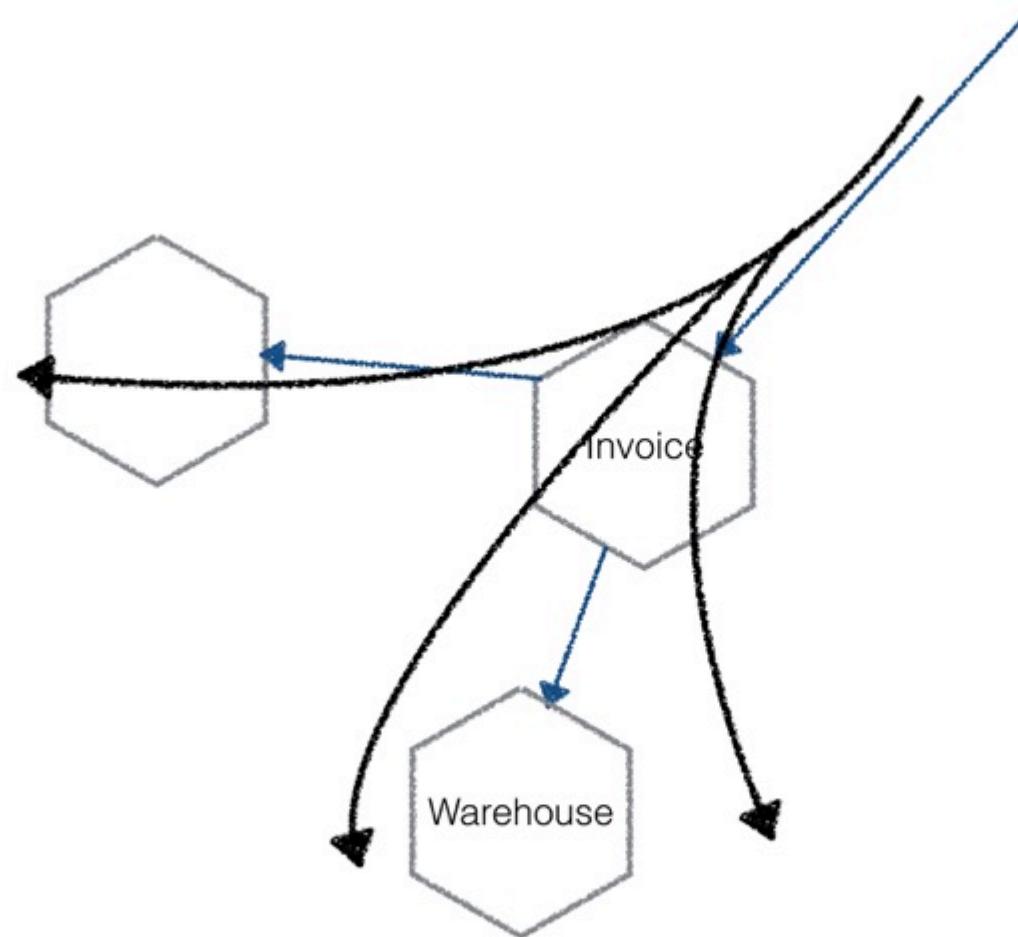
The first thing you should do:

Correlation IDs

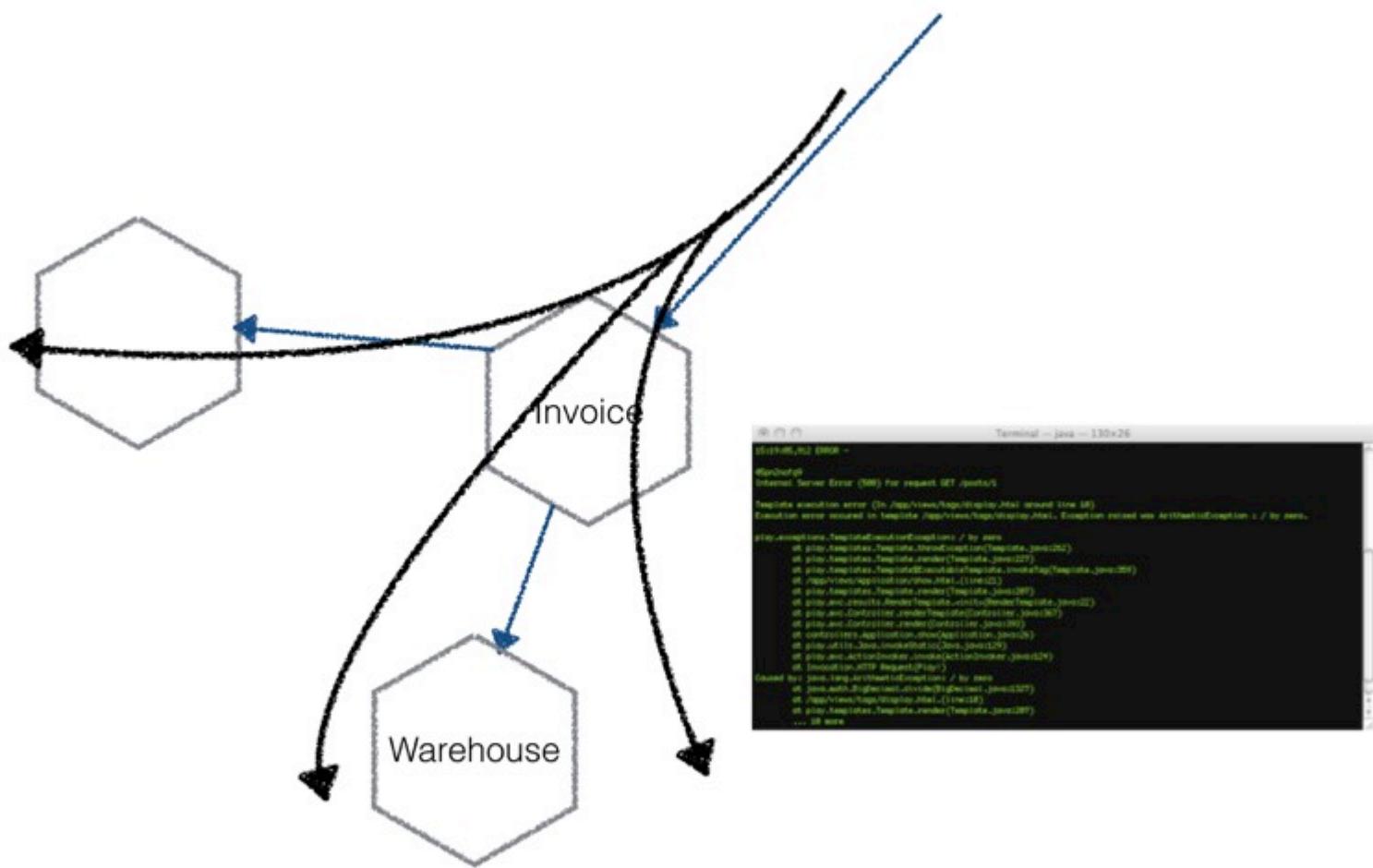
CORRELATION IDS



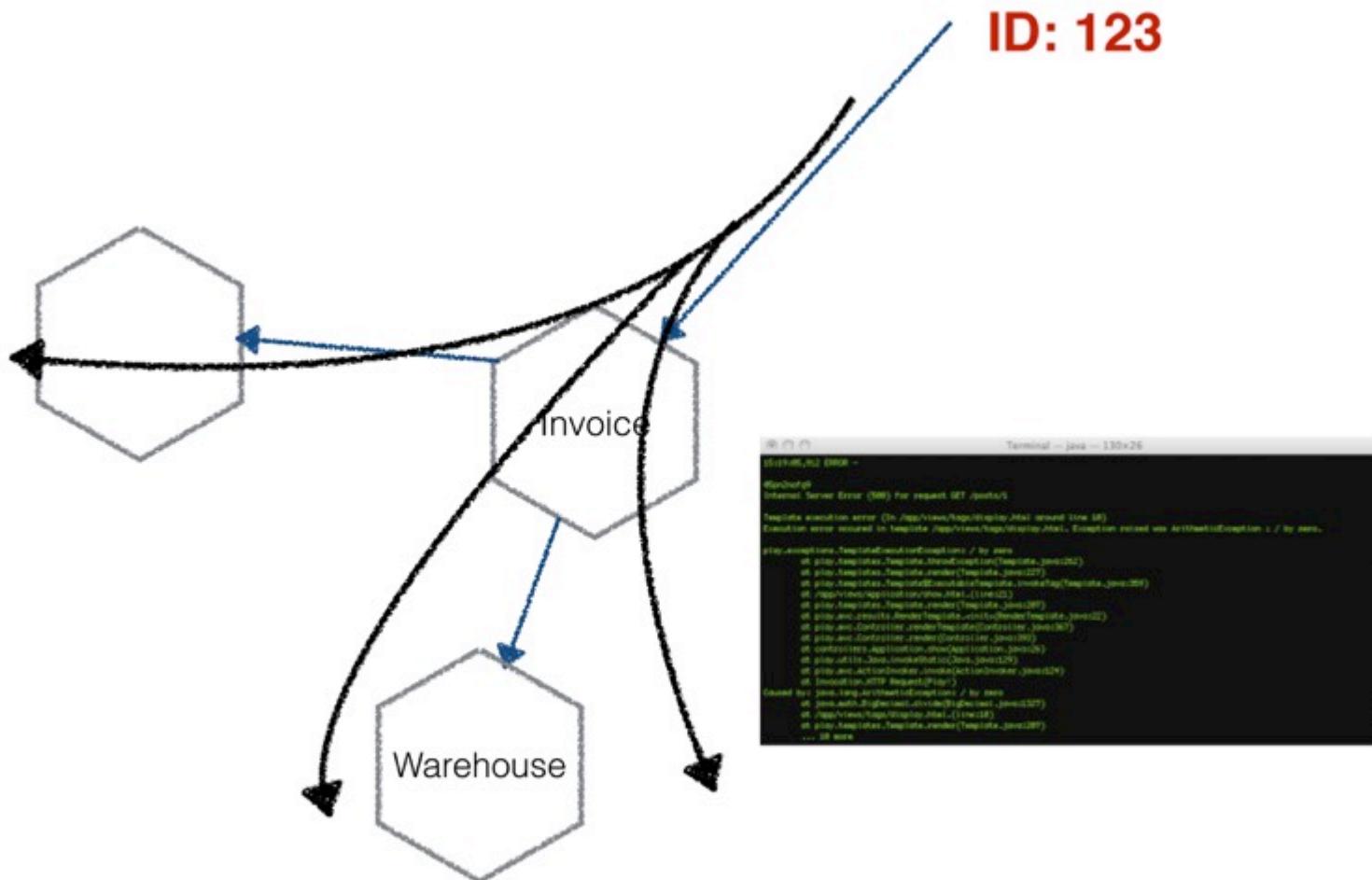
CORRELATION IDS



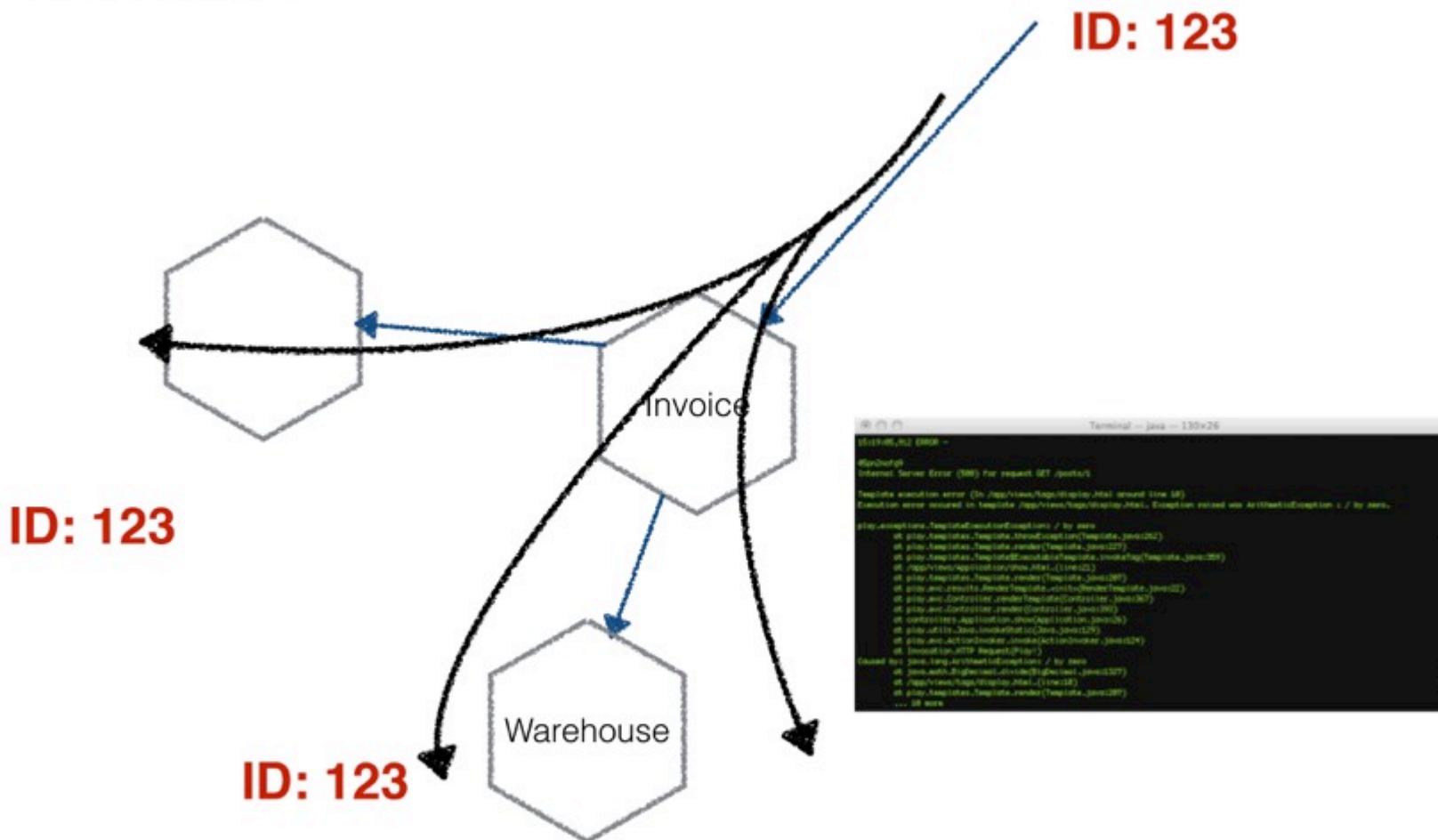
CORRELATION IDS



CORRELATION IDS



CORRELATION IDS



12:00 23/07/2017 [Invoice] WARN [123-gda] Unknown country code

12:01 23/07/2017 [Warehouse] ERROR [] Invalid country code

Correlation ID

12:00 23/07/2017 [Invoice] WARN [123-gda] Unknown country code

12:01 23/07/2017 [Warehouse] ERROR [] Invalid country code

Correlation ID

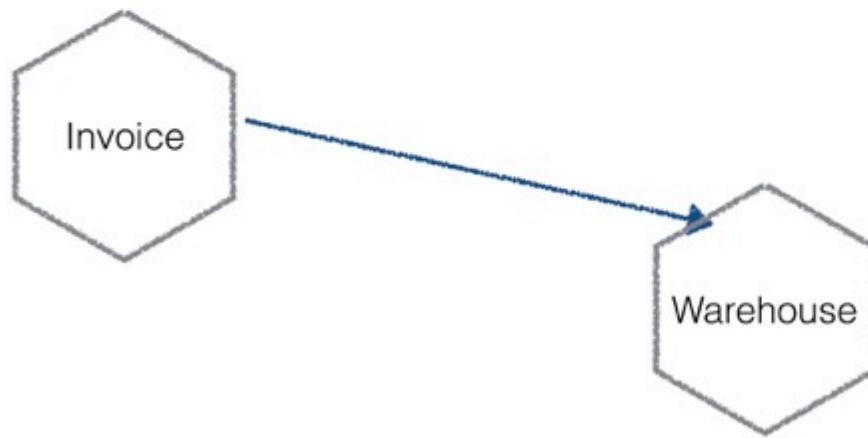
12:00 23/07/2017 [Invoice] WARN [123-gda] Unknown country code

12:01 23/07/2017 [Warehouse] ERROR [] Invalid country code

No correlation ID!

