FROM MONOLITH TO CONTAINERS

Verizon’s Journey To Modernize Legacy Apps on OpenShift

Malik Sayed
Senior Manager, Digital Architecture @Verizon
Summit 2017

Zohaib Khan (@zeebluejay)
App Modernization Practice Lead @RedHat
Summit 2017
A LOOK AT CLOUD ADOPTION

By Simon Wardley http://enterpriseitadoption.com/

#redhat #rhsummit
WHY MODERNIZE?

- Enable experimental approach to product development.
- Optimize for Speed vs Cost
- Catalyst for Organizational transformation
BUT CHANGING SYSTEMS AT SCALE IS HARD!

- Organizational Inertia: Skills, Processes and Mindset
- Sizable Investments in IT existing assets
- Often times it is unchartered territory
- Business Perception: Modernization efforts are IT project with large amounts of risks and little perceived value
IT MUST EVOLVE TO STAY AHEAD OF DEMANDS
IS THERE A STRUCTURED WAY TO MODERNIZE?
PATTERNS FOR WORKLOAD MODERNIZATION

LIFT AND SHIFT
- Containerize existing components
- Deploy them on a PaaS
- Keep external integrations and data on legacy
- Legacy applications have to be well written and suited

AUGMENT / REFACTOR
- Legacy remains intact
- New layer enables new capabilities
- Deploy on PaaS
- New integration points between legacy and new layers

COMPLETE REWRITE
- Legacy is totally replaced
- New interfaces and data
- Use PaaS to run
- Some data and features can be re-wrapped, but mostly are retired.
3 PATTERNs FOR MODERNIZATION

Starting Points

LIFT AND SHIFT
- Have a well Architected set of applications with clean separation of concerns.
- Components can be easily deployed independently.

AUGMENT / REFACTOR
- Legacy system that is hard to change piecemeal.
- Changes are not generally isolated.
- Hard to change parts.
- Composite architecture over time.

COMPLETE REWRITE
- Legacy end-of-life system.
- Built on technology going out of support / skillsets do not exist anymore.
- Benefits of Rewrite outweigh its costs.
LIFT AND SHIFT MODERNIZATION

STARTING POINT
CONFIGS
MESSAGING
DATA STORE

SCALE OUT WITH CONTAINERS
CONFIGS
MESSAGING
DATA STORE

MICROSERVICES & APIs
SINGLE CONCERN API
SINGLE CONCERN API

PAAS
BUSINESS
PERSISTENCE
COMPARING THE APPROACHES

- Lift and Shift
- Augment / Refactor
- Complete ReWrite

- Generally the most expensive and longest
VERIZON’S JOURNEY
VZSEARCH

- Tier-1 Verizon Application
- Built on Monolithic COTS platform
- Challenges: Environments lockdown (Dev, QA), Scaling, Proprietary, Large Releases, Application Quality etc.
- Little automation for development and deployments.
VZSEARCH MODERNIZATION OBJECTIVES

- Can we isolate pieces in separate containers without massive rewrite?
- How fast can we get to:
  - Disposable non-Prod environments
  - Auto scaling
  - Self service to eliminate wait times
  - Integrated CI & CD

- Can we do this with Open Source technology to avoid vendor lock-in?
- Wishlist: Can we make it portable across Public and Private cloud?
ARCHITECTURE - BEFORE
TARGET ARCHITECTURE - OPENSHIFT
#1 - DOCKER CONTAINERS FROM DOCKERFILES
#2 - OPENSHIFT POD TEMPLATES
OUTCOMES
VZSEARCH OPENSSHIFT OUTCOMES

- Can we isolate pieces in separate containers without massive rewrite?
VZSEARCH OPENSHIFT OUTCOMES

- Can we isolate pieces in separate containers without massive rewrite?
- How fast can we get to:
  - Disposable non-Prod environments
  - Auto scaling
  - Self service to eliminate wait times
  - Integrated CI & CD
VZSEARCH OPENSShift OUTCOMES

- Can we isolate pieces in separate containers without massive rewrite?
- How fast can we get to:
  - Disposable non-Prod environments
  - Auto scaling
  - Self service to eliminate wait times
  - Integrated CI & CD
- Can we do this with Open Source technology to avoid vendor lock-in?
Can we isolate pieces in separate containers without massive rewrite?

How fast can we get to:
- Disposable non-Prod environments
- Auto scaling
- Self service to eliminate wait times
- Integrated CI & CD

Can we do this with Open Source technology to avoid vendor lock-in?

Wishlist: Can we make it portable across Public and Private cloud?
THANK YOU

plus.google.com/+RedHat
linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHatNews
DEPLOY PHASE: ITERATIVELY DIVIDE, SCALE & DELIVER

**METHODOLOGY & PROCESS**

Red Hat experts, Customer's SMEs, Project Management, Migration team leads

**CENTER OF EXCELLENCE**

**KNOWLEDGE BASE**

**MIGRATION TEAMS (1...N)** Migration Factory partners and/or customer team

**CHALLENGE BACKLOG**
# Application Modernization Pathways

Enabling Modernization from Legacy Applications

<table>
<thead>
<tr>
<th></th>
<th>Starting Point</th>
<th>Open Source &amp; Enablement</th>
<th>Cloud Enablement</th>
<th>Desired State</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lift and Shift Modernization</strong></td>
<td>Non Open Source Middleware Applications</td>
<td>Migrate to Red Hat JBoss Open Source stack</td>
<td>Enable on Red Hat Container platform</td>
<td>Modernize applications to API and Microservices cloud architecture on containers</td>
</tr>
<tr>
<td><strong>Augment with new functionality</strong></td>
<td>Monolithic / Mainframe applications that are hard to change without breaking functionality</td>
<td>Enable integrations to expose data and functionality using Red Hat Open Source stack</td>
<td>Enable on Red Hat Container platform</td>
<td>Modernize applications to API and Microservices cloud architecture on containers</td>
</tr>
<tr>
<td><strong>Complete Re-write</strong></td>
<td>Monolithic or applications on non Open Source middleware about to be retired</td>
<td>Architecture and Design mapping old system capabilities to new architecture.</td>
<td>Setup, train on and get hands-on experience with modern container platform from Red Hat</td>
<td>Create new set of applications on modern cloud-native architectures</td>
</tr>
</tbody>
</table>