

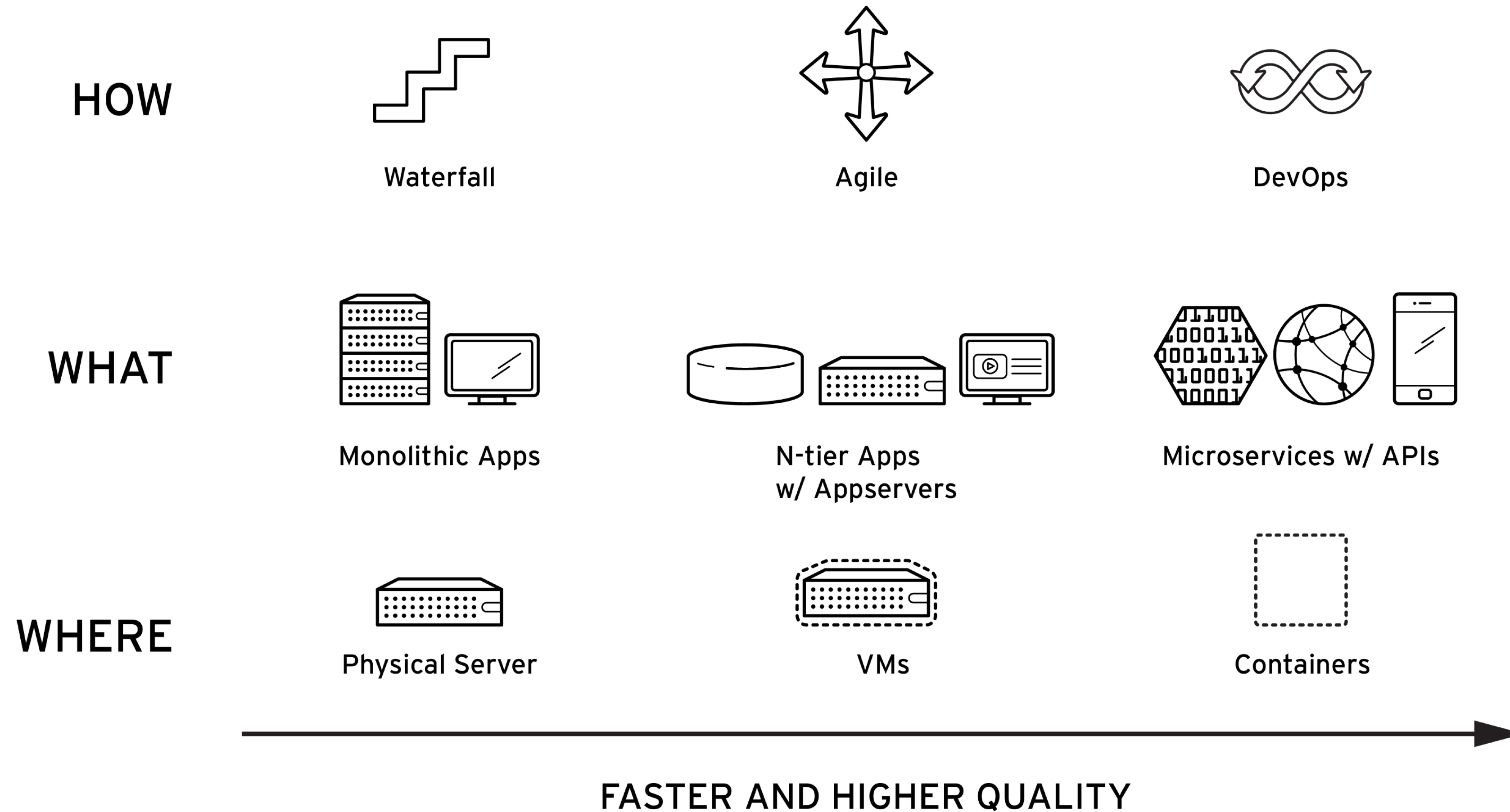
# A DEVOPS STATE OF MIND

Chris Van Tuin  
Chief Technologist, West  
[cvantuin@redhat.com](mailto:cvantuin@redhat.com)



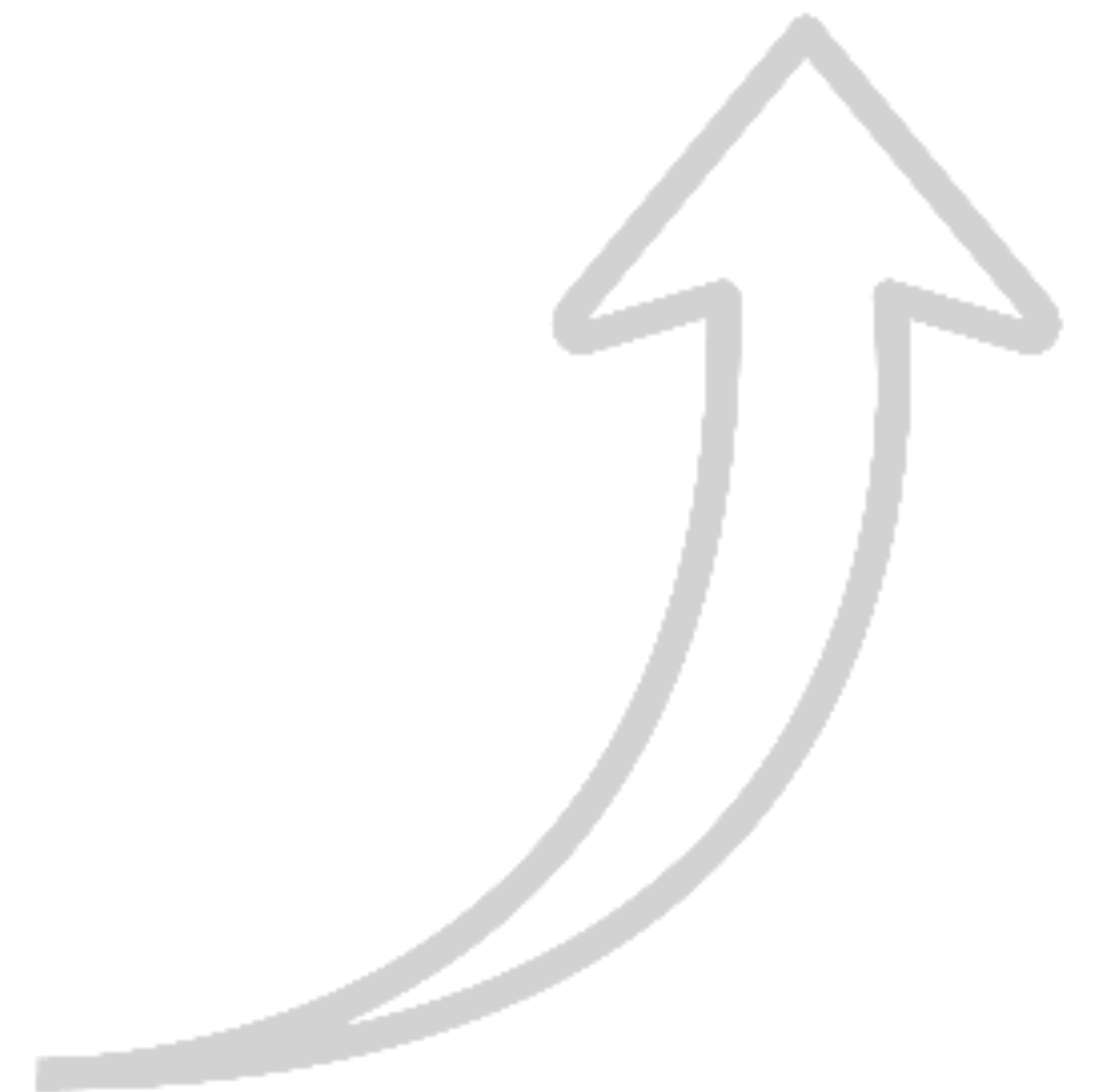
# THE NEED FOR SPEED

## THE ACCELERATION OF APPLICATION DELIVERY FOR THE BUSINESS



“In short, software is eating the world.”