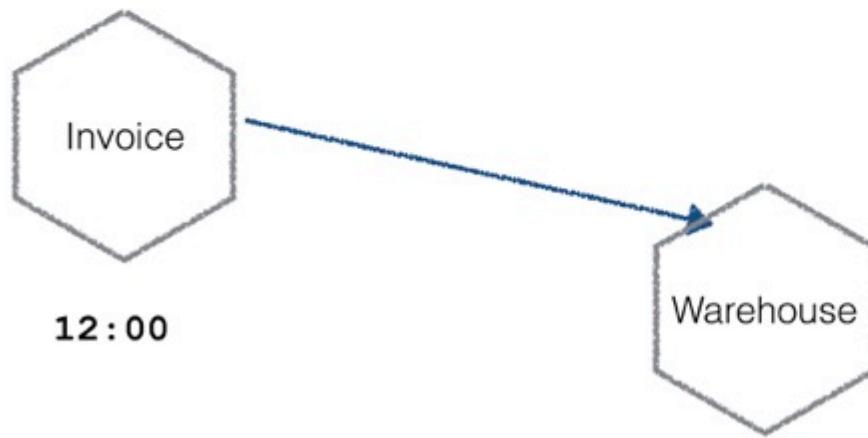
12:00 23/07/2017 [Invoice] WARN [123-gda] Unknown country code

12:01 23/07/2017 [Warehouse] ERROR [123-gda] Invalid country code



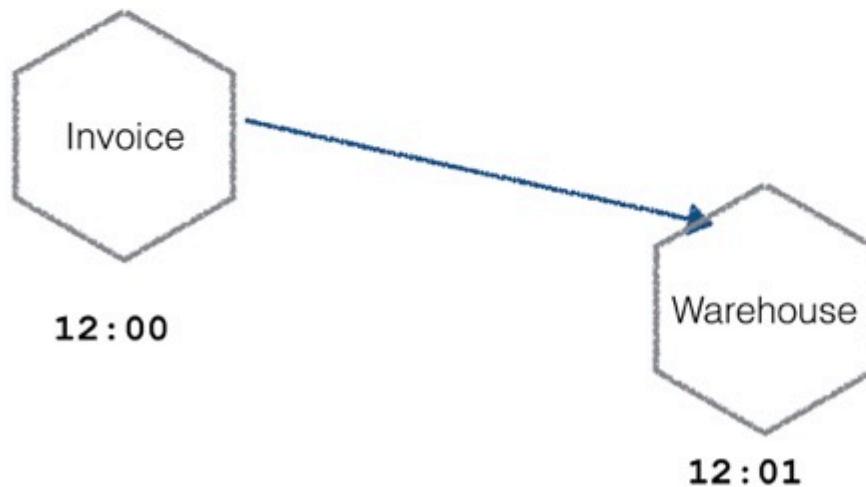
12:00 23/07/2017 [Invoice] WARN [123-gda] Unknown country code

12:01 23/07/2017 [Warehouse] ERROR [123-gda] Invalid country code



12:00 23/07/2017 [Invoice] WARN [123-gda] Unknown country code

12:01 23/07/2017 [Warehouse] ERROR [123-gda] Invalid country code



Time varies between machines!

You can't infer causality from time
in logs

Operating
Systems

R. Stockton Gaines
Editor

Time, Clocks, and the Ordering of Events in a Distributed System

Leslie Lamport
Massachusetts Computer Associates, Inc.

The concept of one event happening before another in a distributed system is examined, and is shown to define a partial ordering of the events. A distributed algorithm is given for synchronizing a system of logical

A distributed system consists of a collection of distinct processes which are spatially separated, and which communicate with one another by exchanging messages. A network of interconnected computers, such as the ARPA net, is a distributed system. A single computer can also be viewed as a distributed system in which the central control unit, the memory units, and the input-output channels are separate processes. A system is distributed if the message transmission delay is not negligible compared to the time between events in a single process.

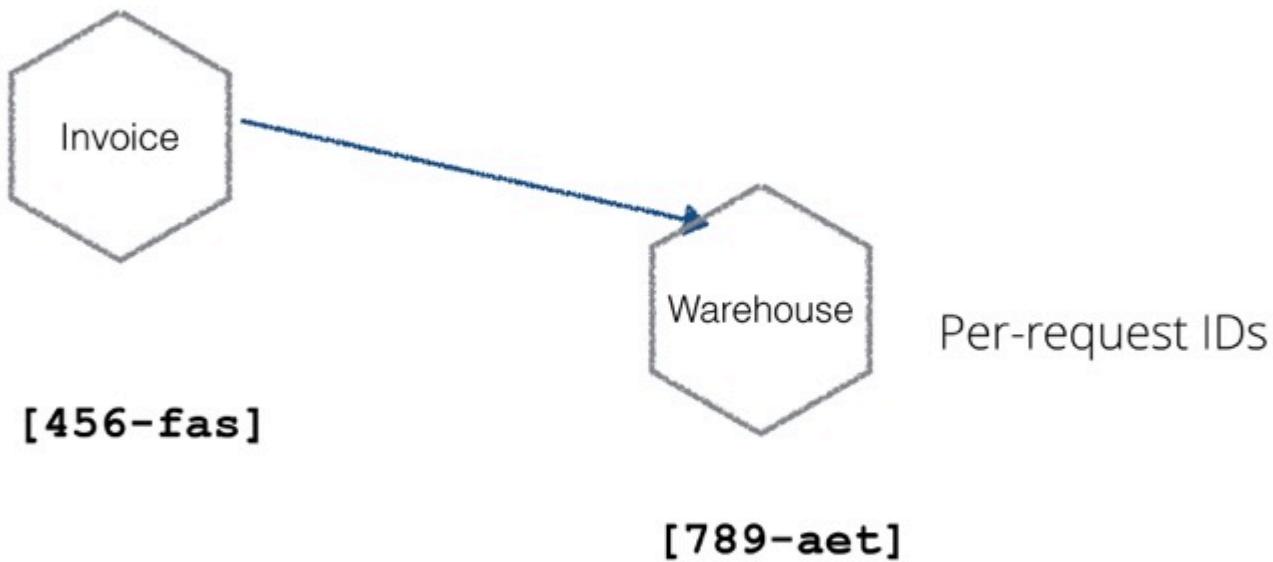
We will concern ourselves primarily with systems of spatially separated computers. However, many of our remarks will apply more generally. In particular, a multiprocessing system on a single computer involves problems similar to those of a distributed system because of the unpredictable order in which certain events can occur.

In a distributed system, it is sometimes impossible to say that one of two events occurred first. The relation "happened before" is therefore only a partial ordering of the events in the system. We have found that problems often arise because people are not fully aware of the

<http://amturing.acm.org/p558-lamport.pdf>

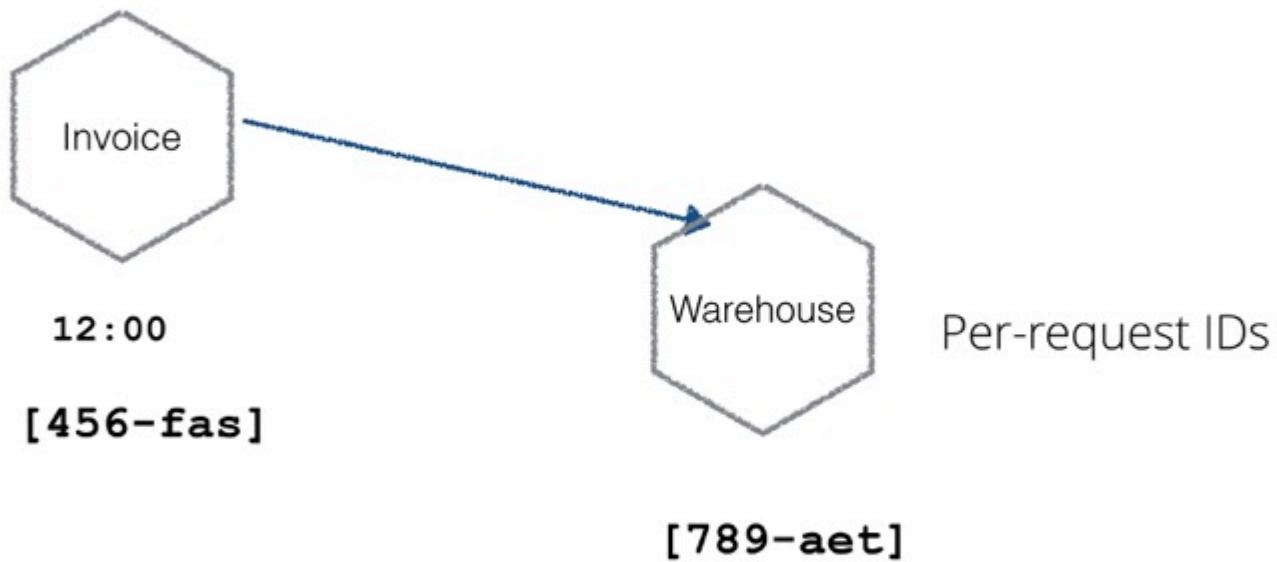
'Global' Correlation ID for whole operation

[123-gda]



