PATH TO CLOUD-NATIVE APP DEV

8 steps to cloud-native app dev

Thomas Qvarnstrom
Technical Marketing Manager
tqvarnst@redhat.com
@tqvarnst

Cesar Saavedra
Product Marketing Manager
csaavedr@redhat.com
@cesar_saavedr

June 2018
The evolution of cloud apps/svcs/fmwks has progressed significantly over the years. Here are some key milestones:

- **1970s**: VANs (EDI) were popularized.
- **1980s**: Colocation became popularized.
- **1990s**: Salesforce SaaS launched in 1999, RosettaNet was introduced in 1998, and Compaq coined the term “Cloud computing” in 1996.
- **2000s**: Amazon was founded in 1994, Java EE 6 was released in 2009, and Heroku announced the Twelve-factor app manifesto in 2011.
- **2010s**: Netflix coins the term “cloud-native” in 2010, Microservices were defined by Fowler in 2014, and Kubernetes was released in 2014.
- **Recent Years**: Docker was introduced in 2013, and DropWizard announced its v1.0 GA in 2011.

These developments have shaped the landscape of cloud computing and have led to more agile and efficient software development practices.
EXISTING AND NEW APPS MUST CO-EXIST

The Digital Economy has brought new demands

Existing IT processes

New IT agility processes

IT needs to adapt
DIGITAL LEADER PAYOFF
Digital leaders enjoy significant advantage

- 26% More profitable than their average industry competitors
- 12% Higher market valuation
- 9% More revenue with existing capacity and products

Capgemini and MIT Center Research: The Digital Advantage - How Digital Leaders Outperform their Peers in Every Industry
Application services speed up the development of a cloud-native application.

Infrastructure services speed up the delivery/deployment of a cloud-native application.

"Yin and yang can be thought of as complementary (rather than opposing) forces that interact to form a dynamic system in which the whole is greater than the assembled parts"

https://en.wikipedia.org/wiki/Yin_and_yang
WHAT IS CLOUD-NATIVE APP DEVELOPMENT

A modern approach to building and running applications

- Service-based
- API-driven
- Containers
- DevOps

Development concerns

Architecture

Communication

Infrastructure

Process

Delivery/deployment concerns
# TRADITIONAL VS. CLOUD-NATIVE APP DEV

<table>
<thead>
<tr>
<th></th>
<th>TRADITIONAL</th>
<th>CLOUD-NATIVE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FOCUS</strong></td>
<td>Longevity and stability</td>
<td>Speed to market</td>
</tr>
<tr>
<td><strong>DEV METHODOLOGY</strong></td>
<td>Waterfall, semi-agile development</td>
<td>Agile development, DevOps</td>
</tr>
<tr>
<td><strong>TEAMS</strong></td>
<td>Isolated dev, operations, QA, and security teams</td>
<td>Collaborative DevOps teams</td>
</tr>
<tr>
<td><strong>DELIVERY CYCLE</strong></td>
<td>Long</td>
<td>Short and continuous</td>
</tr>
<tr>
<td><strong>APPLICATION ARCHITECTURE</strong></td>
<td>Tightly-coupled Monolithic</td>
<td>Loosely coupled Service-based API-based communication</td>
</tr>
<tr>
<td><strong>INFRASTRUCTURE</strong></td>
<td>Server-centric Designed for on-premise Infrastructure-dependent Scales vertically Pre-provisioned for peak capacity</td>
<td>Container-centric Designed for on-premise &amp; cloud Portable across infrastructure Scales horizontally On-demand capacity</td>
</tr>
</tbody>
</table>
PATH TO CLOUD-NATIVE APP DEV: 8 STEPS
More than half (51%) of large organizations have already adopted DevOps.

However, most currently use DevOps for only 10-40% of apps (20% typical).

