Deployment – Done Right!

Eberhard Wolff
@ewolff
Fellow

innoQ

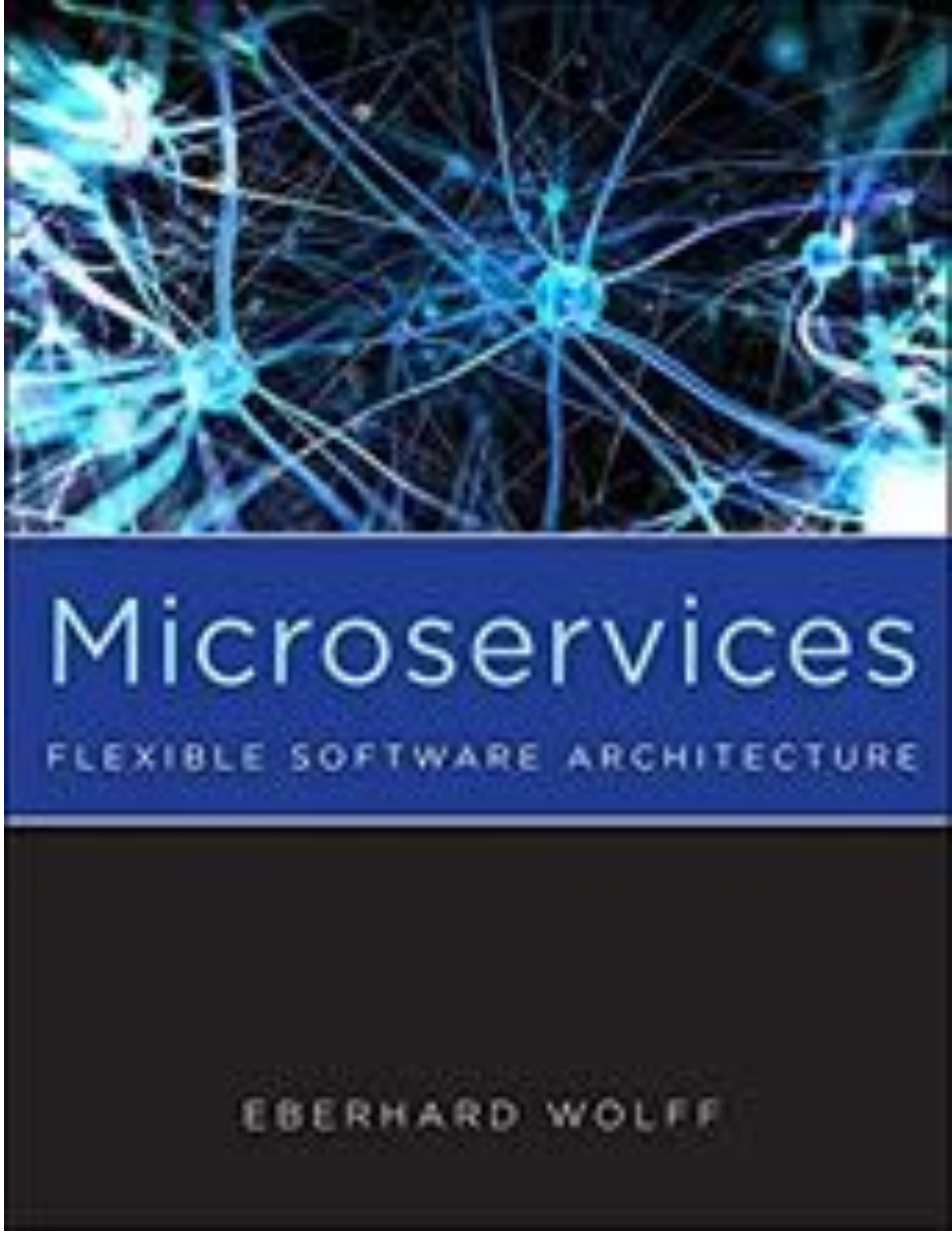
http://continuous-delivery-buch.de/

Eberhard Wolff

Microservices

Grundlagen flexibler Softwarearchitekturen

dpunkt.verlag



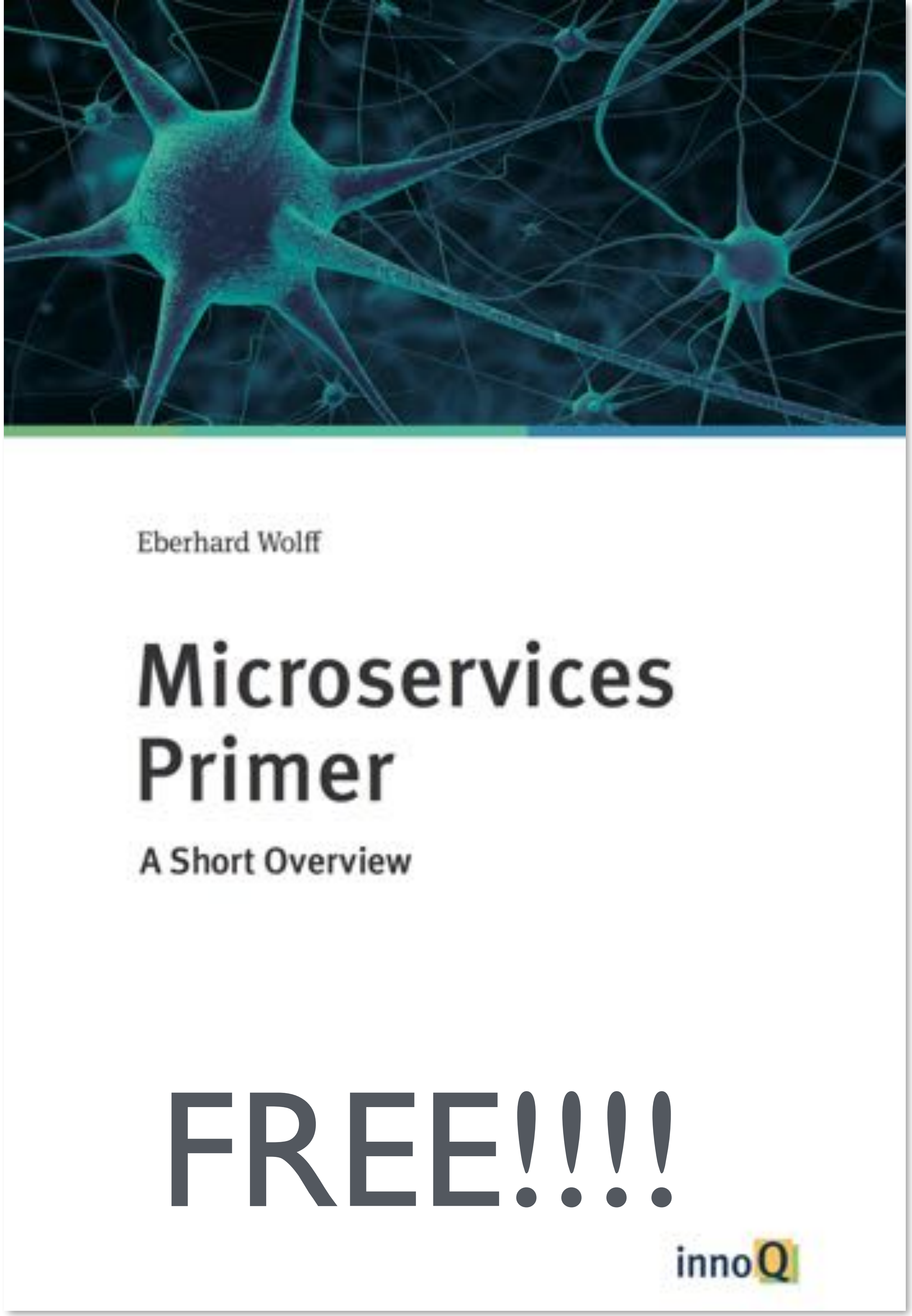
Microservices

FLEXIBLE SOFTWARE ARCHITECTURE

EBERHARD WOLFF

http://microservices-buch.de/