- Marc Andreessen, Wall Street Journal, August 2011





# BUT DEMANDS ON I.T. ARE INCREASING AS BUSINESSES ARE REIMAGINED



Online, Mobile



Software Defined Networks

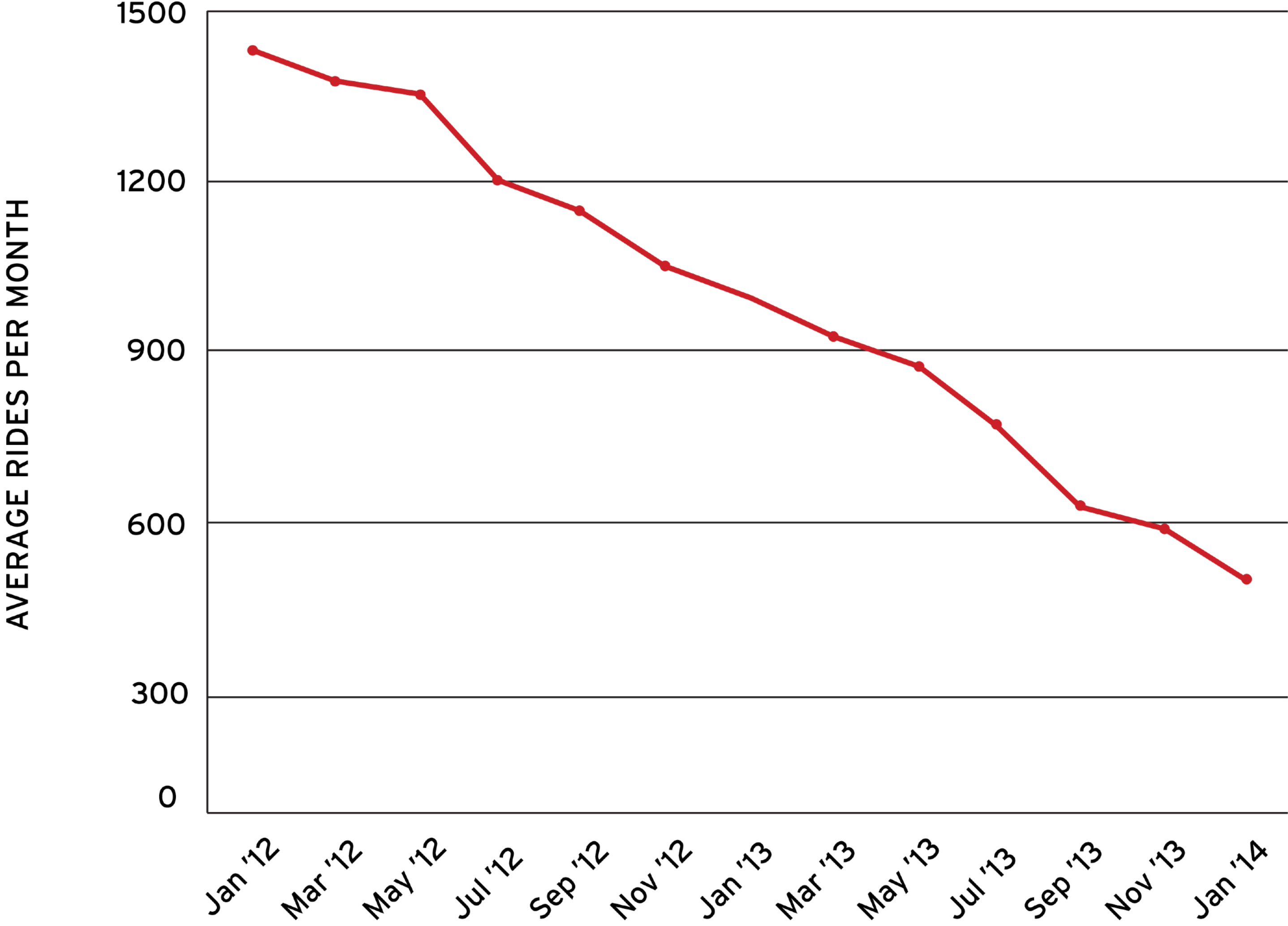


Desktop to Cloud

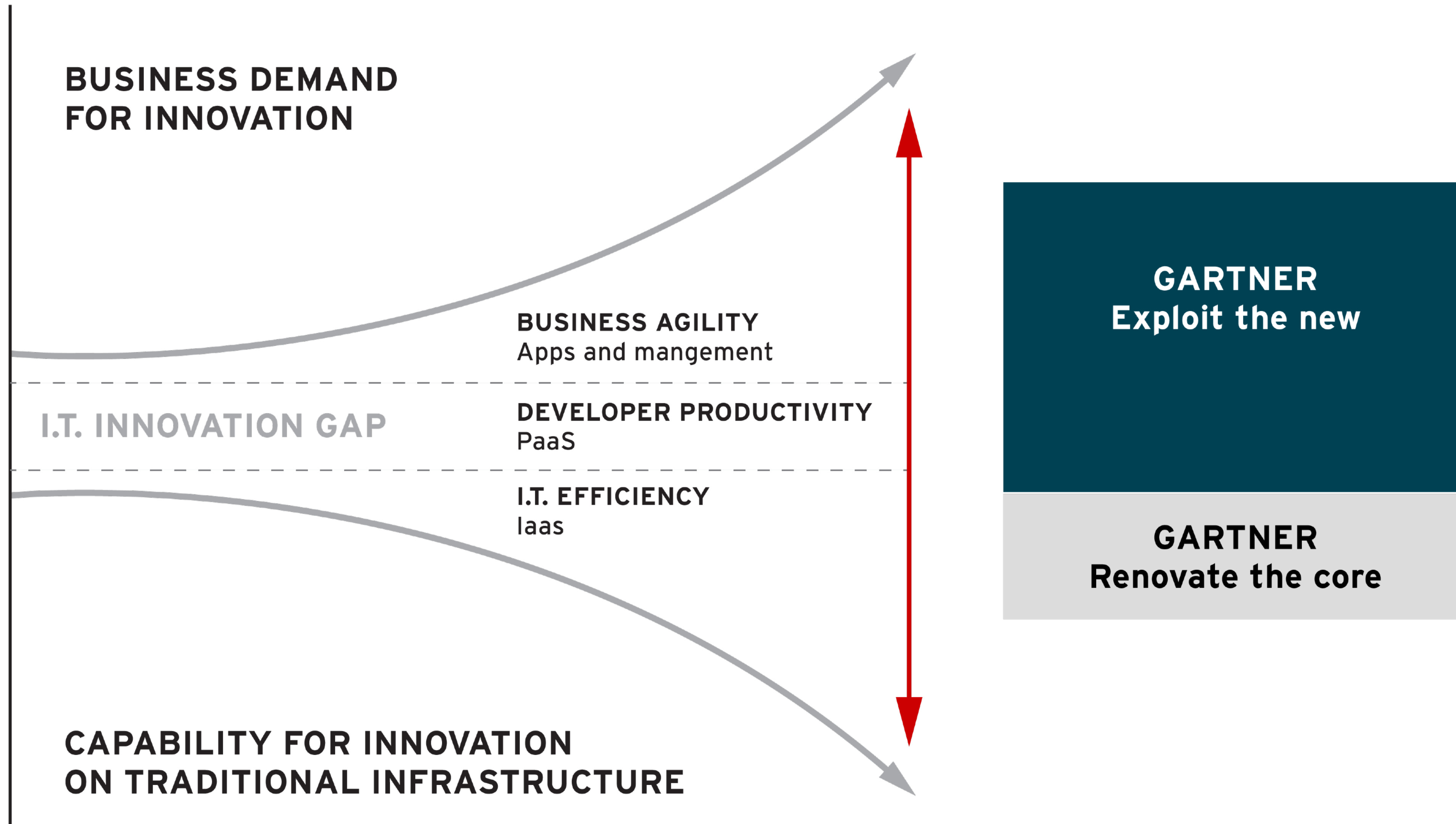


Online, Mobile

# UBER, LYFT FALLOUT: TAXI RIDES PLUNGE 65% IN SAN FRANCISCO



# CREATES AN I.T. INNOVATION GAP



# DELIVERING SOFTWARE TODAY: THE REALITY VS. THE GOAL

**BUDGET**

**45%**  
OVER

**TIME**

**7%**  
OVER

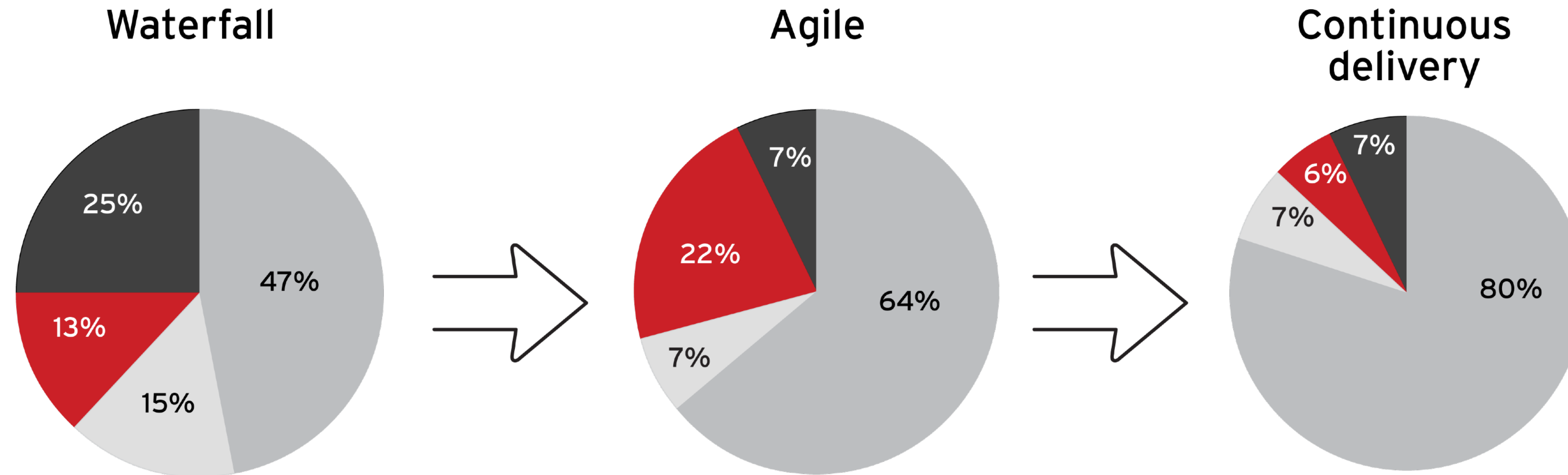
**VALUE**

**56%**  
LESS

Source: Delivering large-scale IT projects on time, on budget, and on value  
McKinsey & Company, October 2012



# DEVELOPMENT LIFECYCLE TRENDS

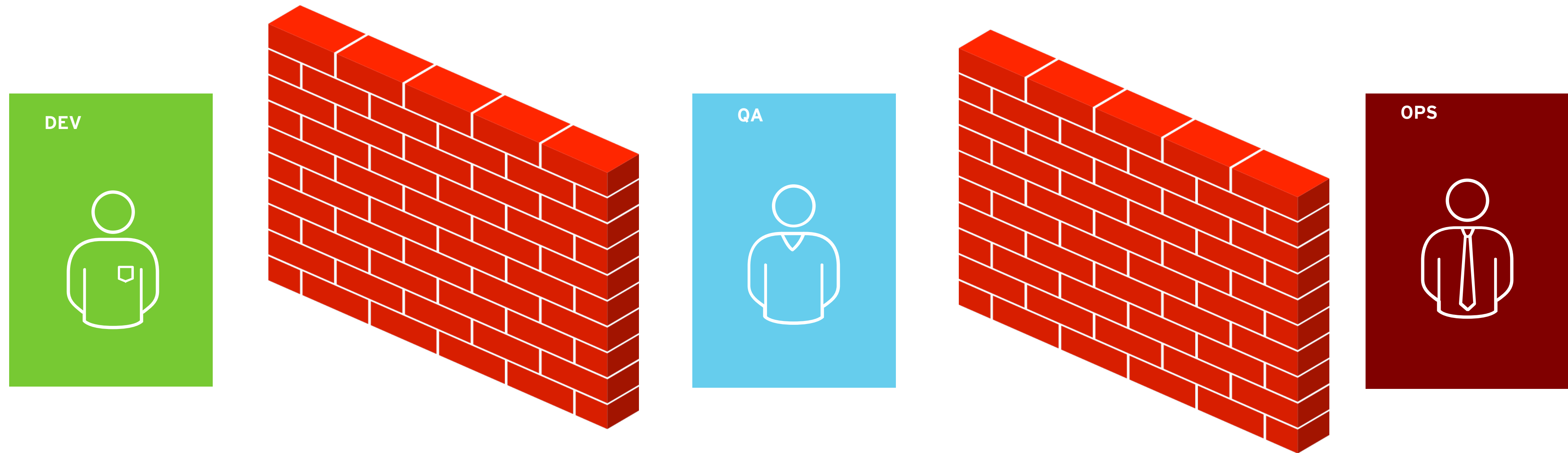


- Design**  
Design and architecture
- Development**  
Development and documentation
- Test**  
Automated and manual tests
- Deployment**  
Deployment to test and production

**More time spent on development**

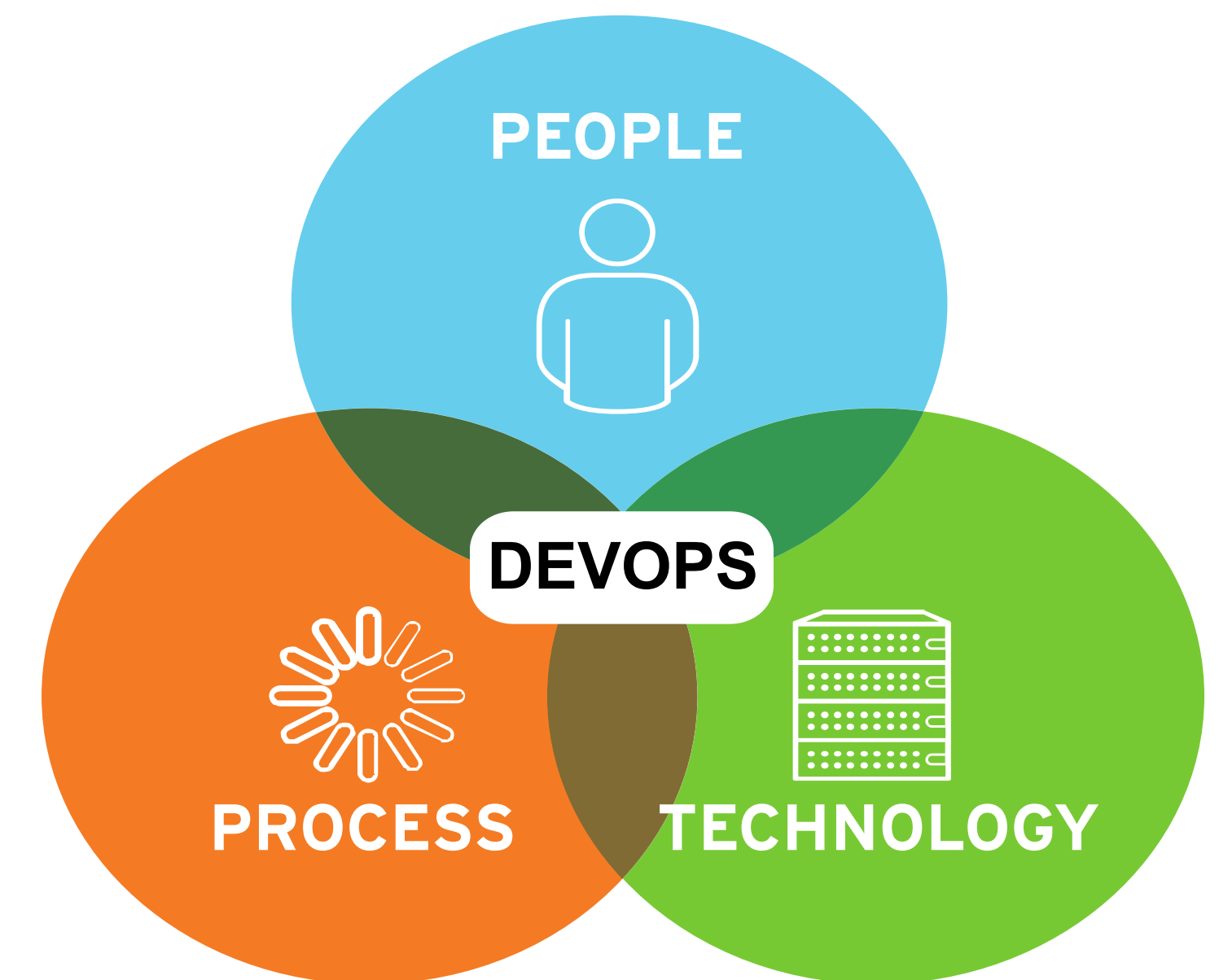
# “THROW IT OVER THE WALL”

Walled off people, walled off processes, walled off technologies



# WHAT CAN I.T. DO? I.T. CAN TURN OPS AND DEV INTO DEVOPS

“DevOps is a software development method that stresses communication, collaboration and integration between software developers and information technology (IT) professionals.”[1]



Applying many of the principles of Agile software development to the full application lifecycle and incorporating automation and monitoring with just a touch of Lean Manufacturing theory.