To take advantage of new tech, faster approaches and tighter collaboration.
## DEVOPS CULTURE BRINGS ABOUT

The culture of open source software projects - a good guide to building DevOps culture

<table>
<thead>
<tr>
<th>INTEGRATED &amp; COLLABORATIVE APPROACH</th>
<th>ENCOURAGES EXPERIMENTATION</th>
<th>FAST FAILURE</th>
<th>TRANSPARENT DECISION-MAKING</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOST TRUST AND COOPERATION</td>
<td>CATALYZES INNOVATION</td>
<td>TEAMS RAPIDLY BUILD PROTOTYPES</td>
<td>ADOPT AGILE WORKFLOWS</td>
</tr>
</tbody>
</table>
2: SPEED UP EXISTING APPLICATIONS

Fast monoliths - migrate app server to container-based platform
2: SPEED UP EXISTING APPLICATIONS

- Expose old components using API's
- Strangulation pattern: replace an existing app piece by piece instead of rewriting the whole
- Add new services
- Using a router (OpenShift | 3scale | Istio) we can send incoming traffic to new services

Ideally use 3scale and/or Fuse
2: SPEED UP EXISTING APPLICATIONS
Retire your monolith as microservices take over

HTML Javascript

OpenShift | Rhoar
3: USE APPLICATION SERVICES

To speed up development - application services optimized for cloud and containers

```java
package com.example.demo.rest;
import javax.enterprise.context.ApplicationScoped;
import javax.ws.rs.Path;
import javax.ws.rs.core.Response;
import javax.ws.rs.GET;
import javax.ws.rs.Produces;
@ApplicationScoped
@Path("/hello")
public class HelloWorldEndpoint {
    @GET
    @Produces("text/plain")
    public Response doGet() {
        return Response.ok("Hello from MicroProfile!").build();
    }
}
```
4: CHOOSE RIGHT TOOL FOR RIGHT TASK

Multiple runtimes and frameworks

“Most applications today are being constructed by teams of developers employing multiple languages.”

Jakarta EE Developer Survey - April 2018

“87% of respondents are using or considering multiple technologies for microservices.”

DEMO - RIGHT TOOL FOR RIGHT TASK
COOLSTORE SERVICES

Red Fedora

Official Red Hat Fedora

$34.99

Forge Laptop Sticker

JBoss Community Forge Project Sticker

$8.50

Solid Performance Polo

Moisture-wicking, antimicrobial 100% polyester design wicks for life of garment. No-curl, rib-knit collar; special collar band maintains crisp fold; three-button placket with dyed-to-match buttons; hemmed sleeves; even bottom with side vents; Import. Embroidery. Red Pepper.
CHAINING OF SERVICE

- Browser
  - Web (1)
    - Catalog microservice
      - Catalog database (1)
  - gateway (8)
    - Inventory microservice
      - Inventory database
5: SELF-SERVICE ON-DEMAND INFRA
To allow developers to access infra when they need it with IT Ops control & visibility
6: AUTOMATE IT TO ACCELERATE DELIVERY
Integrated full-scale enterprise-wide automation

<table>
<thead>
<tr>
<th>AUTOMATION SANDBOXES</th>
<th>COLLABORATIVE DIALOG</th>
<th>SELF-SERVICE CATALOGS</th>
<th>POLICIES AND PROCESSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>For learning the automation language and process</td>
<td>Across organizations for defining service requirements</td>
<td>That empower users and speed delivery</td>
<td>For metering, monitoring, and chargeback</td>
</tr>
</tbody>
</table>
7: IMPLEMENT CI/CD & ADV DEPLOY TECH

CI/CD deployment pipelines - to provide updates w/o operational capacity & reduced risk

- Dev
- Source Repository
- CI/CD Engine
- Container
- Physical
- Virtual
- Private cloud
- Public cloud
7: IMPLEMENT CI/CD & ADV DEPLOY TECH
Advanced deployment techniques

- Rolling upgrades
- Blue/Green deployment
- Canary releases
- A/B testing
8: EVOLVE A MORE MODULAR ARCH

Microservices Architecture and its alternatives

MICROSERVICES
MONOLITHFIRST
MINISERVICES
CUSTOMER SUCCESSES
ACCELERATE APPLICATION DELIVERY

Containers, container orchestration, DevOps, CI/CD, automation

Reduction in deployment time: 12 to 1 weeks
MODERNIZE EXISTING APPS

Migrate, evolve solutions - app svcs integrated & optimized to underlying container platform

60x faster application restart speeds

DECREASED platform support costs and time, freeing resources for innovative service development
FREE e-BOOK

Building Microservices with Enterprise Java
A Practical Guide to Eclipse MicroProfile

v1.0(DRAFT)
June 2018

bit.ly/MP-ebook
THANK YOU

plus.google.com/+RedHat
linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHat