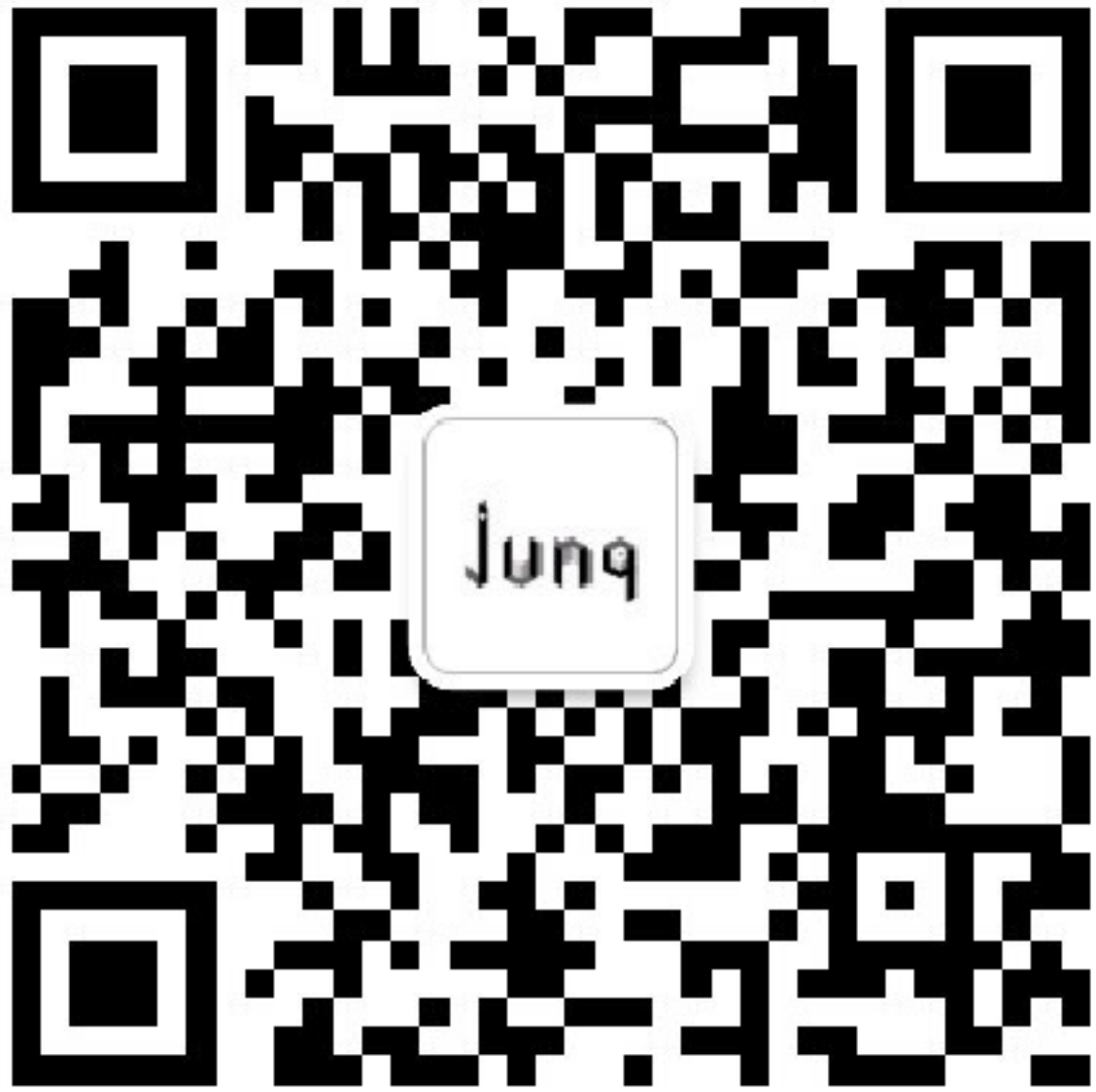
StuQueue
斯达克学院

跟我做一个Java微服务项目

Week #7 微服务部署 / 刘俊强

StuQueue
斯达克学院

欢迎关注StuQ微信公众号



欢迎关注我的微信公众号 😋

大纲

- 微服务部署模式
- 部署流程与工具

微服务部署关注点

- 语言多样性：服务可使用适当的语言/框架进行编写
- 可用性与吞吐量：每个服务拥有多个服务实例
- 速度：构建和部署服务速度要快
- 伸缩性：服务必须独立部署和横向扩展
- 隔离性：服务实例间相互隔离，故障不相互影响
- 资源限制：服务所消耗的资源需要被限制
- 部署需要高性价比