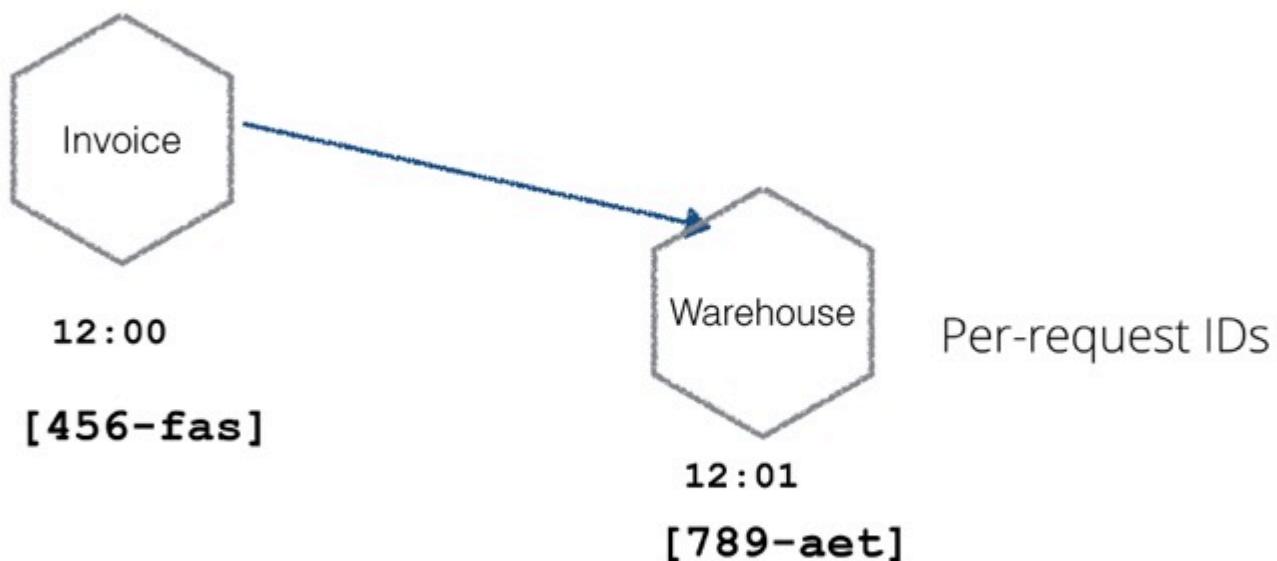
'Global' Correlation ID for whole operation

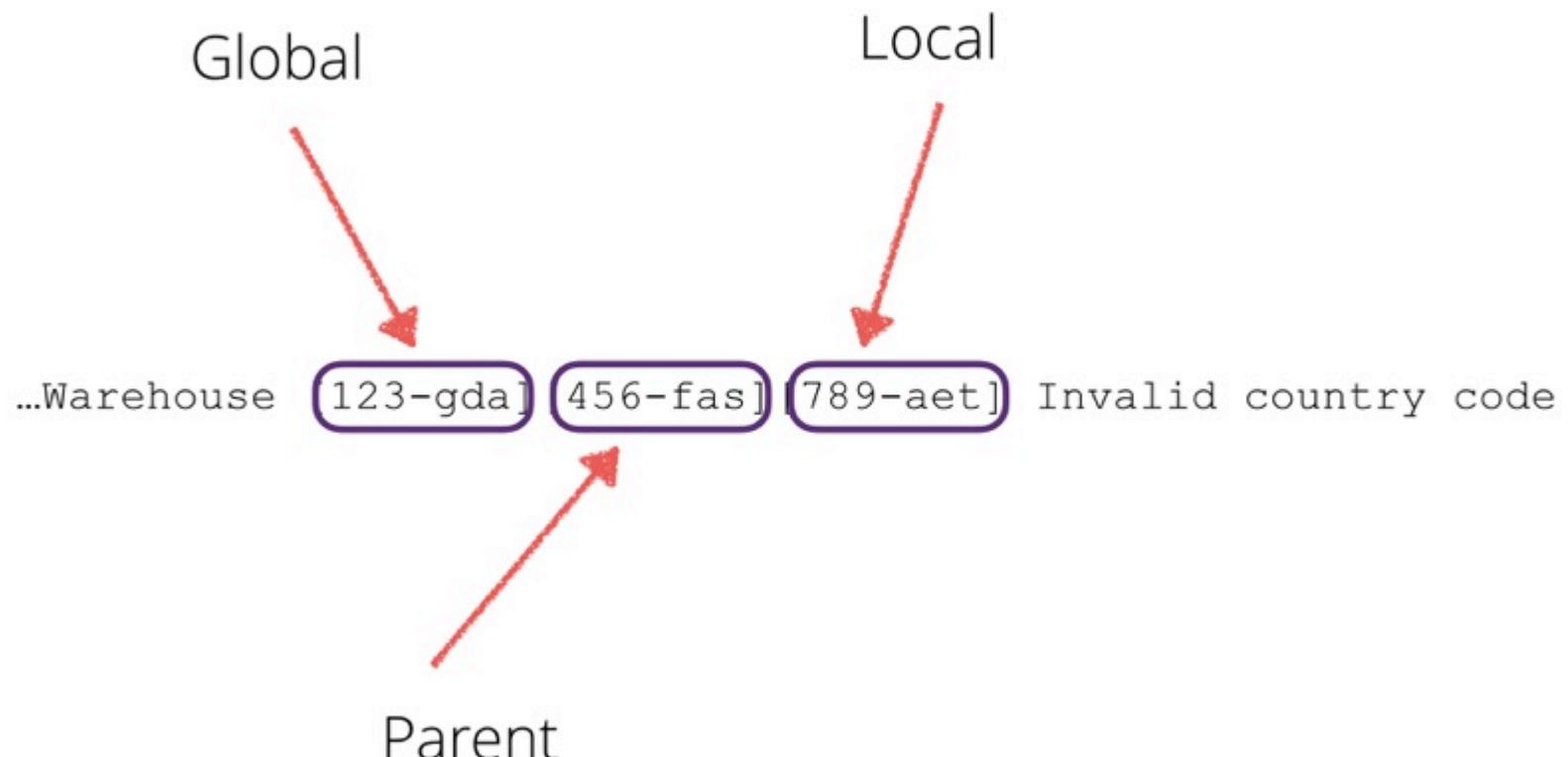
[123-gda]



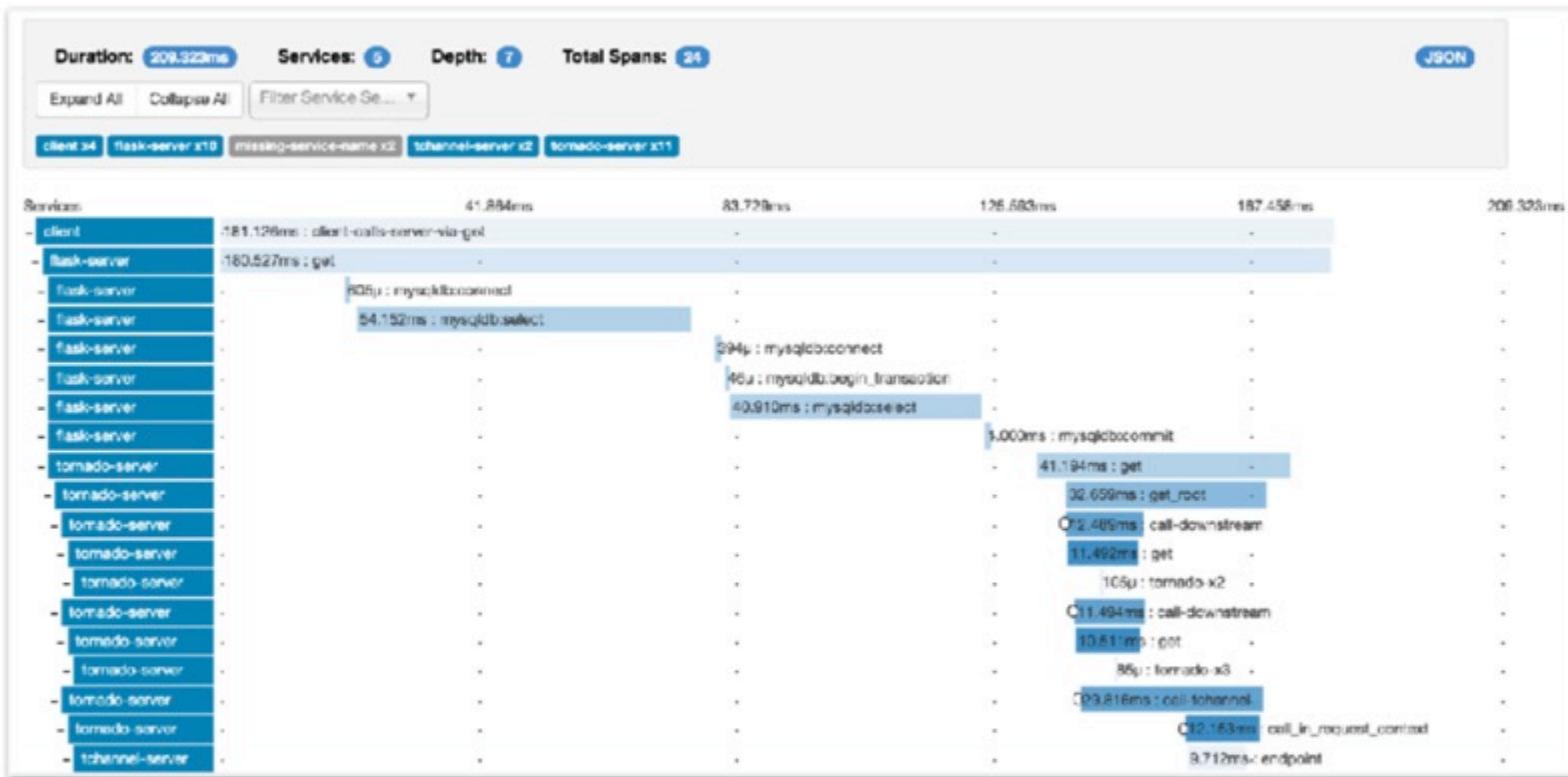
'Global' Correlation ID for whole operation

[123-gda]



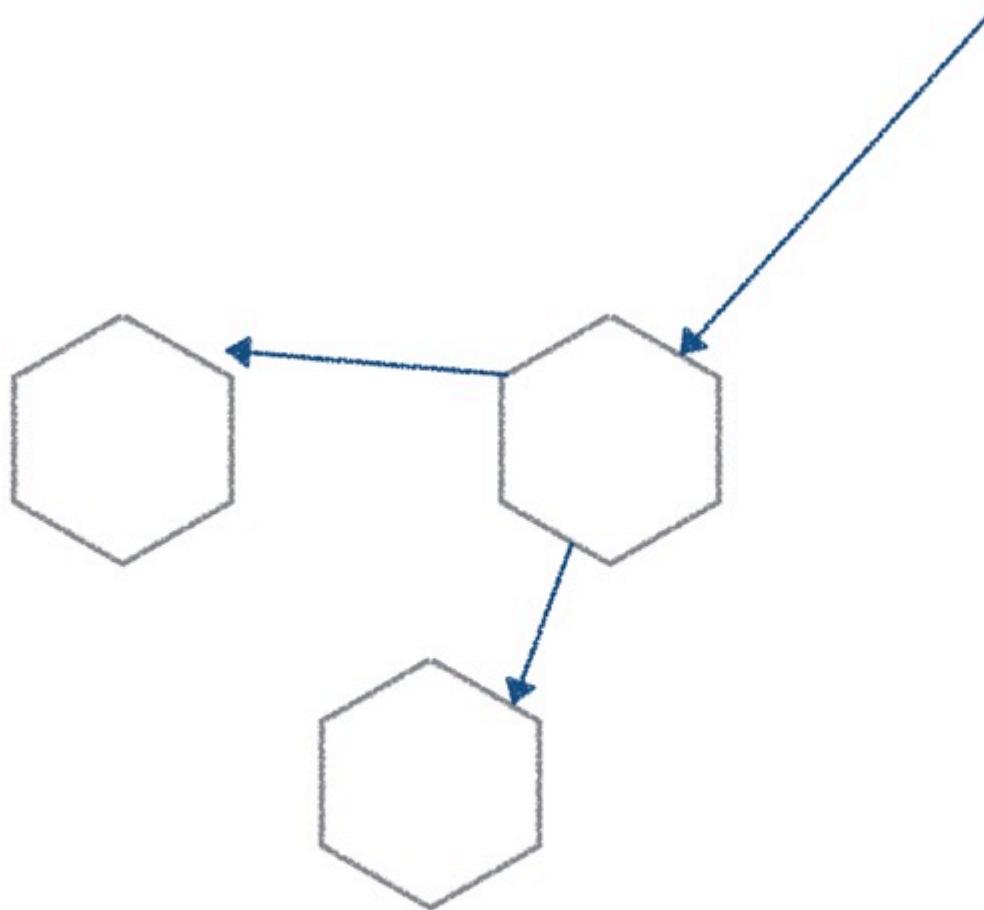


Other mechanisms available

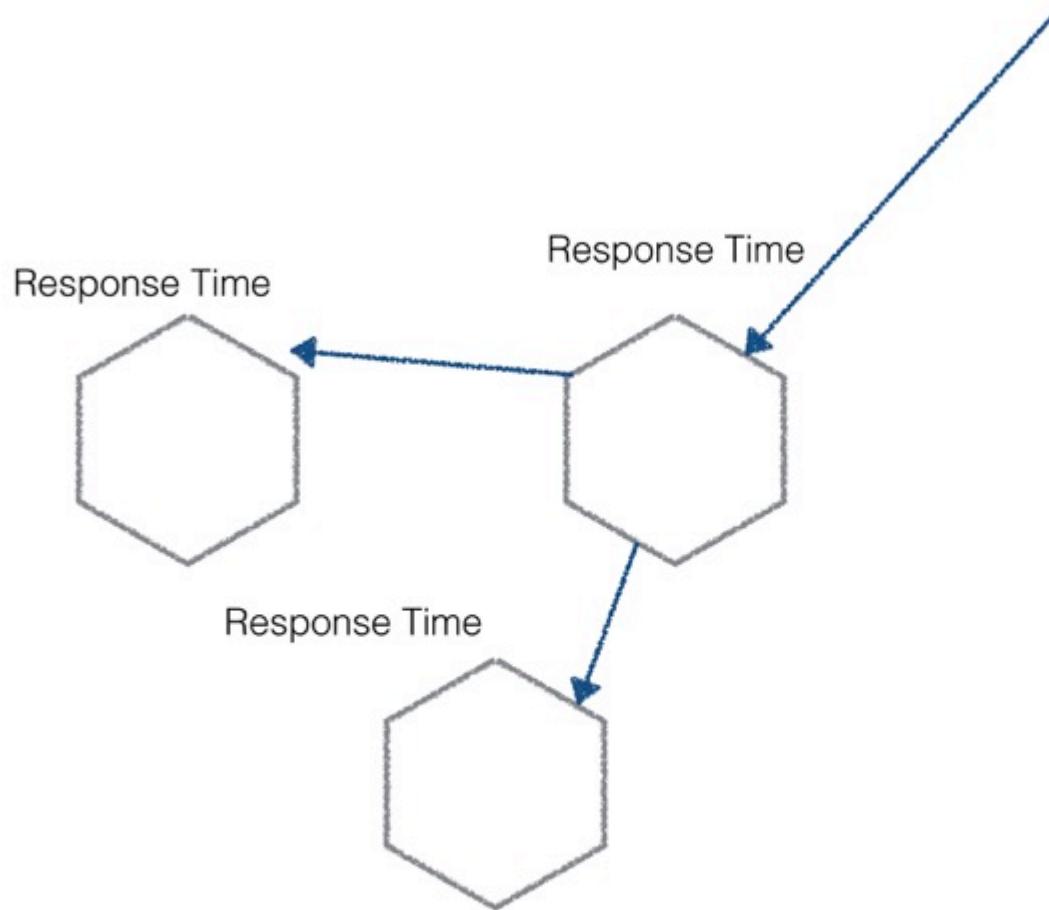


<http://zipkin.io>

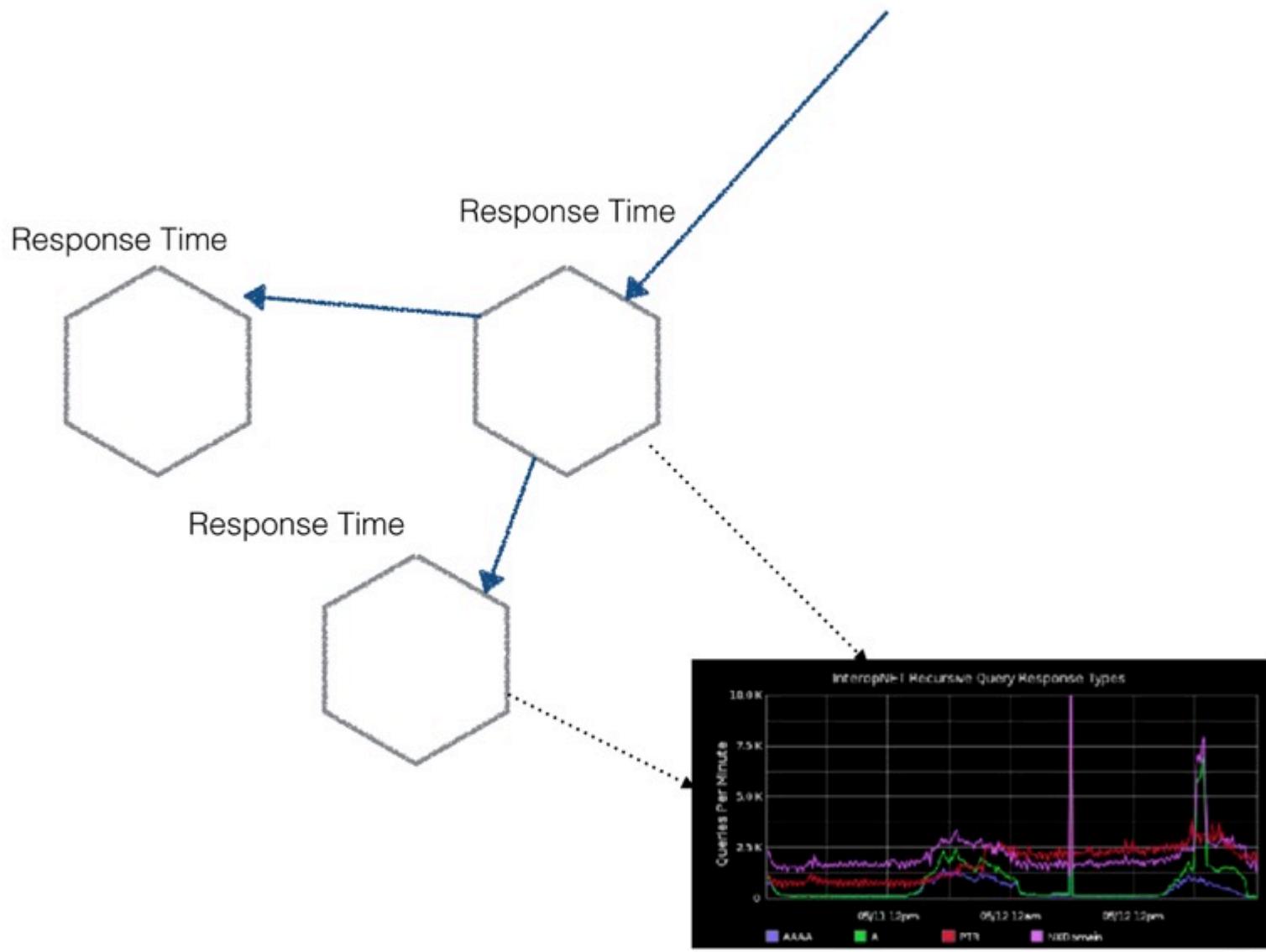
AGGREGATE METRICS



AGGREGATE METRICS



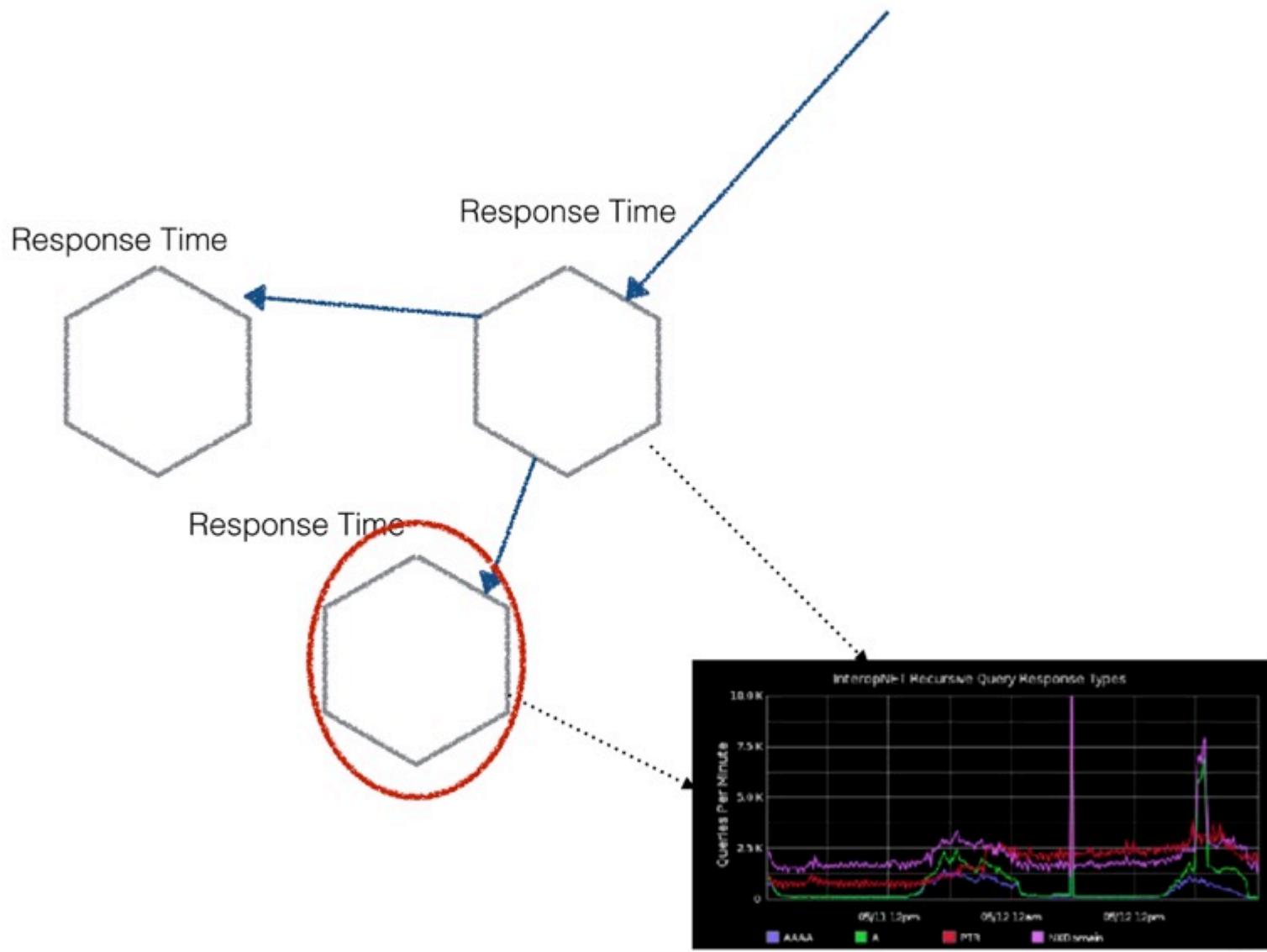
AGGREGATE METRICS



26

342

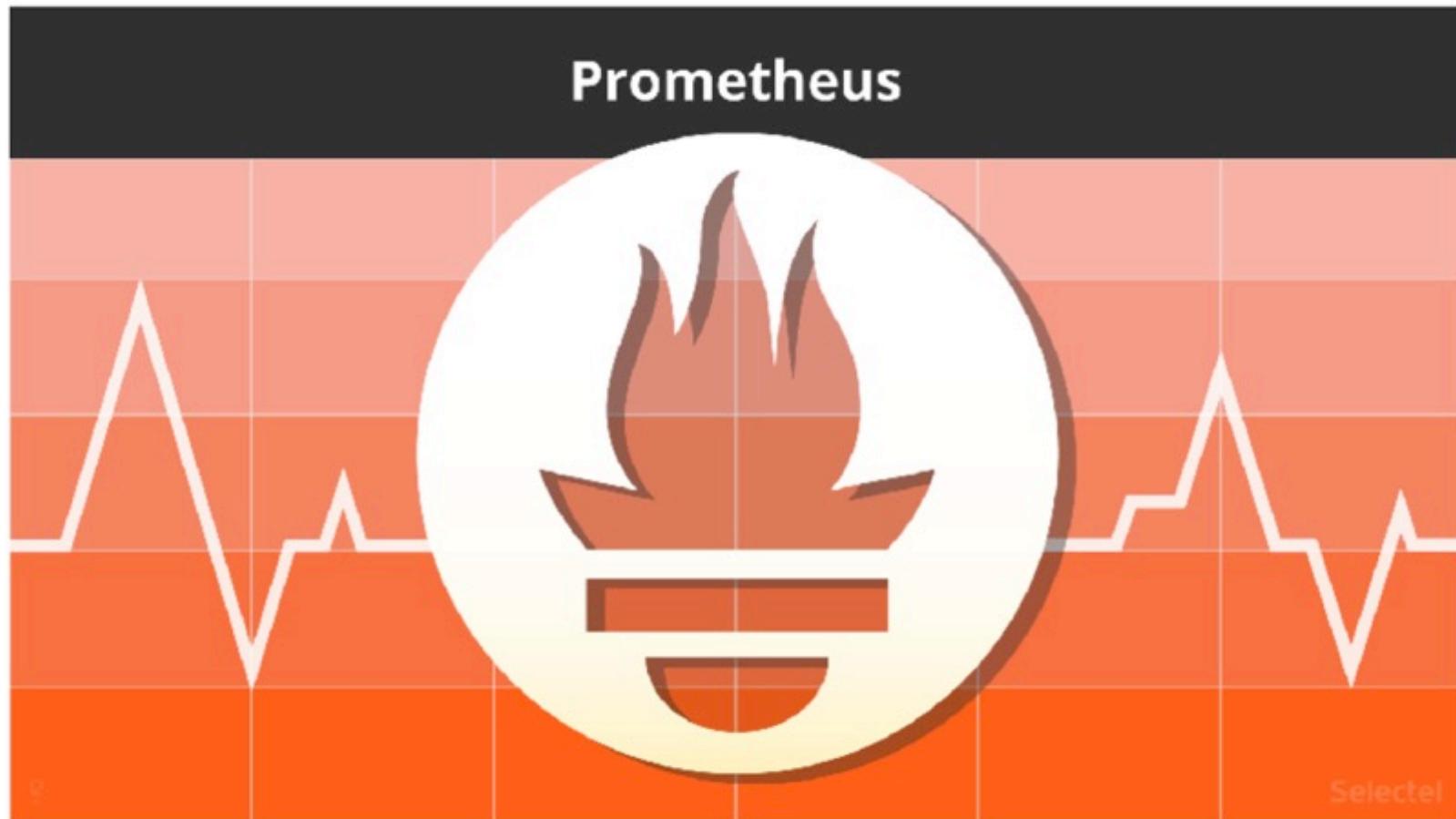
AGGREGATE METRICS



26

343

Prometheus

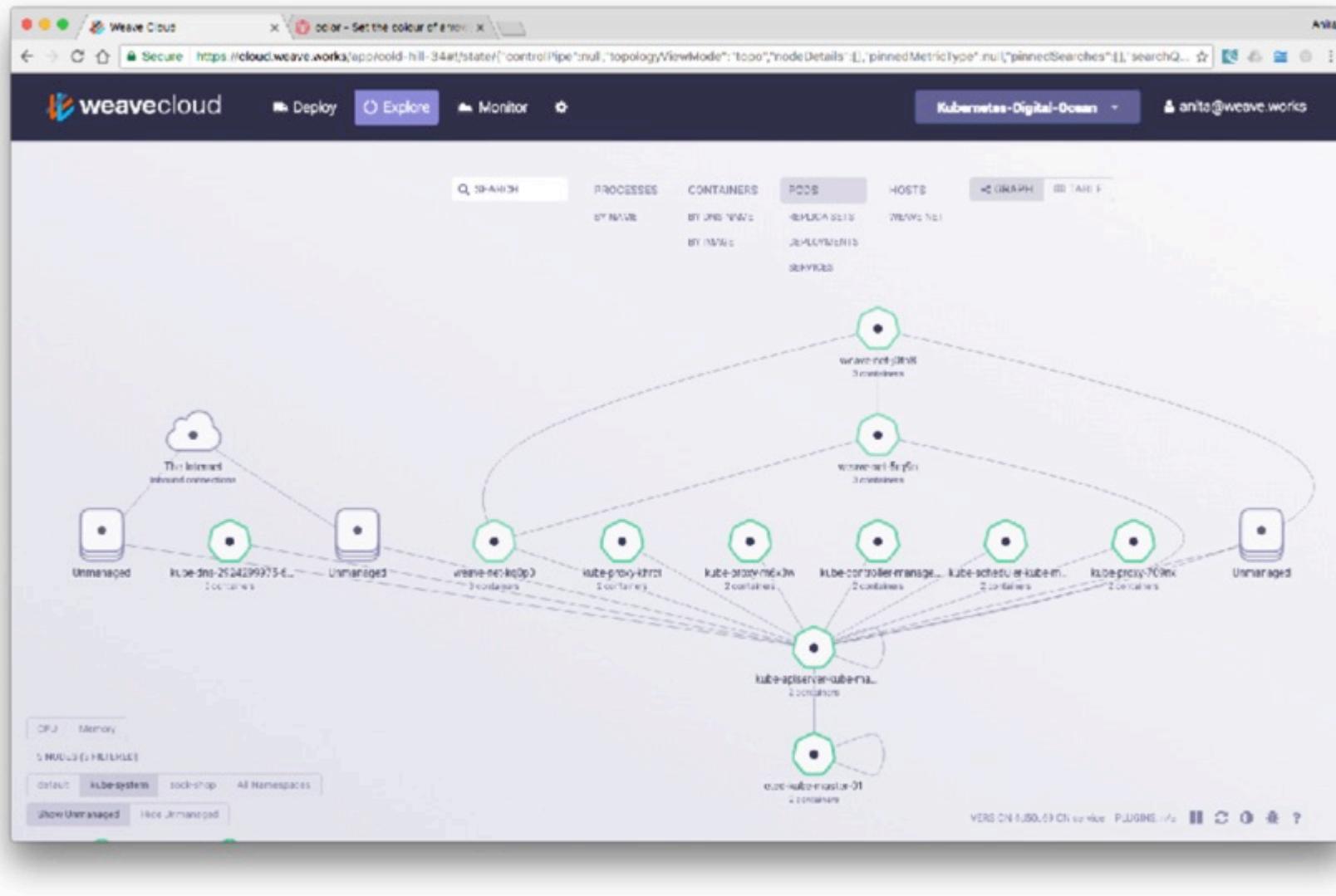


<https://prometheus.io>

The image shows a screenshot of the Weaveworks website. At the top left is the Weaveworks logo. On the right is a three-line menu icon. Below the header is a large video thumbnail showing a person working at a computer with multiple screens displaying code and monitoring data. Overlaid on the video is the text: "Monitor your entire environment and quickly pinpoint issues". A blue button labeled "WATCH DEMO" is visible. The main content area below the video contains the text: "No matter how **dynamic** your microservices environment, Weave Cloud's hosted, horizontally **scalable** Prometheus service lets you **quickly** identify issues with your app."

No matter how **dynamic** your microservices environment, Weave Cloud's hosted, horizontally **scalable** Prometheus service lets you **quickly** identify issues with your app.