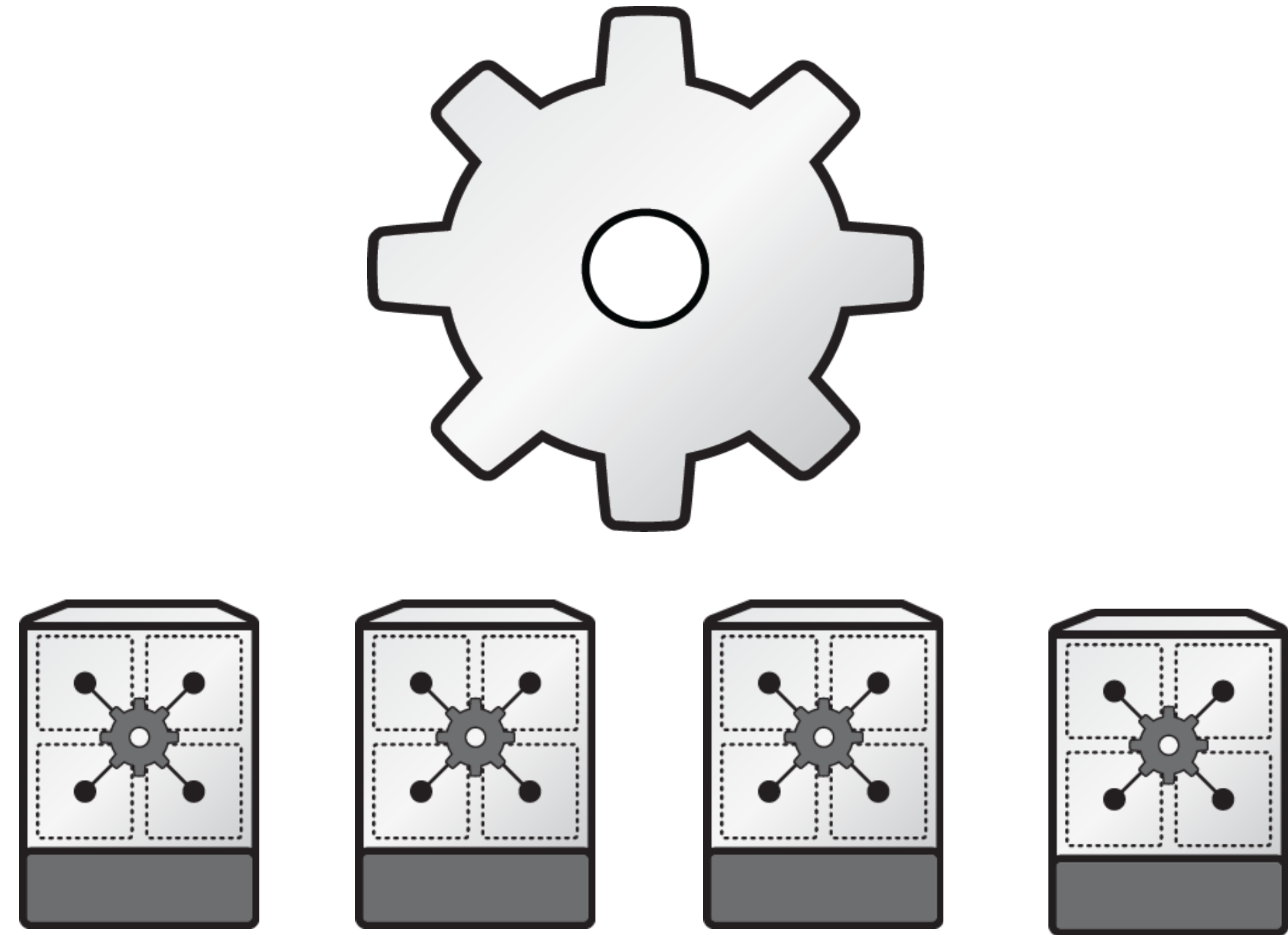
# COLLABORATION



# WHAT ENABLES DEVOPS?

## “Configuration in code”

- Standardized environments
- Linux containers
- Automated provisioning

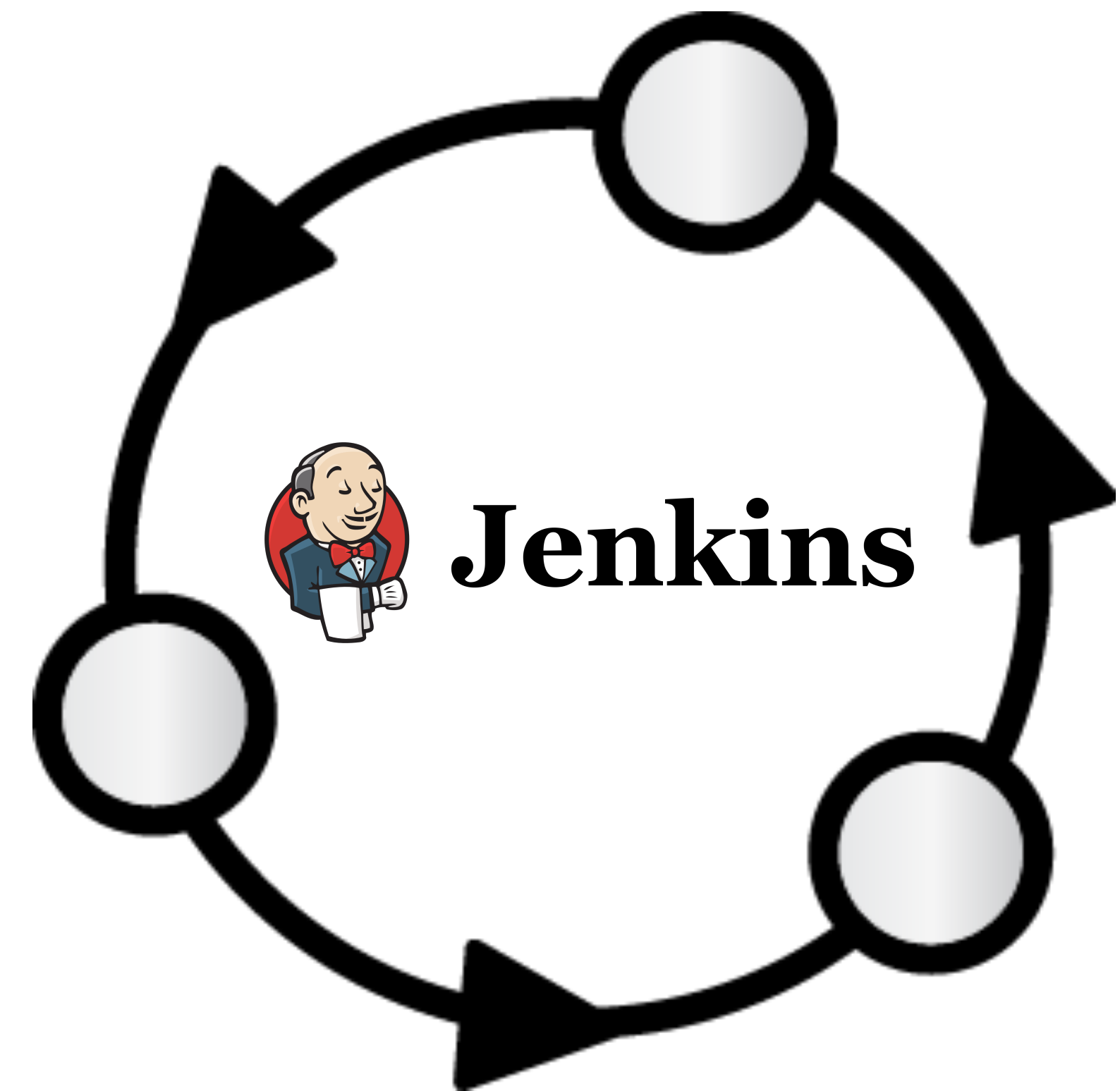


**KEY CONCEPT: FAIL FAST AND RECOVER VS. NEVER FAIL**

# WHAT ENABLES DEVOPS?

## CI/CD: Automated testing and deployment

- Continuous integration
- Continuous delivery

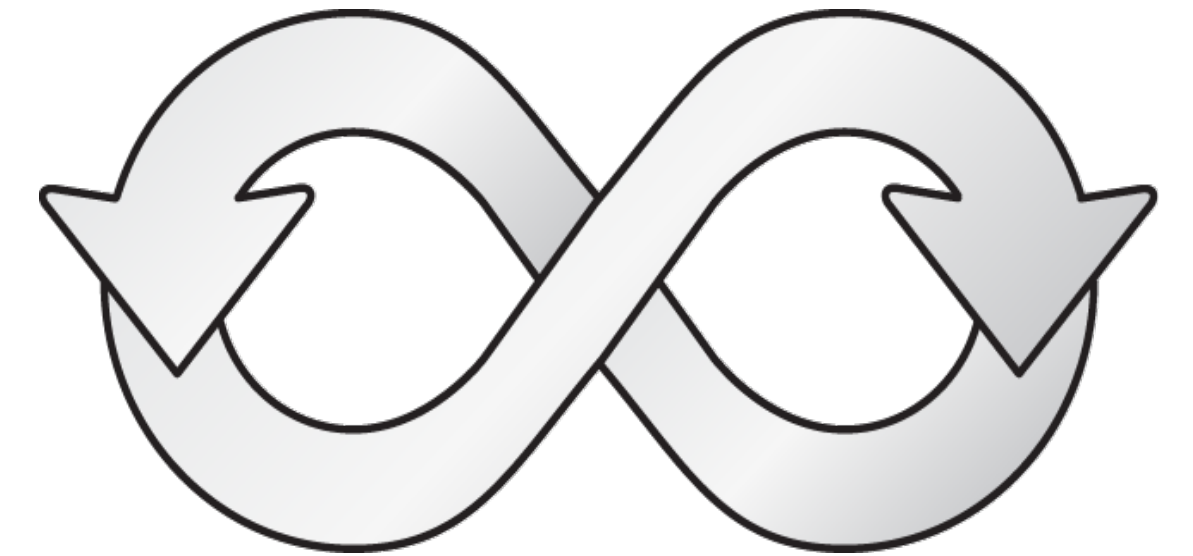


**KEY CONCEPT: SMALL CHANGES = LESS RISK**

# WHAT ENABLES DEVOPS?

## Continuous innovation

- Developer self-service
- Rapid prototyping



**KEY CONCEPT: CULTURE CHANGE = ACCEPTANCE OF FAILURE**

# ORGANIZATIONS IMPLEMENTING DEVOPS

**Better deployment quality**

**63%**

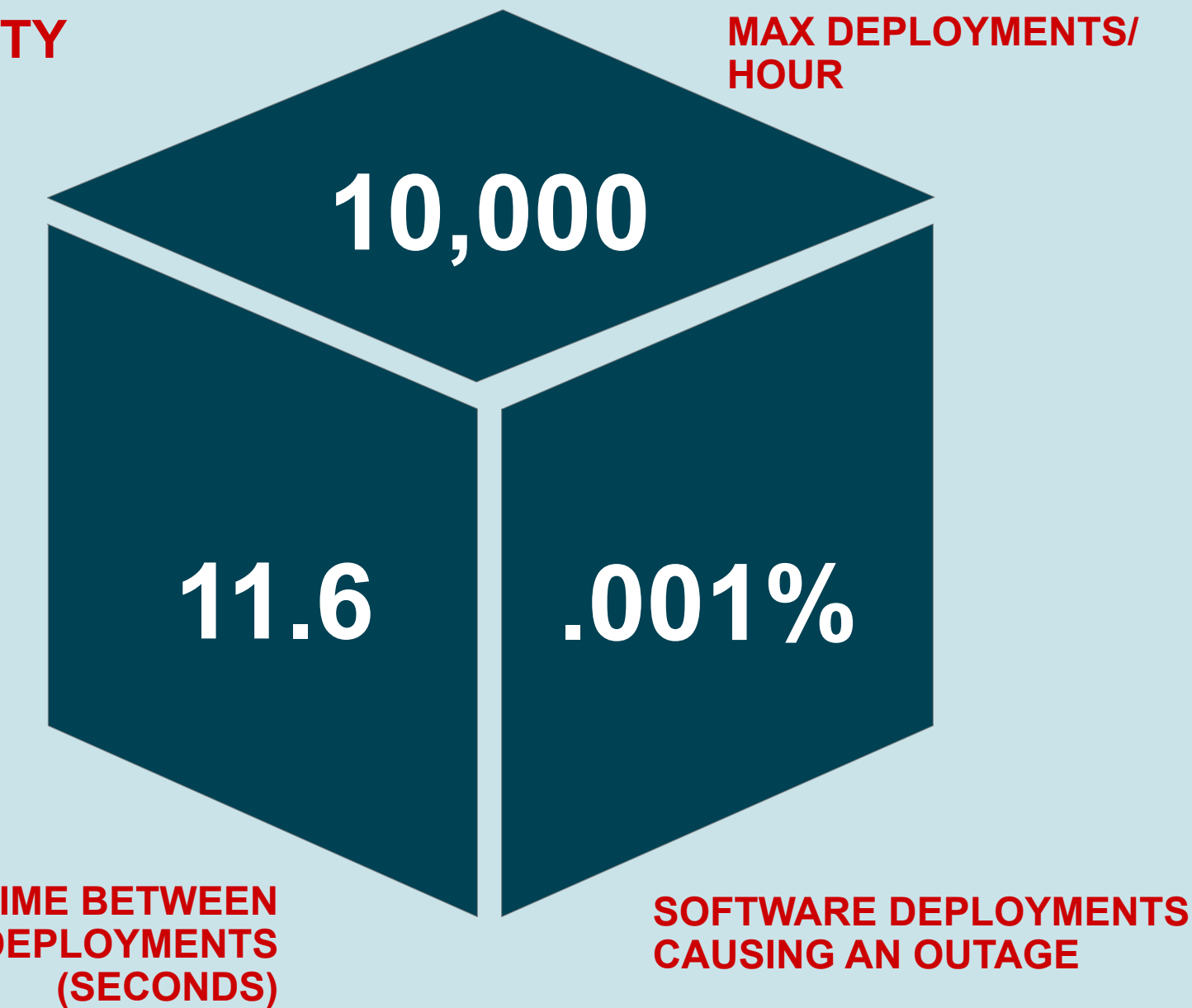
**Faster release frequency**

**63%**

**Improved process visibility**

**61%**

**DEVOPS VALUE  
IN ACTION: VELOCITY  
AT AMAZON AWS**



Source: 2014 State of DevOps Report  
Puppet Labs, IT Revolution Press, ThoughtWorks



# ORGANIZATIONS IMPLEMENTING DEVOPS

“**30** innovations to the website deployed each day, ... sometimes adding **millions** of dollars in sales”  
Forbes, Apr'14

Etsy

“Taking a system that required a **full month** to release new features and turning it into one that pushes out updates **multiple times per day.**”  
Wired



# THREE KEY QUESTIONS FOR I.T. AND BUSINESS

1

How to quickly and reliably deliver new capabilities?

2

What kinds of new apps and services to deliver and support?

3

Where to create and run new apps and services?

# DEVOPS IS PART OF A LARGER SHIFT

**HOW?**

**DEVOPS**

**WHAT?**

**CLOUD APPS  
+  
MICROSERVICES**

**WHERE?**

**CONTAINERS**

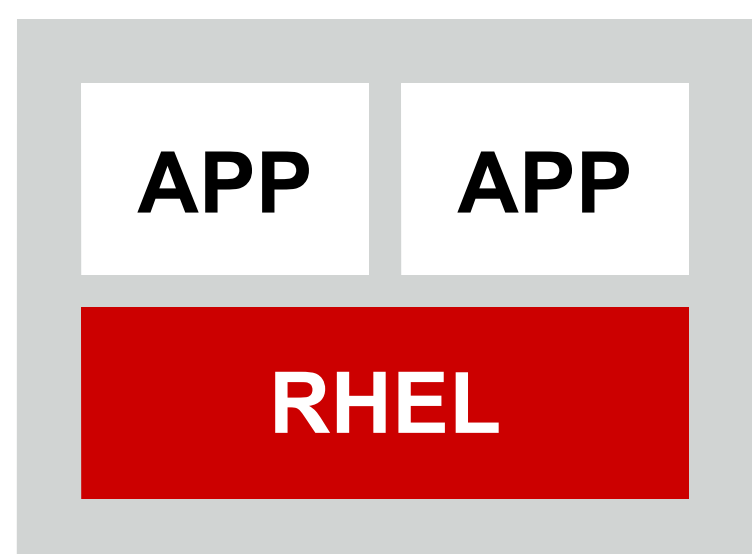
# DEVOPS + CLOUD = INDUSTRIALIZE



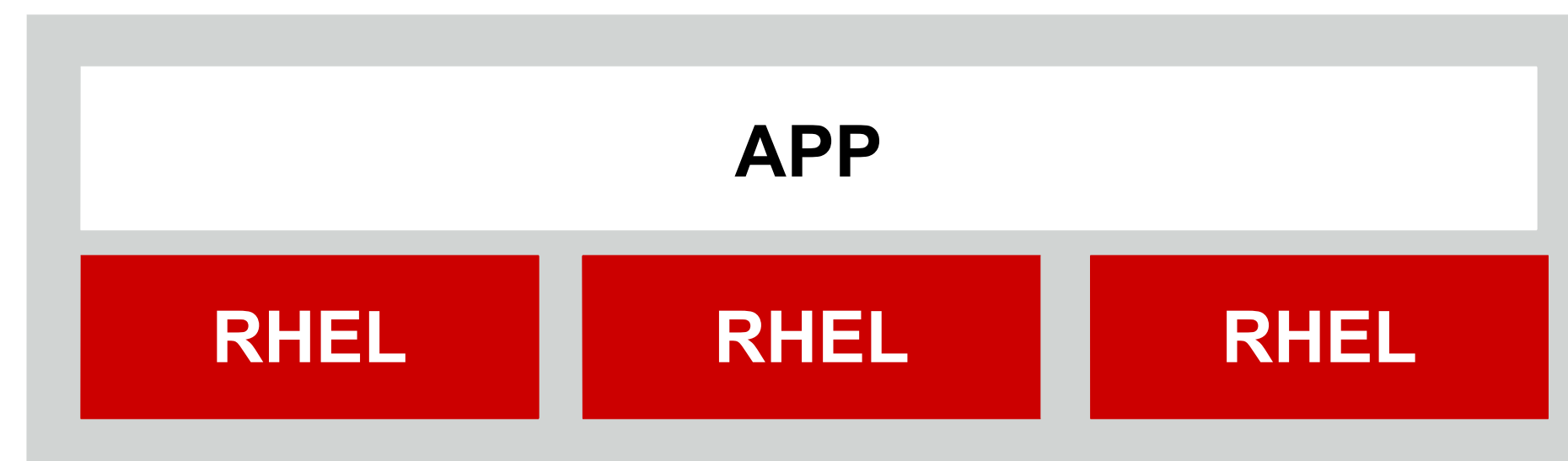
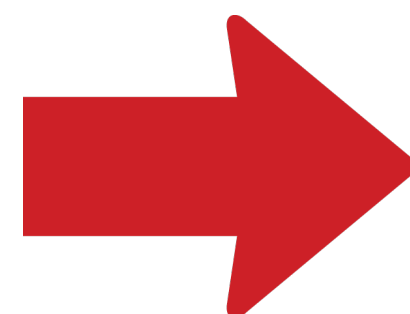
# THE NEW APPLICATION

- Monolithic app container
- Scale up by adding hardware resources
- Limited scale out through clustering

- Distributed, networked, containerized services
- Scale out by orchestrating services
- *Faster iteration and release*
- *More robust*



**SINGLE-HOST APPS**



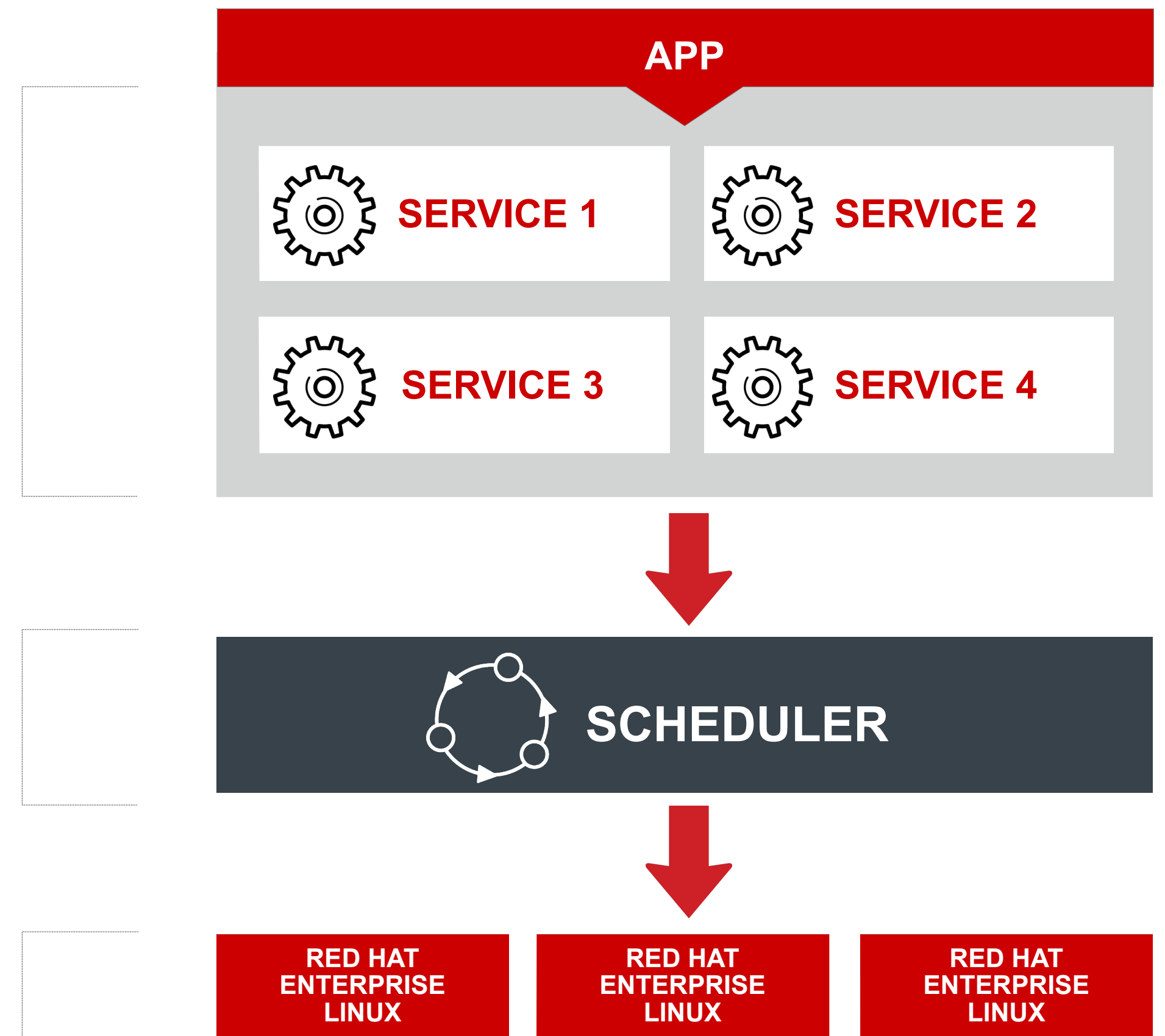
**MULTI-HOST APPS**

# THE NEW OPERATING SYSTEM

Orchestrator (Kubernetes): Model the app across multiple hosts/containers

Scheduler (Kubernetes): Provide service and APIs for placing the app onto resources