http://microservices-book.com/



Eberhard Wolff

Microservices Primer

A Short Overview

innoQ

FREE!!!!

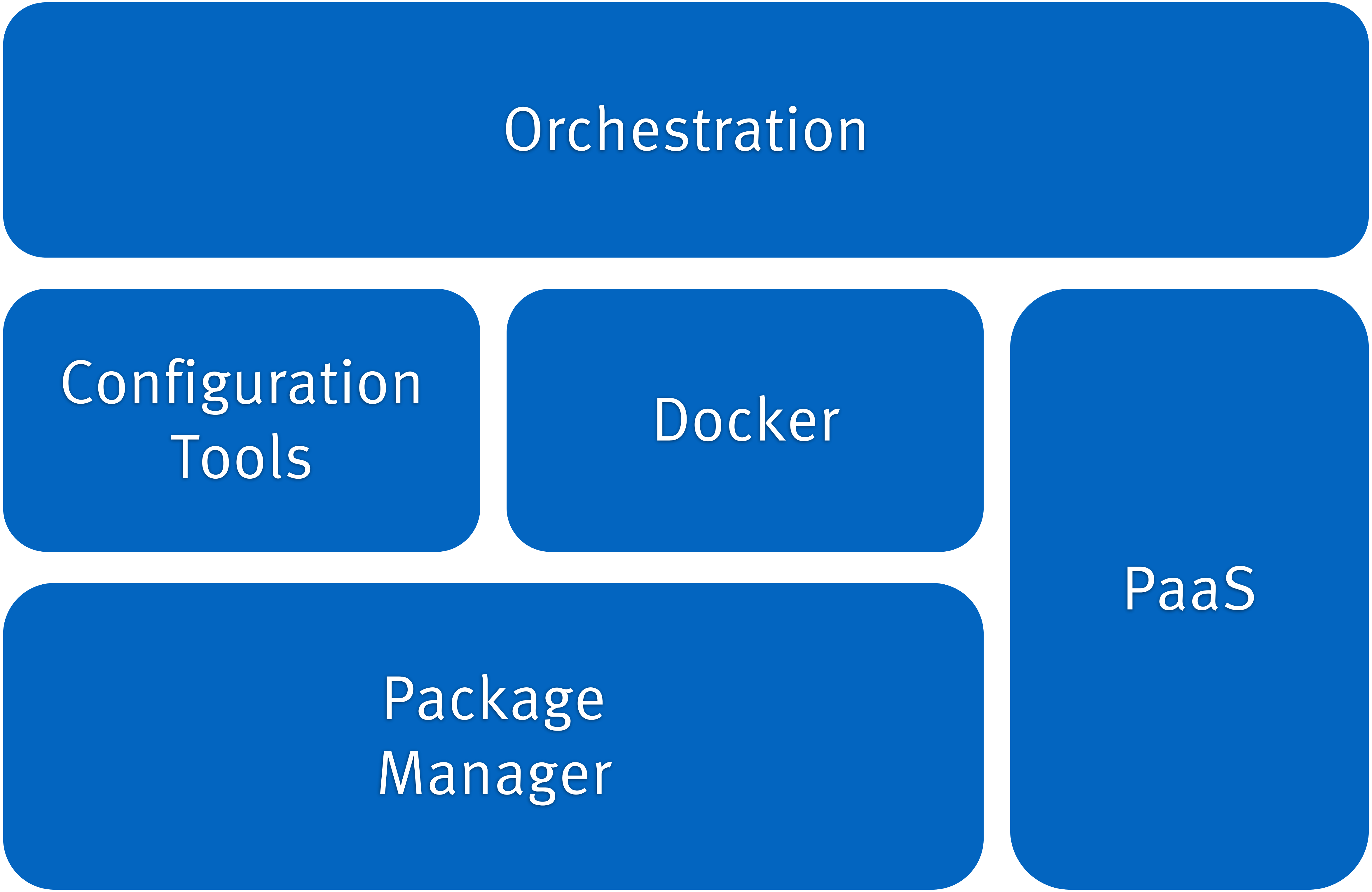
http://microservices-book.com/primer.html

Why This Talk?

- Numerous technologies
- Deployment: pretty old problem
- High productivity gains possible
- Microservices: A lot more to deploy

Why This Talk?

