Role of Microservices in BiModal IT

Arun Gupta, Red Hat

Arun Gupta

Director, Technical Marketing & Developer Advocacy

@arungupta
blog.arungupta.me
arungupta@redhat.com

BiModal IT

Mode 1 is traditional, emphasizing predictability, accuracy, stability

Mode 2 is exploratory, emphasizing agility and speed

Mode 2

- Trial and error
- Fail-fast and recover
- Minimum viable product
- Early feedback
- Iterate often

BiModal is NOT

- Dividing into two teams
- Just Agile development
- An IT capability
- Organization chart change
- Shadow IT

Gartner

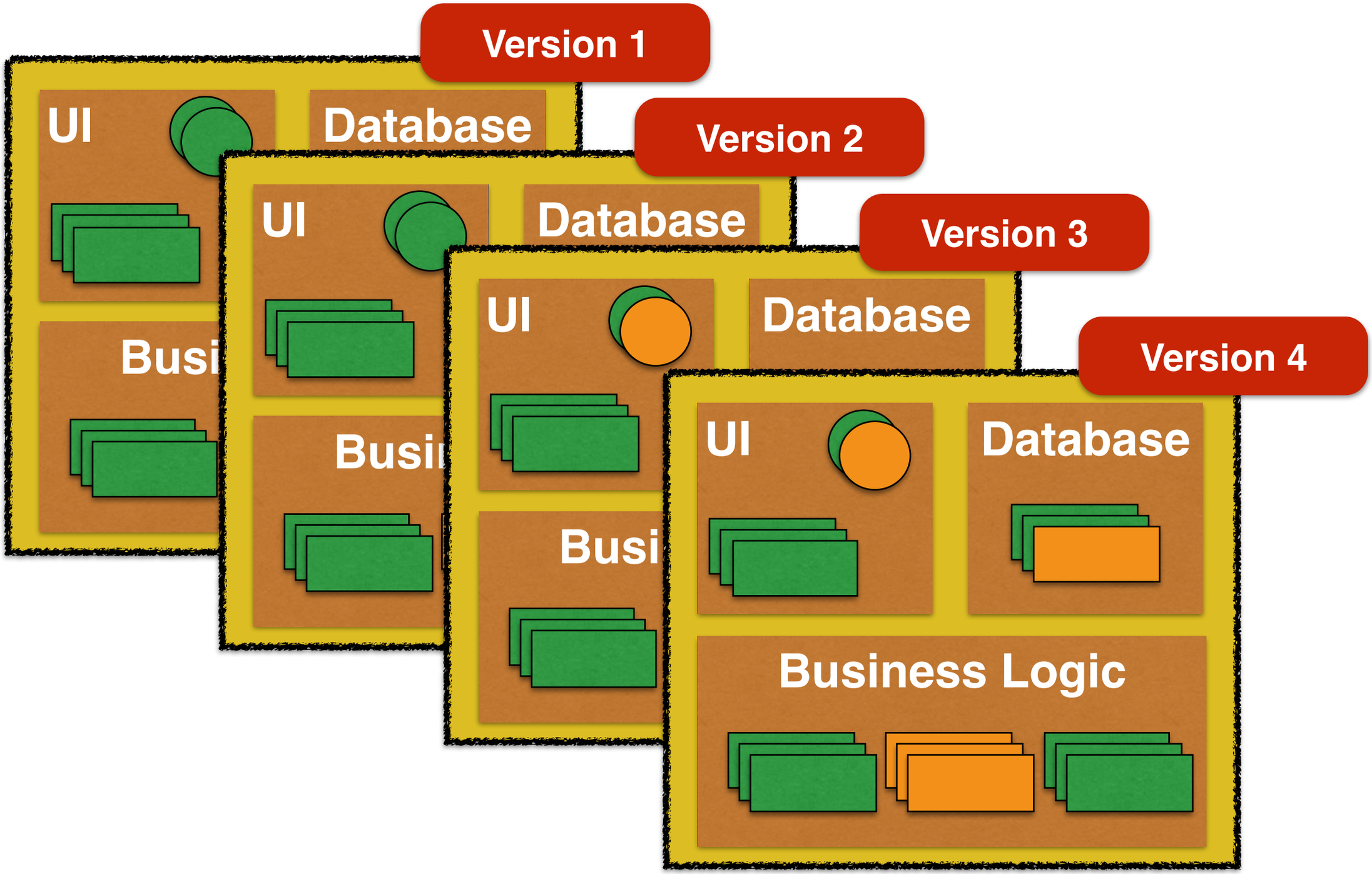
Deeply Different, Both Essential

Think Marathon Runner

Mode1		Mode 2
Reliability	**Goal**	Agility
Price for performance	**Value**	Revenue, brand, customer experience
Waterfall, V-model, high-ceremony IID	**Approach**	Agile, Kanban, low-ceremony IID
Plan-driven, approval-based	**Governance**	Empirical, continuous, process-based
Enterprise suppliers, long-term deals	**Sourcing**	Small, new vendors, short-term deals
Good at conventional process, projects	**Talent**	Good at new and uncertain projects
IT-centric, removed from customer	**Culture**	Business-centric, close to customer
Long (months)	**Cycle Times**	Short (days, weeks)