Container pool (Red Hat Enterprise Linux/ Docker): Provide resources to run app



# CONTAINERS: BUILD, SHIP, RUN

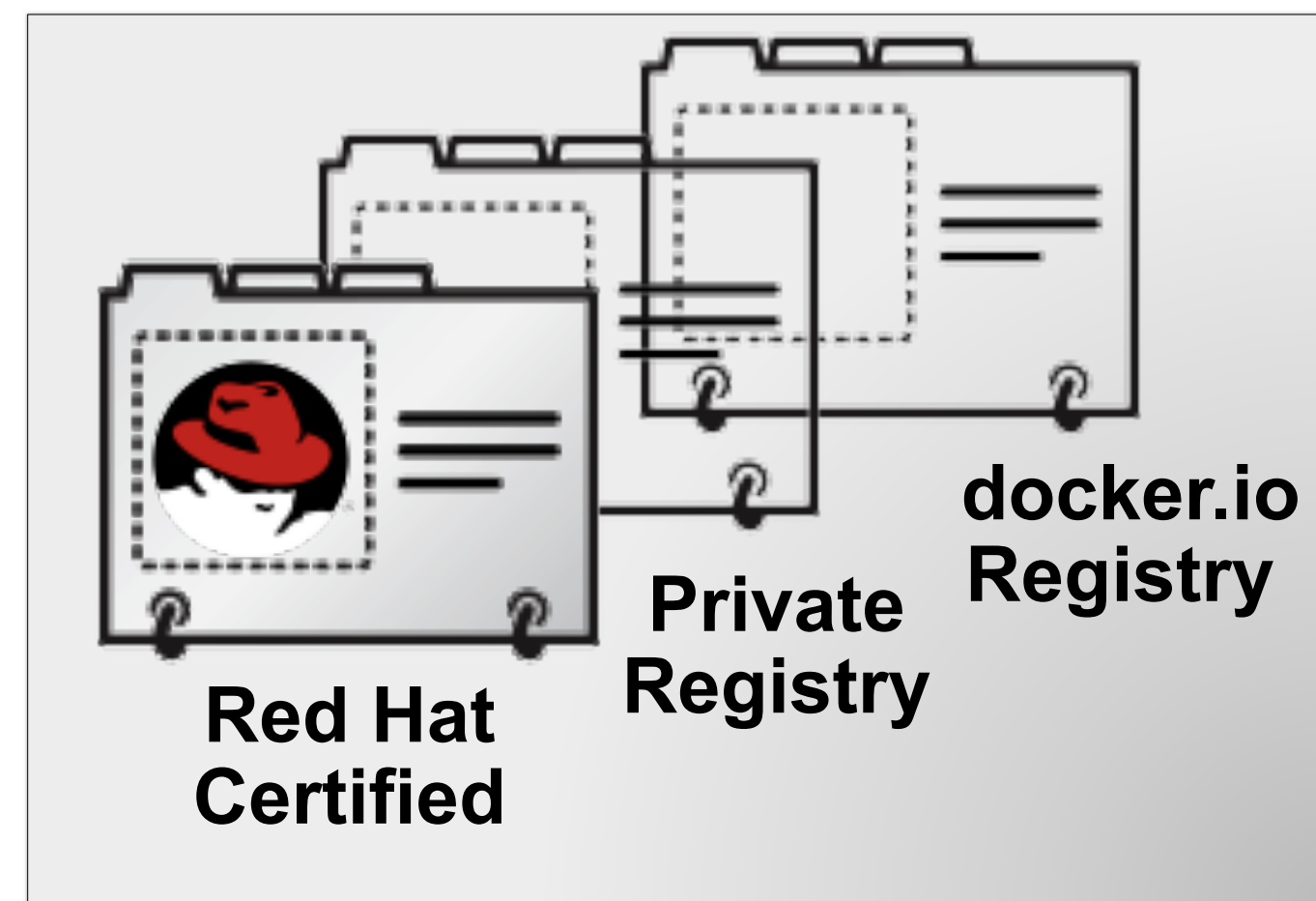
## Dockerfile

```
FROM fedora:latest  
CMD echo "Hello"
```

## Build

“docker build or commit”

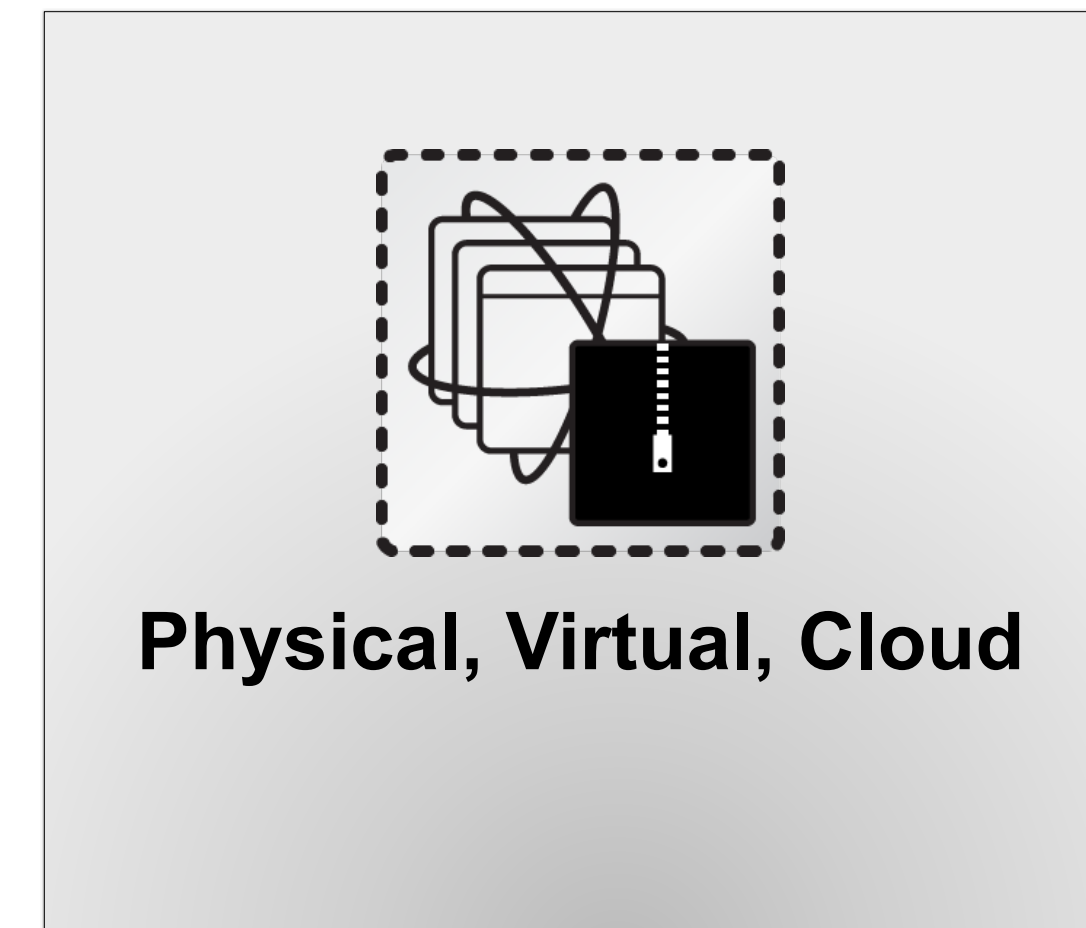
## Image



## Ship

“docker push or pull  
<IMAGE\_ID>”

## Container



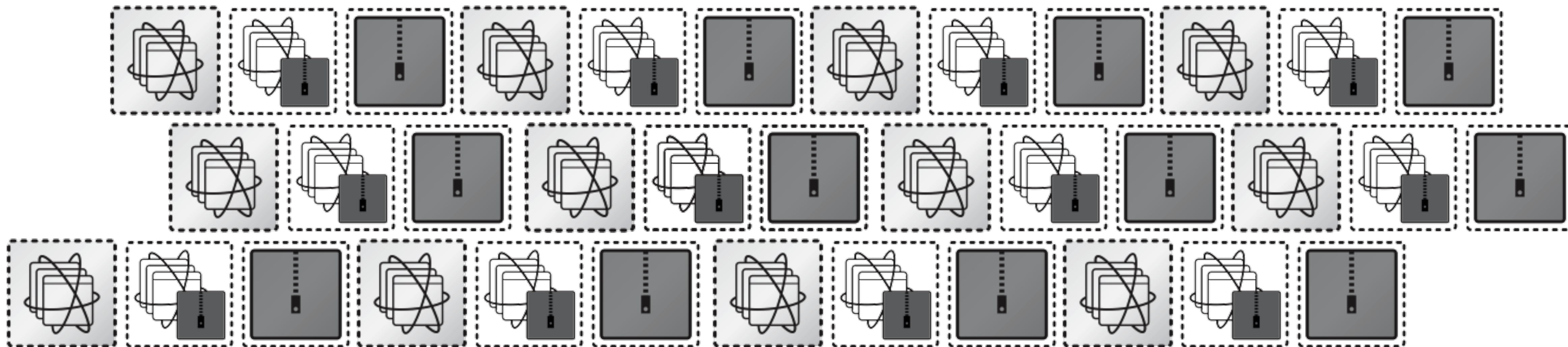
## Run

“docker run  
<IMAGE\_ID>”



# kubernetes

by Google





# KUBERNETES: DEPLOY APPLICATION

The screenshot shows a 'Deploy Application' dialog box overlaid on a blurred background of a Kubernetes management interface. The dialog box contains the following fields:

- Type:** A dropdown menu with 'Kubernetes Manifest' selected.
- Manifest:** A text input field containing 'k8s-sample-app.json'.
- Namespace:** A dropdown menu with 'marmalade' selected.


At the bottom right of the dialog box, there are two buttons: a grey 'Cancel' button and a blue 'Deploy' button. The background interface shows a 'Cluster' tab, a 'Network' section, and a table with columns for 'Machines' and 'Address'. A 'Deploy' button is also visible in the background table.

# KUBERNETES: DEPLOY APPLICATION

Machines Dashboard Cluster

Overview Topology Containers

CPU Memory Network



572.2 MB  
381.5 MB  
190.7 MB  
0

5 min 4 min 3 min 2 min 1 min

Kubernetes Services				✓ Deploy
Name	Address	Containers	Namespace	
database	18.0.10.117:5434	1	marmalade	
frontend	18.0.10.63:80	1	marmalade	

Kubernetes Nodes			+
Machines	Address	Containers	
node-1.rha	node-1.rha	2	

# KUBERNETES: DEPLOY APPLICATION

The screenshot displays a Kubernetes dashboard interface. At the top, there is a navigation bar with 'Machines', 'Dashboard', and 'Cluster' tabs. On the left, a sidebar contains 'Overview', 'Topology', and 'Containers' sections. The main content area shows two pod details cards, both located on 'node-1.rha'.

**database-0crnd** (on node-1.rha)

- Node Address: 18.0.75.4
- Pod Address: marmalade
- Namespace: Running
- Status: Always
- Restart Policy: Always
- Labels: name: database, template: ruby-helloworld-sample

**ruby-helloworld-database** (on node-1.rha)

- Image: mysql
- Image ID: e0db8fe06e30
- Container ID: 0cbbe3f2c25d
- State: running
- Since: 2015-06-09T21:34:38Z
- Restart Count: 0
- Environment: MYSQL\_ROOT\_PASSWORD=●●●●, MYSQL\_DATABASE=root
- Port: 3306/TCP

**frontend-nnxff** (on node-1.rha)

- Node Address: 18.0.75.3
- Pod Address: marmalade
- Namespace: Running
- Status: Always
- Restart Policy: Always
- Labels: name: frontend, template: ruby-helloworld-sample

# KUBERNETES: SCALE SERVICE

The screenshot displays a Kubernetes management interface. A modal dialog titled "Adjust Service" is open, allowing the user to scale the "frontend" service. The dialog contains the following fields and controls:

- Name:** A text input field containing "frontend".
- Replicas:** A section with a dropdown menu currently set to "frontend" and a text input field containing the number "10".
- Buttons:** "Cancel" and "Adjust" buttons at the bottom right of the dialog.

The background interface shows a dashboard with the following components:

- Navigation:** "Machines", "Dashboard", and "Cluster" tabs.
- Overview:** A section with tabs for "CPU", "Memory", and "Network". The "Memory" tab is active, showing a graph with values 572.2 MB, 381.5 MB, and 190.7 MB.
- Containers:** A table titled "Kubernetes Services" with columns: Name, Address, Containers, Namespace, and an edit icon.
- Kubernetes Nodes:** A table titled "Kubernetes Nodes" with columns: Machines, Address, and Containers.

Name	Address	Containers	Namespace
database	18.0.10.117:5434	1	marmalade
frontend	18.0.10.63:80	1	marmalade

Machines	Address	Containers
node-1.rha	node-1.rha	2
node-2.rha	node-2.rha	0
node-3.rha	node-3.rha	0

# KUBERNETES: SCALE SERVICE

The screenshot displays a Kubernetes dashboard interface. At the top, there is a navigation bar with 'Machines', 'Dashboard', and 'Cluster' tabs. On the left side, there is a sidebar menu with 'Overview', 'Topology', and 'Containers' options. The main area shows a network diagram of a cluster. It features several nodes (represented by server icons) at the bottom, connected to a central hub. From this hub, numerous pods (represented by blue cube icons) are distributed across the nodes. A central orange service icon is also visible, connected to the pods. A purple replication controller icon is also present in the cluster. On the right side, there is a sidebar with a search bar and a list of object types: Pod, Service, Replication Controller, and Node. Each type has a brief description of its function.

