Architectures for Successful Continuous Delivery in a Multi-Cloud World

Anders Wallgren | VP Product Strategy, CloudBees
@anders_wallgren
Every business is a software business

“Software is eating the world”

FINSERV
building solutions to deliver better service

RETAIL
building platforms for online sales and support

AUTOMOTIVE
building services for the connected car

FEDERAL
delivering on time and with higher quality

HEALTHCARE
building solutions to improve care

TELECOM
building embedded and online services
Software Is Eating My Bank

Software is eating the cars, too!

New Mercedes-Benz E-Class has more lines of computer code than an Airbus 380
Elite DevOps Performers Do It More Often

- 46x More Software Deployments
- 2555x Less Lead Time Code to Deploy
- 2604x Faster Recovery
- 7x Less Likely for Changes to Fail

Teams that are fully leveraging the cloud are **23x more likely** to be elite DevOps performers.
BUT for most organizations there’s still a gap!

- Only 22% are fully adopting cloud
- 64% take more than two weeks to deploy
- Only 19% of deploy code at least once daily
- Only 21% said they are not employing cloud-native architectures because they are too overwhelmed

Source: https://devops.com/survey-finds-ci-cd-adoption-has-long-way-to-go/
What Does Architecture Have to Do with All This?
Architecture is about the important stuff.

Whatever that is.

Martin Fowler’s definition
It is not the strongest of the species that survives, nor the most intelligent. It is the one that is most adaptable to change.

Charles Darwin

(sort of, see https://quoteinvestigator.com/2014/05/04/adapt/)
Architectural Considerations

• **Application Architecture**
  Monolith → Loosely Coupled (Cloud Native / Microservices)

• **Delivery and Orchestration Architecture**
  Traditional Releases → Continuous Delivery Pipelines
  Manual → Scripted → Automated → Modeled / Declarative (*-as-code)

• **Operational Architecture**
  Bare-metal → VM → Cloud VM → Container → K8S → Serverless
Application Architectures
Monolithic Architectures
The good, the bad, the ugly...

Pros
• Can be easier to develop
• Can be easier to test
• Can be easier to deploy

Cons
• Easier to produce spaghetti code
• Harder to integrate new technologies
• Harder to learn and understand the code
• You have to scale everything to scale anything
• Can’t deploy anything until you deploy everything
Loosely Coupled Allows Teams To...

- Make large scale changes to the system without waiting for other teams
- Reduce the coordination with others required to finish work
- Perform deployments without (or minimal) downtime
- Deploy and Release on-demand, independent of other services
- Do majority of testing on-demand

Microservices Architecture

A suite of services, each focused on doing one thing well

- Independently developed
- Independently deployable
- Exposes an API
- Runs in its own process

Sounds loosely coupled to me!

“Gather together those things that change for the same reason, and separate those things that change for different reasons.”

– Robert Martin
What’s cool about Microservices?

- Divide and conquer complex distributed applications
- Independently developed and deployed, ideal use case for containers
- Freedom to choose the right technology (new or old) for each service
- Smaller more autonomous teams are more productive
Microservices are not a silver bullet!

Monolithic vs Microservices

Monolithic

Microservices

@alvaro_sanchez
Pipeline Architecture
Do You Have That One CI/CD Team?

With one app, releasing into one environment, where anything is possible?
Enter Your (Enterprise) Reality


= “Team N’s DevOps Tooling” (eg, Jira, Jenkins, HPQC, Selenium)
Many architectures, many pipelines, many destinations

“Bi-modal IT is a massive reductionist oversimplification” – Jez Humble
Example

Automated or Manual Approval Gates
Use a Model Based Pipeline Architecture

- Decouple pipeline, applications, environments, cloud infrastructure, and processes and point tools
- Enable system architecture thinking
- Get your databases intimately involved in the pipeline
- Decouple cloud, applications, environments, and configurations from your pipelines.
Architect for Dynamic Environments

- Eliminate Snowflake Environments
- Whenever possible, enable dynamic provisioning of pre-prod environments
- Allow scheduling and reserving of static resources to minimize blocking
- Ensure environment configuration consistency and repeatability to reduce “fear based hording”

Go Dynamic
Maximize utilization of resources
Architect for Self Service

- Give Developers and Testers Keys to the Car
- Invest in creating a reusable wrapper layer
- Consistency > Beauty! Don’t fear promoting existing scripts
- Build For Reuse.
- Control and manage access more effectively to better support governance & auditability

Go Self-Service
On-demand access to environments, resources, and tools
Treat Pipeline as a First Class Feature

- **Pipeline as Code**
  Version control everything, Create templates, Declarative vs Imperative

- **Model everything**
  Abstract and decouple application components from Environment/Config, processes and point tools, Use parameters and shared components

- **Collaborate & Test**
  Share code, Test, refactor, incrementally improve over time

Go Code Native
Architect for Visibility

- Enable visibility from code check-in to production deploy!
- Use cloud APIs to track and optimize pre-production environments across platforms
- Enable more frequent deployments to DEV and QA environments
- Measure all aspects of the pipeline! Identify and eliminate bottlenecks
Build Security and Compliance into Pipeline

- Delivery Team
- Version Control
- Build
- Test
- Release

Pipeline Orchestration

(credit: botchagalupe)
Promote Loosely Coupled Architectures

By creating loosely-coupled architectures and designing team boundaries to enable developer productivity and safety, we can improve deployment outcomes.
Closing Thoughts

• **Cloud is Critical to Success.** It’s not enough to just "move to the cloud"! **Doing Cloud Right** is both an opportunity and a challenge.