Think Sprinter

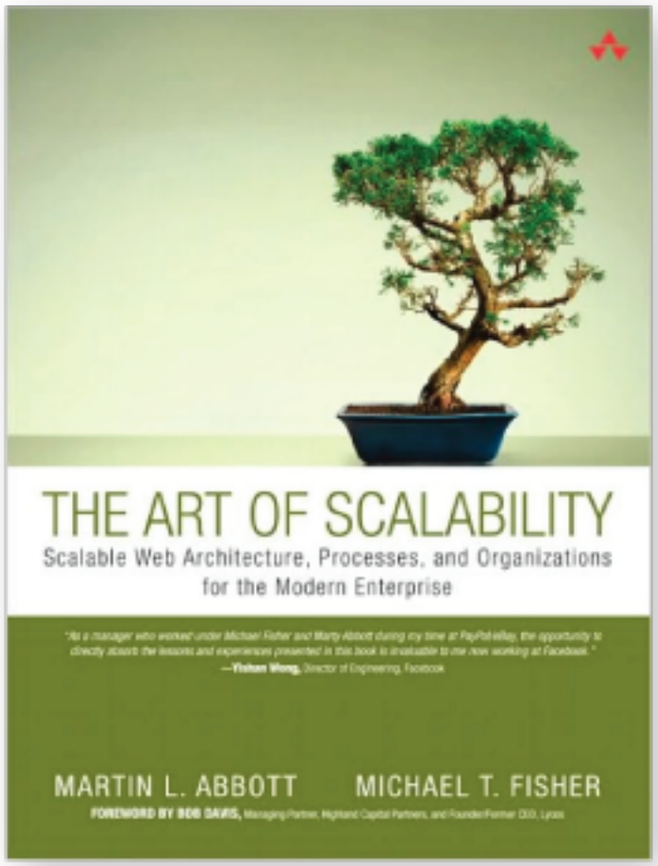
Mode 1 Application

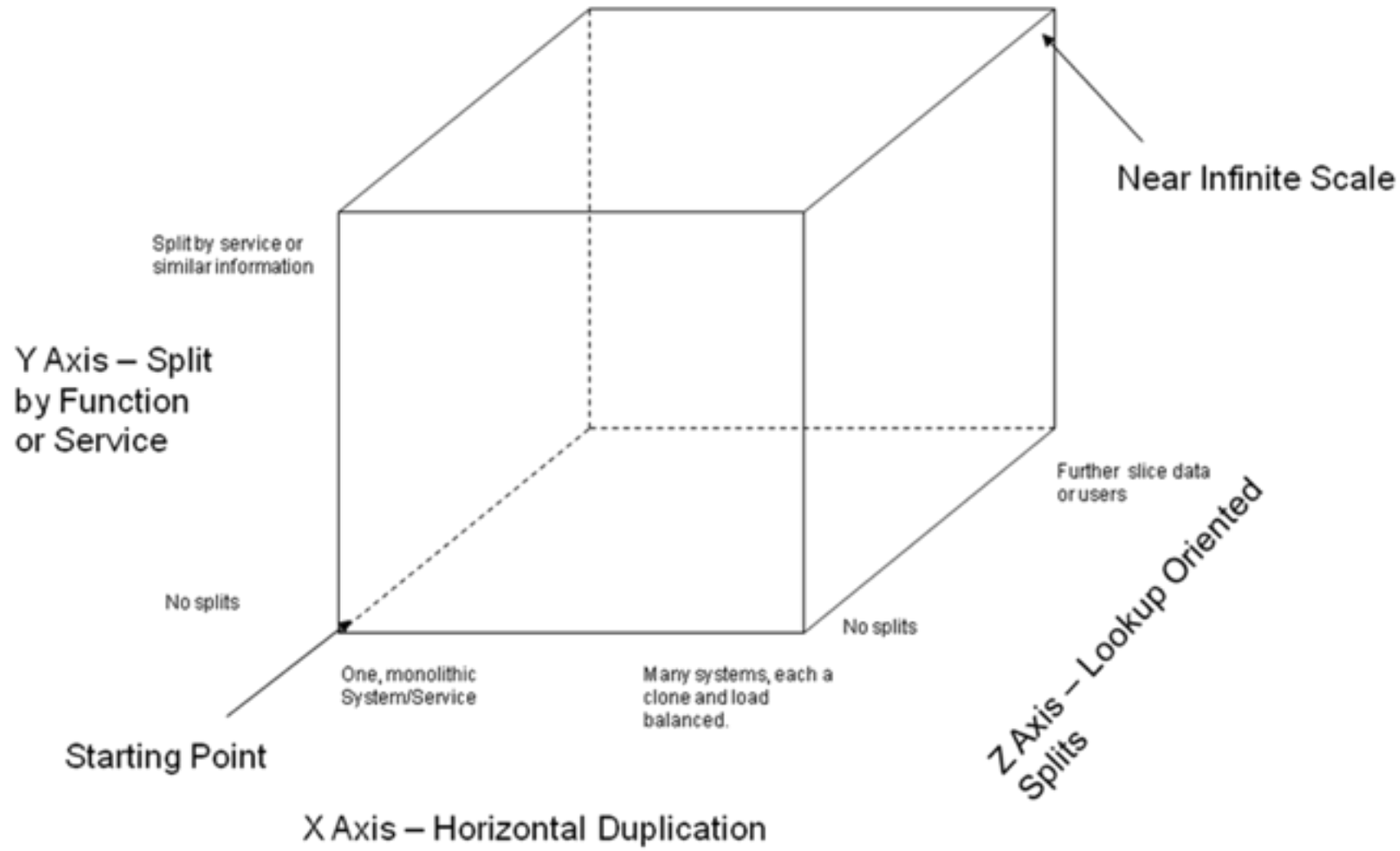


redhat

Disadvantages of Mode 1 Application

- Difficult to deploy and maintain
- Obstacle to frequent deployments
- Makes it difficult to try out new technologies/ framework

