Cloud Native Applications

Microservice Architectures in the Enterprise

Christian Lewis
Platform Architect – Pivotal
clewis@pivotal.io
@acrlewis
Linkedin.com/in/christianlewis
Monolithic Applications Drive Complex, Manual Deploys & Waterfall Release Cycles

Can you deliver full CI/CD for every major app in your portfolio today, or are you doing 75+ step manual deployments?
Productivity: Software developers spend too much time on administration

Task Breakdown

- Design and Coding
- Brainstorming
- Administrative Tasks
- Managing Environments
- Waiting for Test/Build

Average Hours / Week

Sources: “Electric Cloud LinkedIn Survey to software developers”
Trend towards new lightweight architectures

Microservices addressing speed to market and cloud scale
What are Microservices?

Loosely coupled service oriented architecture with bounded contexts

If every service has to be updated in concert, it's not loosely coupled!

If you have to know about surrounding services you don't have a bounded context.
Both speed and safety are required

- Addressing customer needs faster, with less downtime retains your customers
- Being able to make quick experiments (that aren’t destabilizing) gives you more chances at success
- Protecting market share from disruptors or something unforeseen
- The first to successfully rollout the innovation tends to win, defines the category
Continuous Delivery

Development

Day 0
Specify
Automated

Code

Build

Test / Verify

Package Repository

Day 1
Deploy
Automated

Day 2
Production Concerns
Manual

CI/CD

Operations
Ops Loop

Global Product Data Interoperability Summit | 2016

Dev Loop

1. Design
2. Build
3. Integrate
4. Monitor
5. Deploy
6. Run
7. Upgrade

Biz

1. Provision
2. Authorize
3. Integrate
4. Audit
5. Scale
6. Recover
7. Upgrade
Cloud adoption with non-cloud native applications, unopinionated tools and adhoc automation

- Spring Boot
- Mesos
- Dropwizard
- Kubernetes
- Puppet
- SALT
- Chef
- Ansible
- Docker
- Seneca
- VMware vSphere
- Chronos
- Docker
- SALT
- Amazon Web Services
- Ruby on Rails
- Azure
- OpenStack
- Cloud Infrastructure
- Application Frameworks
- Container Schedulers
- Automation Tools
Cloud Technology Stack

Application Framework

Language framework for microservice-based architectures including components for service discovery, metrics and circuit breakers.

Contract: 12 Factor Application

Platform Runtime

Application container runtime with attachable backing services, automated CI/CD, routing, health management and logging.

Contract: BOSH Release

Infrastructure Automation

A single deployment API for provisioning for bit-for-bit, consistent, self-healing deployments across any private or public cloud.
Rich, production ready library based on Netflix OSS for cloud native components, security and management.

- core
- boot
- Cloud Services
- Service Registry
- Config Server
- Circuit Breaker

Dev

Ops + Platform
Cloud Native

- Microservices architecture
- API-first design

Cloud Resilient

- Fault-tolerant and resilient design
- Cloud-agnostic runtime implementation
- Bundled metrics and monitoring
- Proactive failure testing

Cloud Friendly

- 12 Factor App methodology
- Horizontally scalable
- Leverages platform for high availability

Cloud Ready

- No permanent disk access
- Self-contained application
- Platform-managed ports and networking
- Consumes platform-managed backing services
The Cloud Native Journey

Cloud Native
- Believer in opinionated platforms, enforcing cloud-native design
- Shipping microservices with full CI/CD, automated lifecycle management
- Struggling with modernizing applications, reskilling

Containers
- Believer in reproducible builds, operational visibility, open-source
- Shipping a container runtime, microservices, automated provisioning
- Struggling with adhoc operations, manual CI/CD, unintegrated tools, curating OSS

DevOps
- Believer in automation, opinionated software, microservices, 12 Factor
- Shipping API-first applications,
- Struggling with adhoc operations, lack of tooling and monitoring, security

Agile
- Believer in agile, speed to market, software as a differentiator
- Shipping greenfield projects in public cloud because on-premise is too slow
- Struggling with CI/CD, provisioning environments, lack of operational visibility

Traditional
- Believer in IT as a cost center, large projects, customizing off-the-shelf software
- Shipping completed software projects
- Struggling with failed projects, long lead times, business advantages of software
Traditional Model

- **Discovery**: huge issue: mostly custom solutions, JNDI, etc.
- **Configuration**: manual, error-prone, required app restarts
- **Dependencies & Resilience**: co-bundling, EJBs, complex
- **Operators**: no standardized process for deployment and management
- **Developers**: low productivity due to complex environments and expensive context switching
Microservice Model

Global Product Data Interoperability Summit | 2016

- **Discovery**: Service registry for discovery and credentials
- **Configuration**: Config server for secure, versioned application configuration and updates
- **Dependencies & Resilience**: Circuit breakers, platform services marketplace and 4 layers of health monitoring
- **Operators**: Structured deployment model with a single API to manage operations tasks such as deploys and scaling.
- **Developers**: Rich library of distributed system patterns
Rehost, Refactor & Rearchitect Strategies

Platform Runtime
- API Gateway
- Microservice App
- 12 Factor App

Backing Services
- Database
- Dynamic Router
- Service Discovery

BOSH VMs
- App Server

Off Platform
- ERP

Traditional App Environment
- Load Balancer
- Web Server
- App Server
- Database
- ERP

ERP

Rehost, Refactor & Rearchitect Strategies

Platform Runtime
- API Gateway
- Microservice App
- 12 Factor App

Backing Services
- Database
- Dynamic Router
- Service Discovery

BOSH VMs
- App Server

Off Platform
- ERP
Looking to incubate a cloud native application model?

Application Framework
+ Platform Runtime
+ Infrastructure Automation

Complementary O’Reilly Book