• **Start fast and adapt.** Give stakeholders access to resources they need ASAP. Pipeline framework needs to support legacy and cloud-native as you transition.

• **Don’t Forget Control!** Make sure your release platform provides flexible governance so the kids don’t crash the car.

• **Cloud demands visibility** – focus on centralized system visibility to integrating data from all your cloud deployment and monitoring tools.

• **Scale and Speed** – moving to cloud will demand more agility and scale from your pipeline and release process.
Resources

• Migrating to Microservices at Netflix
  https://www.infoq.com/presentations/migration-cloud-native

• How we ended up with microservices
  http://philcalcado.com/2015/09/08/how_we Ended up_with_microservices.html

• Confusion in the Land of the Serverless
  https://www.infoq.com/presentations/serverless-issues

• How to Make the Leap: Building Cloud-Ready Applications into the Architecture
  https://www.infoq.com/articles/cloud-ready-applications

• Backend for Frontend
  http://samnewman.io/patterns/architectural/bff

• WTF IS OPERATIONS? #SERVERLESS
  https://charity.wtf/2016/05/31/wtf-is-operations-serverless
  https://charity.wtf/2016/05/31/operational-best-practices-serverless

• Saga: How to implement complex business transactions without two phase commit
  https://blog.bernd-ruecker.com/saga-how-to-implement-complex-business-transactions-without-two-phase-commit-e00aa41a1b1b

• Developing Transactional Microservices Using Aggregates, Event Sourcing and CQRS
  https://www.infoq.com/articles/microservices-aggregates-events-cqrs-part-1-richardson

Mobile Needs A Four-Tier Engagement Platform
  https://go.forrester.com/blogs/13-11-20-mobile_needs_a_four_tier_engagement_platform

• How to Make the Leap: Building Cloud-Ready Applications into the Architecture
  https://www.infoq.com/articles/cloud-ready-applications

• What Is "Cloud-Native" Data and Why Does It Matter?
  https://www.infoq.com/articles/cloud-native-data

• Low-risk Monolith to Microservice Evolution
  http://blog.christianposta.com/microservices/low-risk-monolith-to-microservice-evolution
Thank You!

Anders Wallgren | VP Technology Strategy, CloudBees
@anders_wallgren
“Just because cloud is a hammer, doesn’t mean everything is a nail.”

Anders Wallgren
VP Technology Strategy,
CloudBees
APPLICATION ARCHITECTURE

How the Fibonacci paradigm applies.
CONTINUOUS DELIVERY

How are you testing and what are your business goals?
CONTAINERS & SERVERLESS

Are your apps set up for one or the other?
The curious case of Knative, serverless and server-full.
DEPLOYMENT MESH

What to think about before implementation.
COMMON STRATEGIES

The six “R’s” of cloud migration.
Digital Innovation Drives Market Capitalization

Top 5 Publicly Traded Companies (by Market Cap)

<table>
<thead>
<tr>
<th>Year</th>
<th>#1</th>
<th>#2</th>
<th>#3</th>
<th>#4</th>
<th>#5</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001</td>
<td>GE</td>
<td>Microsoft</td>
<td>EXON</td>
<td>Citi</td>
<td>Walmart</td>
</tr>
<tr>
<td></td>
<td>$406B</td>
<td>$365B</td>
<td>$272B</td>
<td>$241B</td>
<td>$260B</td>
</tr>
<tr>
<td>2006</td>
<td>EXON</td>
<td>GE</td>
<td>TOTAL</td>
<td>Microsoft</td>
<td>Citi</td>
</tr>
<tr>
<td></td>
<td>$446B</td>
<td>$383B</td>
<td>$327B</td>
<td>$293B</td>
<td>$273B</td>
</tr>
<tr>
<td>2011</td>
<td>EXON</td>
<td>Apple</td>
<td>PeterChiao</td>
<td>Shell</td>
<td>ICBC</td>
</tr>
<tr>
<td></td>
<td>$406B</td>
<td>$376B</td>
<td>$277B</td>
<td>$237B</td>
<td>$228B</td>
</tr>
<tr>
<td>2016</td>
<td>Apple</td>
<td>Alphabet</td>
<td>Microsoft</td>
<td>Amazon</td>
<td>Facebook</td>
</tr>
<tr>
<td></td>
<td>$582B</td>
<td>$556B</td>
<td>$452B</td>
<td>$364B</td>
<td>$359B</td>
</tr>
</tbody>
</table>
“Release” is the Enterprise Challenge

• “I need to release multiple software products, and a mix of complex, traditional, as well as cloud-native. I even have mainframes!”