- Easier deployment = easier to change software
- Goal of software architecture
- Dev moving towards DevOps
- Infrastructure as Code
- i.e. deployment becomes part of software architecture



Orchestration

Configuration Tools

Docker

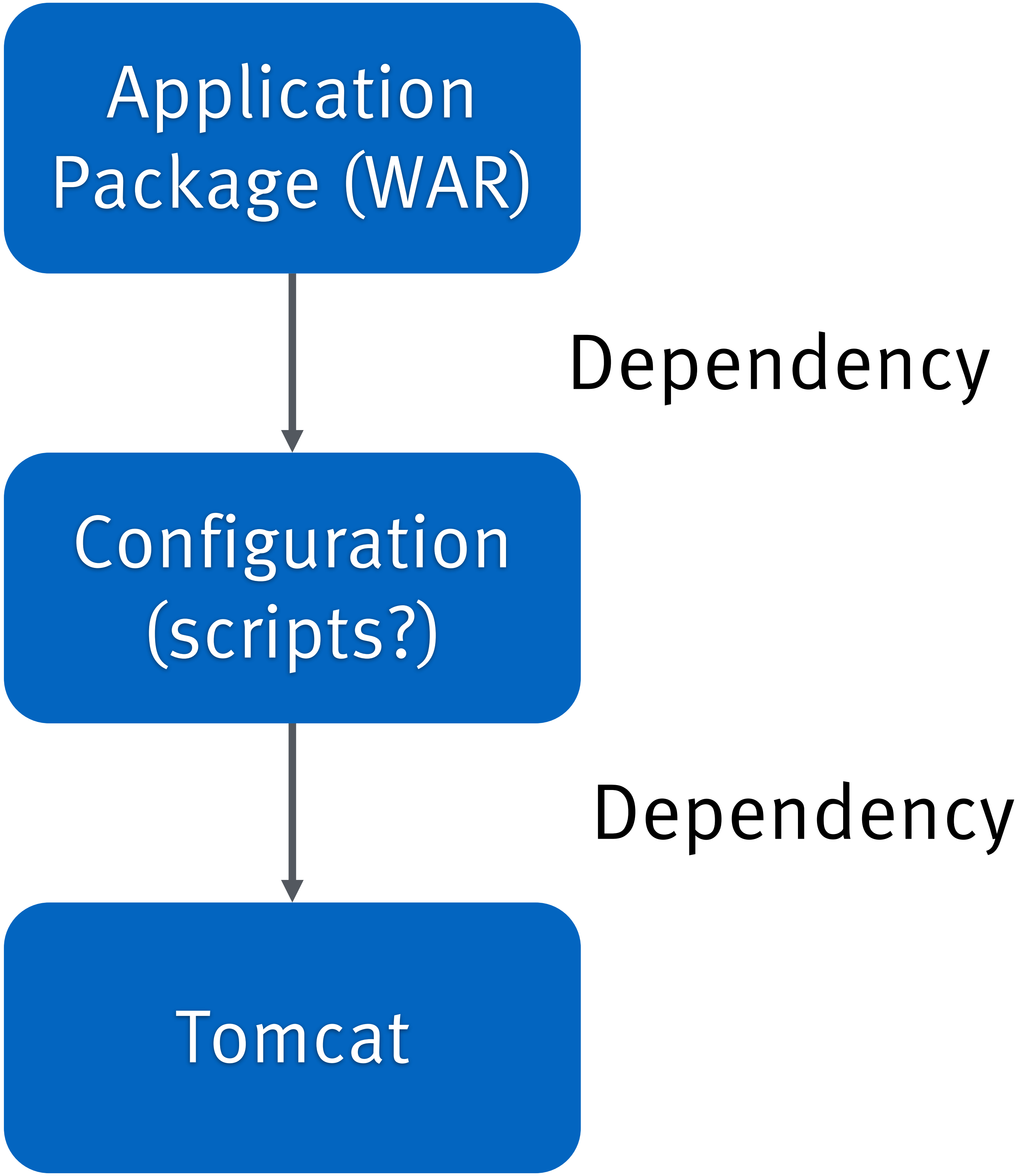
PaaS

Package Manager

Package Managers

Package Manager

- Simple format
- Easy to distribute to many servers
- Packages software
- Defines dependencies
- Can also run scripts
- Can also include configurations



Application Package (WAR)

Dependency

Configuration (scripts?)

Dependency

Tomcat

Package Manager: Challenges

- Usually depend on OS and Linux distro
- deb for Debian, msi for Windows etc.

- What if the installation aborts?
- What if the system is inconsistent?

- Dependencies might cause problems

Package Manager: Challenges

- No ideal tool for configuration

- Manual changes might be needed
- Systems might become inconsistent

- What about wiring the hosts together?

Package Managers are widely used

Package Managers
are widely used
...and only a part of
the full solution

Nix

- Runs on Linux and Mac OS X
- Declarative approach: Define desired state
- Atomic upgrades and rollbacks
- Rollbacks enable experiments

- Fixes some issues

Ubuntu Snappy / Flatpak

- Package application with all libraries
- ...instead of defining dependencies

- Applications are portable across distros
- Enables rollbacks...
- ...independent updates of distro and app

Ubuntu Snappy / Flatpak

- Packaging is all about dependencies
- Basically no packages and dependencies any more
- Self-contained like Docker images

FLATPAK

snappy

What we still need…

- Configuration
- Updates? Rollbacks?
- Make sure installation was successful

Configuration

- Templates

- Configuration database catalogs servers
- E.g. add web servers to load balancer

- ✔ Configuration
- Updates? Rollbacks?
- Make sure installation was successful

Idempotency

- Idea: Start installation as often as you like
- …result is the same
- First run on fresh system: install everything
- Run directly afterwards: Do nothing

Idempotency

- Configuration describes desired state
- Updates become trivial

Idempotency

- Not: Install this package!
- But: This package should be installed
- Package already Installed ➜ do nothing
- Package not installed ➜ install

- Not: Create this file!
- But: File should look like this.