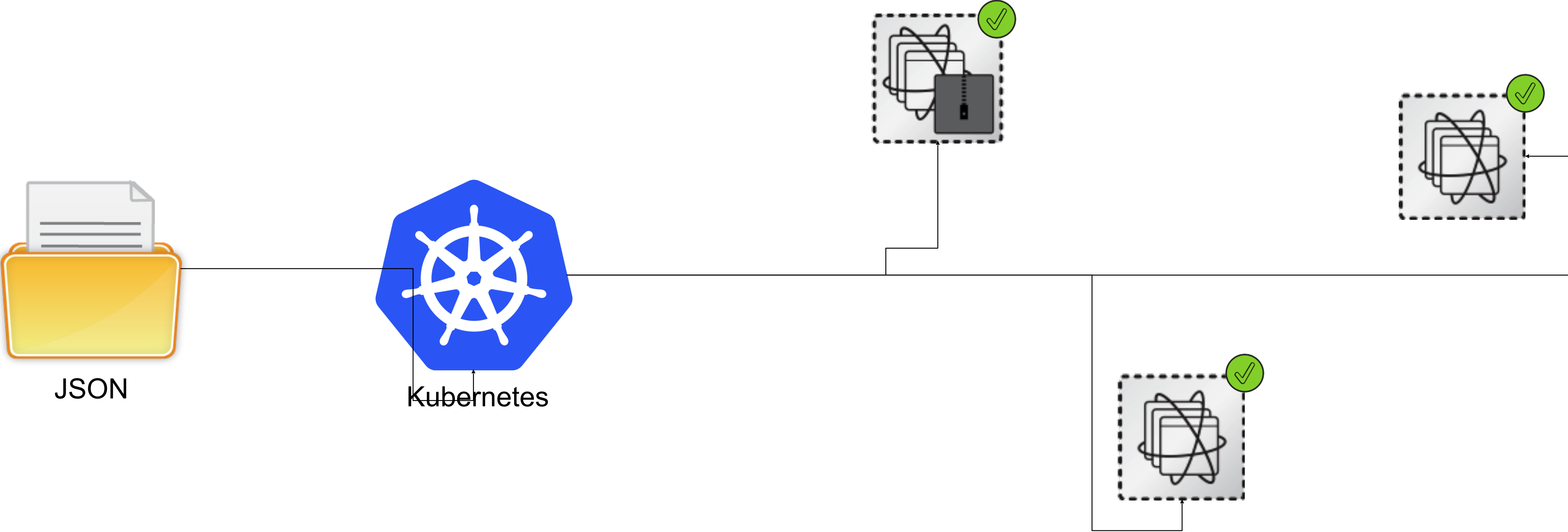
Machines Dashboard Cluster

Overview  
Topology  
Containers

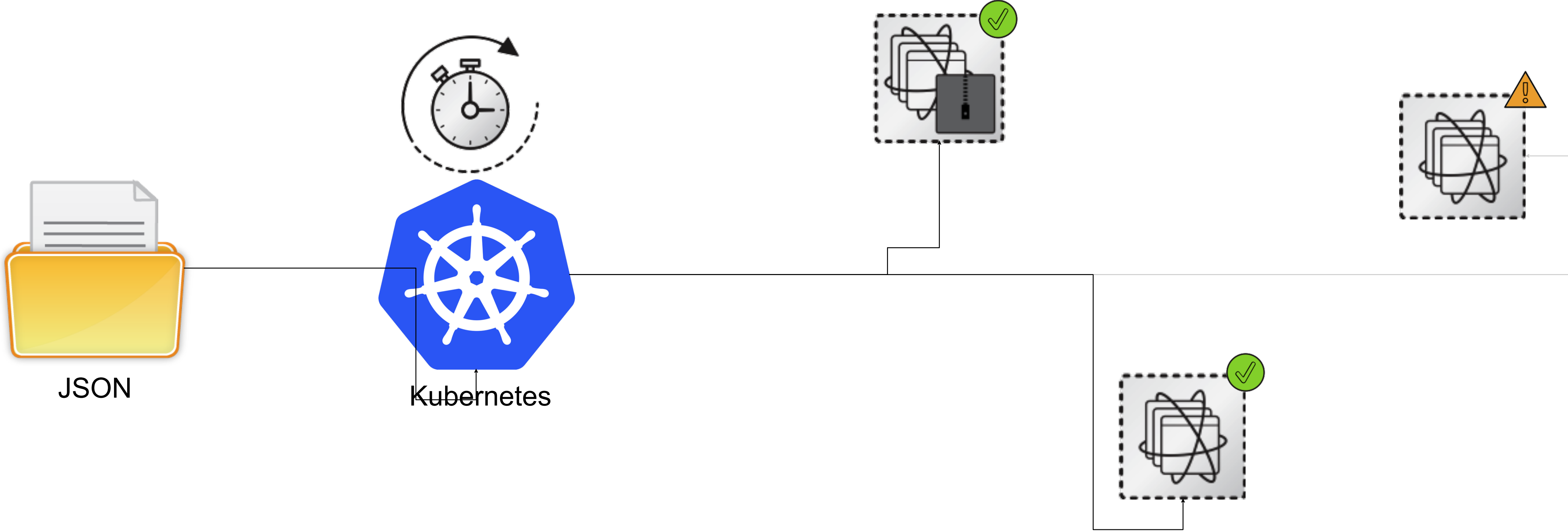
Select an object to see more details.

- Pod**  
Pods contain one or more containers that run together on a node, containing your application code.
- Service**  
Services group pods and provide a common DNS name and an optional, load-balanced IP address to access them.
- Replication Controller**  
Replication controllers dynamically create instances of pods from templates, and remove pods when necessary.
- Node**  
Nodes are the machines that run your containers.

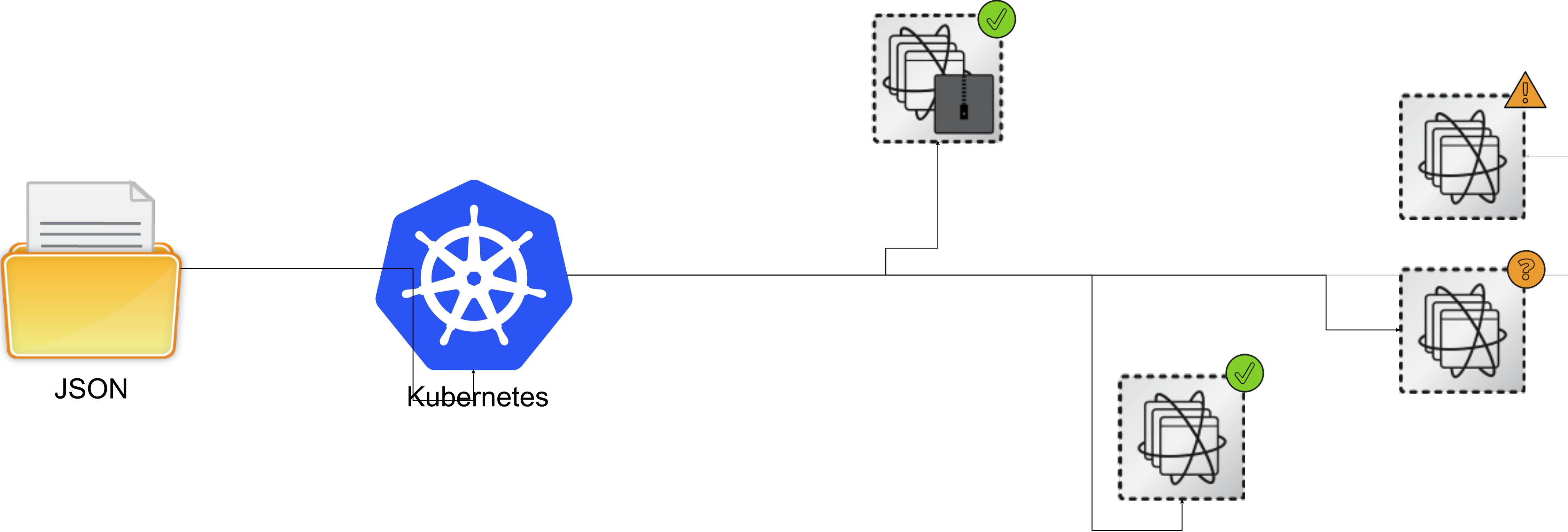
# KUBERNETES: HEALTH CHECK



# KUBERNETES: HEALTH CHECK

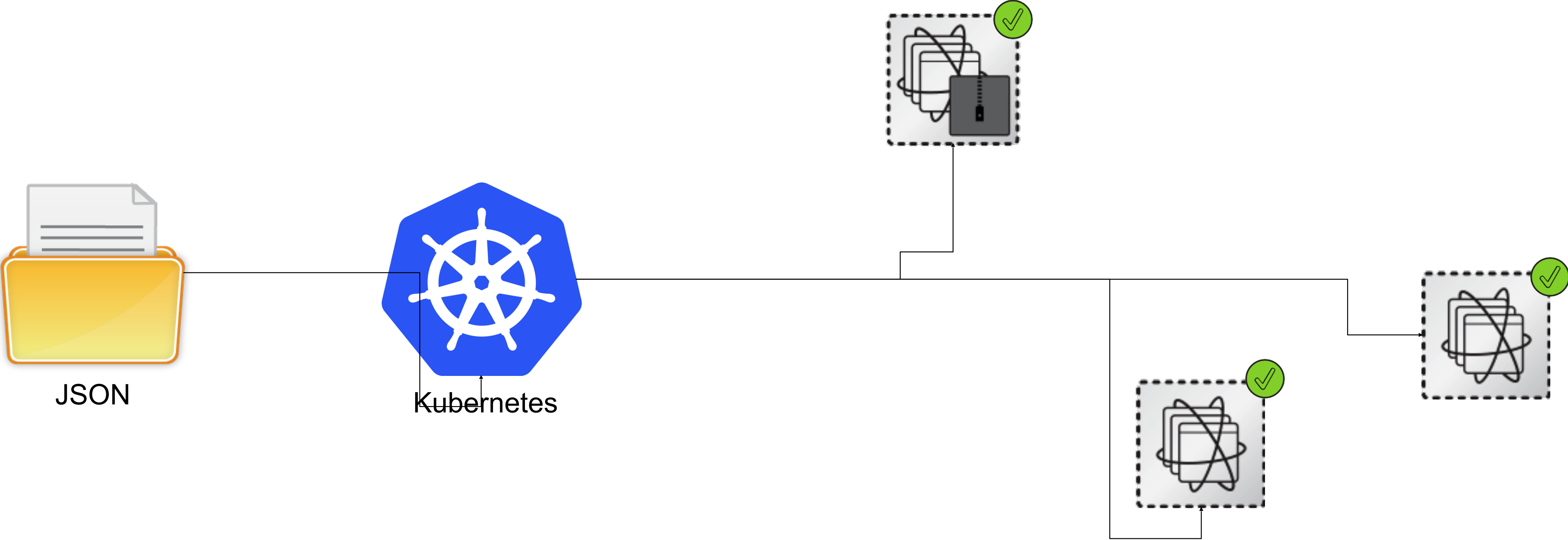


# KUBERNETES: HEALTH CHECK

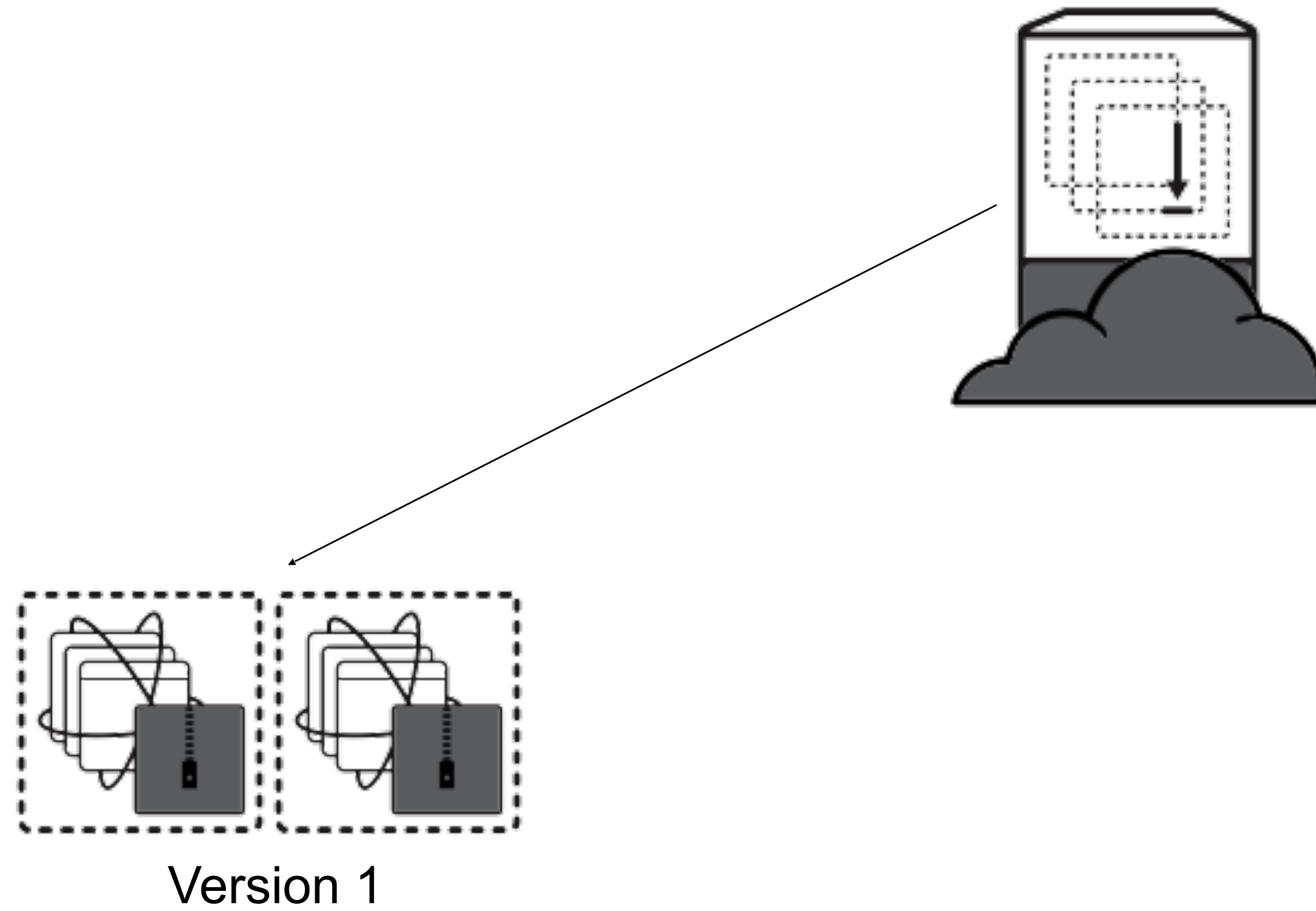




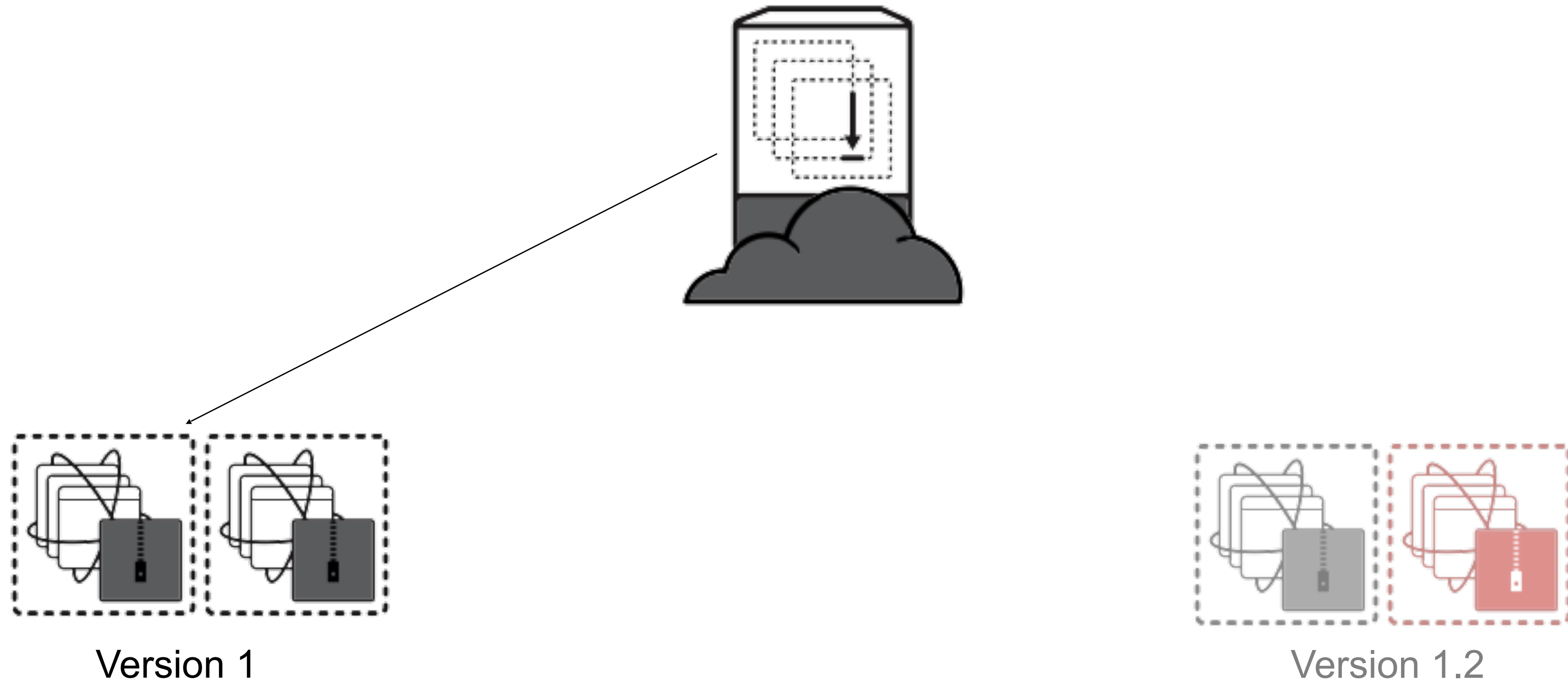
# KUBERNETES: HEALTH CHECK



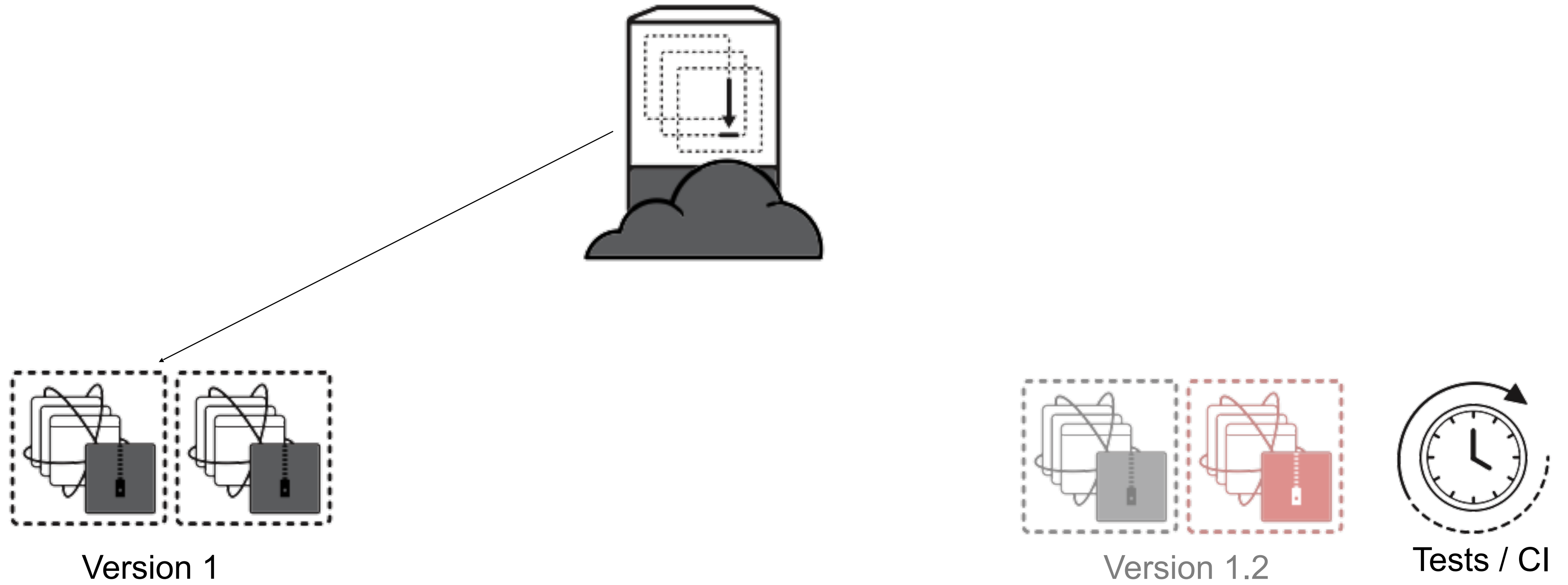
# KUBERNETES: DEPLOYMENT



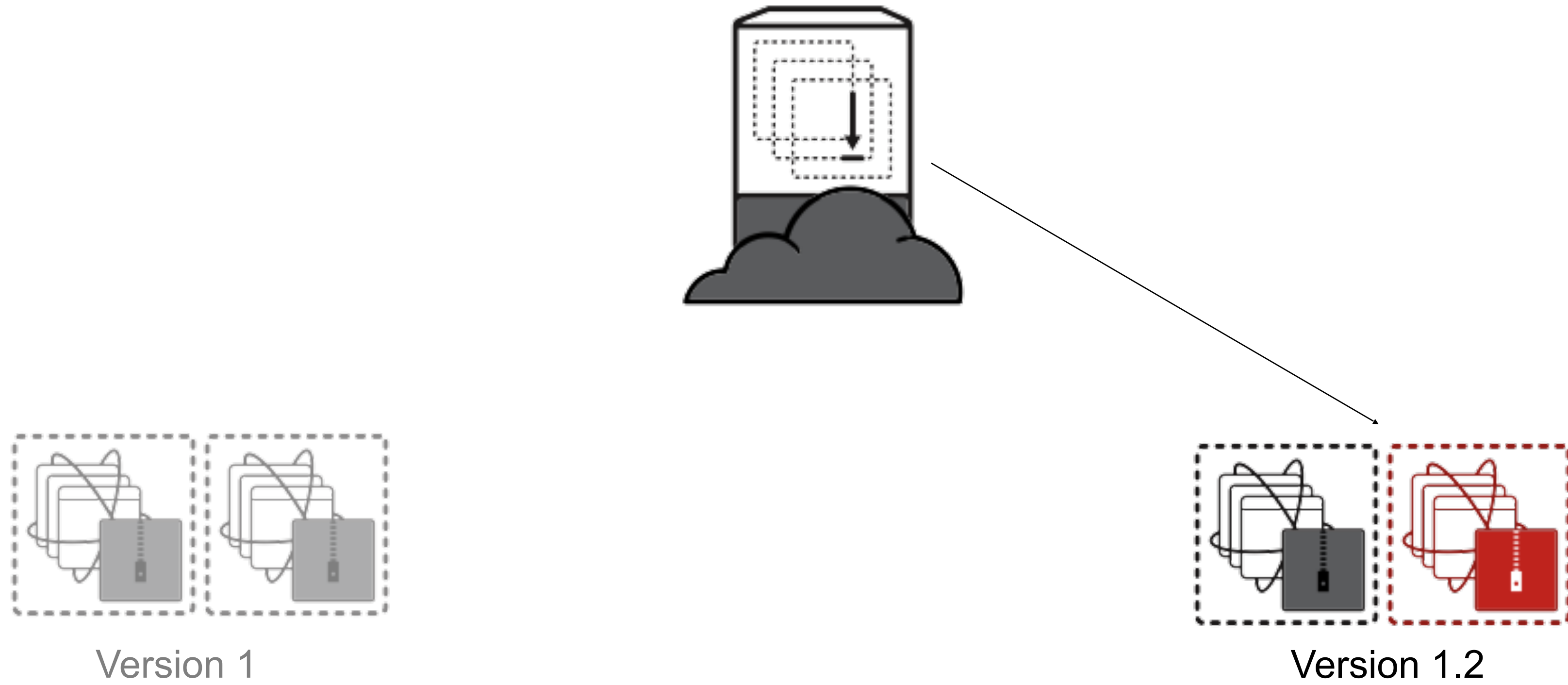
# KUBERNETES: DEPLOYMENT



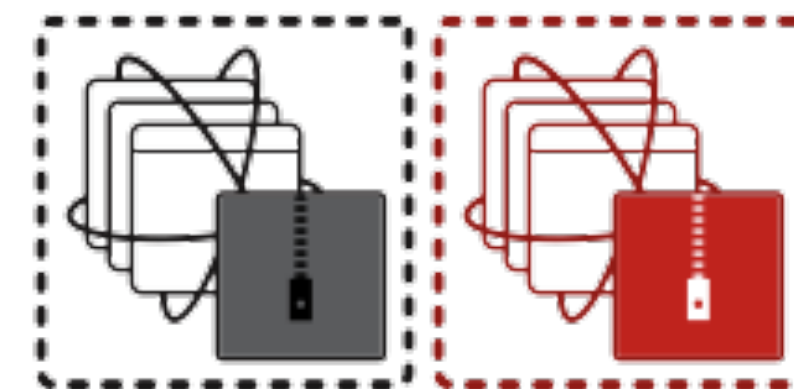
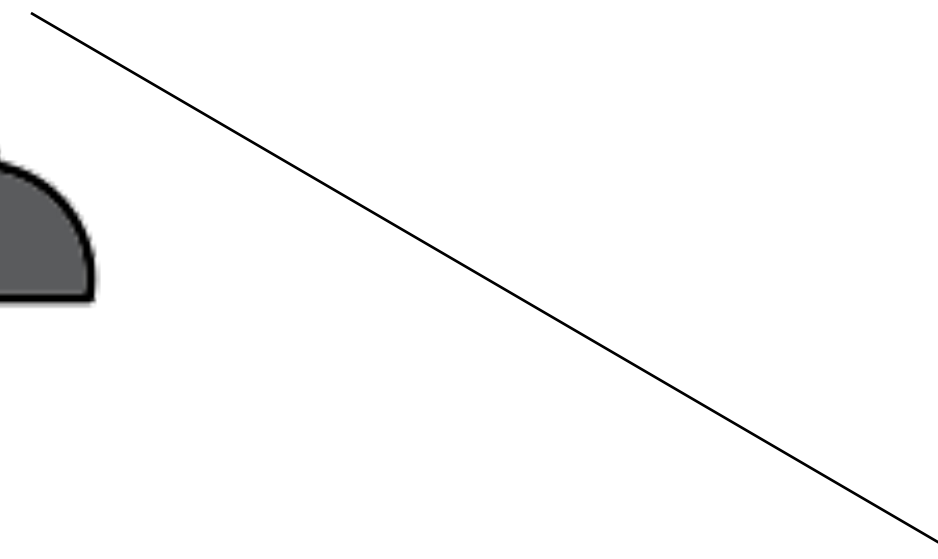
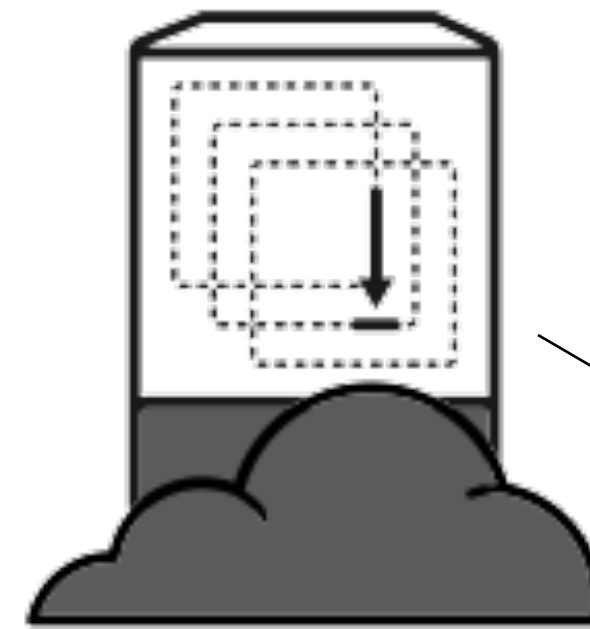
# KUBERNETES: DEPLOYMENT



# KUBERNETES: DEPLOYMENT



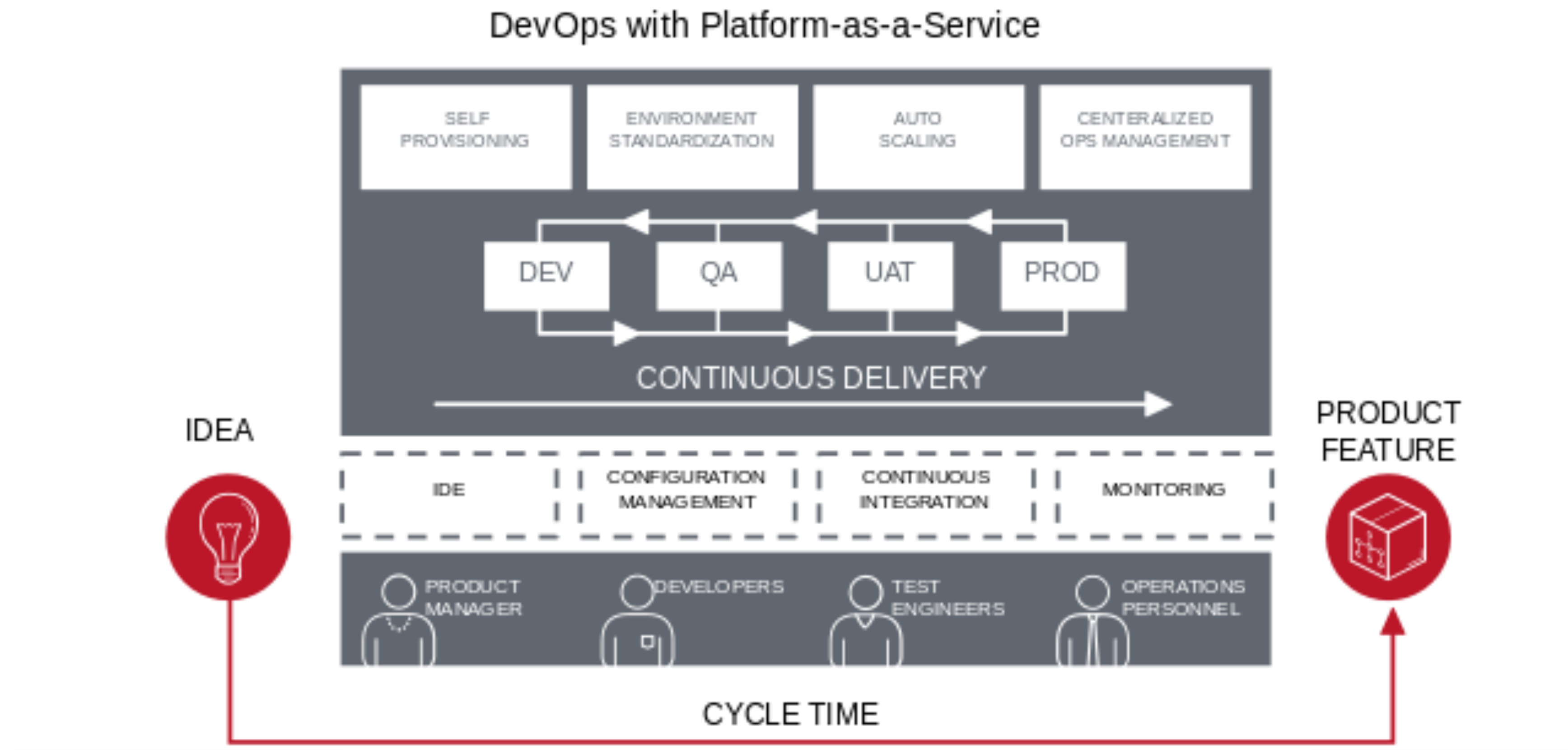
# KUBERNETES: DEPLOYMENT



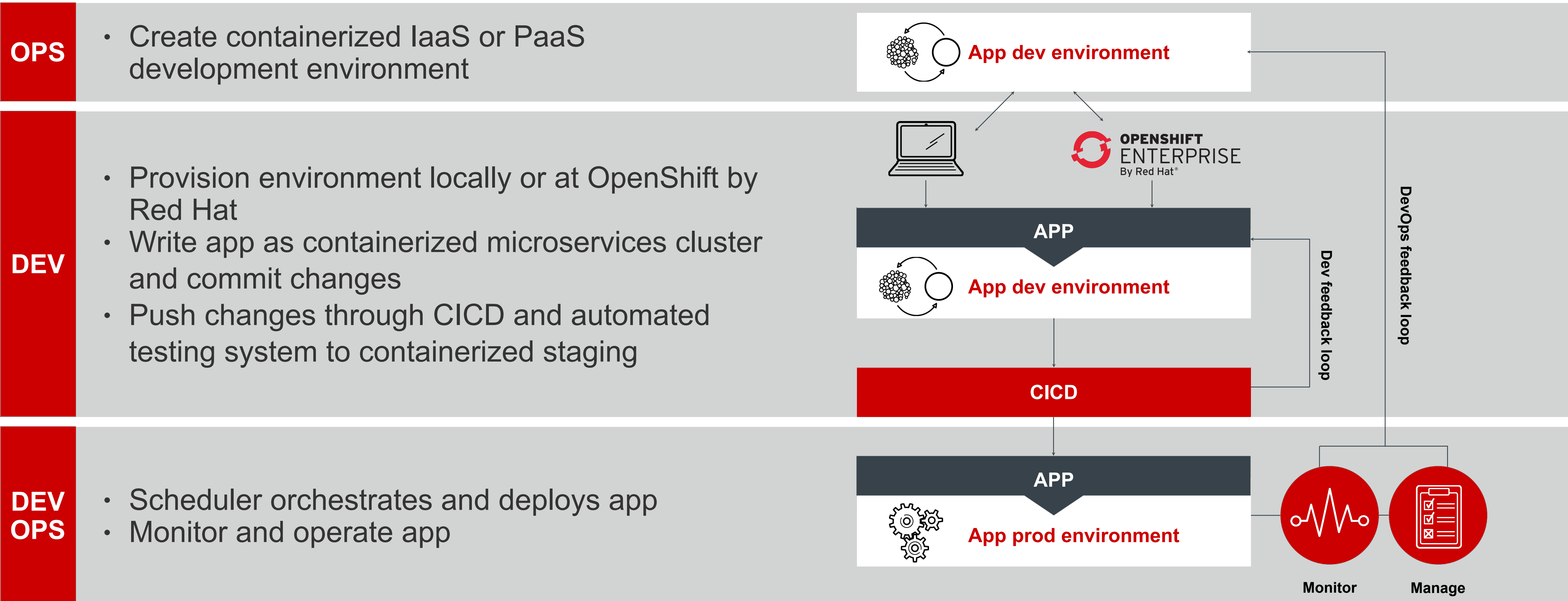
Version 1.2

# REDUCE CYCLE TIME FROM IDEA TO FEATURE

## DEVOPS WITH PaaS



# DEVOPS WORKFLOW





# CUSTOMER CASE STUDY



# FINANCIAL SERVICES COMPANY

“It could take 6 weeks to get a single word changed on the web site.”

“It took 2 years after a competitive start-up launch to get a competing product to market.”

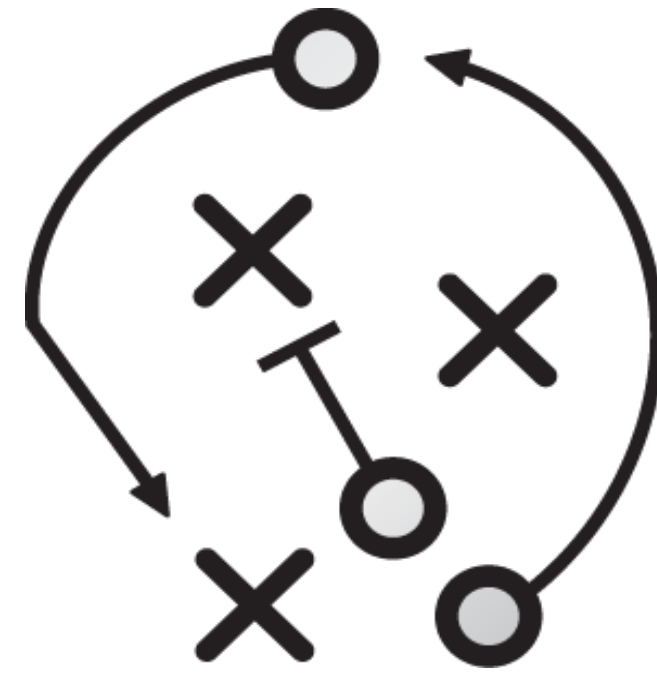
“When developers work in Node.js, they can change the code they’re working on, direct it to run, and see whether it works-in the blink of an eye.”

“The environment, while stable, didn’t use the sexiest technologies, which made recruiting difficult.”

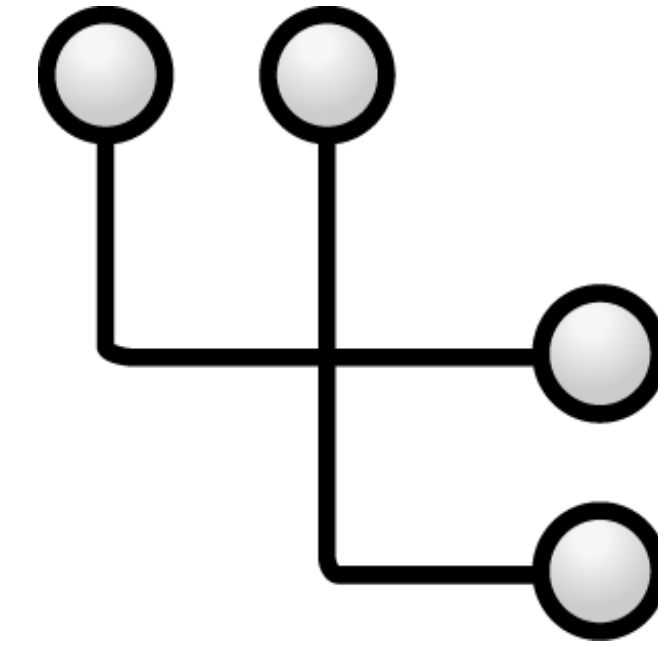
# BUSINESS CHALLENGES



Growth



Competition



Agility

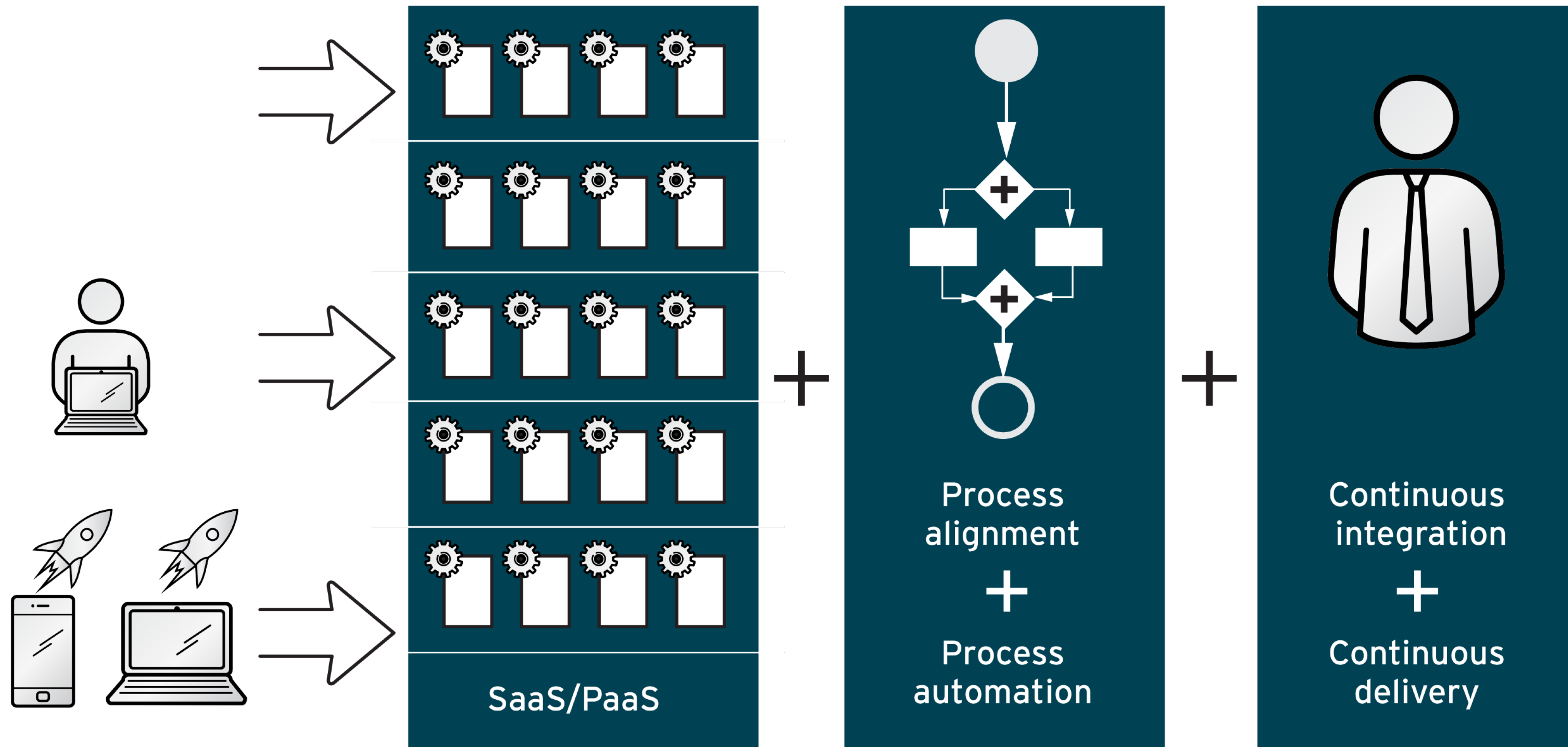


Predictability



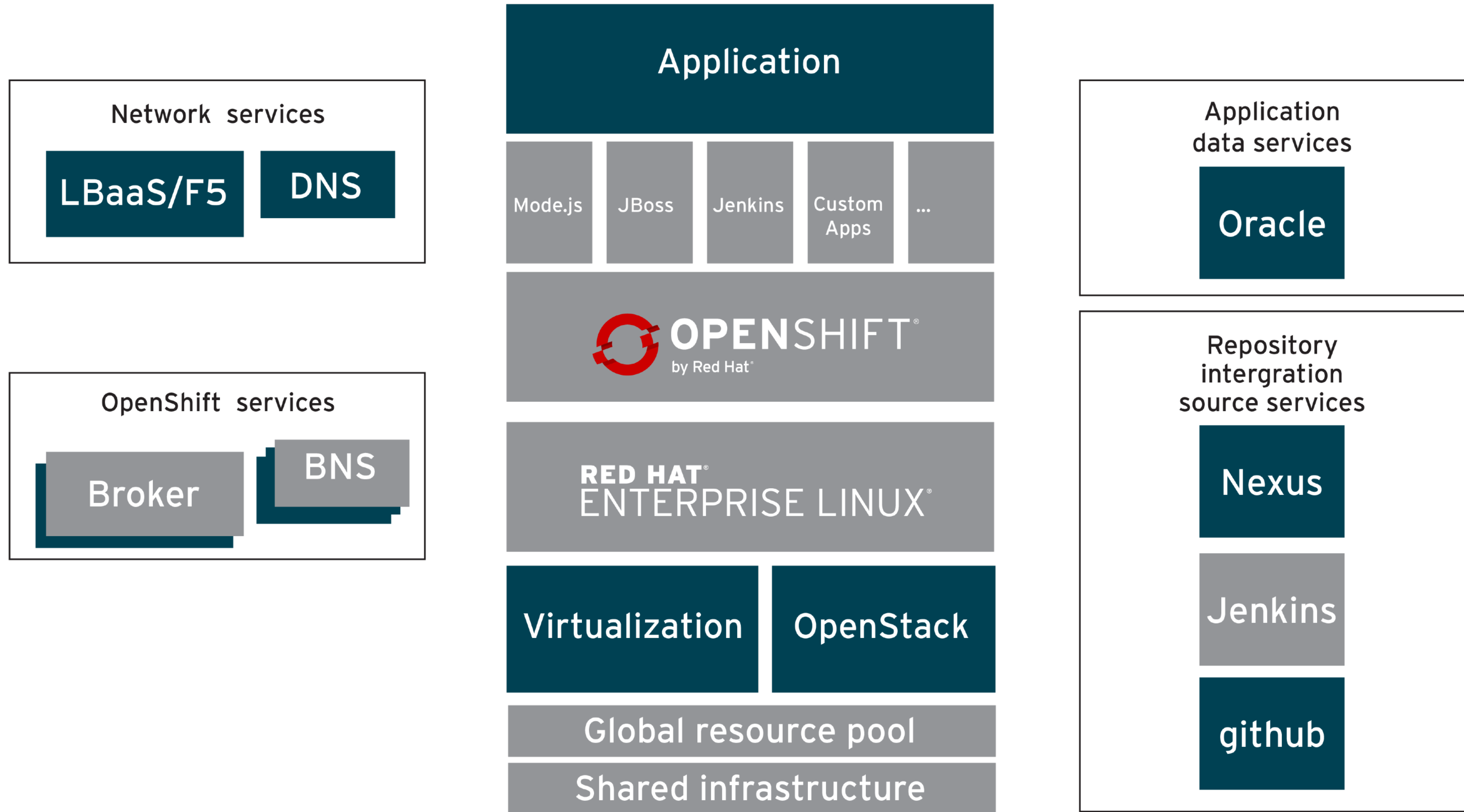
Recruiting

# DEVOPS SOLUTION APPROACH

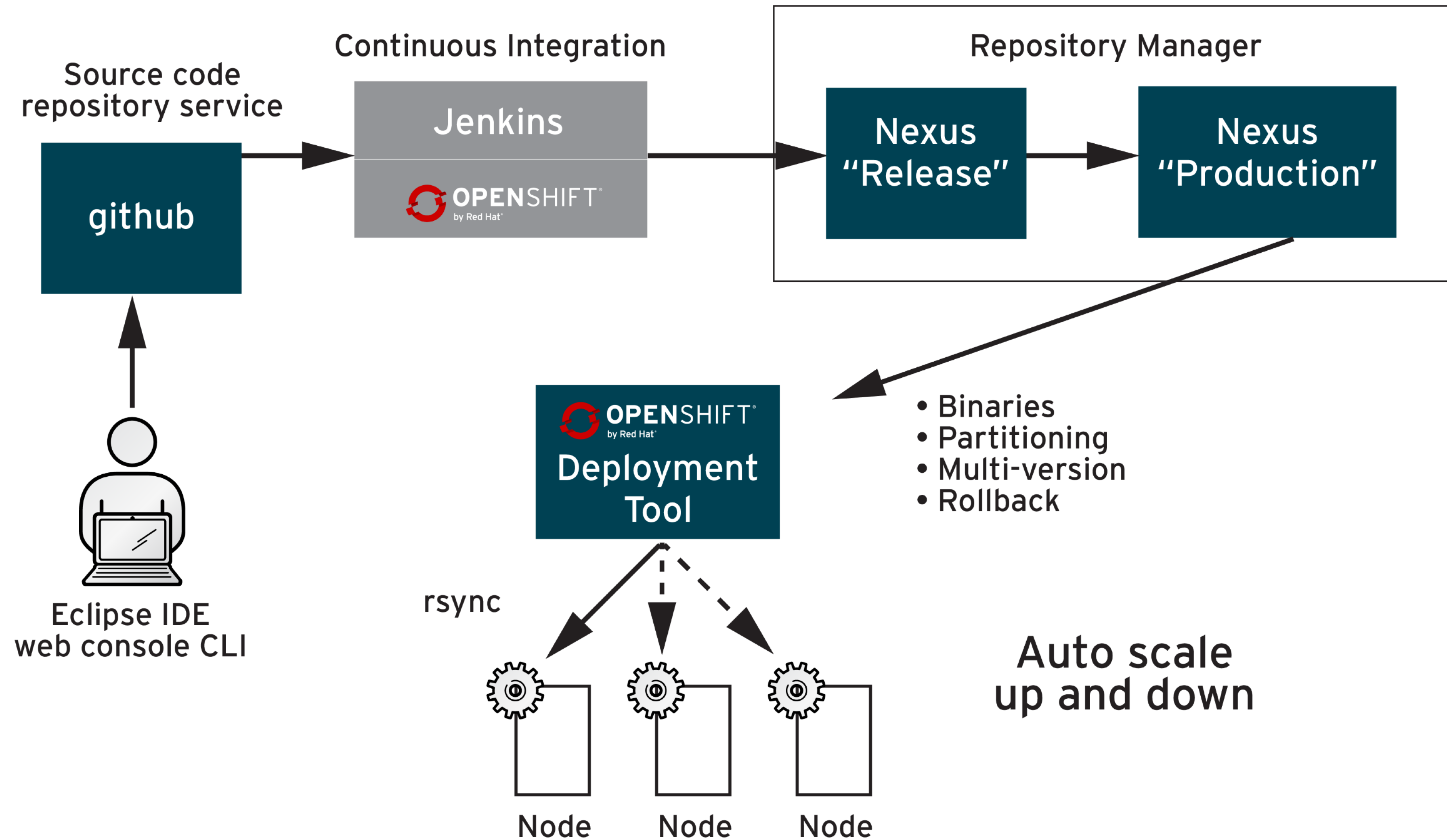


Leverage Automation Technologies Combined with Cloud Architecture

# ARCHITECTURE



# DEVELOPMENT TO PRODUCTION IN <30 MINUTES



# CONTINUOUS INTEGRATION and CONTINUOUS DELIVERY BUILD PIPELINE

The screenshot displays the Jenkins web interface for a Build Pipeline. At the top, the Jenkins logo is on the left, and a search bar, user name 'developer', and 'log out' link are on the right. Below the header, the breadcrumb 'Jenkins > Build Pipeline >' is visible, along with a 'DISABLE AUTO REFRESH' link. The main content area is titled 'Build Pipeline' and features two icons: 'Run' and 'History'. The pipeline is visualized as a horizontal sequence of four stages: 'Commit', 'Acceptance', 'UAT', and 'Production', connected by green arrows. Three pipeline runs are shown as rows:

- Pipeline #68:** The 'Commit' stage is highlighted in yellow, indicating it is the current or most recent stage. It shows a timestamp of 'Aug 28, 2014 4:29:24 PM' and a duration of '8.6 sec and counting' by user 'developer'. The 'Acceptance', 'UAT', and 'Production' stages are shown as blue boxes, indicating they have not yet started.
- Pipeline #67:** The 'Commit' stage is highlighted in green, showing a timestamp of 'Aug 28, 2014 4:22:34 PM' and a duration of '12 sec' by user 'developer'. The 'Acceptance' stage is also highlighted in green, showing a timestamp of 'Aug 28, 2014 4:22:54 PM' and a duration of '33 sec'. The 'UAT' and 'Production' stages are blue.
- Pipeline #66:** The 'Commit' stage is highlighted in green, showing a timestamp of 'Aug 28, 2014 4:18:29 PM' and a duration of '12 sec' by user 'developer'. The 'Acceptance' stage is highlighted in green, showing a timestamp of 'Aug 28, 2014 4:18:49 PM' and a duration of '49 sec'. The 'UAT' and 'Production' stages are blue.

# CONTINUOUS INTEGRATION and CONTINUOUS DELIVERY BUILD PIPELINE

The screenshot shows the JBoss Developer Studio interface. On the left, the Package Explorer displays a project structure for 'ticket-monster-for-devops'. The main editor area shows a 'Commit Changes to Git Repository' dialog box. The dialog box contains a 'Commit message' field with the text 'Changed application name and added high demand web'. Below this, the 'Author' and 'Committer' fields are both set to 'Trevor Quinn <tquinn@redhat.com>'. The 'Files (2/2)' section lists two files to be committed: 'src/main/java/org/jboss/jdf/example/ticketmonster/rest/EventEndpoint.java' and 'src/main/webapp/resources/templates/desktop/home.html'. At the bottom of the dialog, there are three buttons: 'Commit and Push', 'Cancel', and 'Commit'. The background shows the project tree with various Java files and folders like 'dto', 'model', 'rest', and 'service'.

Property	Value
derived	false
editable	true
last modified	July 29, 2014 at 4:23:49 PM
linked	false



# CONTINUOUS INTEGRATION and CONTINUOUS DELIVERY BUILD PIPELINE

The screenshot displays the Jenkins web interface for a 'Build Pipeline'. At the top, the Jenkins logo is on the left, and a search bar, user name 'developer', and 'log out' link are on the right. Below the header, the breadcrumb 'Jenkins > Build Pipeline' is shown on the left, and 'DISABLE AUTO REFRESH' is on the right. The main content area is titled 'Build Pipeline' and features a 'Run' button and a 'History' button. The pipeline is visualized as a horizontal sequence of stages: Pipeline, Commit, Acceptance, UAT, and Production. Three pipeline runs are shown, each with its own set of stage boxes. Run #69 is currently active, with the 'Commit' stage in progress (green box) and 'Acceptance' (yellow box) starting. Run #68 is completed, with all stages (Commit, Acceptance, UAT, Production) shown as blue boxes. Run #67 is also completed, with all stages shown as blue boxes. Green arrows indicate the flow from one stage to the next.

Pipeline #	Stage	Status	Start Time	Duration	User
#69	Commit	In Progress	Aug 28, 2014 4:30:59 PM	12 sec	developer
	Acceptance	Starting	Aug 28, 2014 4:31:19 PM	27 sec and counting	
#68	Commit	Completed	Aug 28, 2014 4:29:24 PM	9.3 sec	developer
	Acceptance	Completed			
	UAT	Completed			
	Production	Completed			
#67	Commit	Completed	Aug 28, 2014 4:22:34 PM	12 sec	developer
	Acceptance	Completed	Aug 28, 2014 4:22:54 PM	33 sec	
	UAT	Completed			
	Production	Completed			

# CONTINUOUS INTEGRATION and CONTINUOUS DELIVERY BUILD PIPELINE

The screenshot displays the Jenkins web interface for a 'Build Pipeline'. At the top, the Jenkins logo is on the left, and a search bar, user name 'developer', and 'log out' link are on the right. Below the header, the breadcrumb 'Jenkins > Build Pipeline >' is visible, along with a 'DISABLE AUTO REFRESH' link. The main content area is titled 'Build Pipeline' and features two icons: 'Run' and 'History'. The pipeline is visualized as a horizontal flow of stages across three rows, each representing a different pipeline run:

- Row 1 (Pipeline #69):** Starts with a grey 'Pipeline #69' box. The first stage is '#69 Commit' (green box) with a timestamp of 'Aug 28, 2014 4:30:59 PM', duration of '12 sec', and user 'developer'. This is followed by '#84 Acceptance' (red box) with a timestamp of 'Aug 28, 2014 4:31:19 PM' and duration of '50 sec'. The final two stages are 'UAT' and 'Production' (both light blue boxes).
- Row 2 (Pipeline #68):** Starts with a grey 'Pipeline #68' box. The first stage is '#68 Commit' (red box) with a timestamp of 'Aug 28, 2014 4:29:24 PM', duration of '9.3 sec', and user 'developer'. This is followed by 'Acceptance', 'UAT', and 'Production' (all light blue boxes).
- Row 3 (Pipeline #67):** Starts with a grey 'Pipeline #67' box. The first stage is '#67 Commit' (green box) with a timestamp of 'Aug 28, 2014 4:22:34 PM', duration of '12 sec', and user 'developer'. This is followed by '#83 Acceptance' (green box) with a timestamp of 'Aug 28, 2014 4:22:54 PM' and duration of '33 sec'. The final two stages are 'UAT' and 'Production' (both light blue boxes).

Green arrows indicate the flow from one stage to the next in each pipeline run.

# CONTINUOUS INTEGRATION and CONTINUOUS DELIVERY BUILD PIPELINE

The screenshot displays the Jenkins web interface for a Build Pipeline. At the top, the Jenkins logo is on the left, and a search bar, user name 'tester', and 'log out' link are on the right. Below the header, the breadcrumb 'Jenkins > Build Pipeline' is shown on the left, and 'DISABLE AUTO REFRESH' is on the right. The main content area is titled 'Build Pipeline' and features two icons: 'Run' and 'History'. The pipeline is visualized as three horizontal tracks, each representing a different build number. Each track starts with a 'Pipeline' block followed by several stages connected by green arrows. Stage #70 is currently running, while #69 and #68 are in a failed state.

Build #	Stage	Status	Time	Duration	User
#70	Pipeline	Running			
	#70 Commit	Success	Aug 28, 2014 4:33:44 PM	13 sec	developer
	#85 Acceptance	Success	Aug 28, 2014 4:34:04 PM	37 sec	
	#32 UAT	Running	Aug 28, 2014 4:37:09 PM	18 sec and counting	tester
#69	Pipeline	Failed			
	#69 Commit	Success	Aug 28, 2014 4:30:59 PM	12 sec	developer
	#84 Acceptance	Failed	Aug 28, 2014 4:31:19 PM	50 sec	
	UAT	Failed			
#68	Pipeline	Failed			
	#68 Commit	Failed	Aug 28, 2014 4:29:24 PM	9.3 sec	developer
	Acceptance	Failed			
	UAT	Failed			

# CONTINUOUS INTEGRATION and CONTINUOUS DELIVERY BUILD PIPELINE

The screenshot displays the Jenkins interface for a 'Build Pipeline'. At the top, the Jenkins logo is on the left, and a search bar, user 'release-eng', and 'log out' link are on the right. Below the header, the breadcrumb 'Jenkins > Build Pipeline >' is visible, along with a 'DISABLE AUTO REFRESH' link.

The main content area is titled 'Build Pipeline' and features two icons: 'Run' and 'History'. Below this, three pipeline runs are shown in a grid:

- Pipeline #70:** Shows four stages: '#70 Commit' (Aug 28, 2014 4:33:44 PM, 13 sec, developer), '#85 Acceptance' (Aug 28, 2014 4:34:04 PM, 37 sec), '#32 UAT' (Aug 28, 2014 4:37:09 PM, 47 sec, tester), and '#19 Production' (Aug 28, 2014 4:40:39 PM, 33 sec and counting, release-eng). All stages are green, indicating success.
- Pipeline #69:** Shows four stages: '#69 Commit' (Aug 28, 2014 4:30:59 PM, 12 sec, developer), '#84 Acceptance' (Aug 28, 2014 4:31:19 PM, 50 sec), 'UAT', and 'Production'. The 'Commit' and 'Acceptance' stages are red, indicating failure.
- Pipeline #68:** Shows four stages: '#68 Commit' (Aug 28, 2014 4:29:24 PM, 9.3 sec, developer), 'Acceptance', 'UAT', and 'Production'. The 'Commit' stage is red, indicating failure.

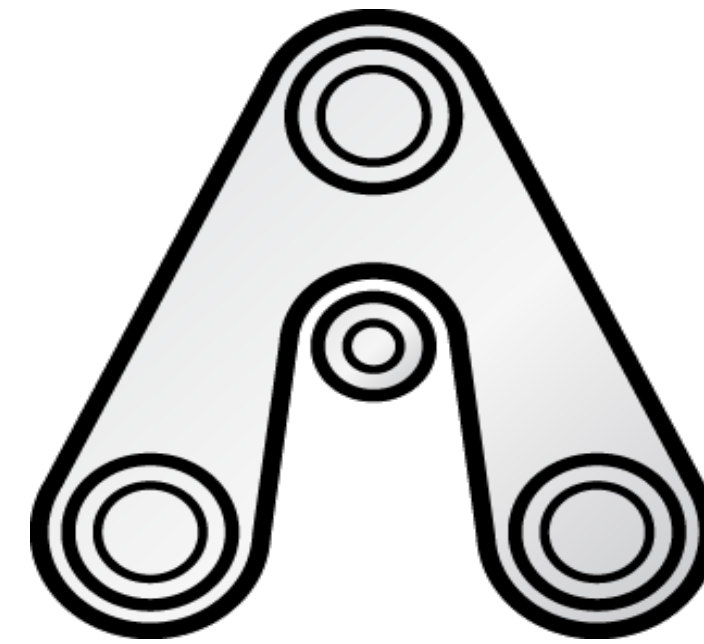
# DEVOPS ROI



Improve business  
agility



Improve developer  
productivity



Improve business  
predictability



Improve operational  
efficiency and costs

# DEVOPS METRICS



**Deployment  
Frequency**



**Change  
Volume**

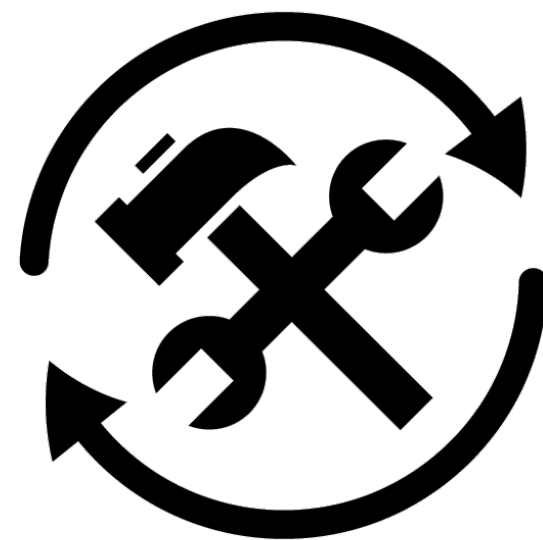


**Lead  
Time**

**404**

Page not found

**Deployment  
Failure Rate**



**Mean Time  
to Recover**

**99.999**

**Service  
Availability**

# DEVOPS IS PART OF A LARGER SHIFT

**HOW?**

**DEVOPS**

**WHAT?**

**CLOUD APPS  
+  
MICROSERVICES**

**WHERE?**

**CONTAINERS**

# THANK YOU

**Chris Van Tuin**

[cvantuin@redhat.com](mailto:cvantuin@redhat.com)