• “I have to support traditional release teams side by side with CI/CD teams that have much faster release cycles.”

• “I need to control and track the exact release payload and version, across all environments.”

• “I need top down visibility and alerting so across entire delivery and release process, including manual tasks, to see where and how to optimize.”

• “I have governance, compliance, and IT business process requirements that Continuous Delivery doesn’t address.”
How do I know if Microservices are right... for me??

- If you already have solid CI, automated testing, and automated deployment, and you’re looking to scale, then **maybe**
- Don’t decompose prematurely
- Are you good at provisioning?
- Are you good at deployment?
- Are you good at monitoring?
- Do you have good domain/system comprehension?

http://martinfowler.com/bliki/MicroservicePrerequisites.html
And Enterprise Release is Already Hard!

• “I need to release not just one single application, but multiple applications to multiple platforms or clouds.”

• “I need to control and track the exact release payload and version, deployed to any cloud/any environment.”

• “I have to support traditional releases side by side with DevOps Initiatives that have much faster release cycles.”

• “I need top down visibility and alerting so I don’t have to dig around to find out how my release is doing, where it’s stuck, or who it’s waiting for.”
DO: Promote Loosely Coupled Architectures

By creating loosely-coupled architectures and designing team boundaries to enable developer productivity and safety, we can improve deployment outcomes.
DO: Make Stuff Available through Self-Service

Invest in creating a reusable wrapper layer

Consistency > Beauty! Don’t fear wrapping and promoting legacy scripts.

Support all types of end users. CLI, API, UI

Build For Reuse.
4 Resource Elasticity

• What does Resource Elasticity mean in the context of cloud? How are businesses leveraging?

• Best Practices for Release Pipeline
  ▪ Scaling out build, test, and CI jobs
  ▪ Leverage cloud for dynamic pre-production environments
  ▪ Shifting left - provision production like environments sooner!

Elasticity
Speed and Scale.
Your Pipeline in a Cloud & Cloud-Native Era

**COMMIT PIPELINE [N]**
- DEV
- BUILD
- TEST

**COMMIT PIPELINE [1]**
- DEV
- BUILD
- TEST

**RELEASE PIPELINE [“N”]**

**INTEGRATION**
- **OPTIONAL ACCEPTANCE**
  - Provision
  - Configure
  - Deploy
  - Functional, Other Tests
  - ... 
  - **OPTIONAL APPROVAL**

**PRE-PROD**
- **OPTIONAL ACCEPTANCE**
  - Provision
  - Configure
  - Deploy (1...N)
  - Smoke, Other Tests
  - ... 
  - **OPTIONAL APPROVAL**

**PROD**
- **OPTIONAL ACCEPTANCE**
  - Provision
  - Configure
  - Deploy (1...N)
  - Smoke, Other Tests
  - ... 
  - **OPTIONAL APPROVAL**

---

**Cloud Helps Here!**

---

© CloudBees
DO: Support Cloud Native Alongside Traditional

→ Addressing Technology Impedance Mismatch
→ Addressing Release Cadence Mismatch

**COMMITE PIPELINE [N]**
- **DEV**
- **BUILD**
- **TEST**

**COMMITE PIPELINE [1]**
- **DEV**
- **BUILD**
- **TEST**

**RELEASE PIPELINE [“N”]**
- **INTEGRATION**
- **PRE-PROD**
- **PROD**

- **OPTIONAL ACCEPTANCE**
- **Provision**
- **Configure**
- **Deploy**
- **Deploy (1...N)**
- **Smoke, Other Tests**
- **OPTIONAL APPROVAL**
- **De-provision**

- **OPTIONAL ACCEPTANCE**
- **Provision**
- **Configure**
- **Deploy (1...N)**
- **Smoke, Other Tests**
- **OPTIONAL APPROVAL**
- **De-provision**

- **OPTIONAL ACCEPTANCE**
- **Provision**
- **Configure**
- **Deploy (1...N)**
- **Smoke, Other Tests**
- **OPTIONAL APPROVAL**
- **De-provision**

© CloudBees
Example Pipeline

Automated or Manual Approval Gates
Electric Cloud University (powered by DOI)

- Subscription-Based Training and Certification, in partnership w/DOI
- Practitioner-focused content to help accelerate success
- Mix of DevOps and Electric Cloud Product Topics
- Delivered Online and Onsite, LMS to track progress

Leading the Transformation
An Executive DevOps Workshop with Gary Gruver

You Build It, You Secure It
A DevSecOps Workshop with John Willis

Experiential DevOps
A Simulation-Enhanced Learning Experience

ElectricFlow 101 Training
A Hands-On Intro to ElectricFlow