✔ › Configuration

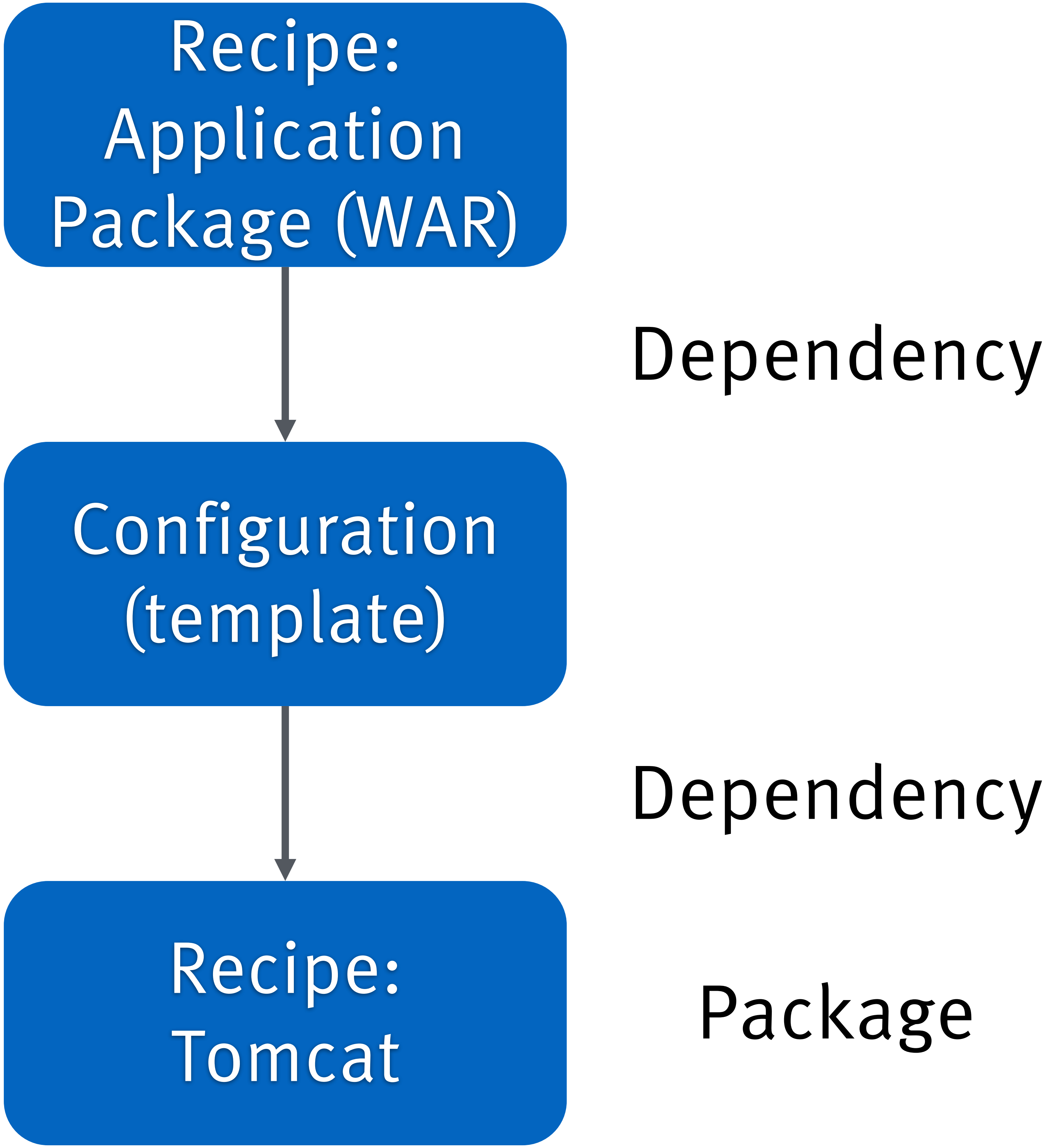
✔ › Updates? Rollbacks?

✔ › Make sure installation was successful

Chef

- Idempotent
- Templates for configuration
- Centralized server to store information

Similar Tools



Recipe: Application Package (WAR)

Dependency

Configuration (template)

Dependency

Recipe: Tomcat

Package

Idempotency: Challenges

- Configuration complex
- No simple script
- …but description of desired state

Idempotency: Challenges

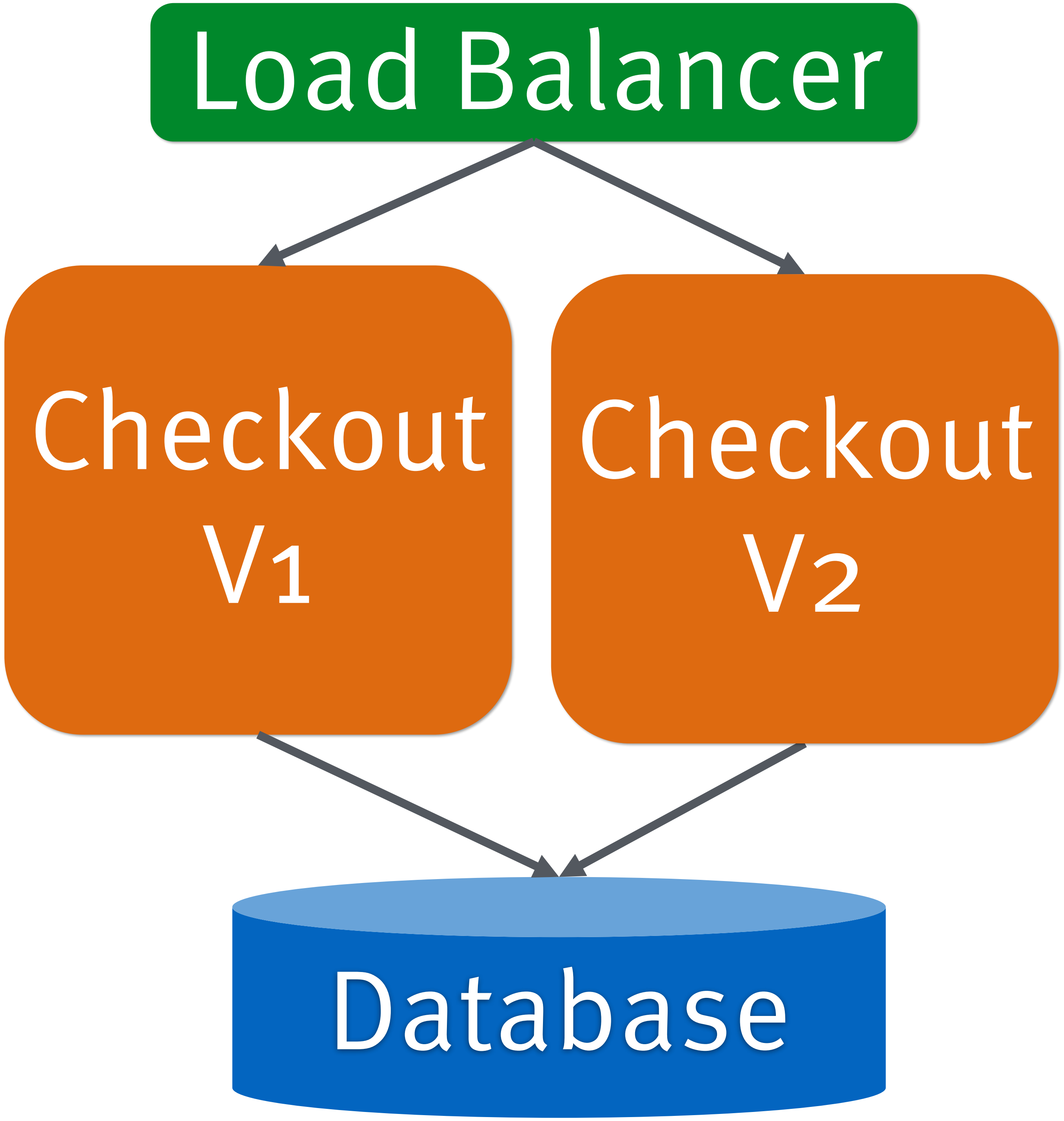
- Not described? Won’t be “repaired”