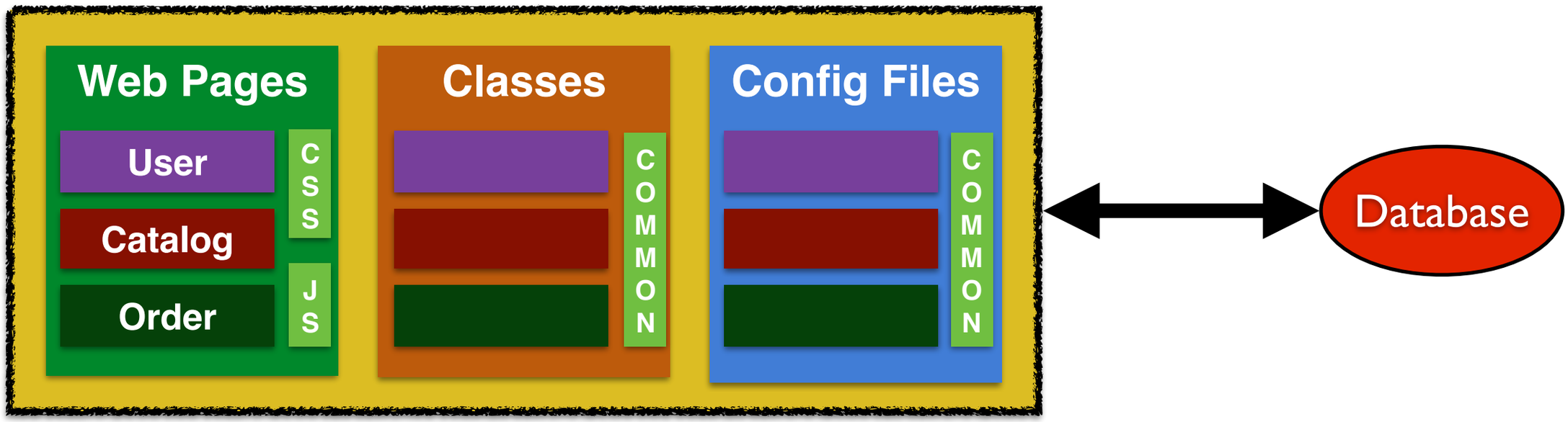


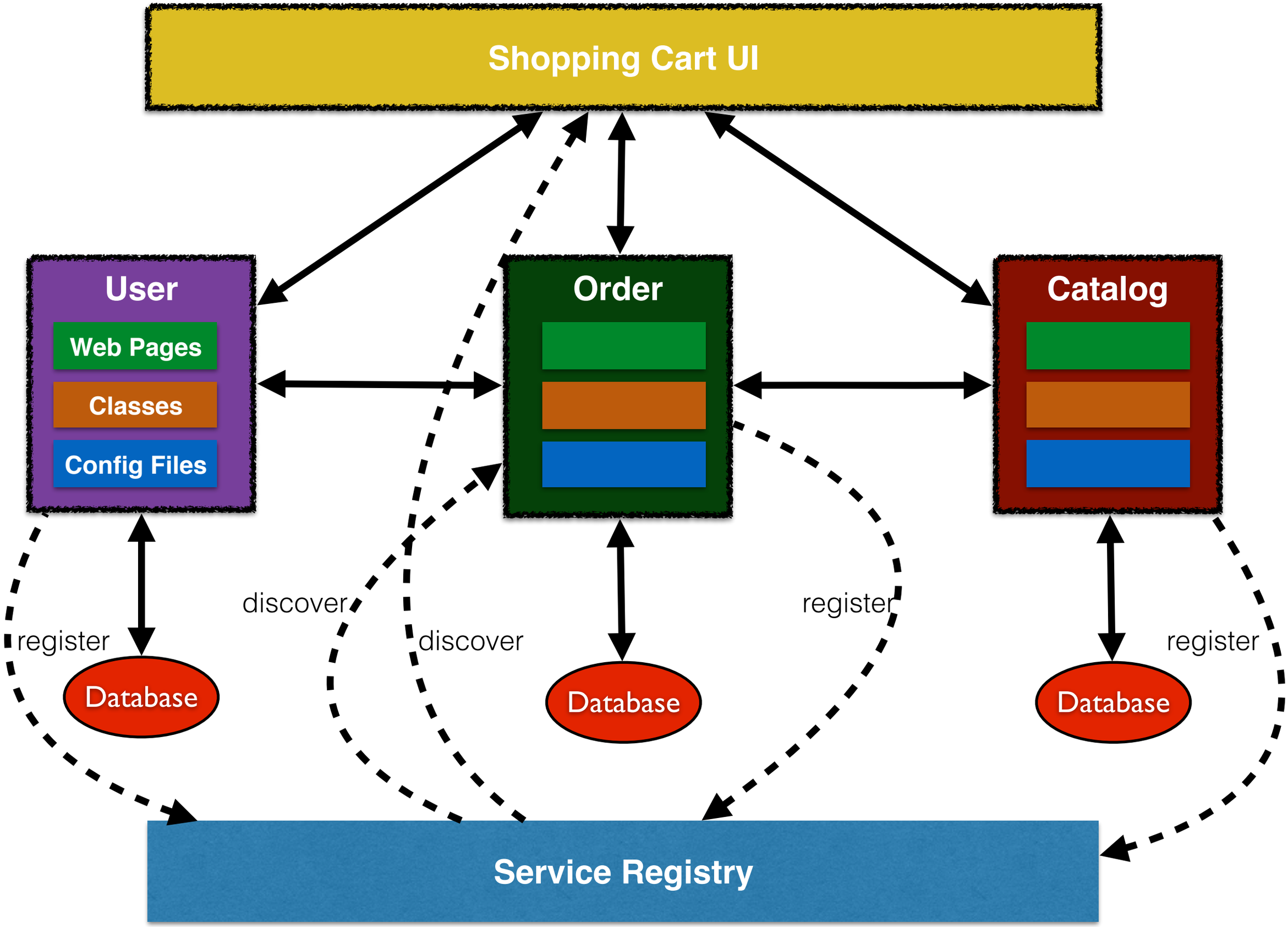


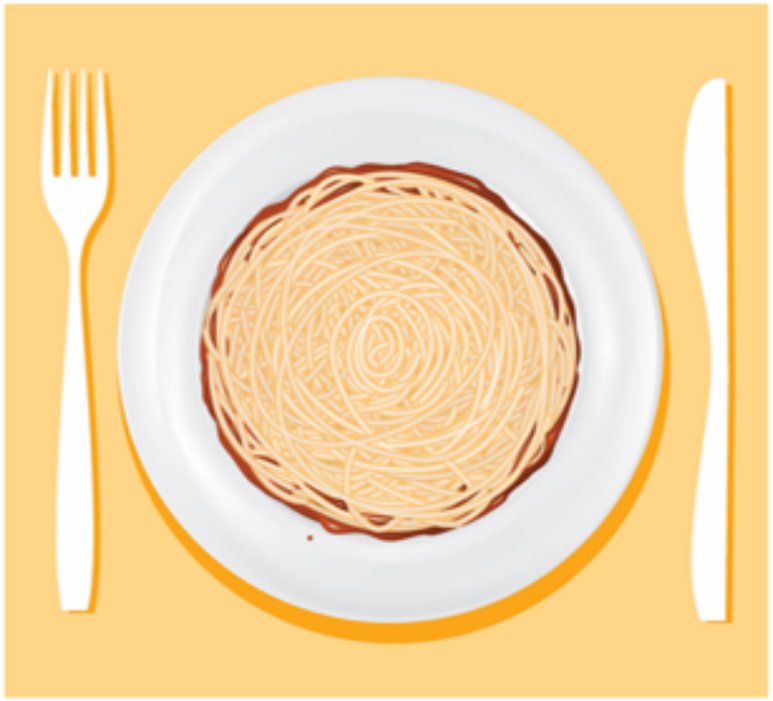
http://akfpartners.com/techblog/2008/05/08/splitting-applications-or-services-for-scale/

redhat

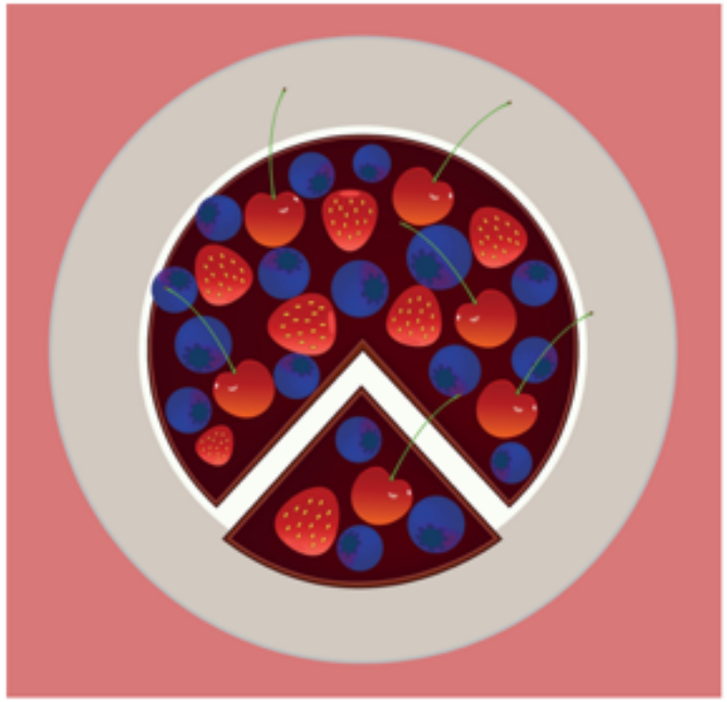
Mode 1 Application




Shopping Cart UI
User
Web Pages
Classes
Config Files
Order
Catalog
register
Database
discover
discover
Database
register
register
Database
Service Registry



With monolithic, tightly coupled applications, all changes must be pushed at once, making continuous deployment impossible.



Traditional SOA allows you to make changes to individual pieces. But each piece must be carefully altered to fit into the overall design.

