redhat.

Cloudy Middleware

MARK LITTLE <MLITTLE@REDHAT.COM>

TOBIAS KUNZE <TKUNZE@REDHAT.COM>

About

Mark Little

Sr Director of Engineering, Red Hat

Tobias Kunze

PaaS Architect, Red Hat

CTO/Co-founder of Makara

Agenda

1. Why PaaS?
2. The JBoss PaaS
3. OpenShift
4. Demo

WHY PAAS?

Why PaaS?

Development

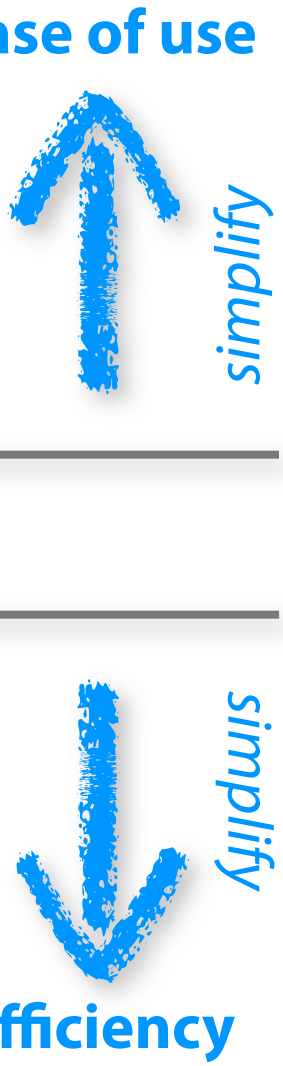
✔ **Focus** on applications

✖ **Stop** dealing with the stack

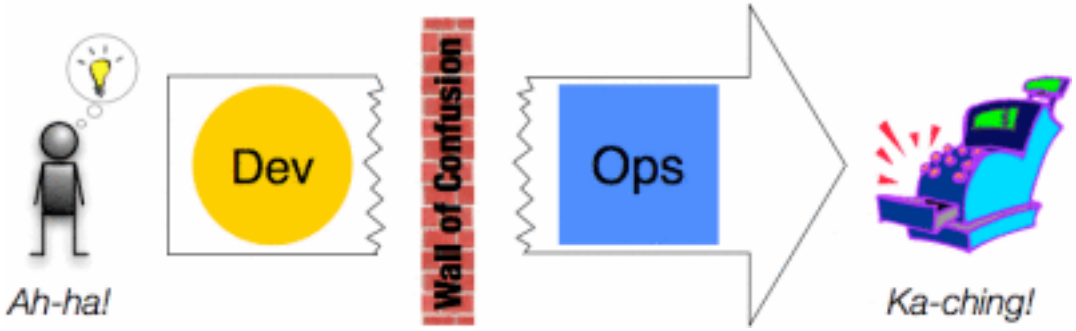
Operations

✖ **Stop** dealing with the application

✔ **Focus** on infrastructure



PaaS: The "Other" DevOps



PaaS is the smart answer to DevOps' problem statement

Source: http://dev2ops.org/blog/2010/11/7/devops-is-not-a-technology-problem-devops-is-a-business-prob.html, accessed 2011-04-29.

Administration is Hard Work

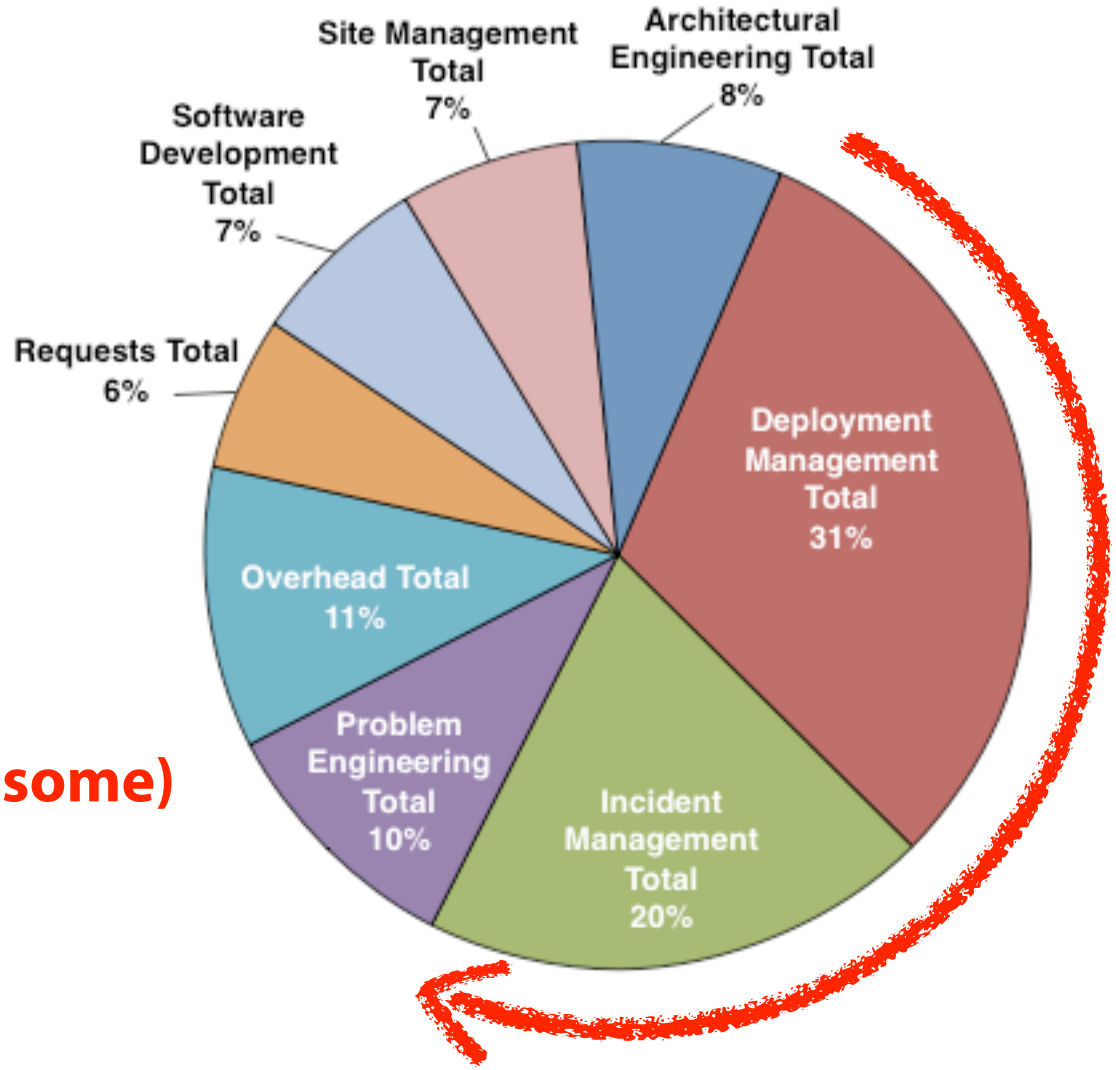
SETUP SERVER	
Setup cloud server	5′
Setup user accounts & keys on cloud server	10′
Install stack	10′
Install stack extensions & libraries	10′
Patch Stack	10′
SCALING	
Setup/configure load balancer	15′
Setup database & replication	30′
Clone application stack to additional server	60′
SECURITY	
Configure security for stack	30′
Configure firewall	10′
Install change audit	30′
MONITORING	
Install system monitoring	30′
Install/configure application monitoring	10′
Install/configure log aggregation	30′
Setup log indexer service for search	30′
Total:	5-8h

SET IT UP ONCE 5-8hrs

AUTOMATE IT 25-40hrs

ADAPT IT TO CHANGE Ugh!
(diff. release/cloud)

Deployment is Hard Work

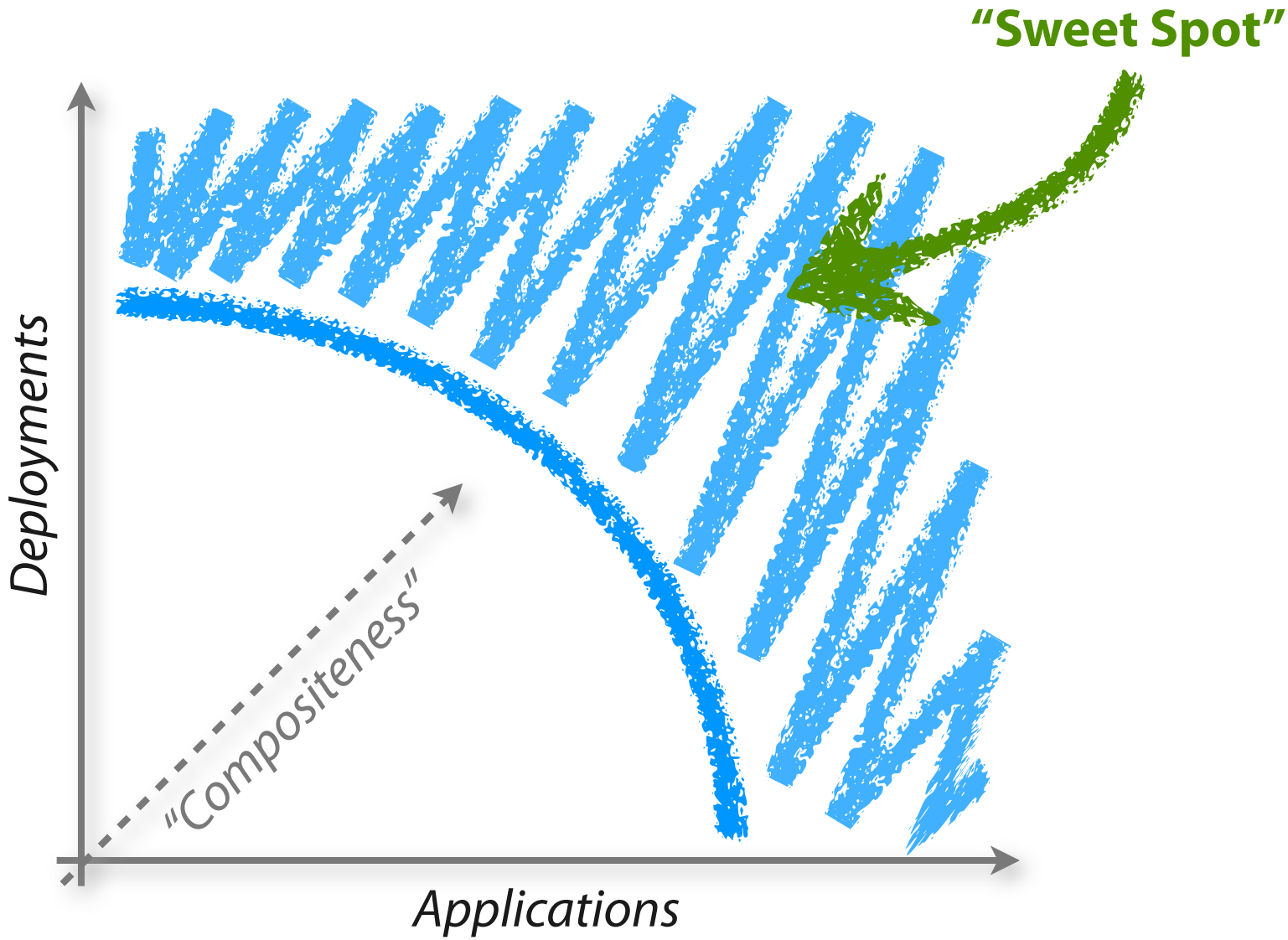


(and then some)

51% deployment-related

Source: Hamilton: “Service Design Best Practices”, AWS, 2009, http://www.mvdirona.com/jrh/TalksAndPapers/JamesHamilton_POA20090226.pdf, accessed 2011-04-01.

PaaS Sweet Spot



Which PaaS?

Development

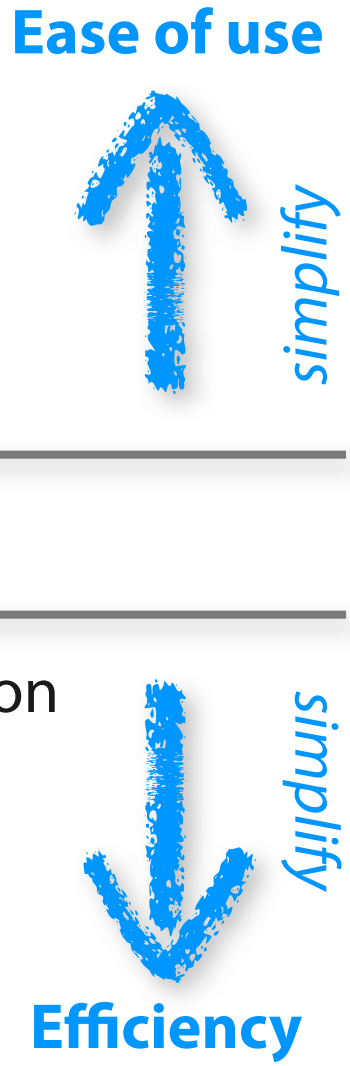
✔ **Focus** on applications

✖ **Stop** dealing with the stack

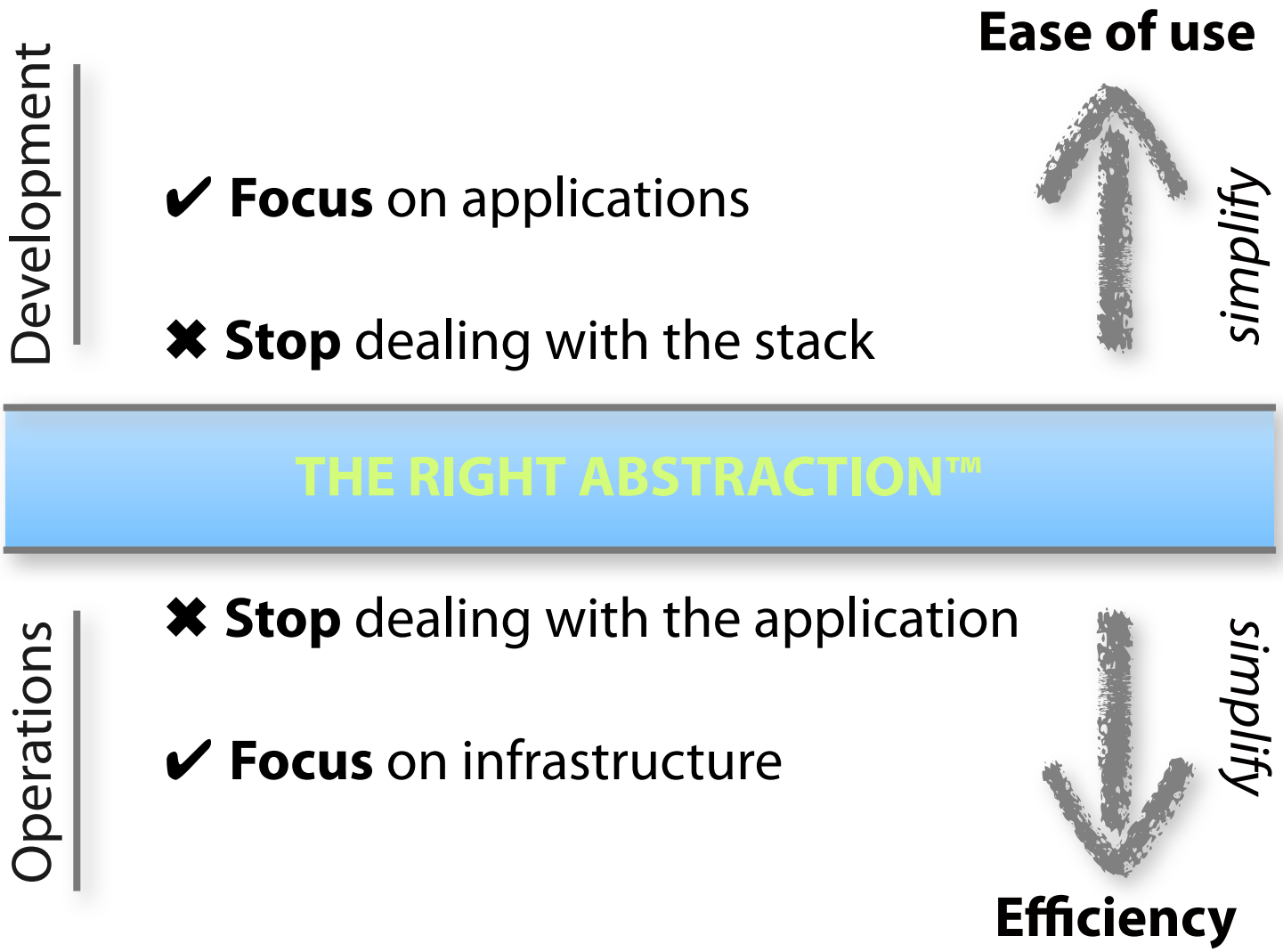
Operations

✖ **Stop** dealing with the application

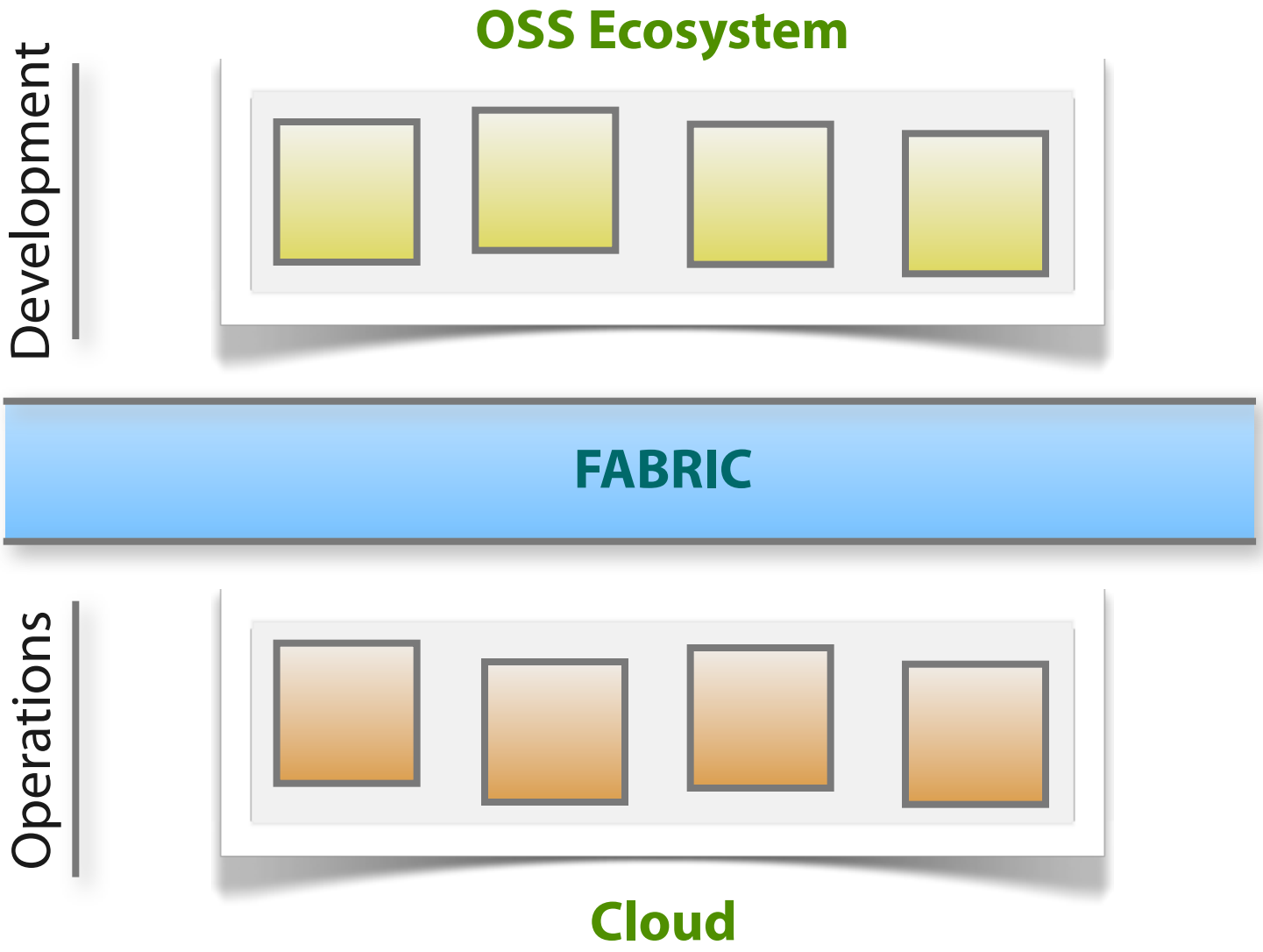
✔ **Focus** on infrastructure



Which PaaS?



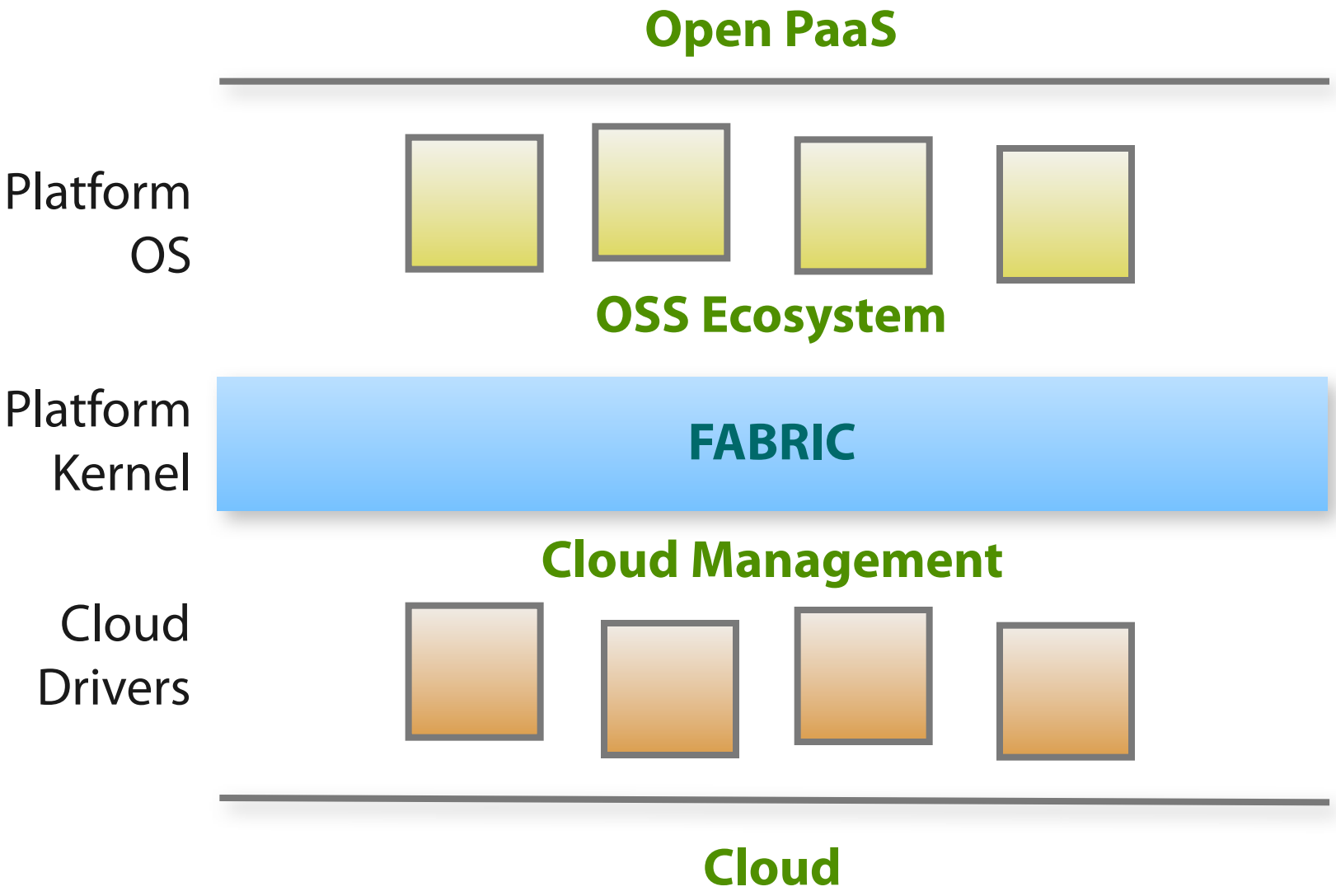
PaaS: The Right Abstraction™



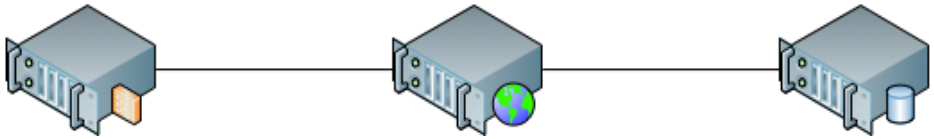
Why Cloud?

1. **Cost**: CAPEX, metered
2. **Convenience**: self-service, on-demand
3. **Allure**: elasticity promises scalability

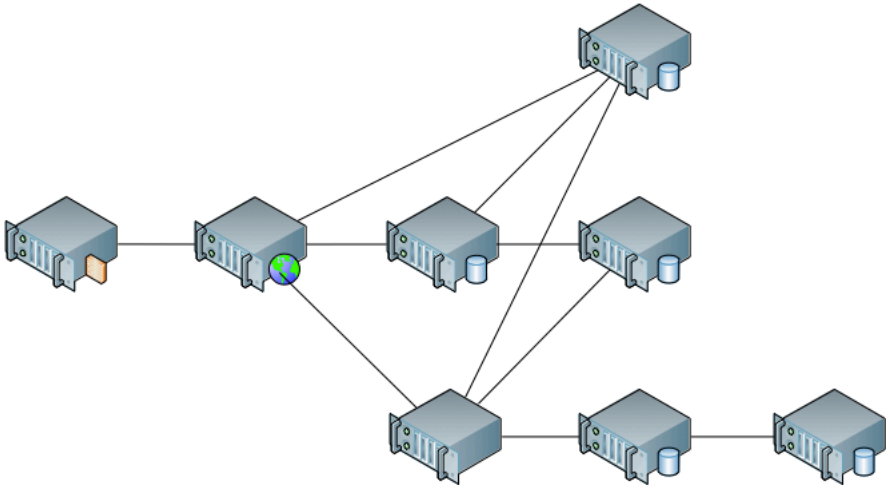
PaaS: The New Linux



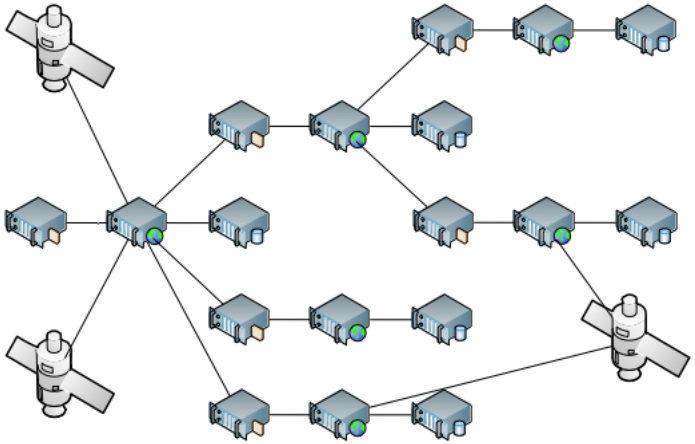
Application Architectures



3 Tiers

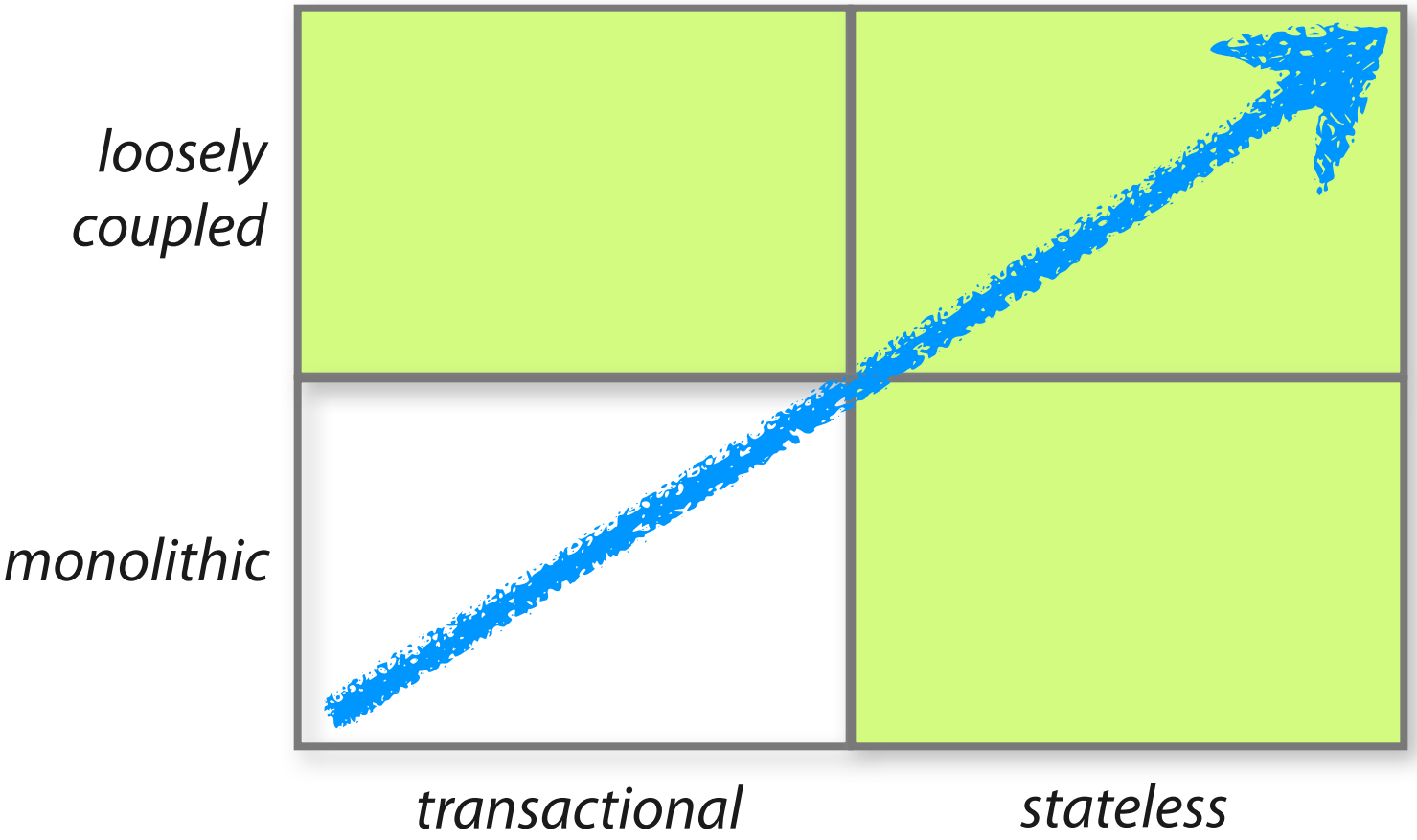


Distributed Apps: NoSQL, KV-Stores



Composite Apps

Architectural Styles



What is JBoss PaaS?

Current PaaS approaches

Far too simplistic

- Not standards based
- Little relevance to existing middleware offerings
- Restricts application capabilities
  - So much for dynamic elasticity
  - Existing s/w investments are irrelevant

Limitations are not based on architectural issues

- Security implications

Vendor lock-in dominates

Present and future directions

Build on our existing implementations

We must provide a natural upgrade path for existing users

- We cannot afford to repeat the DCE/CORBA, DCOM/.NET or CORBA/J(2)EE days

Evolution rather than revolution

If the answer is "Cloud 2010" the question is wrong!

Today "Cloud" means "servers"

- More processors outside of "servers" than inside

JBoss PaaS

Users will want to deploy existing applications more dynamically to cloud environments

- Traditional on-premise, standards-based Java
- Want to be able to use existing programming models to deploy traditionally and within cloud environments
- Interoperability between traditional and cloud

Red Hat will create a PaaS offering that will use standard programming technologies to bridge the worlds of on-premise and cloud deployments

- Support the needs of cloud only PaaS opportunities while protecting customers from programming model and API changes

JBoss PaaS Principles

Address existing JEE/Java deployment plus cloud

Use existing components of the JBoss portfolio – don't reinvent the wheel

Don't invent a closed, proprietary system with new APIs – don't have to change the programming model

Deploy current JEE/Java on-premise plus cloud deployment

- Make them inter-operable (Future proof investment)
- Deployment environment "agnostic" (elastic)

JBoss PaaS Principles

Developer Centric

- PaaS enable from the viewpoint of the developer
  - CDI
- Integrate tooling for cloud D&D with JBDS
- Make easier for developers to use cloud

Standards Based

- Support PaaS and other cloud standards as they exist
- Define or drive standards where they don't exist
  - JSR 347

JBoss PaaS Principles

Optimize for Red Hat IaaS offerings and leverage

Market needs addressed by JBoss PaaS:

- Lower costs – greater utilization of resources on demand
- Make deployment easier and more abstracted
- Elasticity to scale on demand

JBoss PaaS capabilities

Elastic runtime

- Utilize more compute/storage capacity automatically
- JEE services as true services, not co-located within same VM/Container

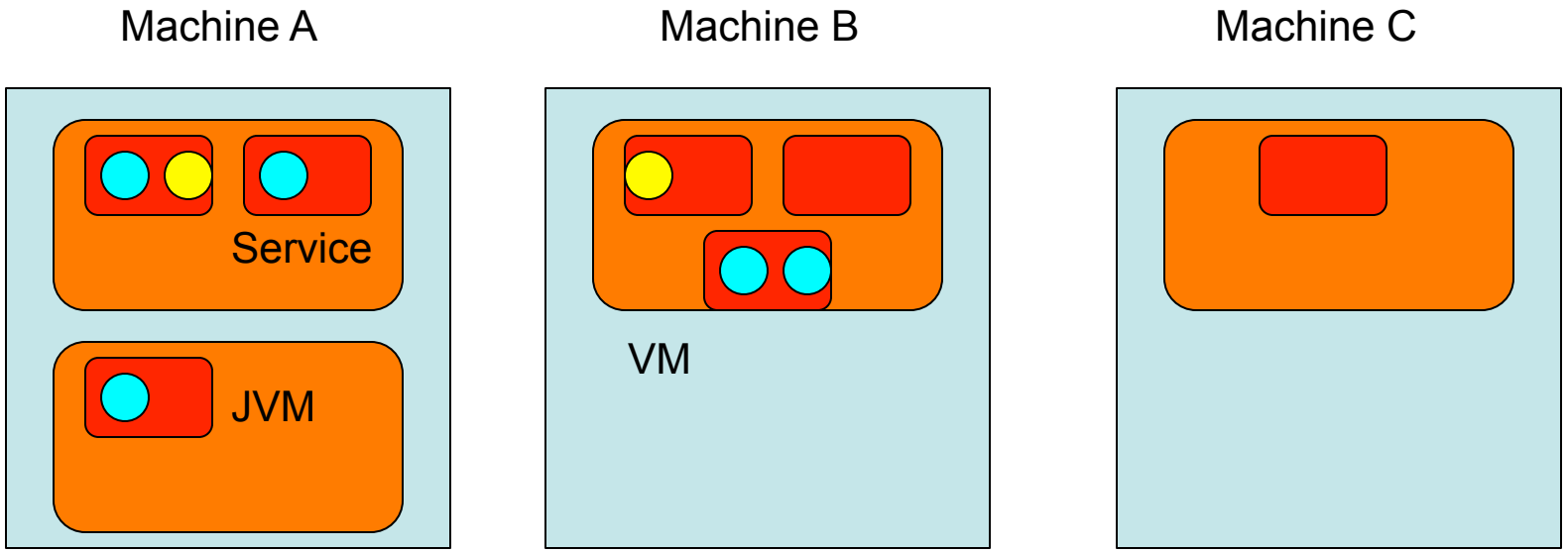
Development

- Runtime hosted as a service in cloud (test deploy)
- Develop (tools) locally – then deploy in Cloud

Abstraction to all configuration considerations

- Data store, clustering, network, security services
- Automatic deployment
- Workflow for deployment

Services, VMs and JVMs



Java EE services (in blue) split across machines, VMs and JVMs

Service instance may abstract more cloud instances

- Hierarchical structure of machines, VMs etc.

Similar deployment structure for applications

JBoss PaaS futures

Lifetime application management (tool chain)

- Works across on-premise & clouds
- Supports heterogeneous deployments
- Build time tools
- Deployment tools
- Ongoing operational & management tools

Developer tools (IDE)

- Expansion of developer tools for easier seamless development

How are we going to do it?

Stage 1

- Begin with JBoss EAP and development tools
- Concentrate on the application platform

Stage 2

- Incorporate other "integration" technology for PaaS enablement – ESB, BPM, presentation technologies
- SOA natural architectural approach for PaaS/SaaS

OPENSHIFT

Makara Vision

PROVISION, DEPLOY, MANAGE, MONITOR & SCALE
EXISTING APPLICATIONS
ON CLOUDS

Makara

Built around monitoring

Control, portability, and *visibility*

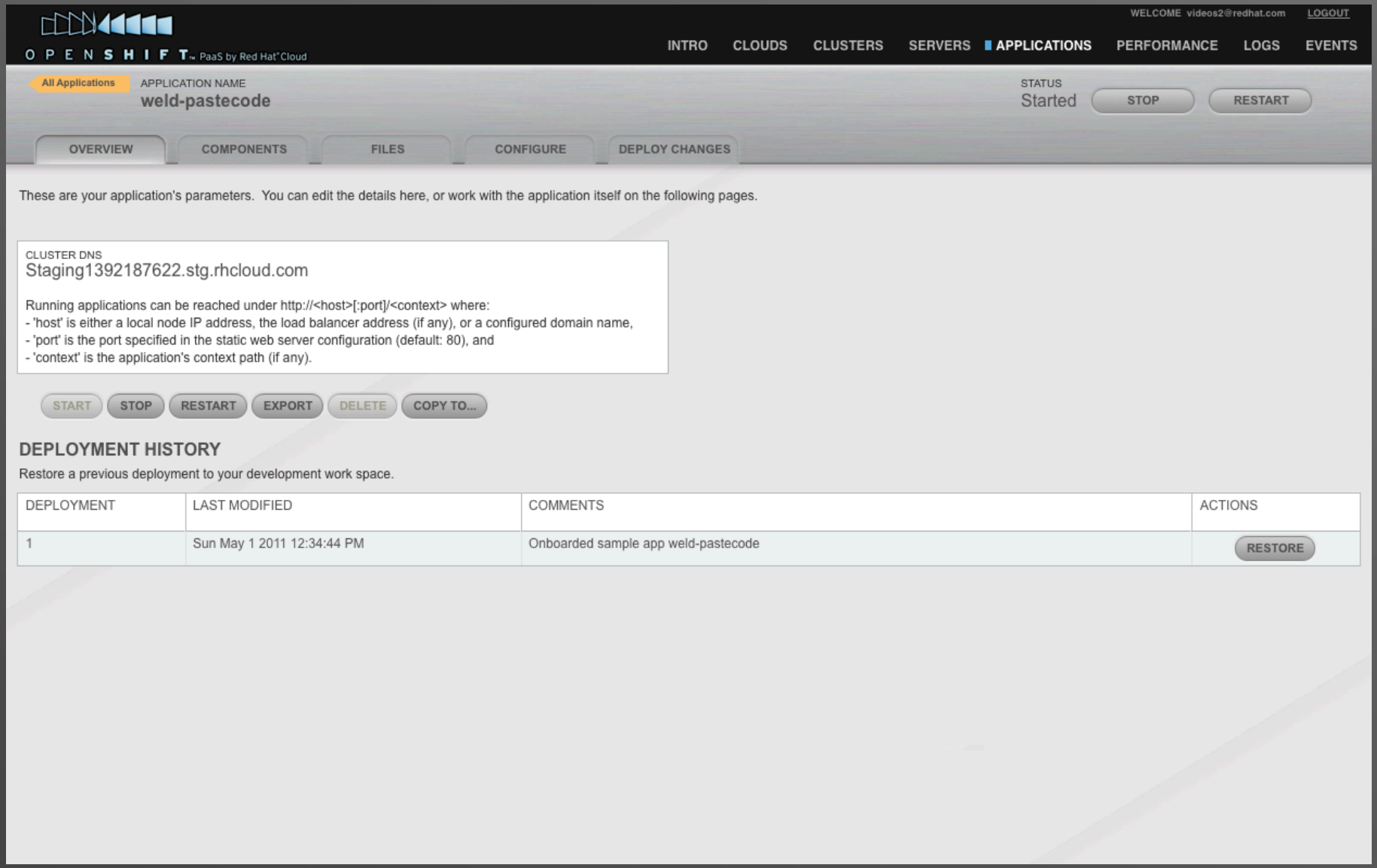
OpenShift

Two interaction models:

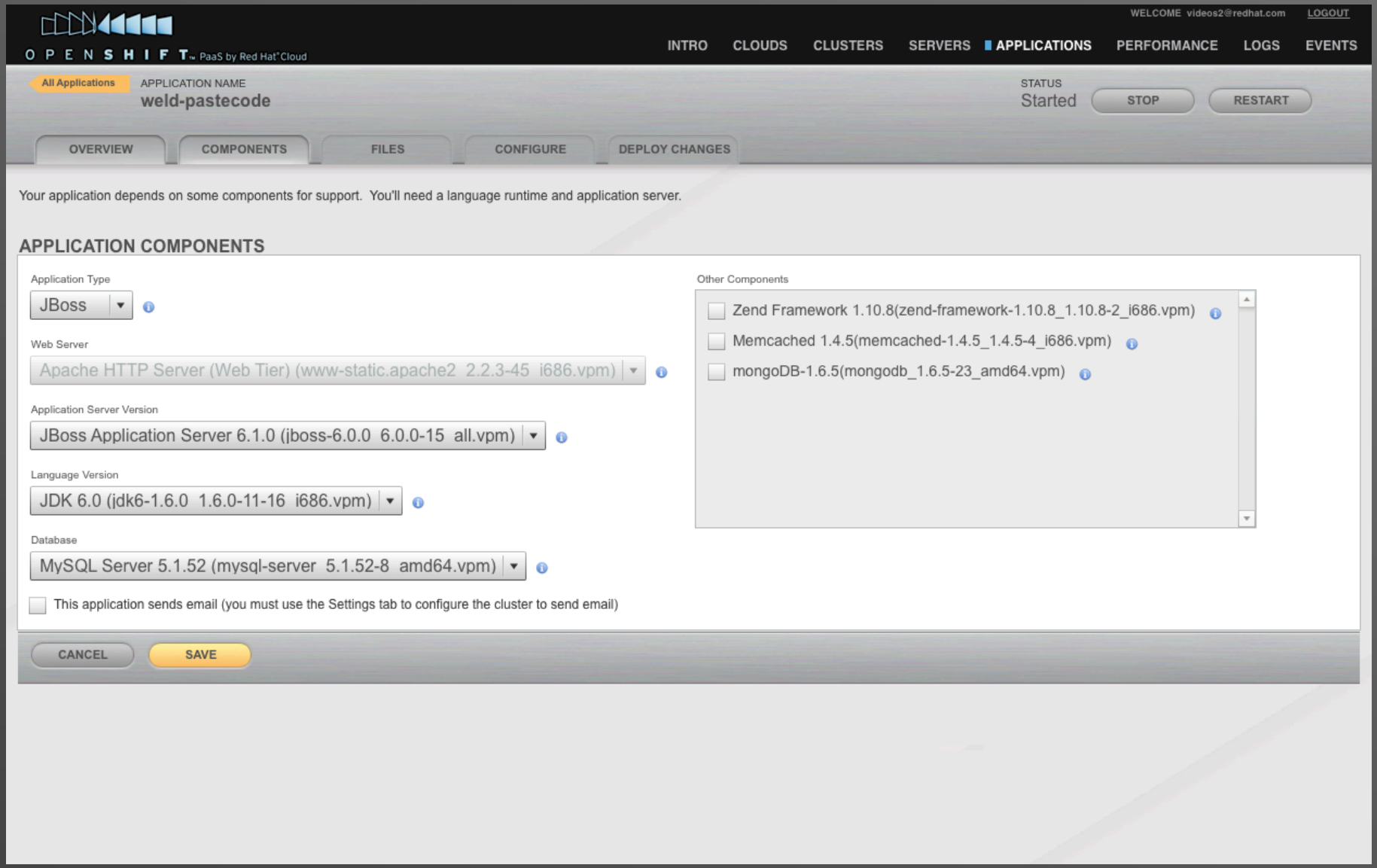
1. OpenShift Express
   - "Runtime-as-a-Service": simple, git-based interaction
2. OpenShift Flex
   - "Middleware-as-a-Service"
   - Optimized for existing models
     - Nodes
     - Middleware, frameworks, services
     - Software ecosystem
   - Operations use cases: self-serve, user management

DEMO

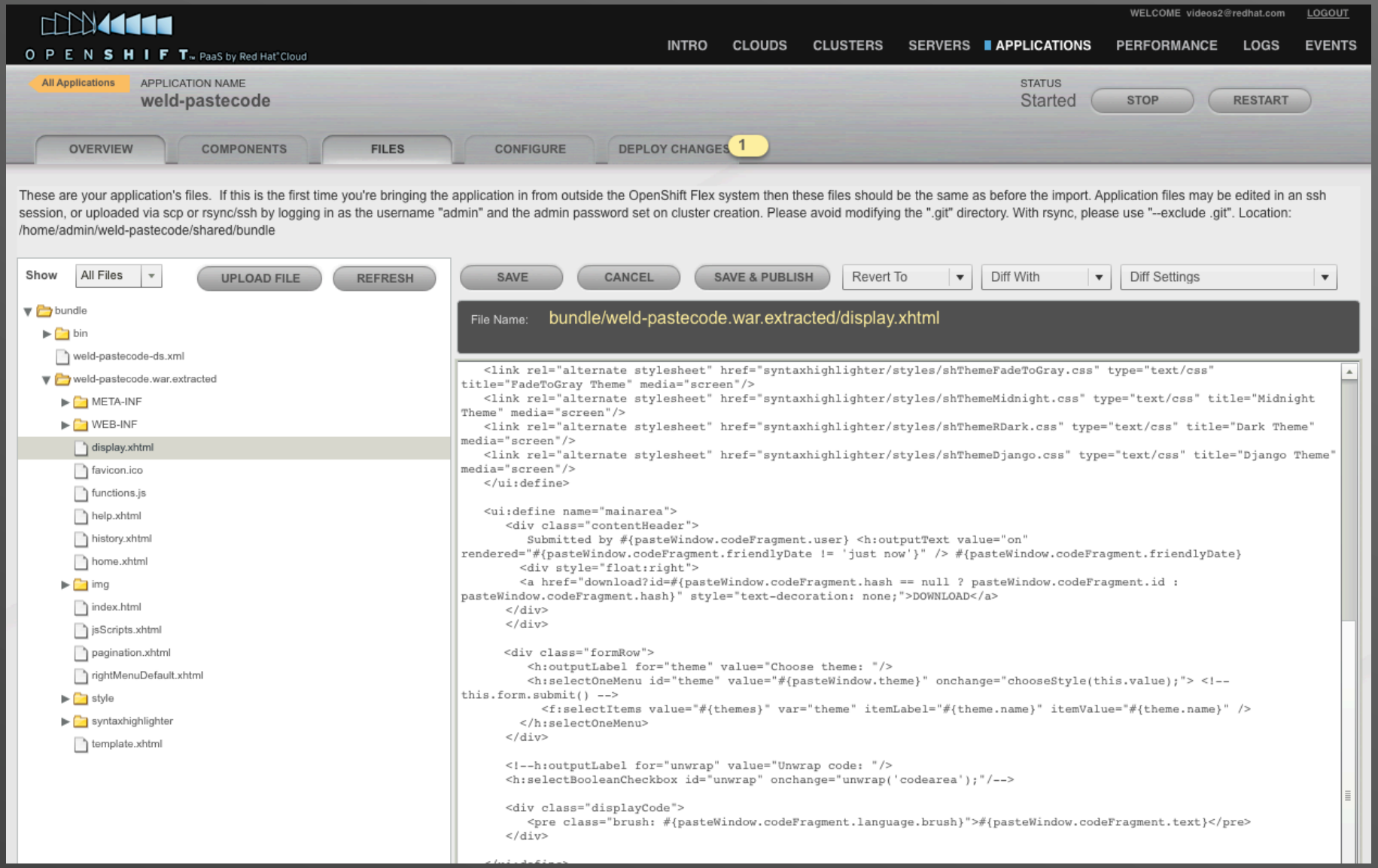
Application Overview



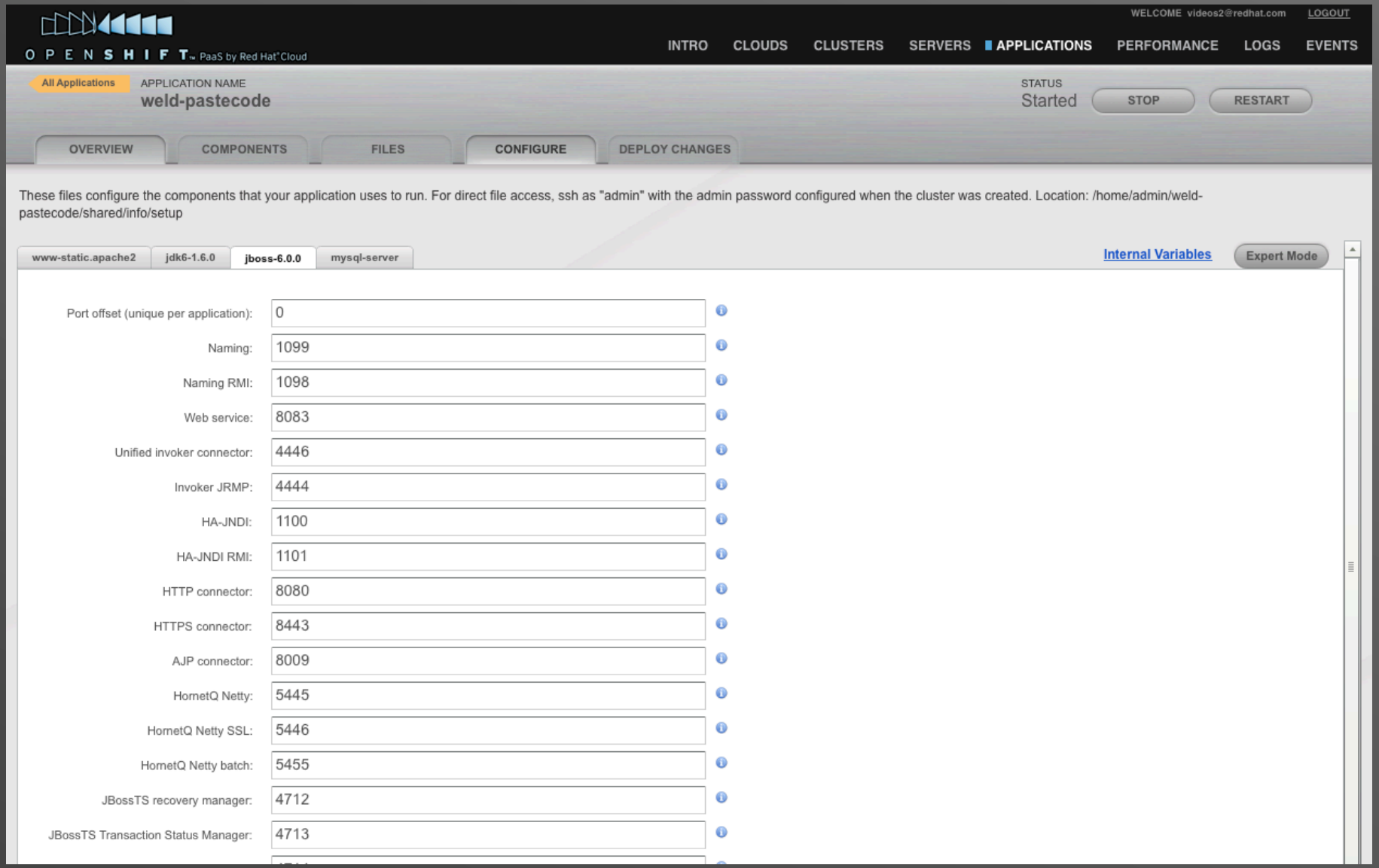
Component Selection



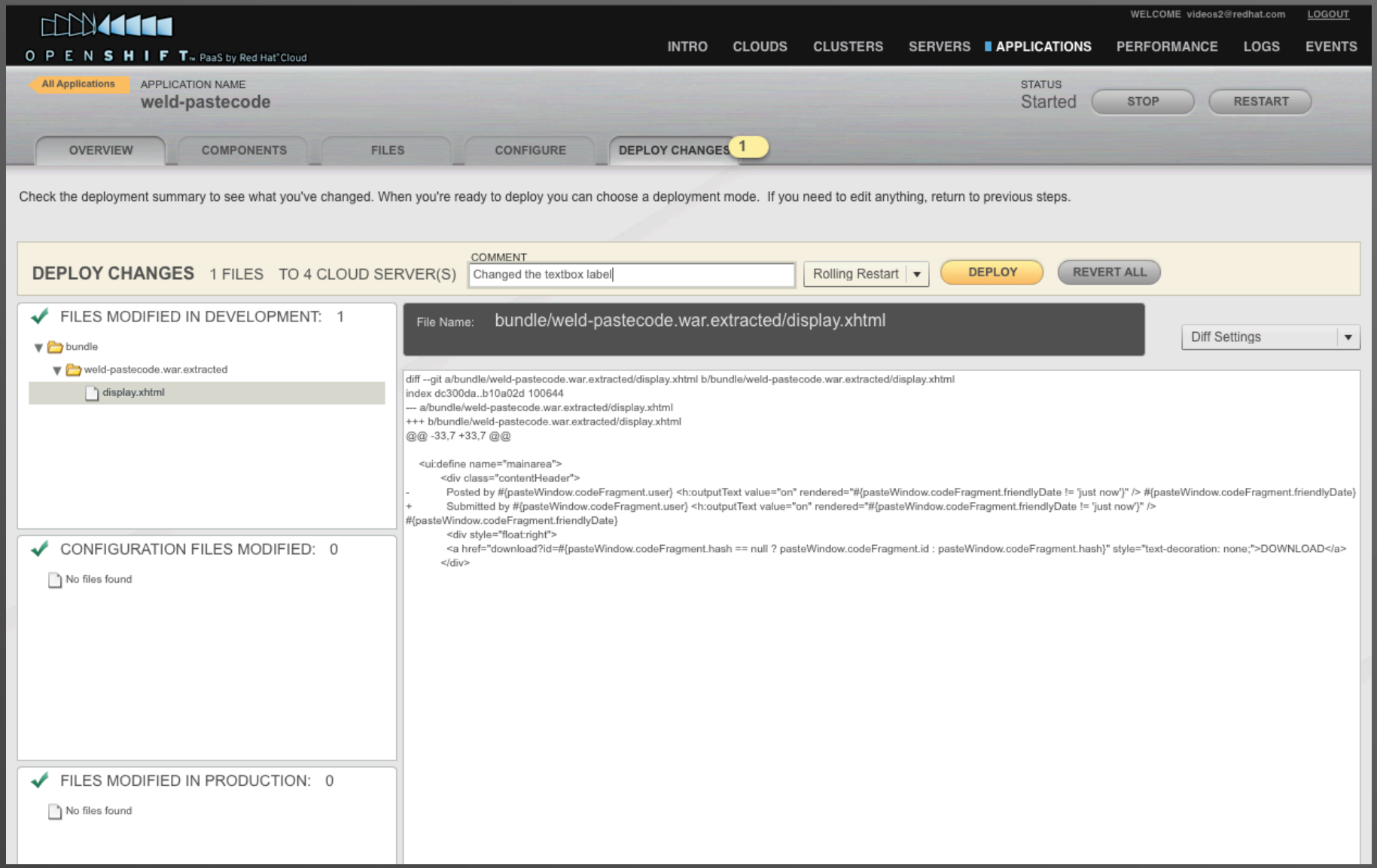
File Management



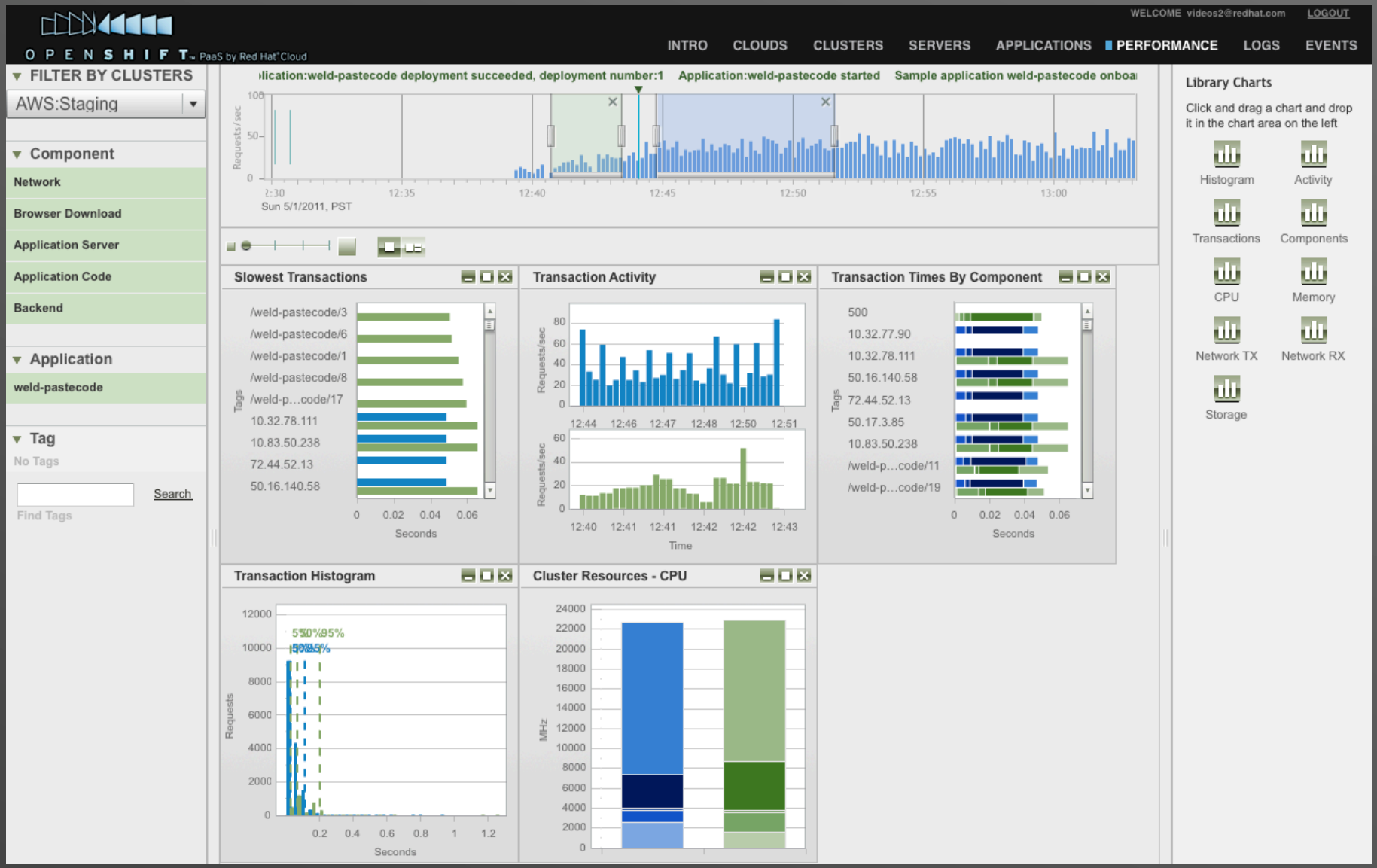
Configuration



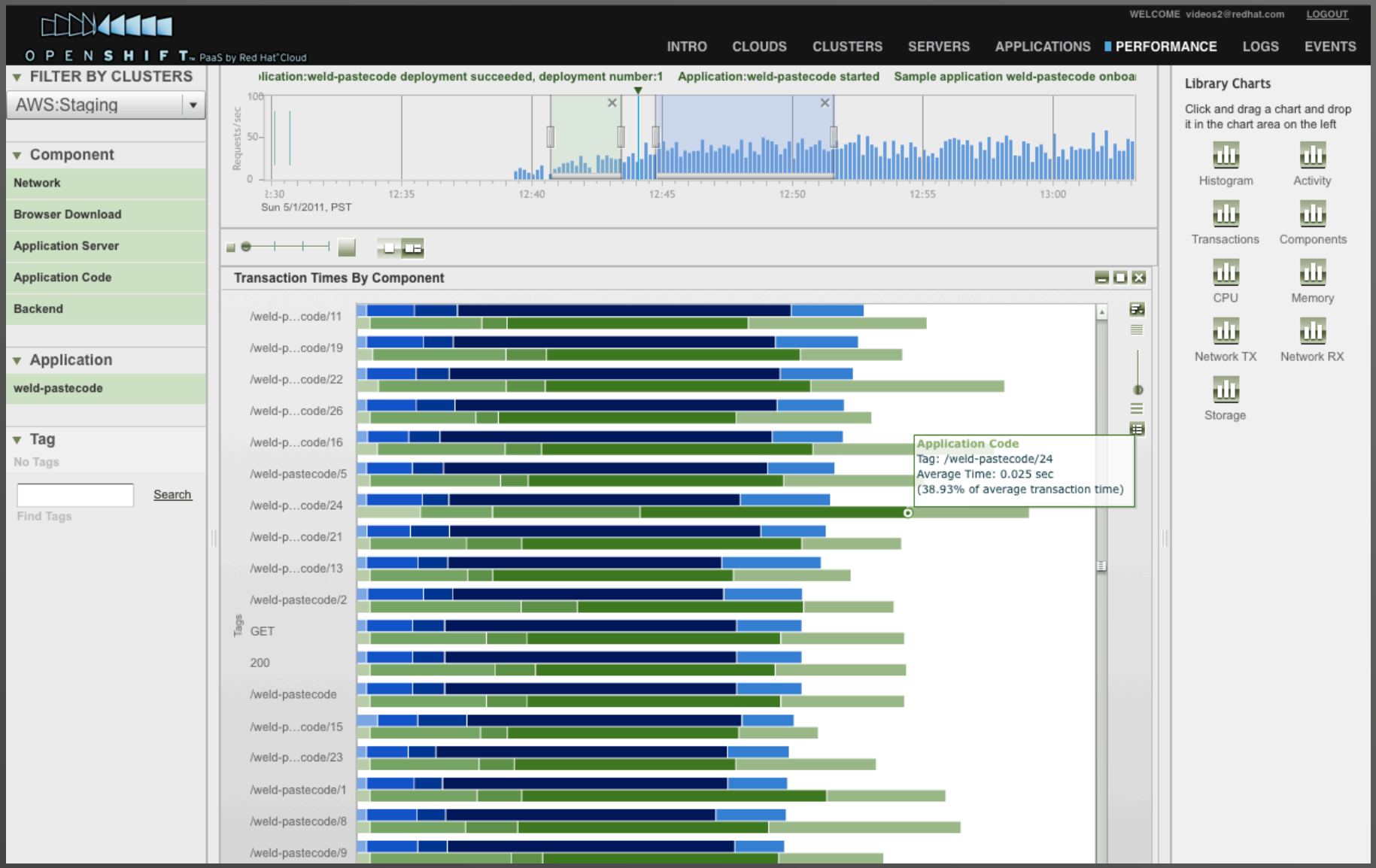
Deployment



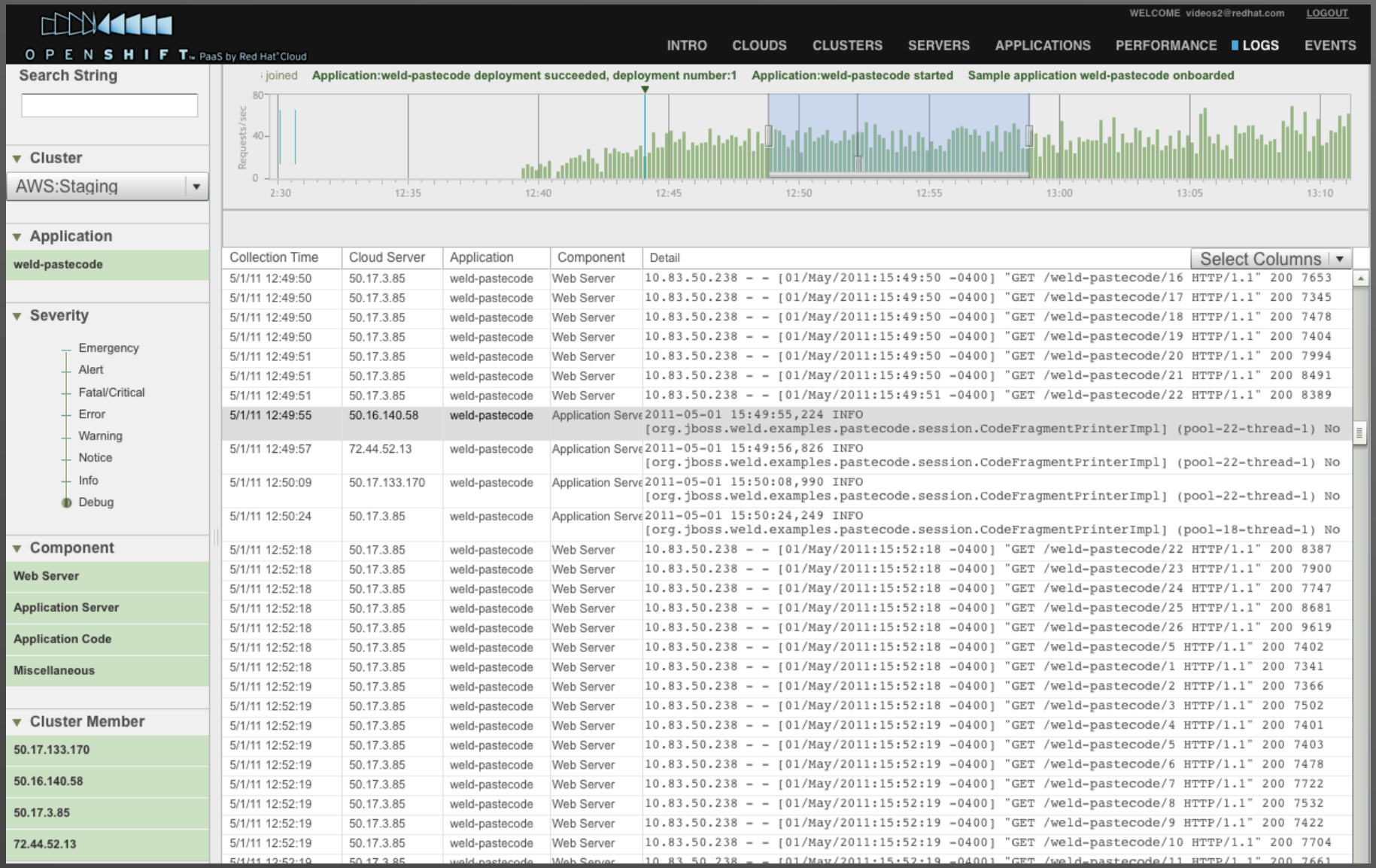
Performance Monitoring



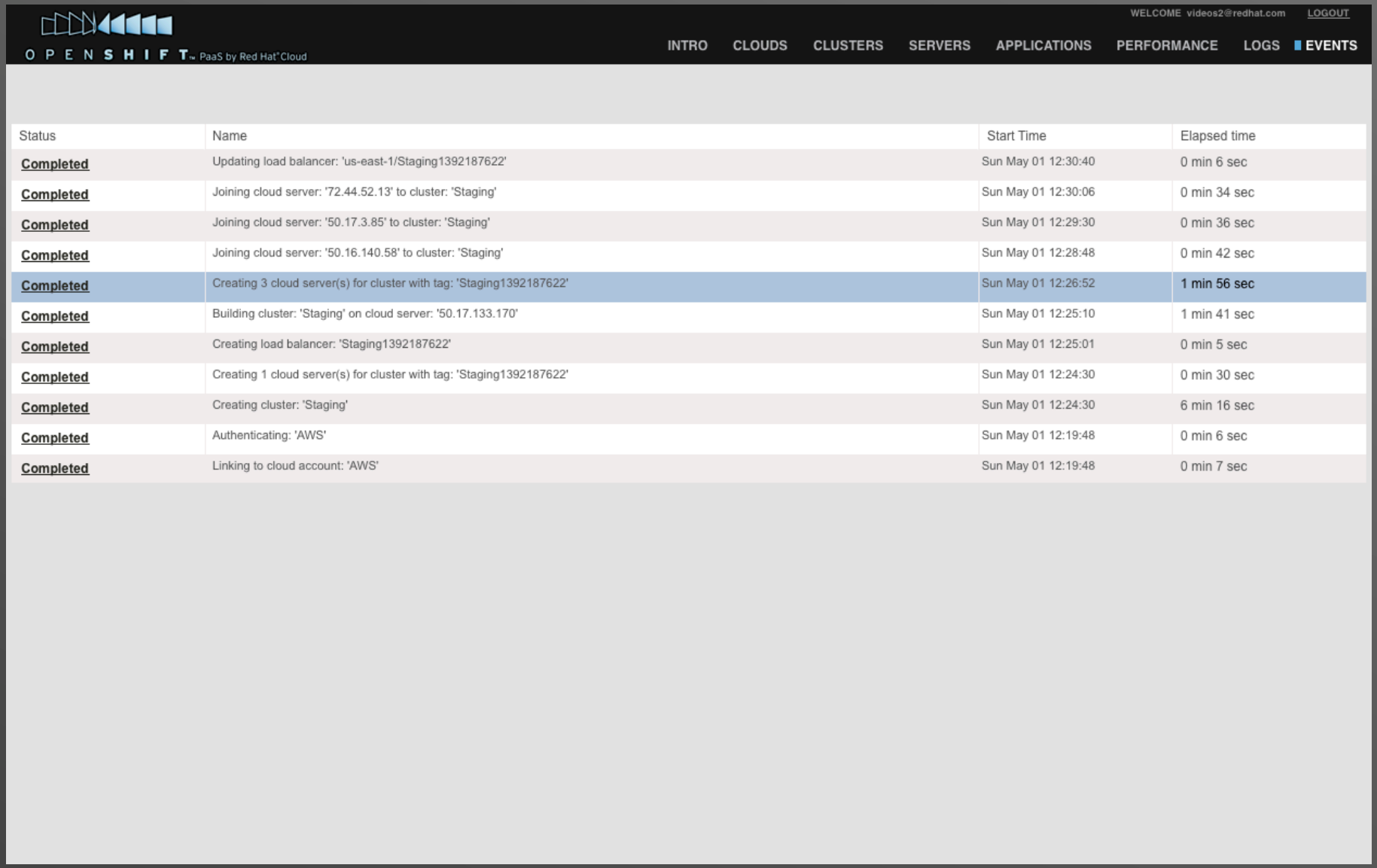
Transaction Monitoring



Log Management

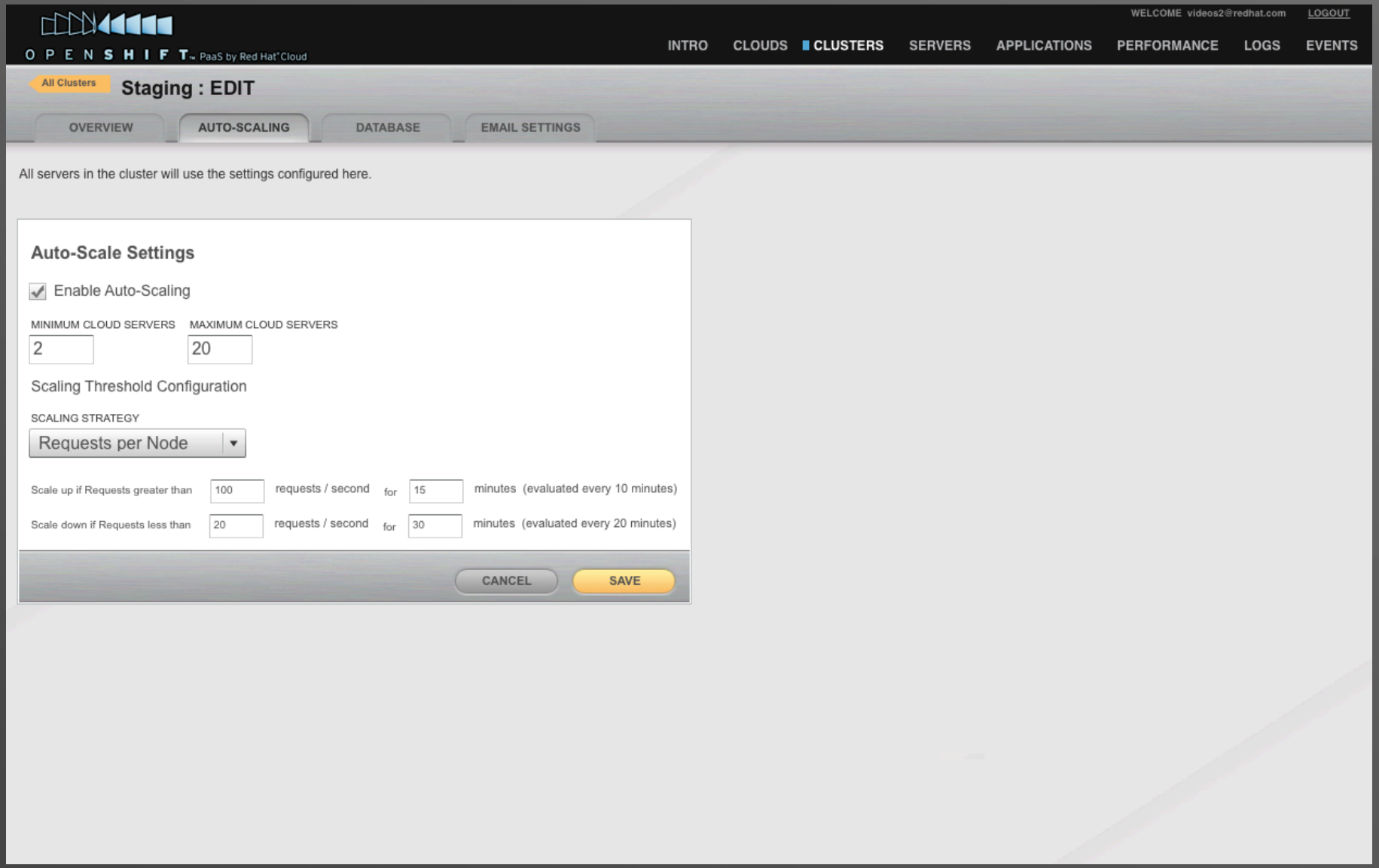


Events

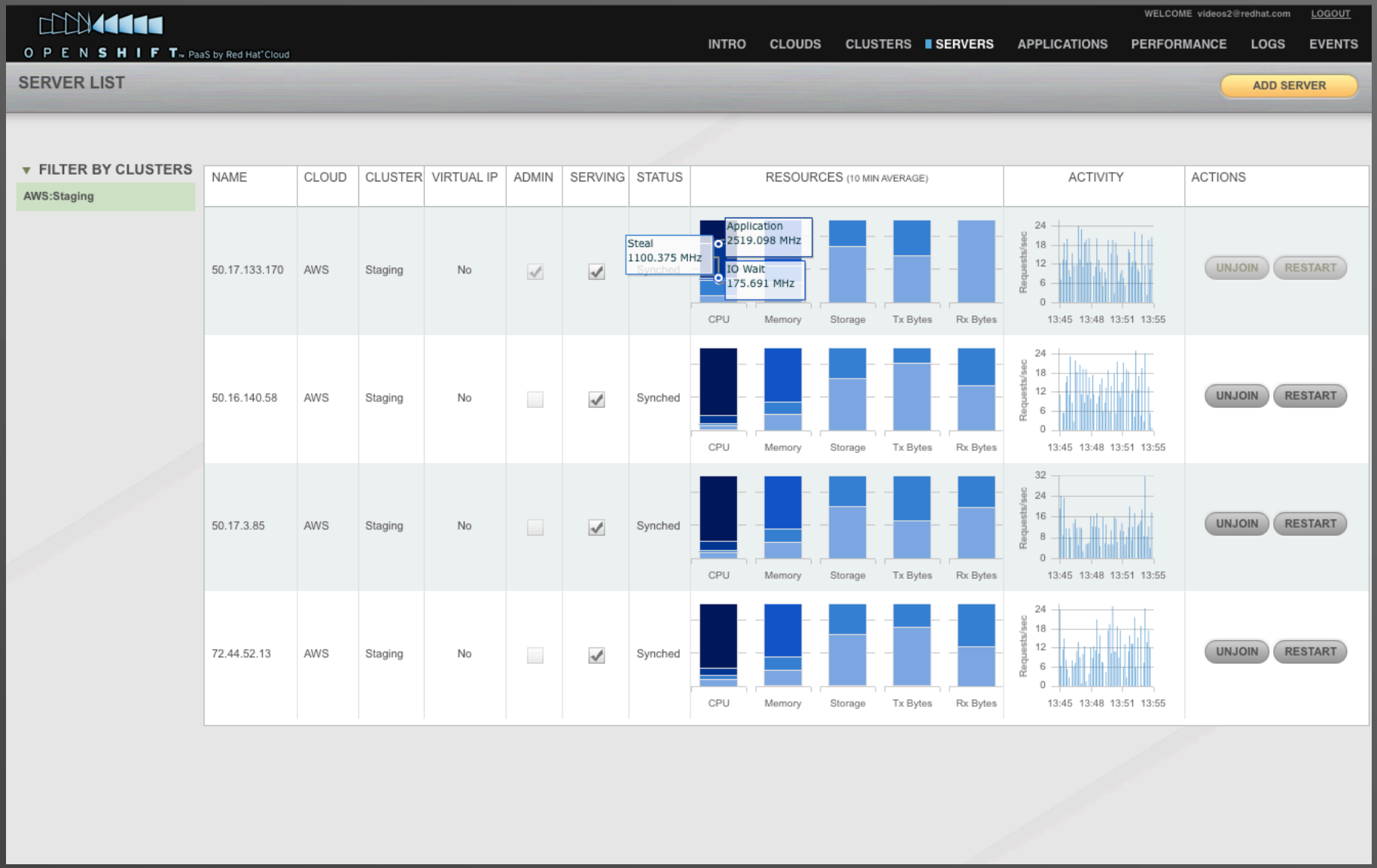


Status	Name	Start Time	Elapsed time
Completed	Updating load balancer: 'us-east-1/Staging1392187622'	Sun May 01 12:30:40	0 min 6 sec
Completed	Joining cloud server: '72.44.52.13' to cluster: 'Staging'	Sun May 01 12:30:06	0 min 34 sec
Completed	Joining cloud server: '50.17.3.85' to cluster: 'Staging'	Sun May 01 12:29:30	0 min 36 sec
Completed	Joining cloud server: '50.16.140.58' to cluster: 'Staging'	Sun May 01 12:28:48	0 min 42 sec
Completed	Creating 3 cloud server(s) for cluster with tag: 'Staging1392187622'	Sun May 01 12:26:52	1 min 56 sec
Completed	Building cluster: 'Staging' on cloud server: '50.17.133.170'	Sun May 01 12:25:10	1 min 41 sec
Completed	Creating load balancer: 'Staging1392187622'	Sun May 01 12:25:01	0 min 5 sec
Completed	Creating 1 cloud server(s) for cluster with tag: 'Staging1392187622'	Sun May 01 12:24:30	0 min 30 sec
Completed	Creating cluster: 'Staging'	Sun May 01 12:24:30	6 min 16 sec
Completed	Authenticating: 'AWS'	Sun May 01 12:19:48	0 min 6 sec
Completed	Linking to cloud account: 'AWS'	Sun May 01 12:19:48	0 min 7 sec

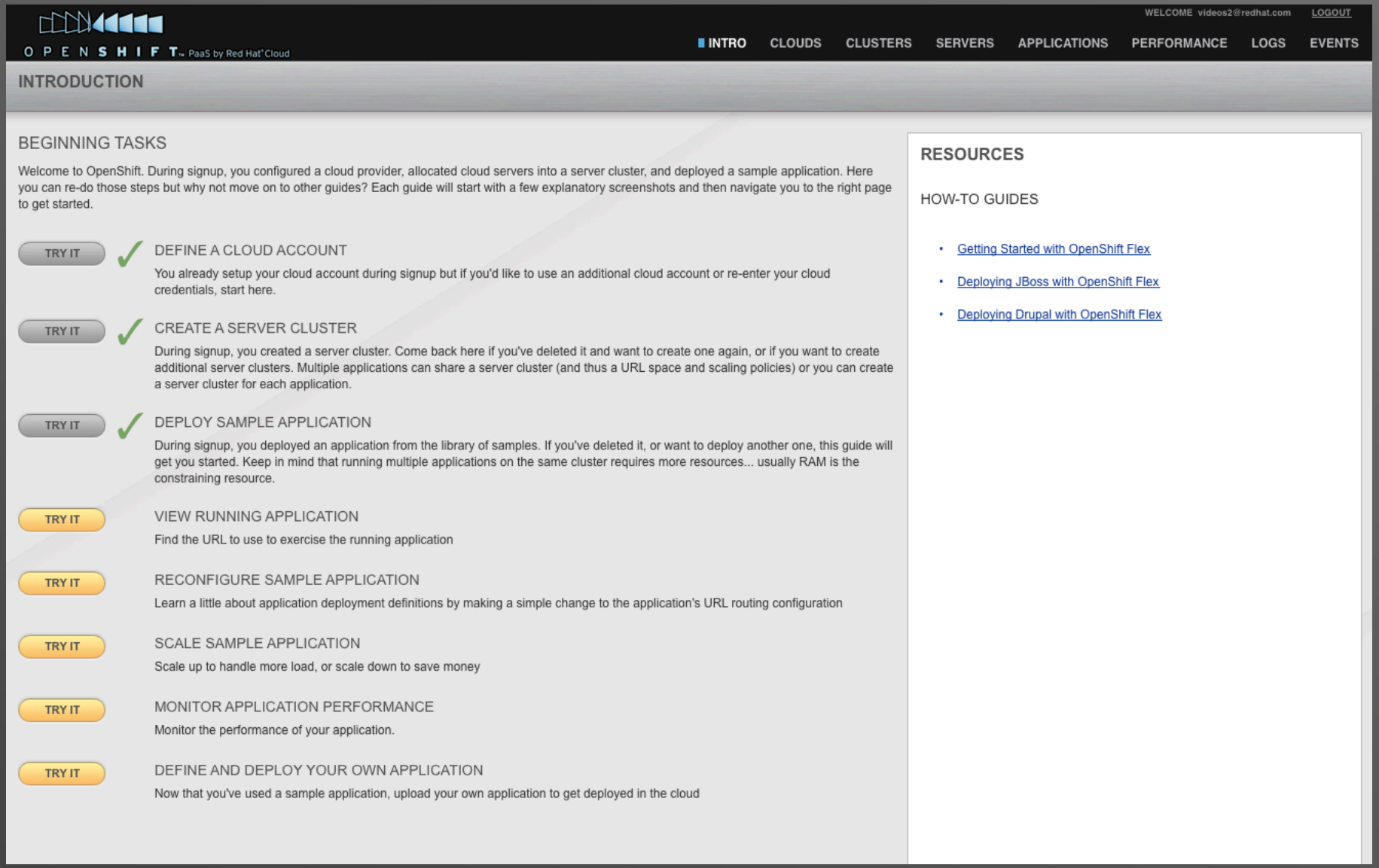
Auto-Scaling



Server Monitoring



Help



Q&A

THANKS!