- Servers are updated stepwise
- …not from scratch
- Not every starting point might work

- Really reproducible?

Why Updates??

Checkout V2


Load Balancer
Checkout V1
Checkout V2
Database

or Cattle?
Replace

Pet
Update

No more updates!

Immutable Server

- Never change a server
- Never update a server
- Always install from scratch

- Fully and reliably reproducible
- Very simple concept
- i.e. no sophisticated tools

Immutable Server seem like a lot of overhead – how could they work?

Application Deployment

- Install operating system
- Install Tomcat
- Change configuration files
- Add application

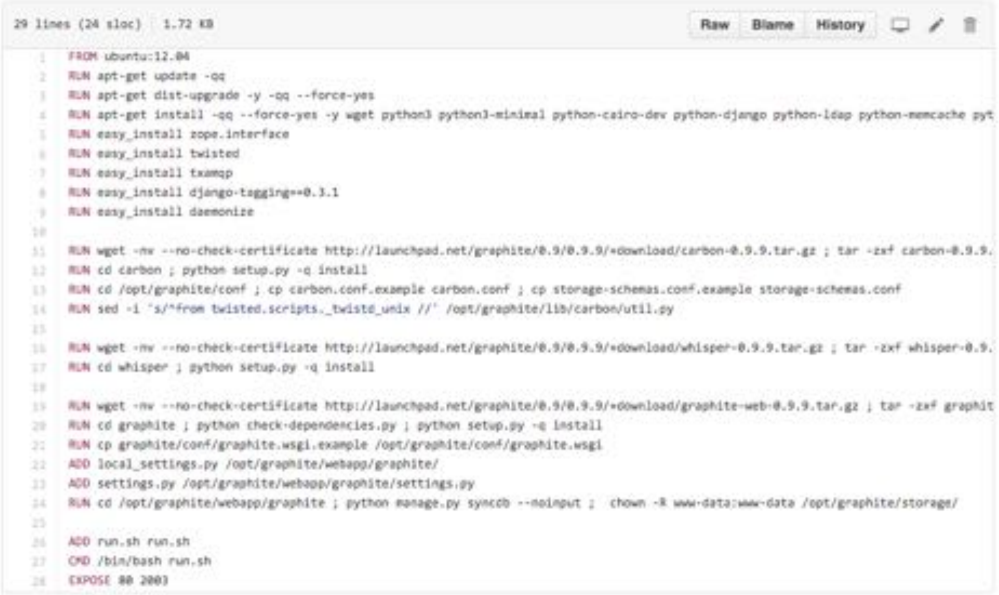
Stable

Volatile

Easy to write a script

Docker = Simple Deployment

- Dockerfile
- Just a shell script
- Installation from scratch

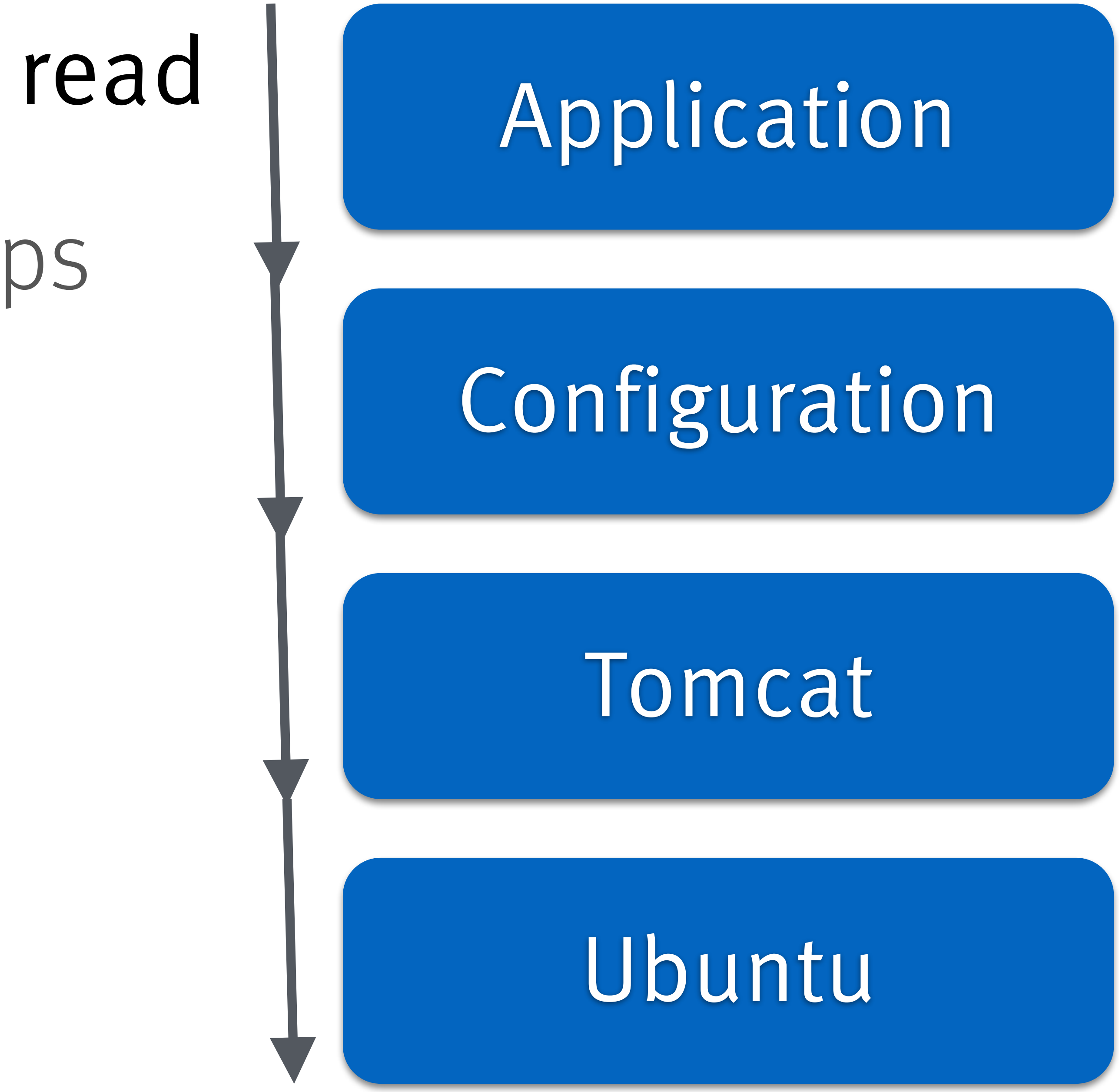


```
FROM ubuntu:12.04
RUN apt-get update -qq
RUN apt-get dist-upgrade -y -qq --force-yes
RUN apt-get install -qq --force-yes -y wget python3 python3-minimal python-cairo-dev python-django python-ldap python-memcache pyt
RUN easy_install zope.interface
RUN easy_install twisted
RUN easy_install txamqp
RUN easy_install django-tagging==0.3.1
RUN easy_install daemonize