With a microservices architecture, developers create, maintain and improve new services independently, linking info through a shared data API.

Kanban Solutions @kanbansolutions kanbansolutions.com

http://www.kanbansolutions.com/blog/microservices-architecture-friend-or-foe/

MSA Characteristics

Domain Driven Design

Explicitly Published Interface

Single Responsibility Principle

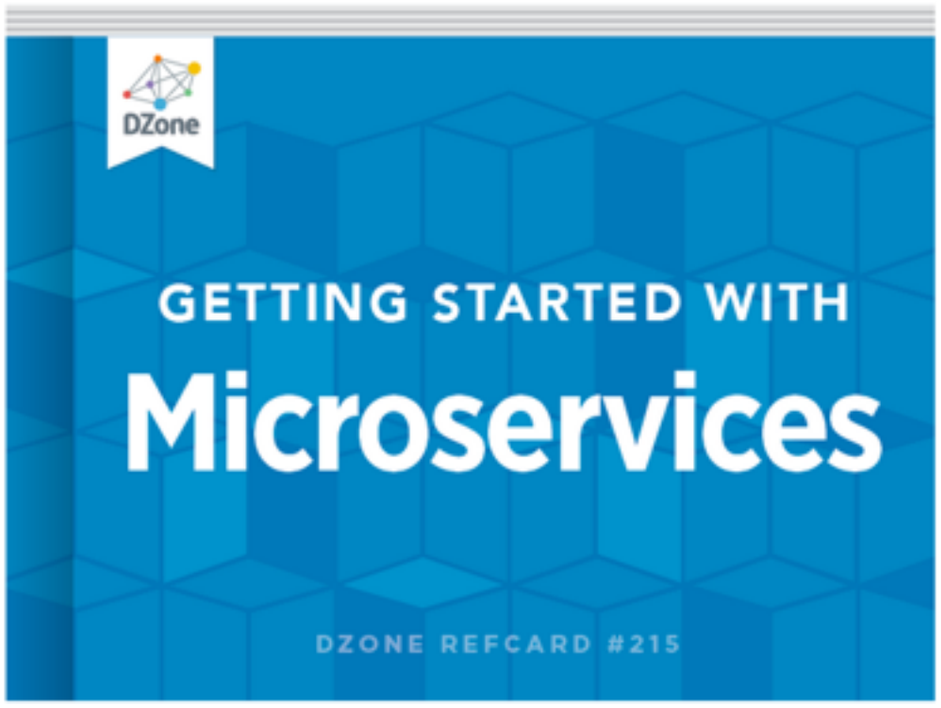
Lightweight Communication

Independent DURS

redhat

Operational Requirements

redhat



Refcard #215

Getting Started With Microservices

Design Patterns for Decomposing the Monolith

by Arun Gupta

Still re-deploying your entire application for one small update? Micros
modular updates and increase the speed of application deployments

Free PDF

DOWNLOAD ☆ SAVE 5,937

dzone.com/refcardz/getting-started-with-microservices

redhat

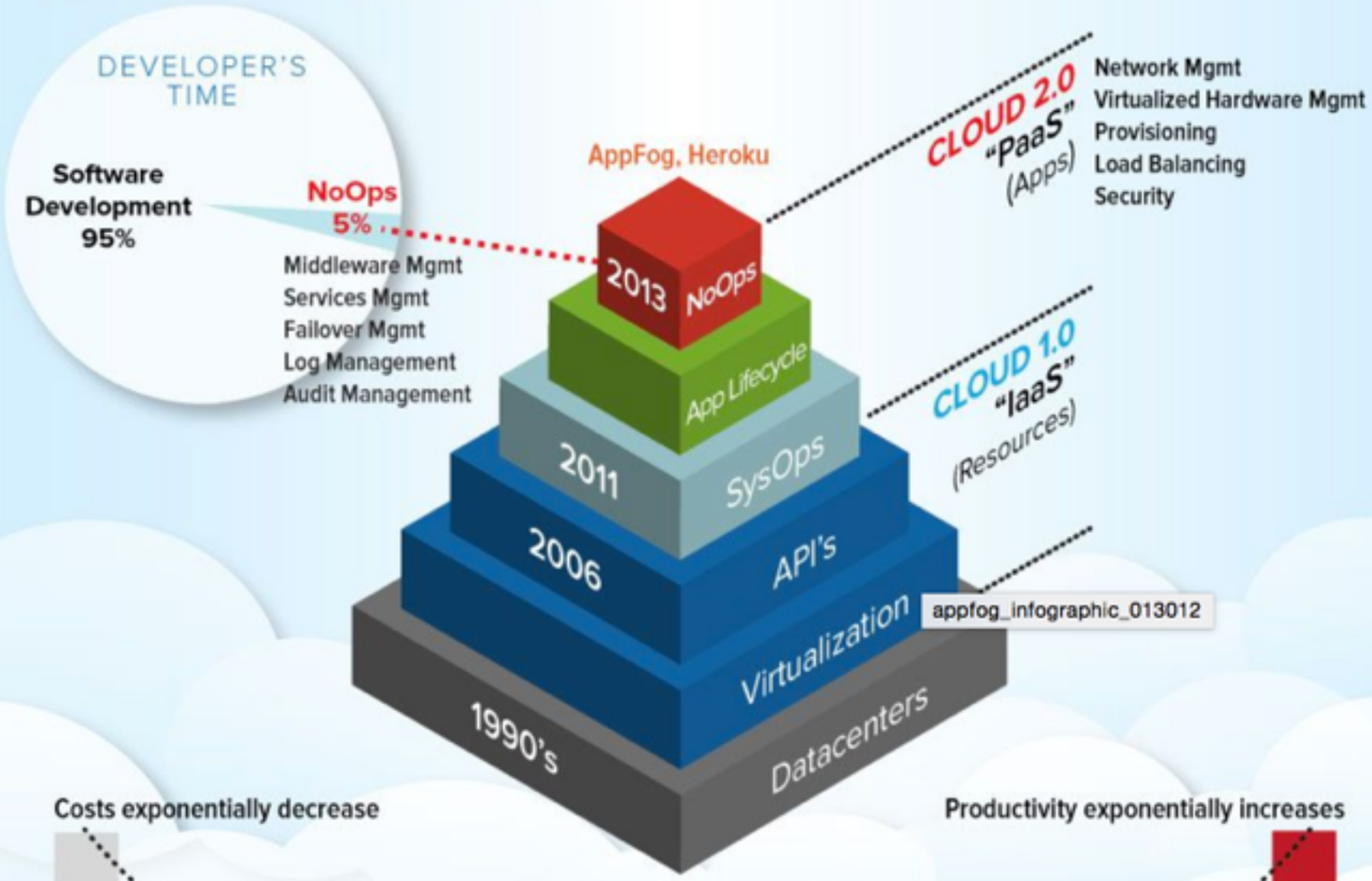
NoOps

- Service replication (k8s, fabric8, etcd, ZK, …)
- Dependency resolution (Nexus, …)
- Failover (Circuit Breaker)
- Resiliency (Circuit Breaker)
- Service monitoring, alerts and events (New Relic, Log stash, …)

redhat

2013: A bright NoOps future

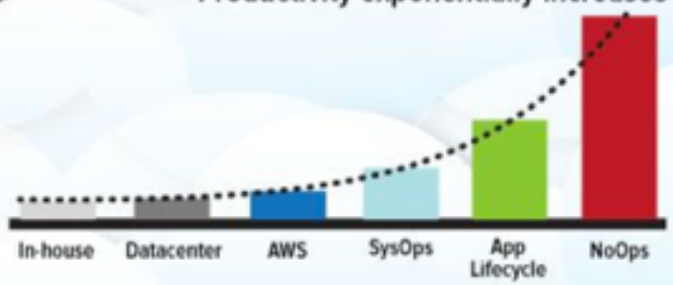
So where does this all lead? The end-game is NoOps. Where building and running an app is purely a developer process — and where developers are not having to spend time doing Ops work.



Costs exponentially decrease

Sources: National Venture Capital Association and Center for Venture Research

Productivity exponentially increases



Sources: National Venture Capital Association and Center for Venture Research

NoOps is the future. But it's a very near future. NoOps, typified by managed Platform-as-a-Service offerings, is the fully realized promise of the cloud. It has the most capex cost reduction of any cloud technology and combines it with the highest amount of productivity, velocity, and development agility.

NoOps takes all of the exponential gains from each layer of this Cloud 2.0 and passing them along to developers as one, rolled-up, massive reduction in cost, time and risk.

By freeing developers to focus on what they want to do — and on what they do best — we are enabling them to even more easily start companies and build products. And that is why AppFog is a *Start-up Enabler.*

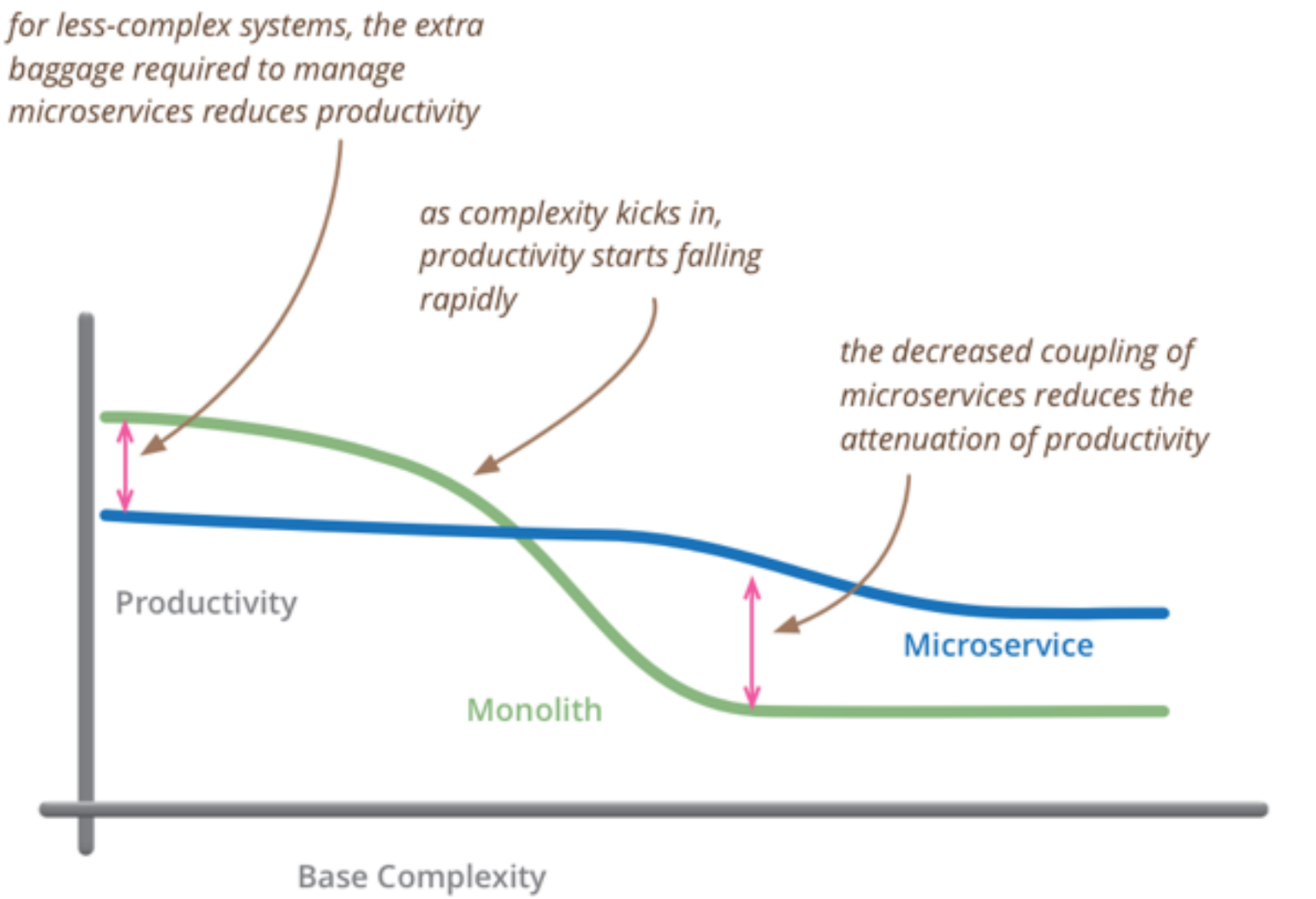
https://gigaom.com/2012/01/31/why-2013-is-the-year-of-noops-for-programmers-infographic/

redhat

Drawbacks of microservices

- Additional complexity of distributed systems
- Significant operational complexity, need high-level of automation
- Rollout plan to coordinate deployments
- Slower ROI, to begin with

Microservice Premium



"don't even consider microservices unless you have a system that's too complex to manage as a monolith"

http://martinfowler.com/bliki/MicroservicePremium.html

redhat

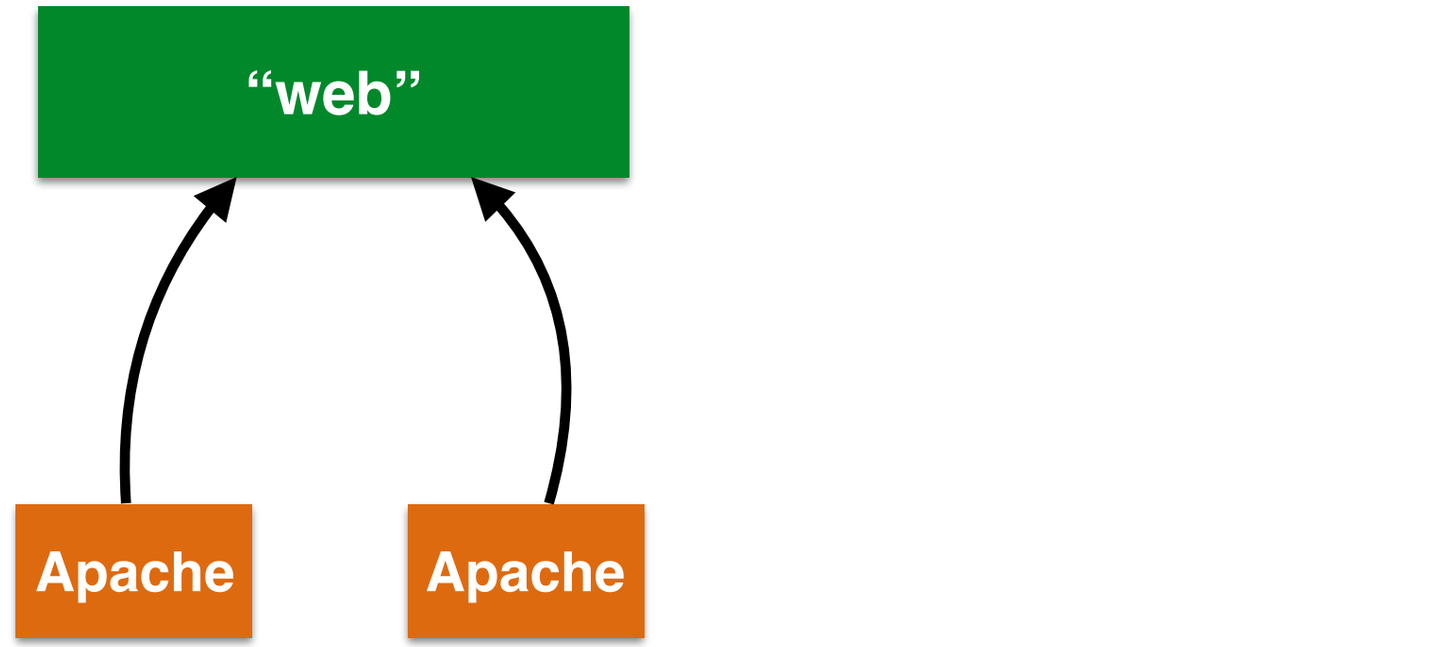
“web”

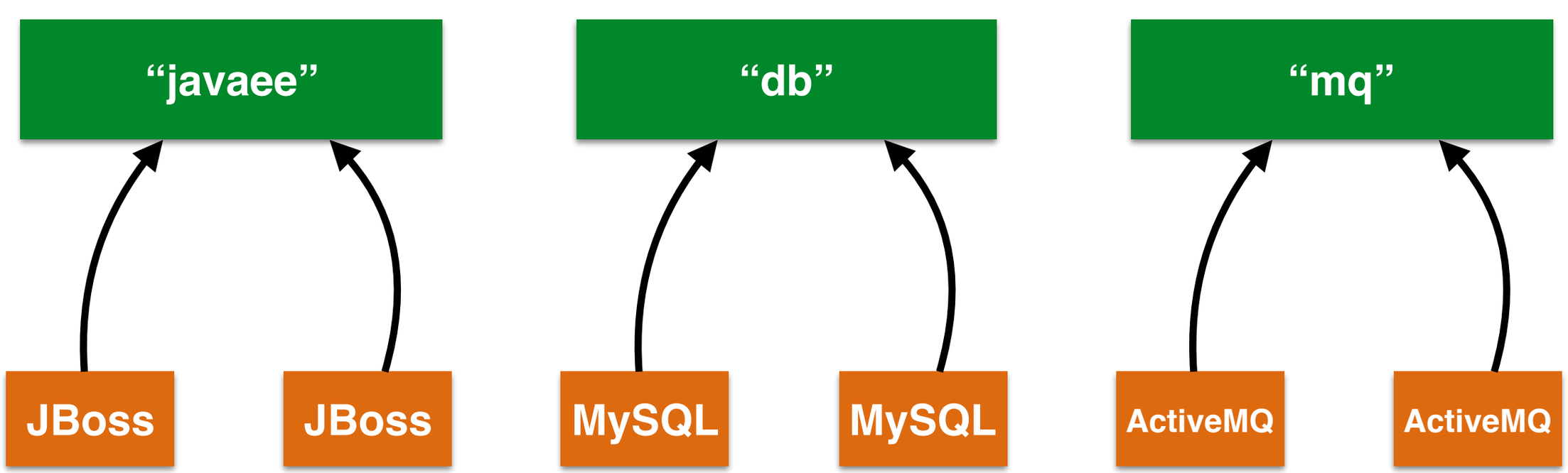
“javaee”

“db”

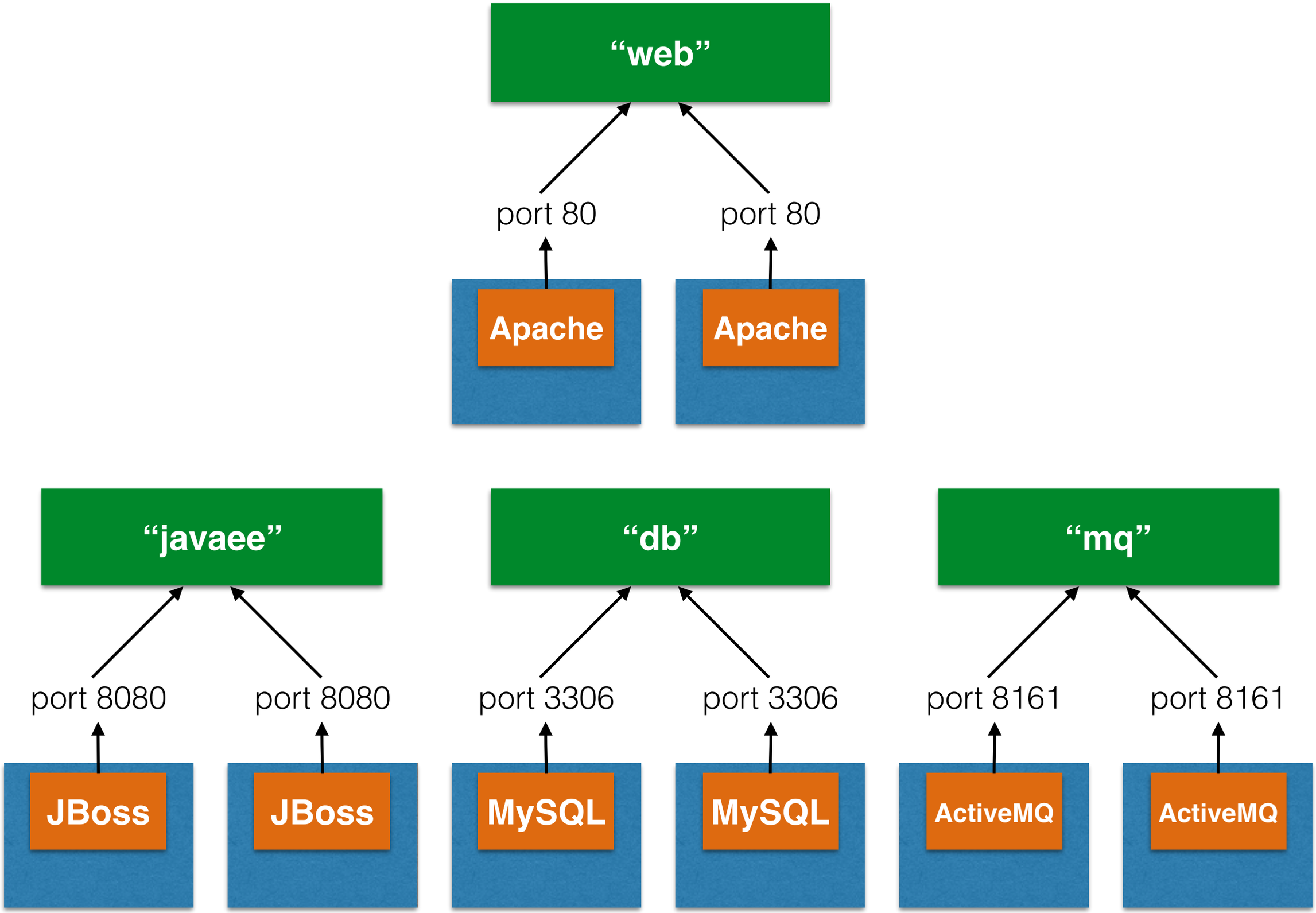
“mq”

redhat


“web”
Apache
Apache

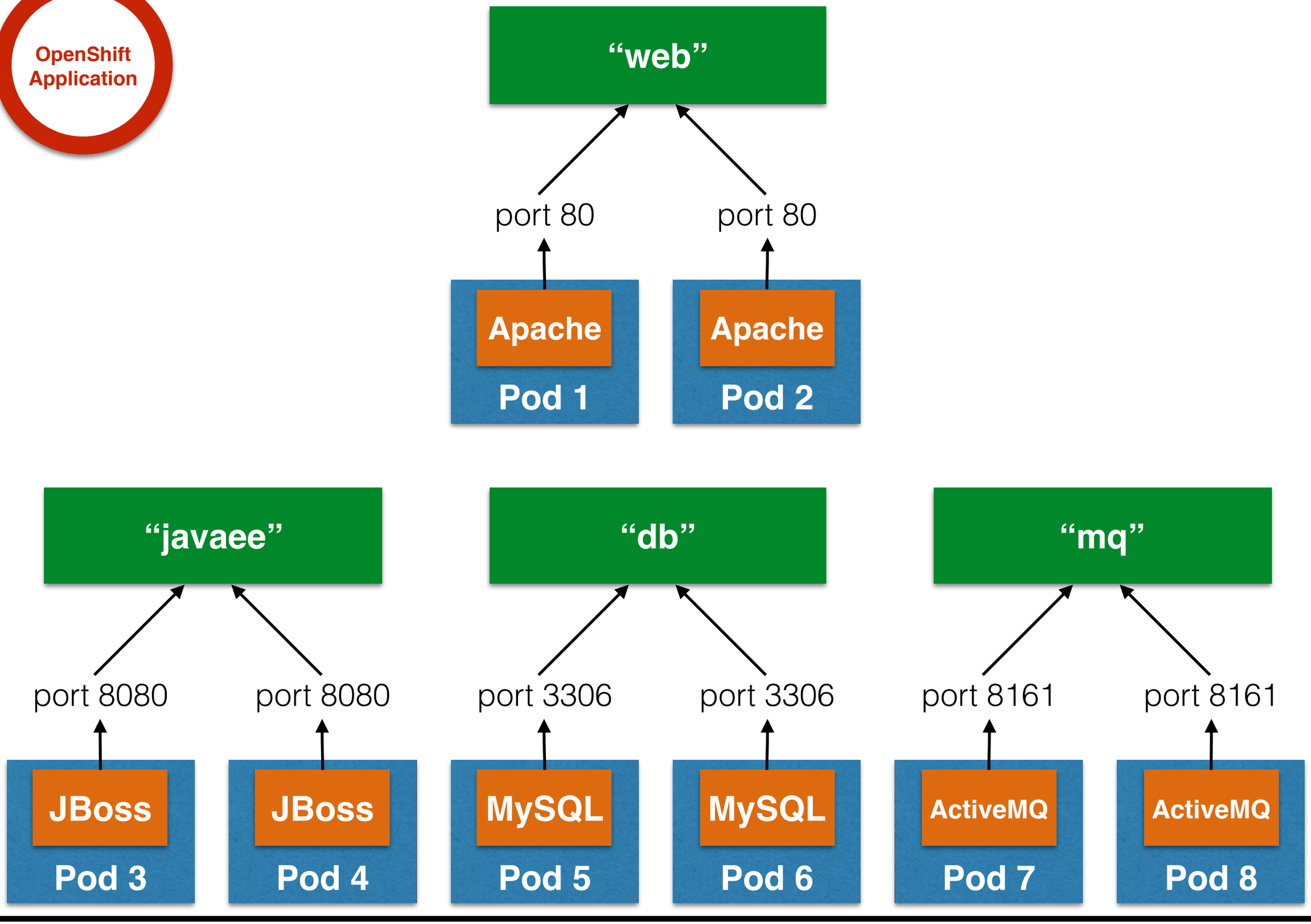

“javaee”
“db”
“mq”
JBoss
JBoss
MySQL
MySQL
ActiveMQ
ActiveMQ

redhat


“web”
port 80
port 80
Apache
Apache
“javaee”
“db”
“mq”
port 8080
port 8080
port 3306
port 3306
port 8161
port 8161
JBoss
JBoss
MySQL
MySQL
ActiveMQ
ActiveMQ

redhat

OpenShift Application



redhat

redhat.

Applications

PaaS

Containers & Orchestration

Container Host

IaaS

xPaaS

RED HAT JBOSS MIDDLEWARE

VERT.X

node.js

OPENSHIFT

kubernetes

docker

RED HAT ENTERPRISE LINUX ATOMIC HOST

openstack