<https://www.weave.works/solution/prometheus-monitoring/>





SEMANTIC MONITORING

Don't look for presence of errors

SEMANTIC MONITORING

Don't look for presence of errors

Define a model for a correctly
operating system

SEMANTIC MONITORING

Don't look for presence of errors

Define a model for a correctly
operating system

Alert if your system doesn't behave
correctly

SEMANTIC MONITORING - EXAMPLE STATEMENTS

People can order CDs

SEMANTIC MONITORING - EXAMPLE STATEMENTS

People can order CDs

We are making at least \$30K per hour

SEMANTIC MONITORING - EXAMPLE STATEMENTS

People can order CDs

We are making at least \$30K per hour

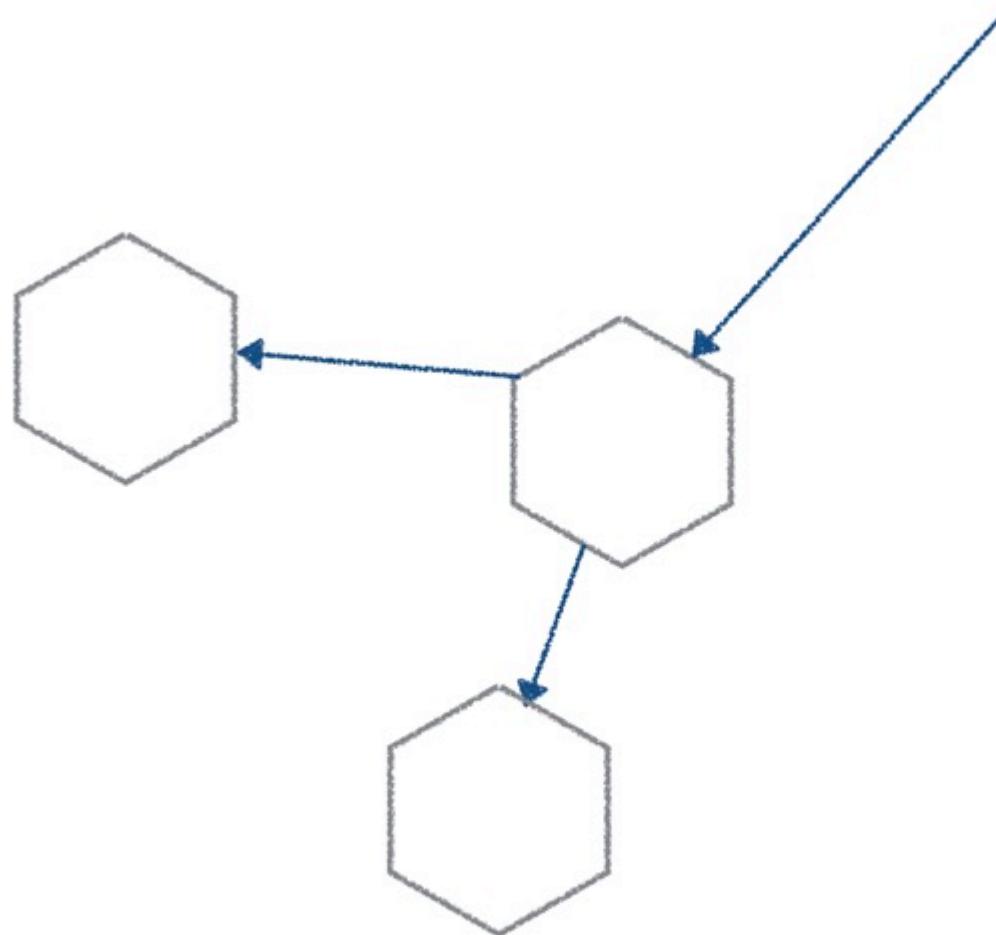
New users can still register

REAL USER MONITORING

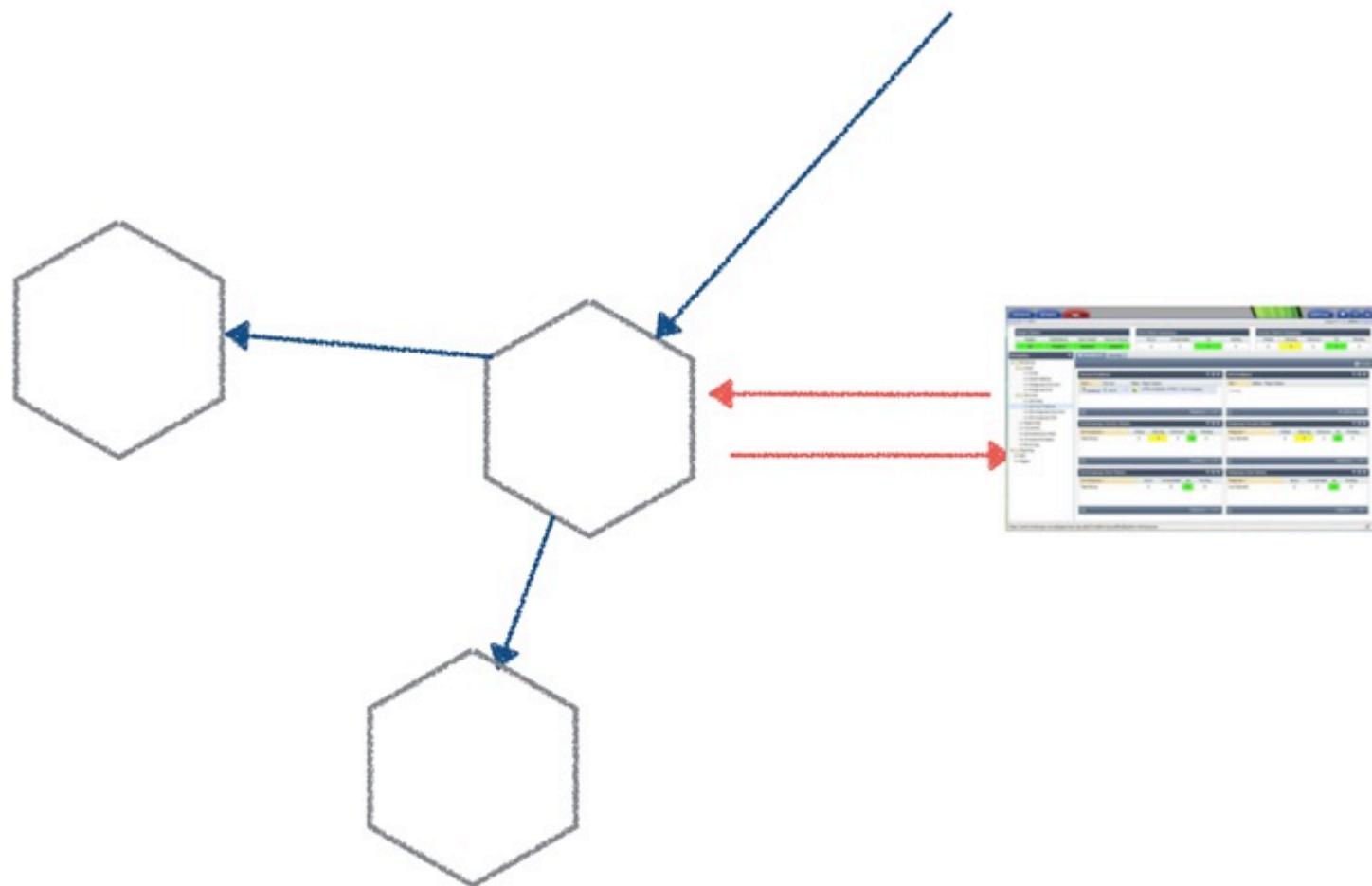


Use metrics to validate the model

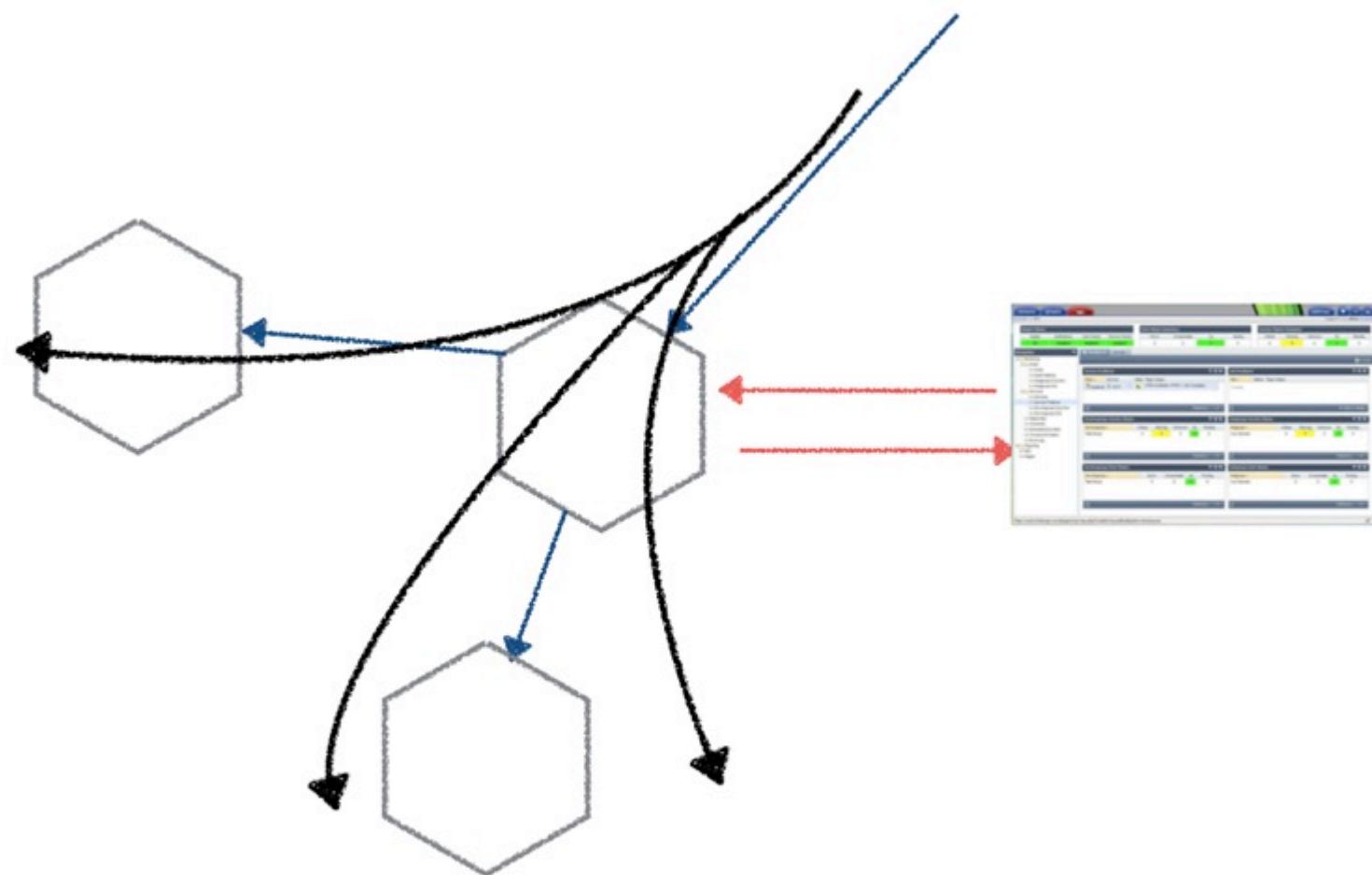
SYNTHETIC TRANSACTIONS



SYNTHETIC TRANSACTIONS



SYNTHETIC TRANSACTIONS





<https://www.flickr.com/photos/yercombe/9033732392/>

358