RUN wget -nv --no-check-certificate http://launchpad.net/graphite/0.9/0.9.9/+download/carbon-0.9.9.tar.gz ; tar -zxf carbon-0.9.9.
RUN cd carbon ; python setup.py -q install
RUN cd /opt/graphite/conf ; cp carbon.conf.example carbon.conf ; cp storage-schemas.conf.example storage-schemas.conf
RUN sed -i 's/^from twisted.scripts._twistd_unix //' /opt/graphite/lib/carbon/util.py

RUN wget -nv --no-check-certificate http://launchpad.net/graphite/0.9/0.9.9/+download/whisper-0.9.9.tar.gz ; tar -zxf whisper-0.9.
RUN cd whisper ; python setup.py -q install

RUN wget -nv --no-check-certificate http://launchpad.net/graphite/0.9/0.9.9/+download/graphite-web-0.9.9.tar.gz ; tar -zxf graphit
RUN cd graphite ; python check-dependencies.py ; python setup.py -q install
RUN cp graphite/conf/graphite.wsgi.example /opt/graphite/conf/graphite.wsgi
ADD local_settings.py /opt/graphite/webapp/graphite/
ADD settings.py /opt/graphite/webapp/graphite/settings.py
RUN cd /opt/graphite/webapp/graphite ; python manage.py syncdb --noinput ;  chown -R www-data:www-data /opt/graphite/storage/

ADD run.sh run.sh
CMD /bin/bash run.sh
EXPOSE 80 2003
```

- Behind the scenes: Optimization
- Every Dockerfile line = filesystem snapshot
- Reuse snapshots for all other Dockerfiles

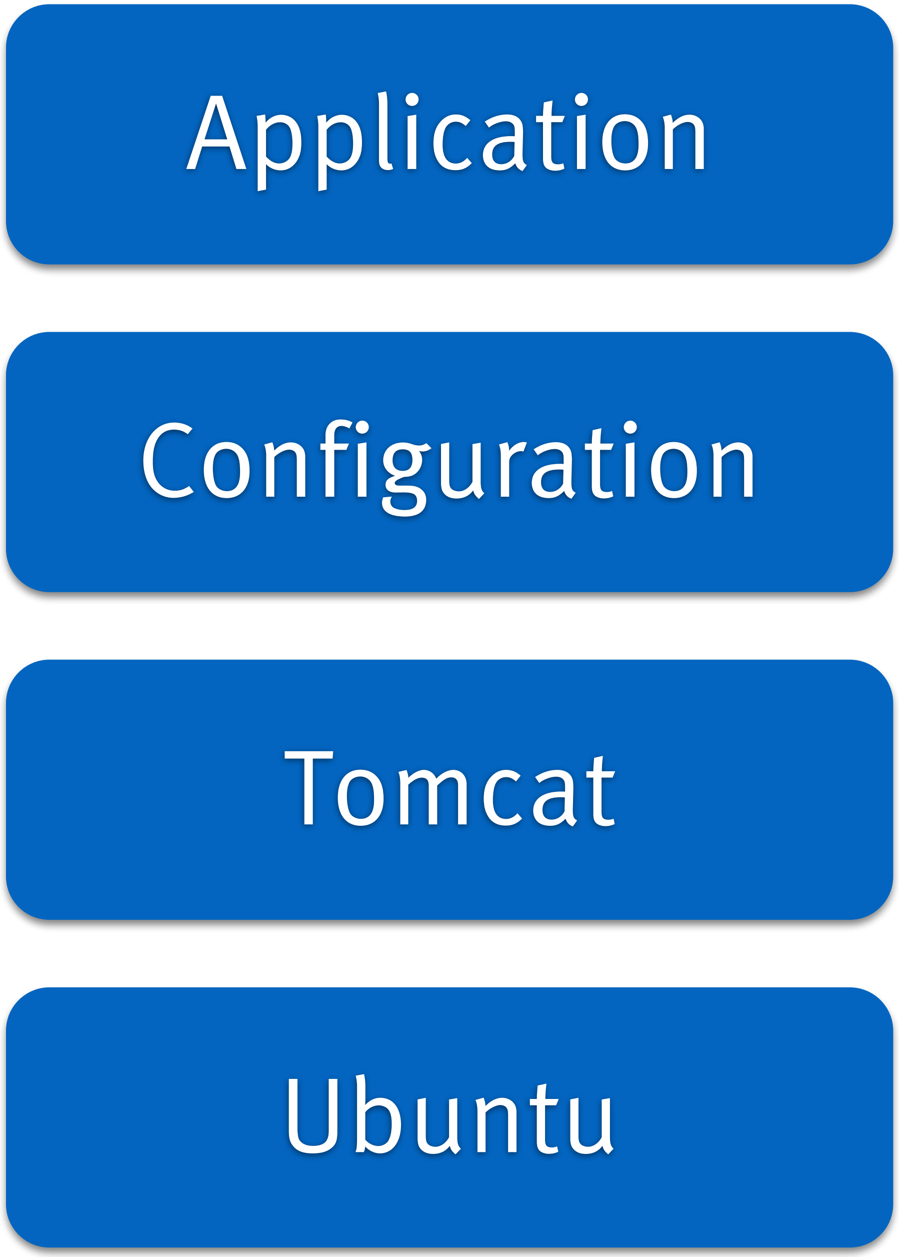
Docker Filesystem

- Each installation steps creates a file system snapshot
- Read go through all layers in order



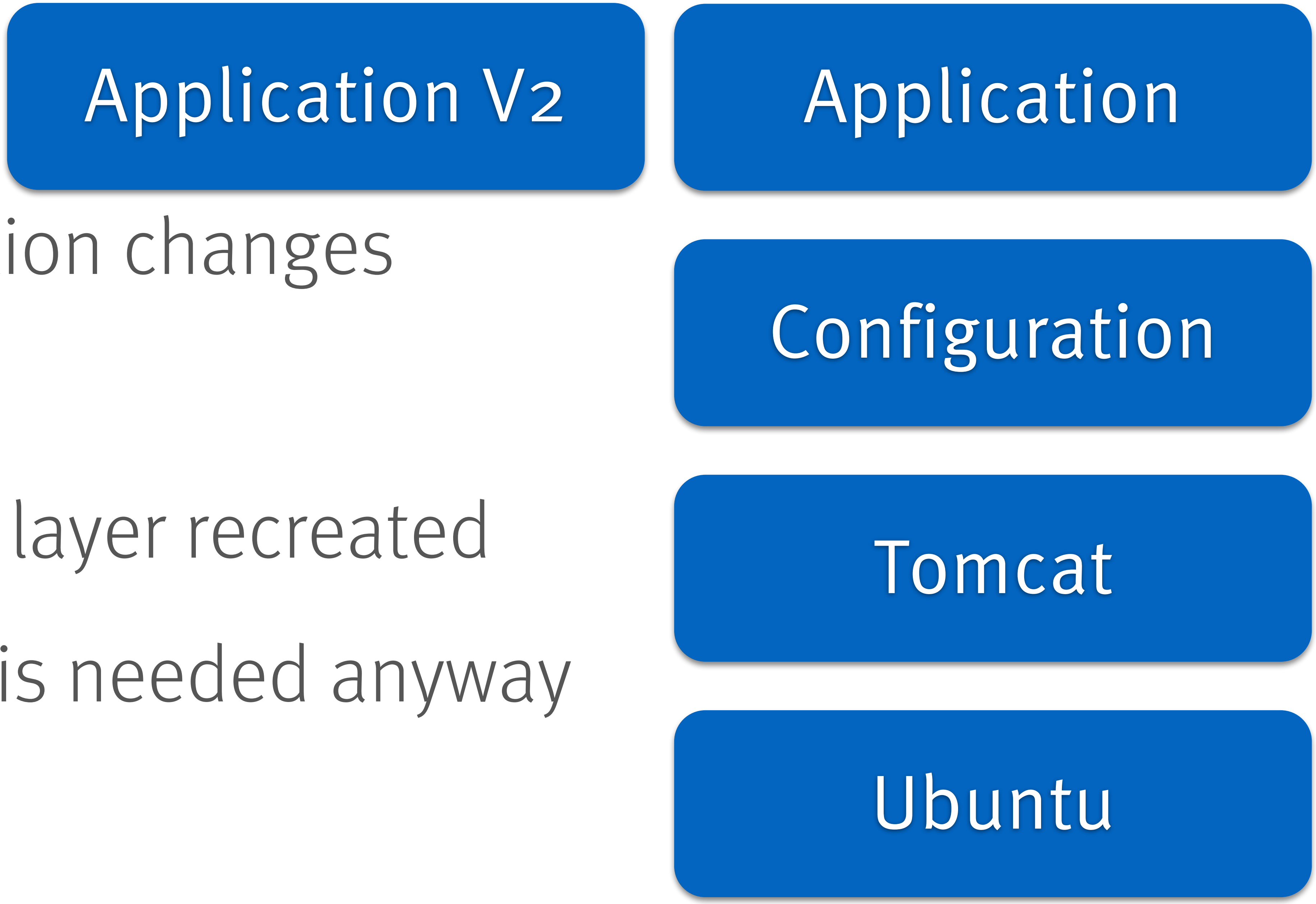
Docker Filesystem

- Rerun installation
- Nothing happens
- Everything is cached



Docker Filesystem

- Application changes
- Just last layer recreated
- ...which is needed anyway



You could use
Chef + Docker

You could use
Chef + Docker
...but why?

Chef… might be useful to deploy Docker hosts.

You think you are creating a server from scratch…

You think you are creating a server from scratch…

…but everything is cached

Why care about servers?

Why Care About Server?

- Applications are what provides value

- Standardize environment
- …and deploy application on the environment

Issues With PaaS

- Standardized infrastructure
- Not flexible
- Hard to migrate existing applications

- Installing PaaS on premise hard
- Enterprise=On-Premise

- Huge success for apps in the Public Cloud

On Premise

OPENSHIFT

CLOUD FOUNDRY™

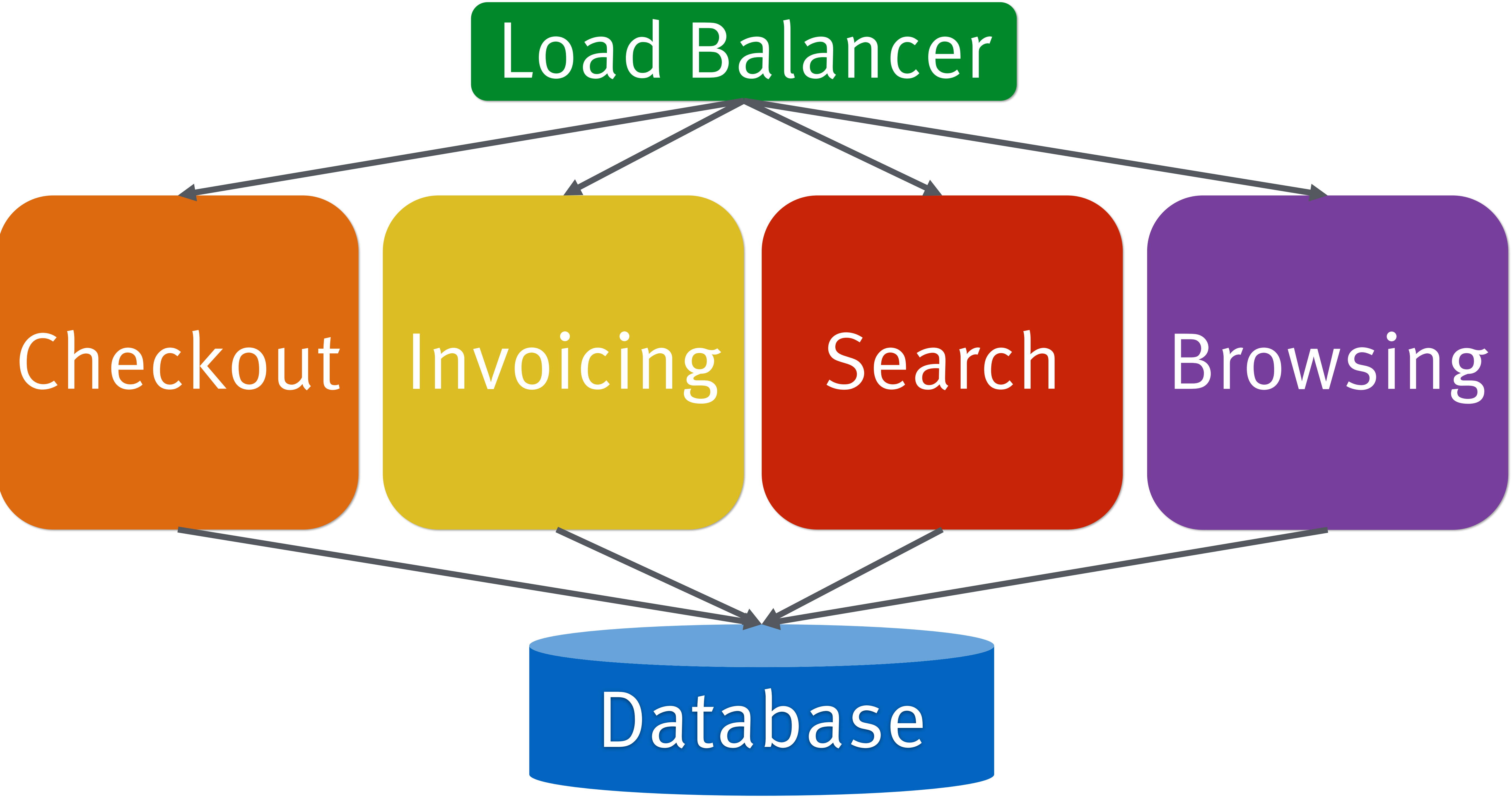
Public Cloud

Amazon Elastic Beanstalk

HEROKU

Orchestration

Individual Boxes



Orchestration

Orchestration

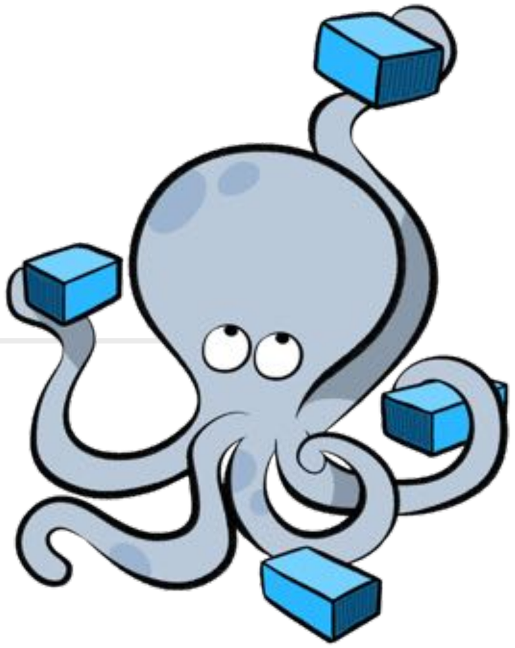
- Deploy many machines
- Create network
- Support many different offerings
- E.g. PaaS, Docker, IaaS…

Terraform

- Infrastructure e.g. cloud, virtualization
- PaaS
- Network
- SaaS (DNS, CDN)

- Similar: Amazon Cloud Formation

Docker Compose



- Coordinate many containers
- Define containers…
- …and networking

- Might use Docker Swarm for cluster
- Limited to Docker, though

Kubernetes

- Pods: Wired Docker containers
- Service discovery
- Configuration
- Load balancing
- Limited to Docker

Conclusion

Package Managers

☀ Well established

☀ Can deploy large server farms

☂ Updates?

☂ Configuration?

New approaches are self-contained and enable rollbacks

Idempotent Tools

☀ Updates

☀ Configuration / templates

☀ Book keeping of servers

☂ Truly reproducible?

☂ Complex?

Updates = pets = anti-pattern

Docker

☀ Enable immutable server

☀ Simple scripts

☀ Tools for coordination

☂ New infrastructure (e.g. Kubernetes)

Updates = pets = anti-pattern

Orchestration

- Needed to create the rest of the environment

- Terraform
- ...or a Docker tool

EMail conli2016@ewolff.com to get:

Slides

+ Microservices Primer

+ Sample Microservices Book

+ Sample of Continuous Delivery Book

Powered by Amazon Lambda & Microservices