微服务部署模式

微服务部署模式

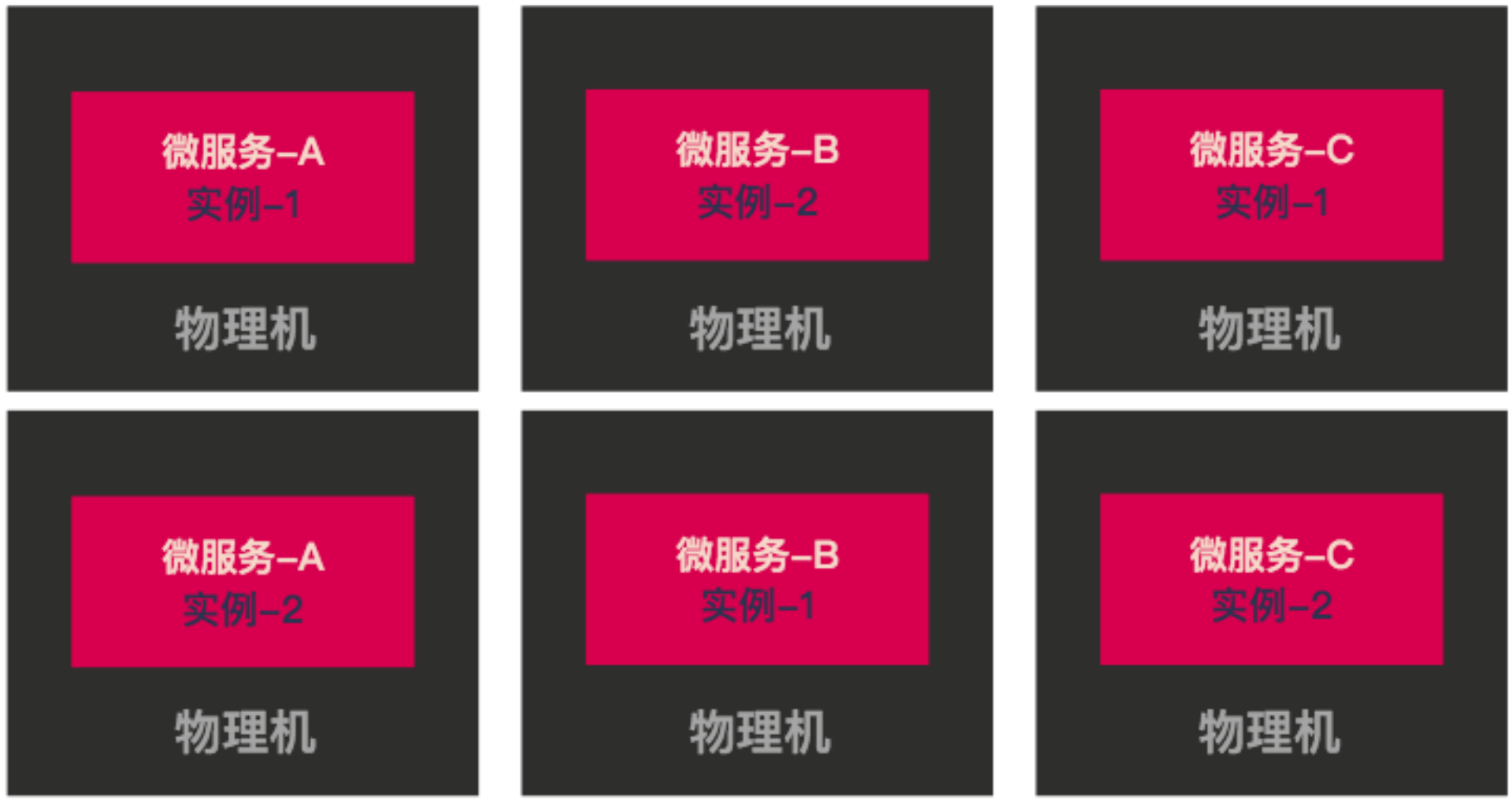
- 单主机部署多个服务实例
- 单物理机部署单个服务实例
- 单虚拟机部署单个服务实例
- 单容器部署单个服务实例

单主机部署多个服务实例

单主机部署多个服务实例

- 资源有效利用
- 部署速度快
- 隔离性差
- 限制资源使用难
- 版本依赖版本冲突风险
- 封装技术老

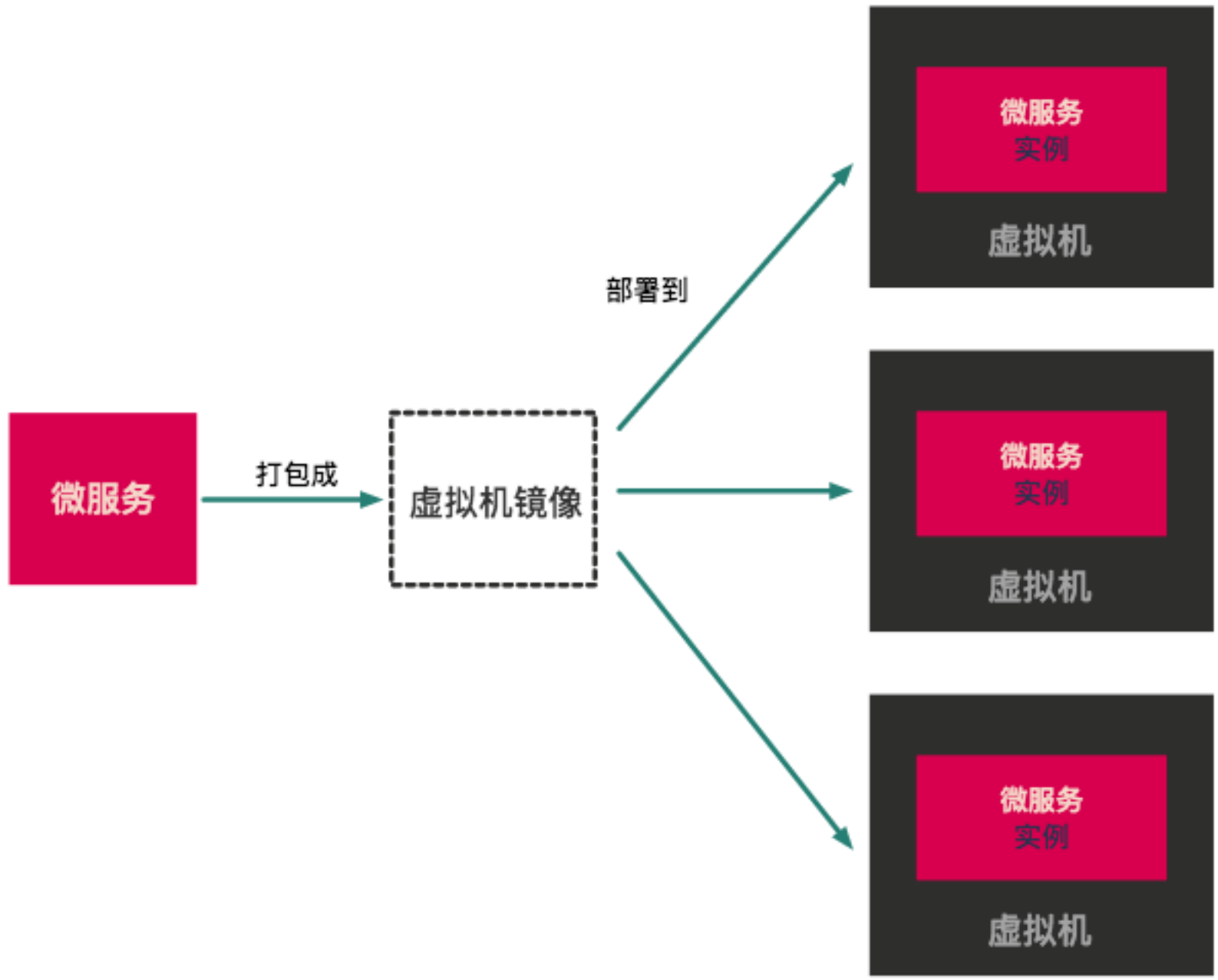
单物理机部署单个服务实例



单物理机部署单个服务实例

- 部署速度快
- 隔离效果好
- 资源利用率不高
- 封装技术老

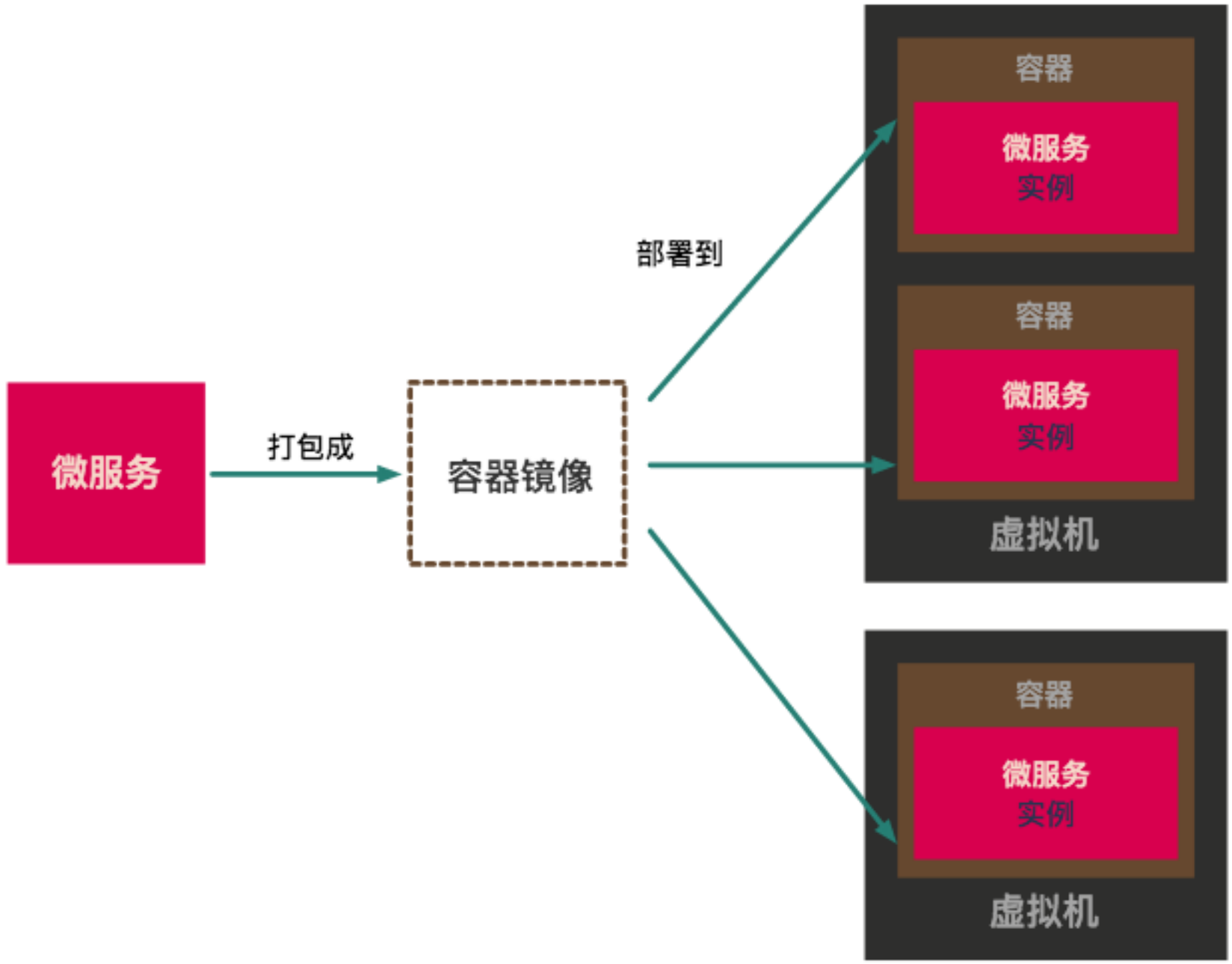
单虚拟机部署单个服务实例



单虚拟机部署单个服务实例

- 隔离效果好
- 可管理性好
- 虚拟化封装技术
- 便于自动伸缩
- 资源利用率不高
- 部署速度慢

单容器部署单个服务实例



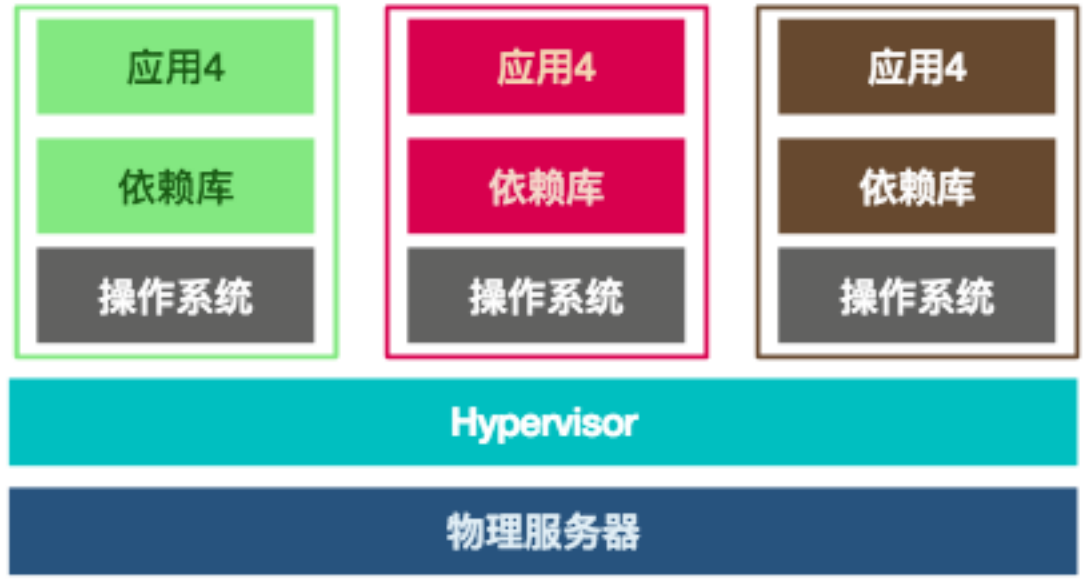
单容器部署单个服务实例

- 隔离效果好
- 可管理性好
- 容器化封装技术
- 资源利用率高
- 部署速度快

虚拟机 vs. 容器

虚拟机 Virtual Machine	容器 Container
内核隔离	共享Linux内核
设备驱动隔离	共享设备驱动
内存隔离	内存隔离
文件系统隔离	文件系统隔离
CPU使用限制	CPU使用限制
分钟级启动	秒级启动
几十个虚拟机	上百个容器

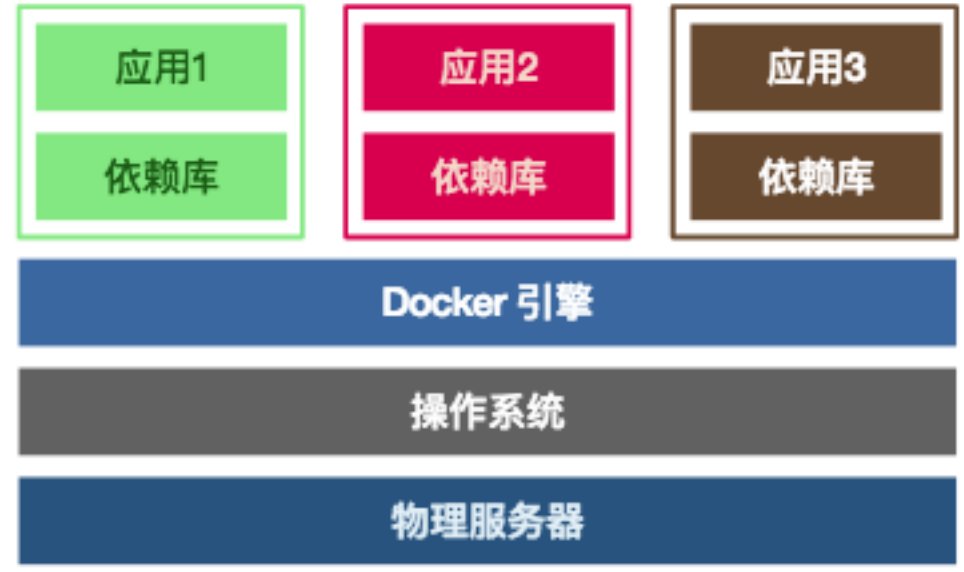
虚拟化 vs. 容器化



虚拟化

操作系统级别隔离

被证实、安全、资源密集

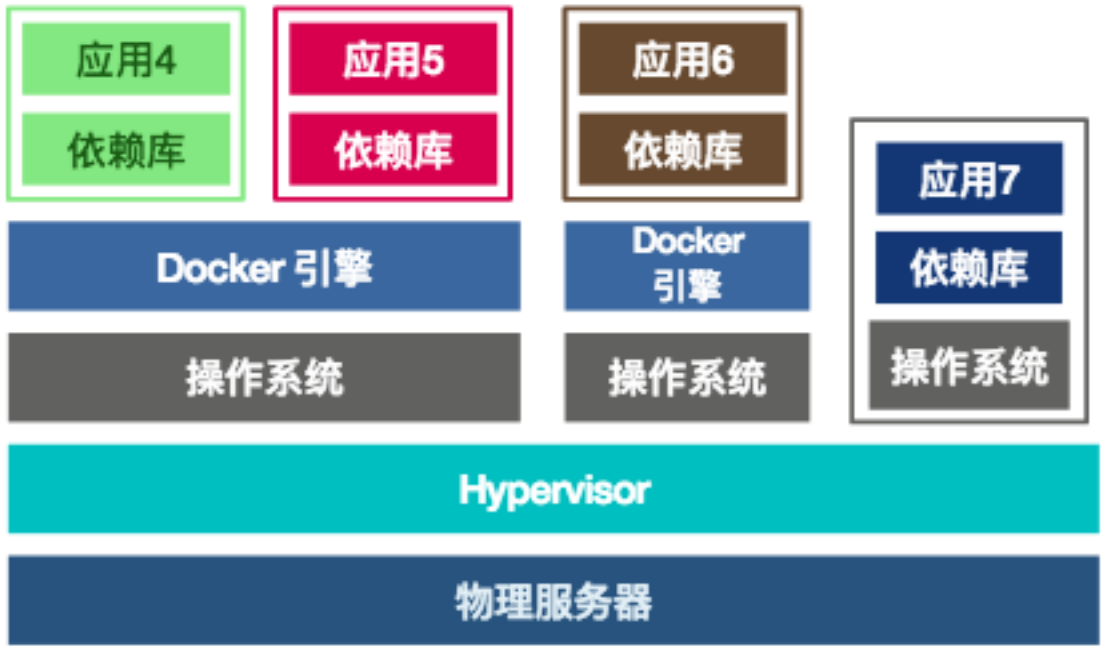


容器化

进程级别隔离

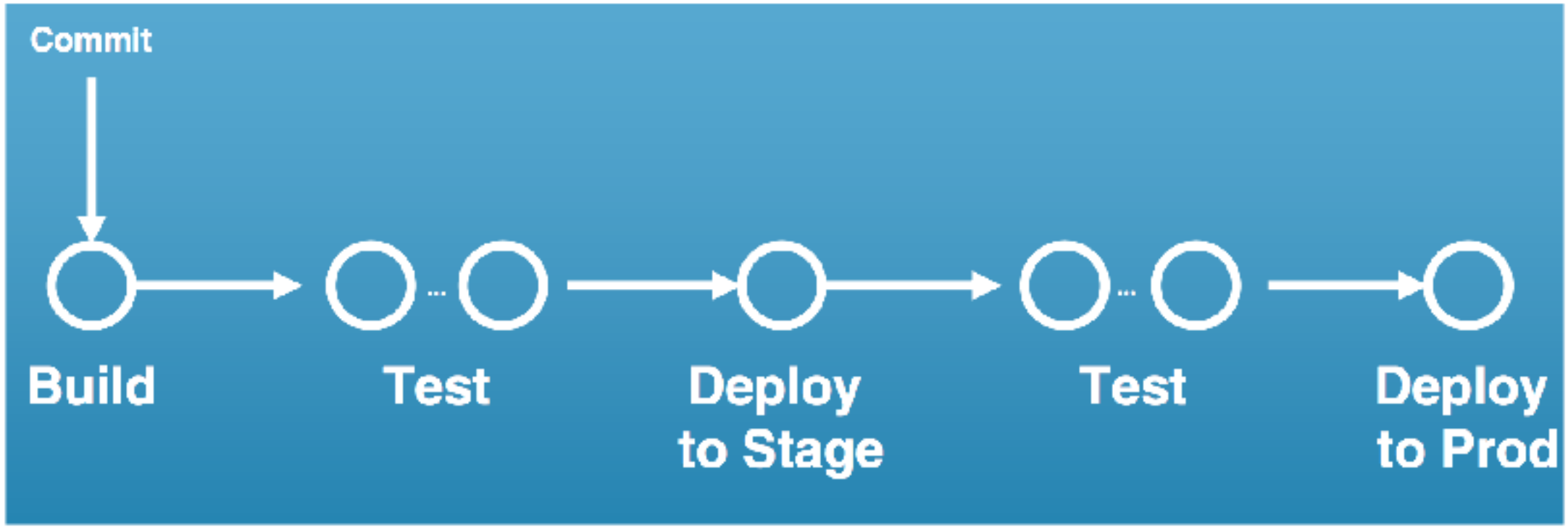
快速、简洁、开发者友好

虚拟化 and 容器化



部署流程与工具

从开发环境到生产环境

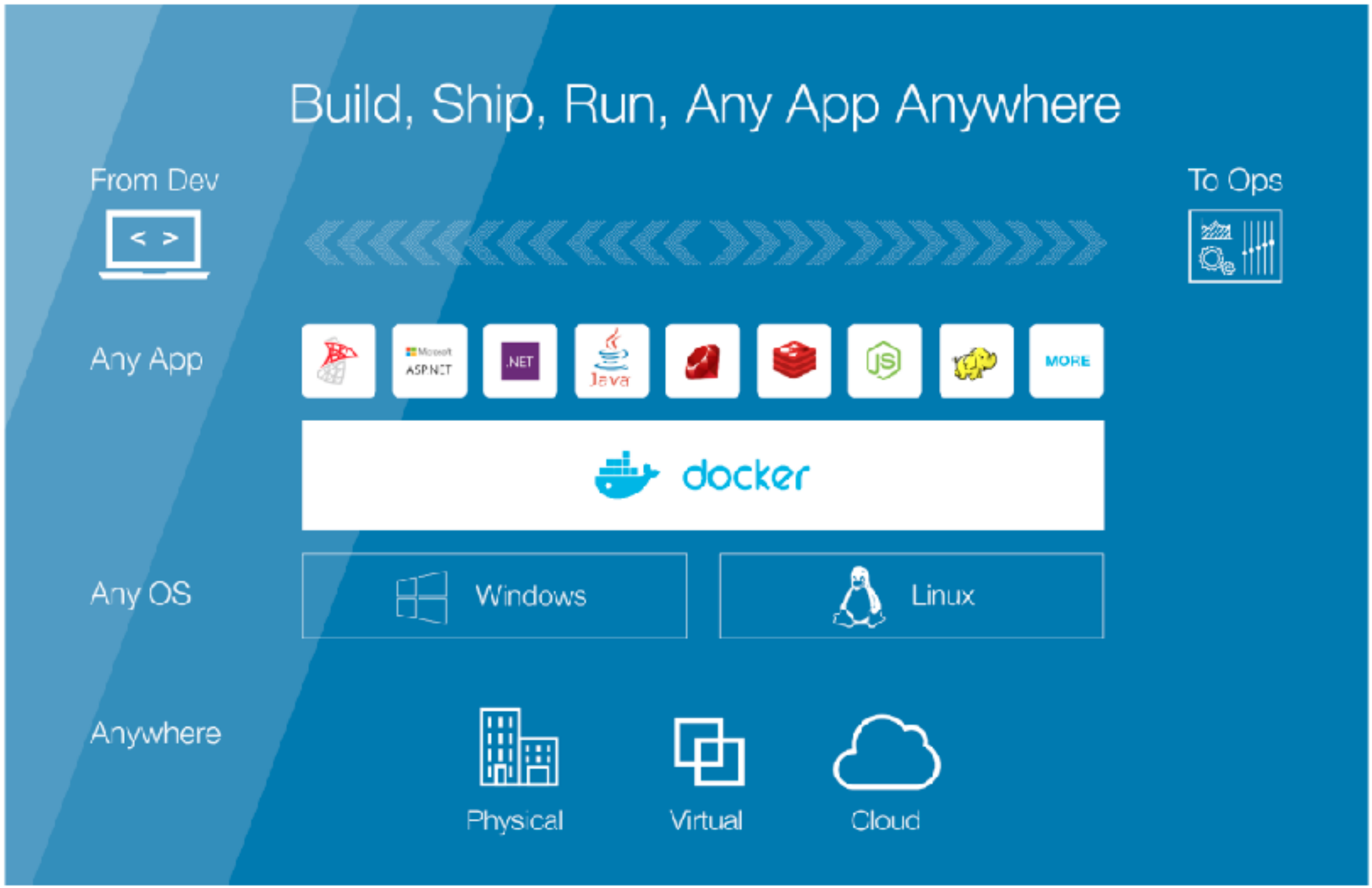


打包选项

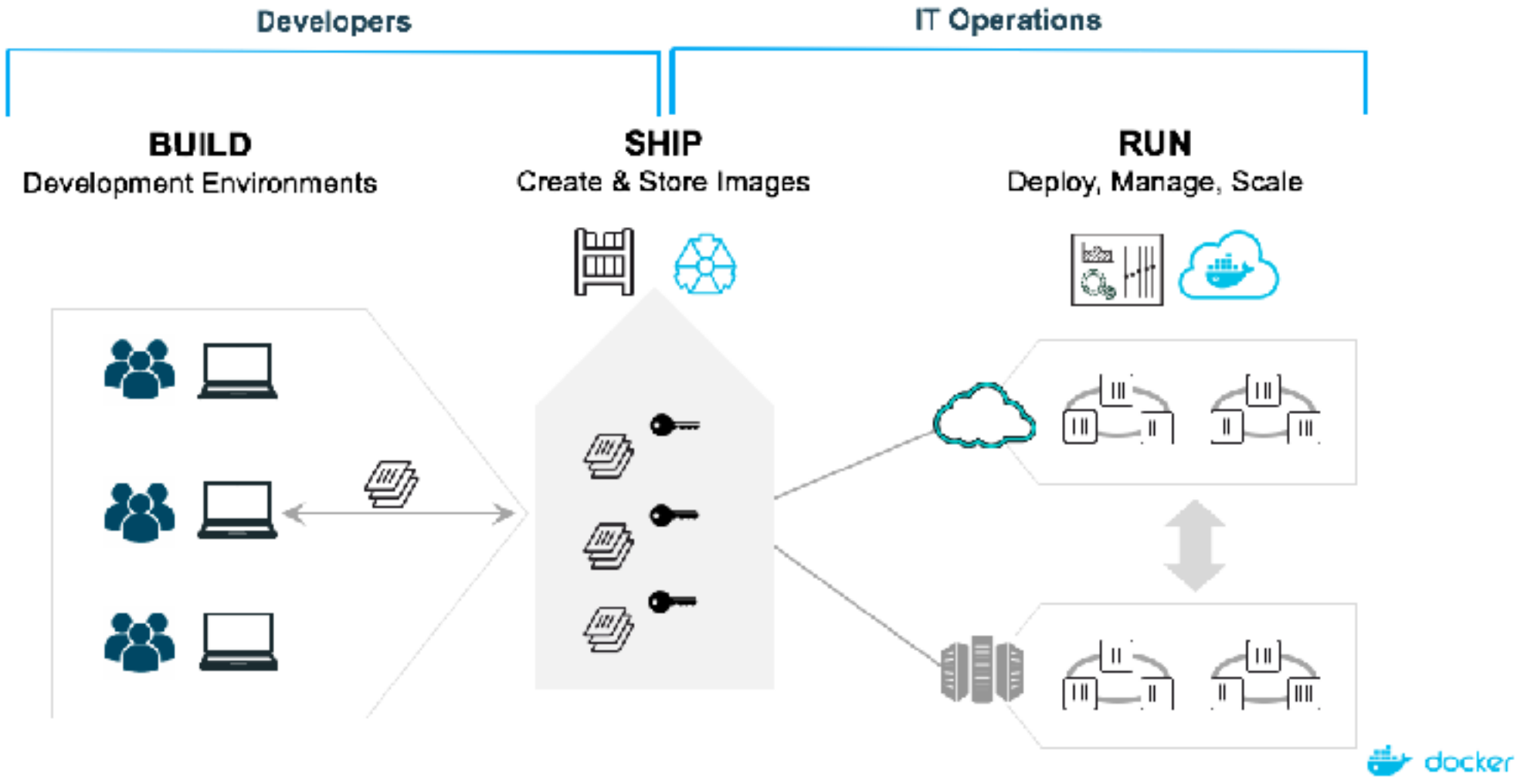
- 压缩包
- WAR、EAR
- 可执行JAR
- 虚拟机镜像
- 容器镜像

Docker基本概念

- 镜像 Image：Docker容器的主要构成成分，包含有运行需要的文件。
- 容器 Container：Docker的运行组件，容器是一个隔离环境，多个容器之间不会相互影响。
- 仓库 Hub/Registry：共享和管理Docker镜像，用户可以上传或者下载上面的镜像，用户可构建私有仓库。


Build, Ship, Run, Any App Anywhere
From Dev
To Ops
Any App
MORE
docker
Any OS
Windows
Linux
Anywhere
Physical
Virtual
Cloud

Put it all together: Build, Ship, Run Workflow



添加Docker支持，Dockerfile:

```
# 基础镜像 jdk8
FROM frolvlad/alpine-oraclejdk8:slim
VOLUME /tmp
# 添加生成的应用可执行jar
ADD target/cart-service-1.0.0.jar  \
                          /emall/cart-service-app.jar
ADD src/main/resources/application-sandboxing.yml \
                          /emall/config/application.yml
ADD data/schema-mysql.sql \
                          /emall/data/schema-mysql.sql
# 声明运行时容器提供服务端口
EXPOSE 9904 9905
RUN sh -c 'touch /emall/cart-service-app.jar'
ENV JAVA_OPTS="-Djava.security.egd=file:/dev/./urandom \
  -Dspring.config.location=file:/emall/config/application.
ENTRYPOINT [ "sh", "-c", \
    "java $JAVA_OPTS -jar /emall/cart-service-app.jar" ]
```

构建镜像并运行

构建应用:

```
$ mvn clean package -Psandboxing
```

构建Docker镜像:

```
$ docker build -t stuq-1160-emall/cart-service .
```

运行容器:

```
$ docker run --name cart-service-1 \
stuq-1160-emall/cart-service -P
```

Demo：打包为Docker镜像

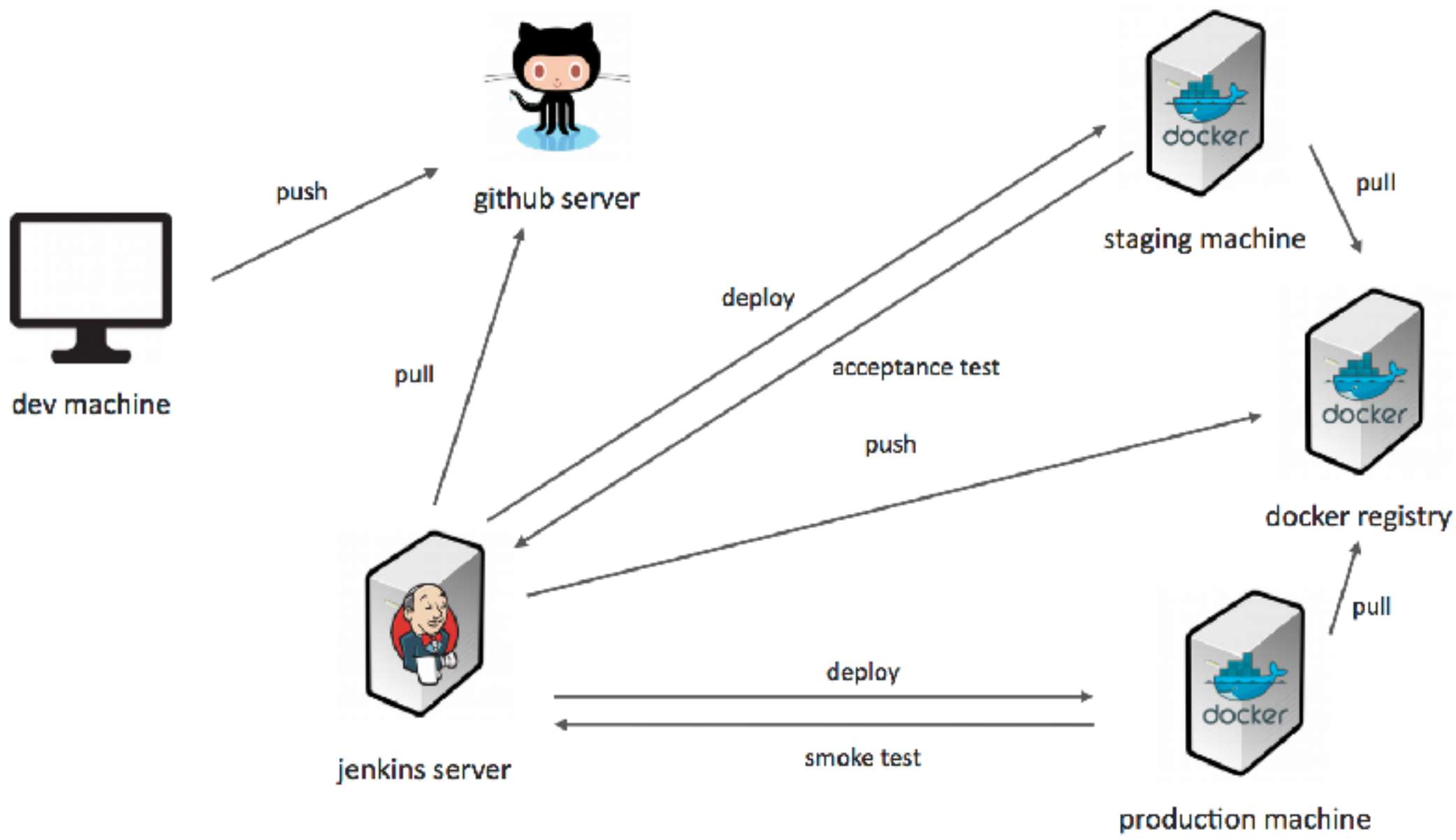
常用持续集成工具

- Jenkins CI
- Bamboo
- Gitlab CI
- Travis CI

常用部署工具

- Chef
- Puppet
- Ansible
- Saltstack

持续部署流程


push
github server
pull
deploy
acceptance test
staging machine
pull
dev machine
push
docker registry
pull
deploy
smoke test
jenkins server
production machine

Docker编排/集群管理工具

- Docker Compose
- Docker Swarm
- Kubernetes
- Mesos
- Nomad

Docker Compose：运行

运行（默认）

```
$ docker-compose up
```

运行（指定文件）

```
$ docker-compose -f deploy/docker-compose/docker-compose.y
```

Docker Compose：停止

停止（默认）

```
$ docker-compose down
```

停止（指定文件）

```
$ docker-compose -f deploy/docker-compose/docker-compose.y
```

Demo： Docker Compose

StuQ 斯达克学院

THANKS!

扫码了解更多

StuQ 斯达克学院