Java EE, WebLogic, Microservices… and some myths busting

JavaCro'16

David Delabassee
@delabassee
Oracle

Java
ORACLE

About me…

JavaOne
CG
DAVID DELABASSEE
SYSTEM ENGINEER
BELGACOM
DDELABASSEE97
2296362
http://java.sun.com/javaone/
Sun microsystems
David
Delabassee
116115
JavaOne
Moscone Center, San Francisco
March 25-29, 2002
DAVID
DELABASSEE
TECHNOLOGY SE
SUN MICROSYSTEMS
BADGE ID: 02152002128884
ALUMNI
J1
ORACLE OPEN WORLD
ORACLE
2014 SAN FRANCISCO
BACKSTAGE
PASS
JavaOne
June 4
Moscone Center
DAVID
DELABASSEE
PRODUCT MANAGER
BELGACOM
BADGE ID: 02152001815
JAVAPOLIS '06
Delabassee David
JavaPolis Steering
Full Conference Pass
ORACLE
David
Delabassee
NCI
NETWORK COMPUTER, INC.

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Learn French!

It's much easier than understanding French speaking English

Java EE, WebLogic, Microservices... and some myths busting

Java EE, WebLogic, Microservices... and some “clarifications”

Java EE, WebLogic, Microservices... and some myths busting

Agenda

- Architectural style
  - MSA vs. Monolith
- WebLogic

Agenda

- Not a rant on MSA!
- Not a preach on MSA!
- Componentization, Deployment & DevOps
- WebLogic
- IMHO

Architecture styles

- **Monolith**
  - Business functionalities are embedded into a single (large) artefact
  - Single Deployment Unit
- **Microservices**
  - Business functionalities are divided into multiple (smaller) artefacts
  - Multiple Independent Services
  - Goal : Easier to develop and maintain applications

MSA Characteristics

- Componentization via Services
- Organized around Business Capabilities
- Products not Projects
- Smart endpoints and dumb pipes
- Design for failure

- Decentralized Governance
- Decentralized Data Management
- Infrastructure Automation
- Evolutionary Design

http://martinfowler.com/articles/microservices.html

Componentization

Advantages

- Independent services
  - Should be independent
    - Eg. deployment
  - Impose clear boundaries
    - and well defined interfaces
- Should be easier to develop, maintain and update
  - Limited & well-defined business context
- Polyglot friendly

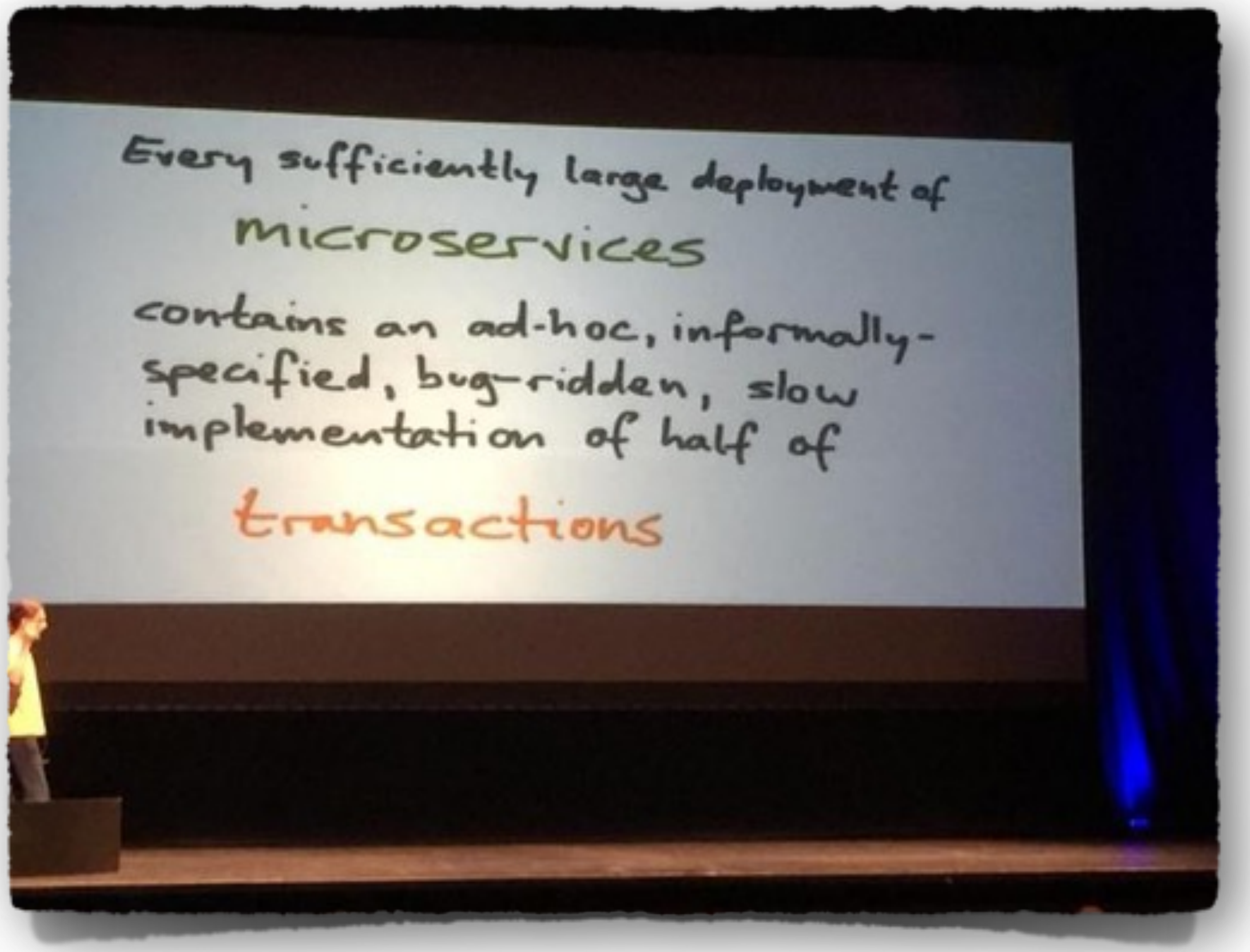
Componentization

Drawbacks

- Out-of-process intra-services exchanges
  - More expensive
  - More brittle
  - More complex
- New exchange patterns
- Data Management
- TX

Componentization

Drawbacks



Every sufficiently large deployment of microservices contains an ad-hoc, informally-specified, bug-ridden, slow implementation of half of transactions

Organized around Business Capabilities

Divide & Conquer

- How to split a complex problem?
  - Skills, Orgs, Geo, …
- Cross-functional teams organised around business capability
  - UX, DB, PM, …

Organized around Business Capabilities

Divide & Conquer

- Nothing prevent a Monlith to be modularized around business capabilities
- Processes & Methodologies
  - Eg. Design Driven Development
- Culture & Organisation

Product not Project

When are we really done?

- A team should own a product over its full lifetime
  - Aka *“You build it, you run it!”*
- An application is not just a set of functionality to release
  - How can the application assist its users to enhance the business capability?
- Culture & Organisation

Smart endpoints and dumb pipes

- Leverage Web principles & protocols
  - Infrastructure friendly, caching, etc.
- New communication patterns
  - Move from fine-grained to coarser-grained exchanges
  - Messaging
- New tools
  - Circuit Breaker, Messaging Solutions…

Design for Failure

Many potential PoF

- Risk of failures are increased
  - Design accordingly and cope with failure!
  - Will induce additional complexity
- Pro-active monitoring
  - Detect failures quickly
  - Automatically restore service

Decentralized Governance

The right tool for the right job

- No platform constraints
- No language constraints

Decentralized Data Management

Decentralized decisions

- Conceptual model focused around the service context
- No Data Management constraints
- Polyglot persistence

Decentralized Data Management

Drawbacks

- Transactions
- Eventual consistency
- Data duplication

Infrastructure Automation

Make deployment boring

- Continuous Delivery & DevOps
- Deployment and Management
- On-premises & Cloud

Evolutionary Design

- Services should be independent
  - Should be easy to replace
  - Should be easy to upgrade

Deployment

Deployment

- *“Java EE Deployments are* [redacted] *!”*
  - No granularity
  - Slow
  - …
  - *[insert your preferred drawback]*

Deployment

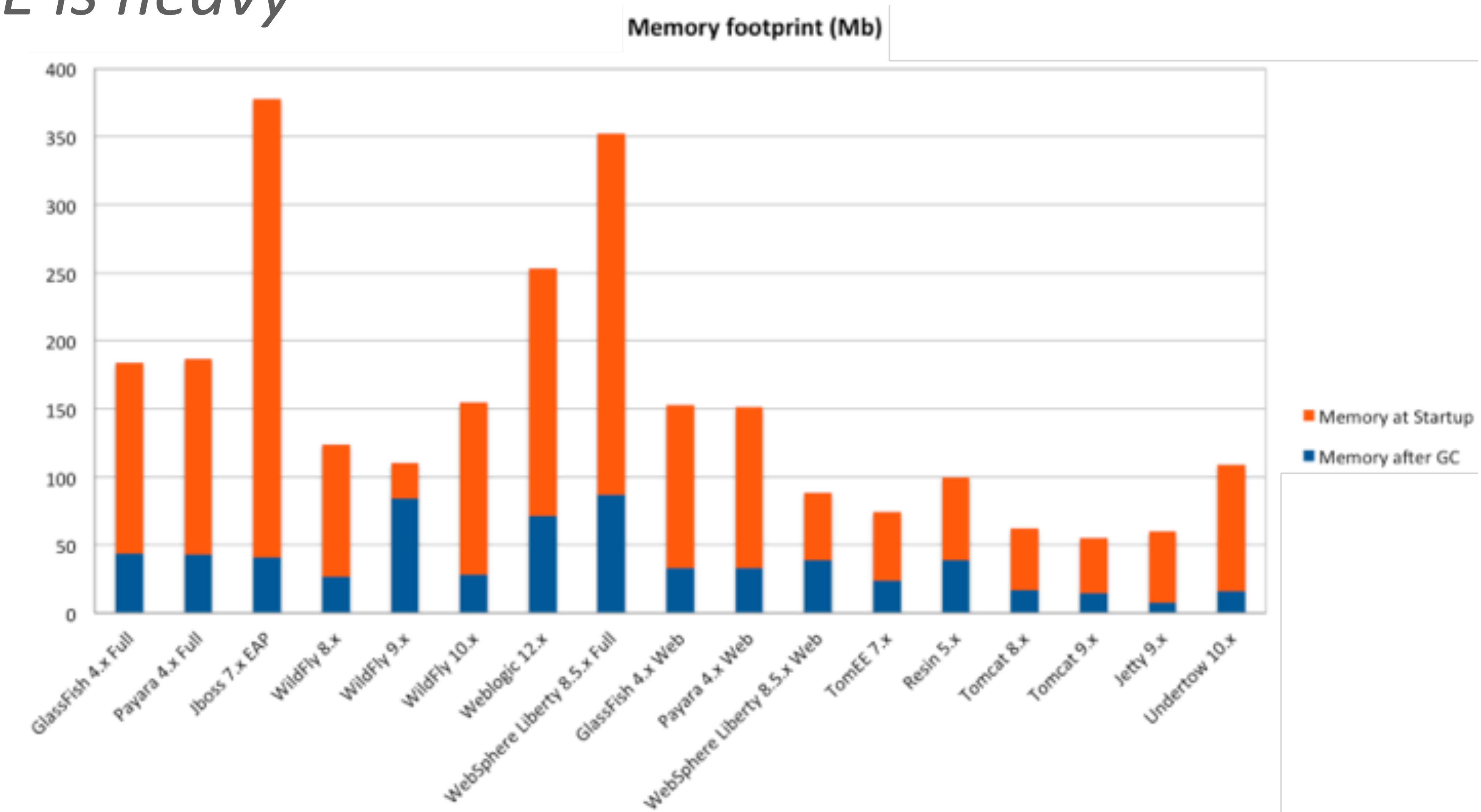
- FastSwap
  - Minimize deployment time during the development phase
- Partial Redplployment
  - Allows redeploying individual modules of a deployed enterprise application
- Module-Level Targeting
  - Enables to add a new application module without having to redeploy already deployed modules
- 3rd party solution

(Re)Deployment

- Production Redeployment
  - Deploy a new version of a service without stopping the current version
  - Clients already connected during the redeployment continue to use the older version of the service until they complete their work
  - Allows roll-back

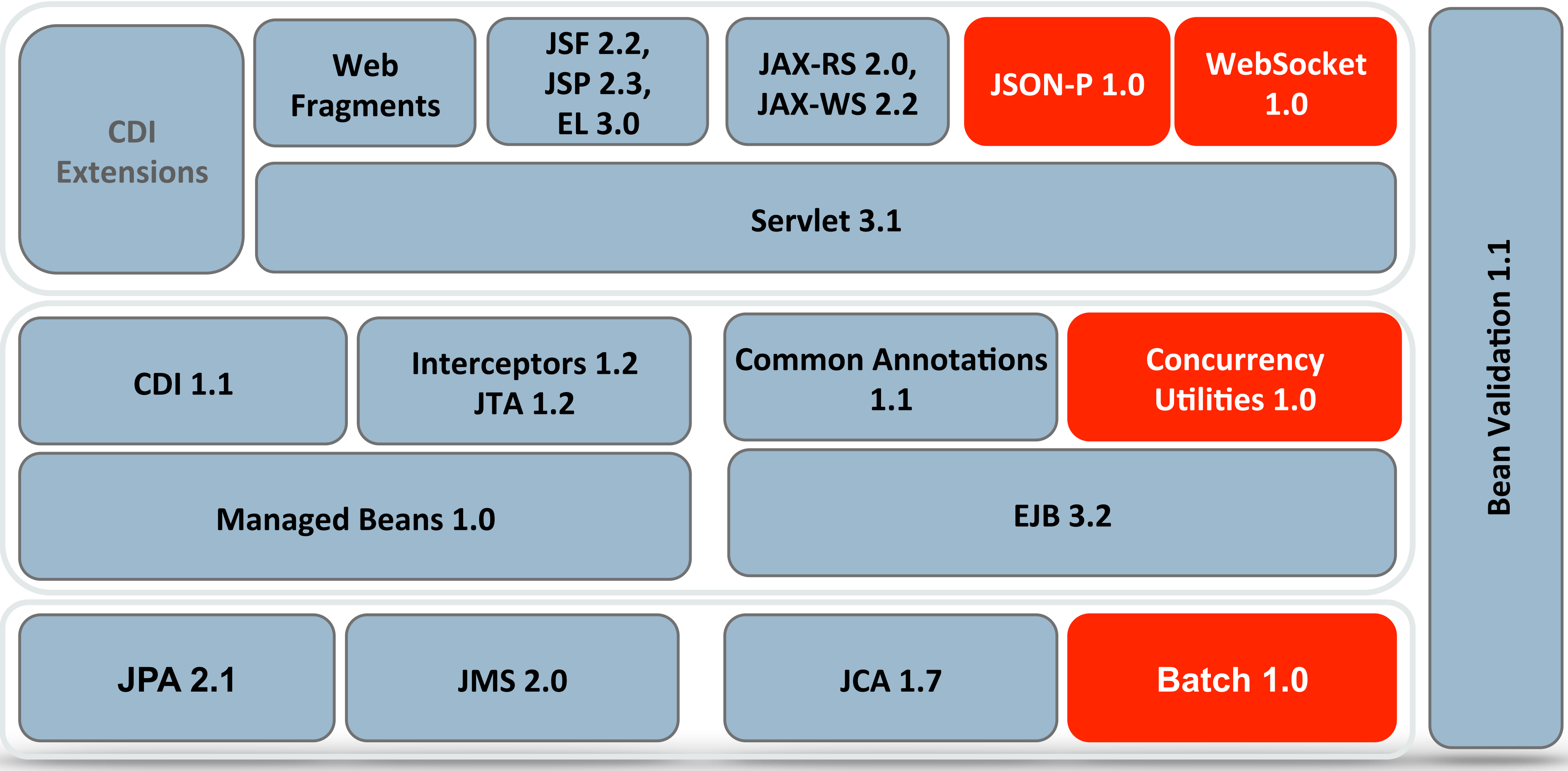
Componentization

“Java EE is heavy”



http://antoniogoncalves.org/2016/02/02/o-java-ee-7-application-servers-where-art-thou/

"Java EE is heavy"



CDI Extensions

Web Fragments

JSF 2.2, JSP 2.3, EL 3.0

JAX-RS 2.0, JAX-WS 2.2

JSON-P 1.0

WebSocket 1.0

Servlet 3.1

CDI 1.1

Interceptors 1.2 JTA 1.2

Common Annotations 1.1

Concurrency Utilities 1.0

Managed Beans 1.0

EJB 3.2

JPA 2.1

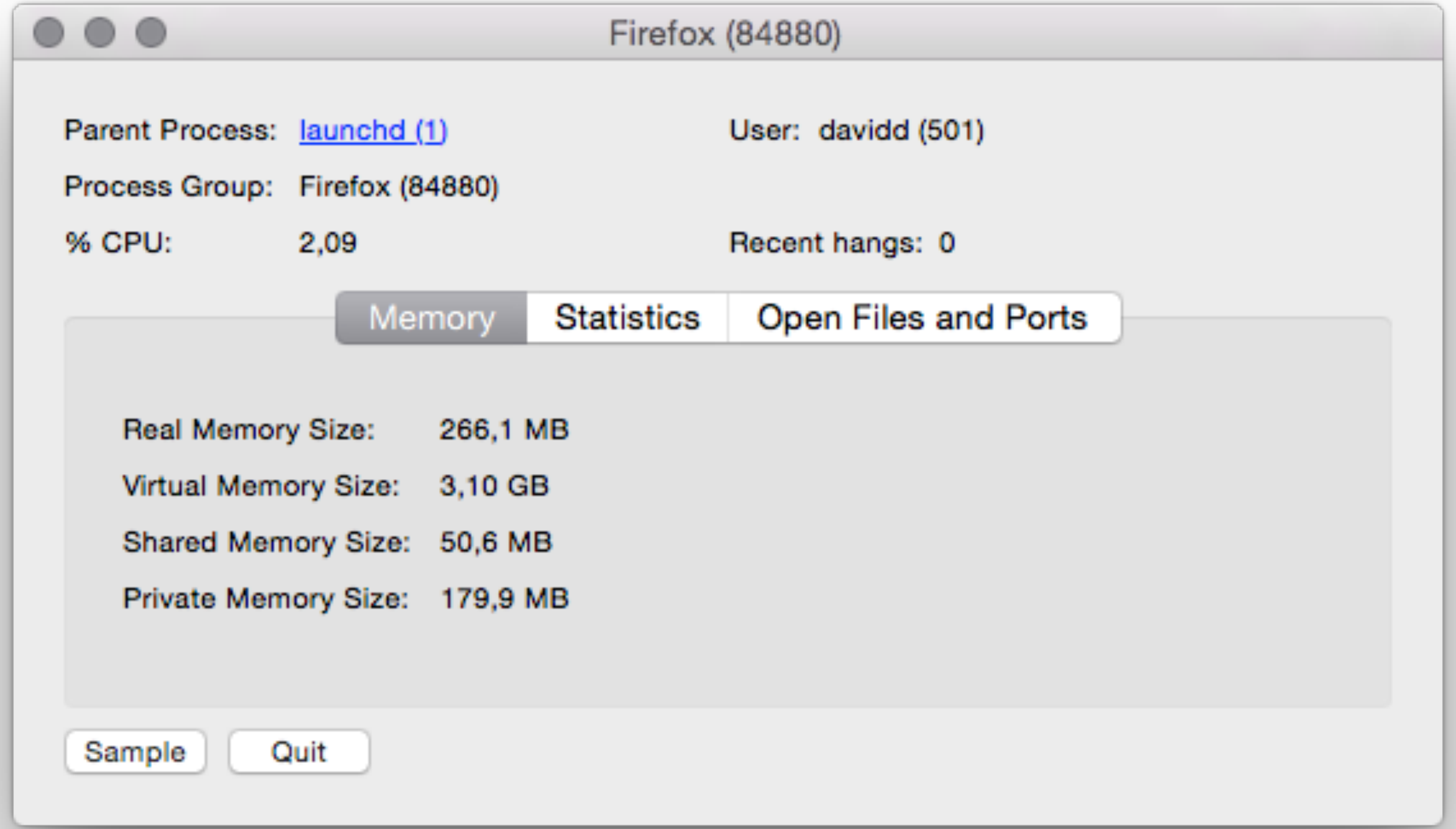
JMS 2.0

JCA 1.7

Batch 1.0

Bean Validation 1.1

"Java EE is heavy"



Firefox (84880)

Parent Process: launchd (1)

User: davidd (501)

Process Group: Firefox (84880)

% CPU: 2,09

Recent hangs: 0

Real Memory Size: 266,1 MB

Virtual Memory Size: 3,10 GB

Shared Memory Size: 50,6 MB

Private Memory Size: 179,9 MB

Firefox 44.0.2 on 10.10.5

“Java EE is heavy”

• Safari	433 MB
- GMail tab	566 MB
- Google tab	158 MB
• Text Wrangler	92 MB
• Twitter App	272 MB
• Keynote	466 MB
• Thunderbird	422 MB

WebLogic Server Multitenant

Density

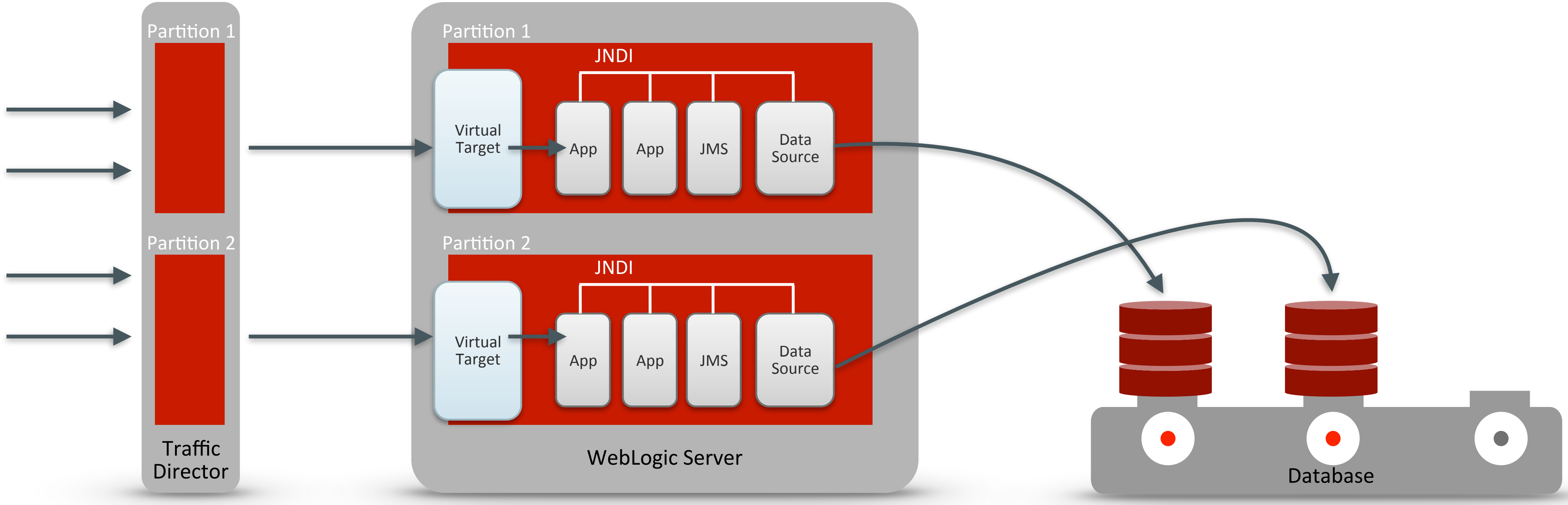
- Sharable infrastructure for use by multiple tenants
  - **Micro Containers**
- Tenant = custom conceptual grouping

WebLogic Server Multitenant

Domain Partition

- Applications and resources deployed for each partition
- No application changes required
- WebLogic infrastructure shared among partitions
- Provides resource isolation within a partition

WebLogic Server Multitenant



Isolation for Pluggable Partitions

Independence and Autonomy for Microcontainers

Runtime Isolation

- JDK and WebLogic partnership
- Heap, CPU, threads, requests…

Administrative Isolation

- Admin roles, lifecycle, troubleshooting

Traffic/Data Isolation

- Dedicated JNDI, segregated data
- Dedicated and shared Coherence caches

Security/Identity Isolation

- Realm, users per partition

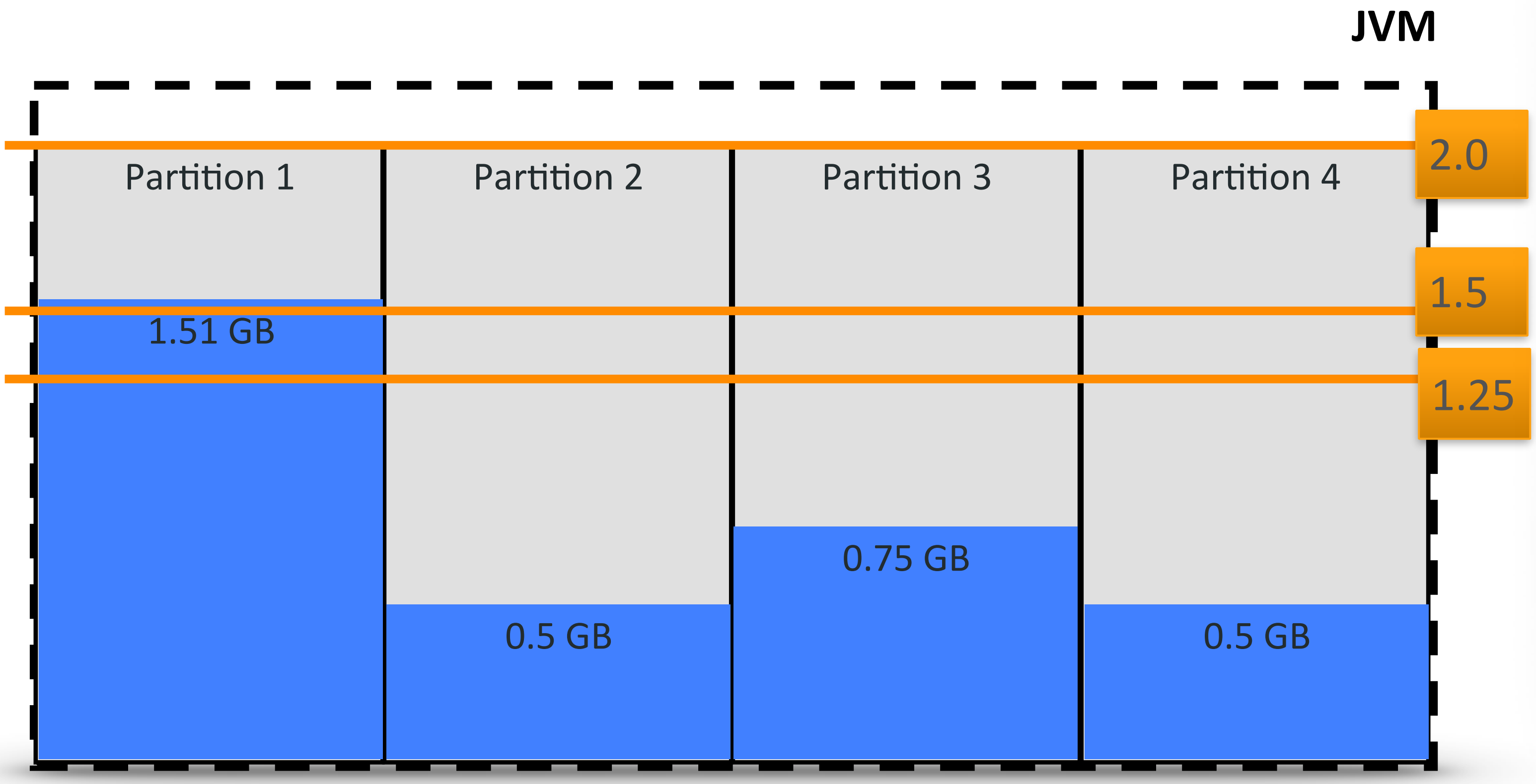
Resource Consumption Managers

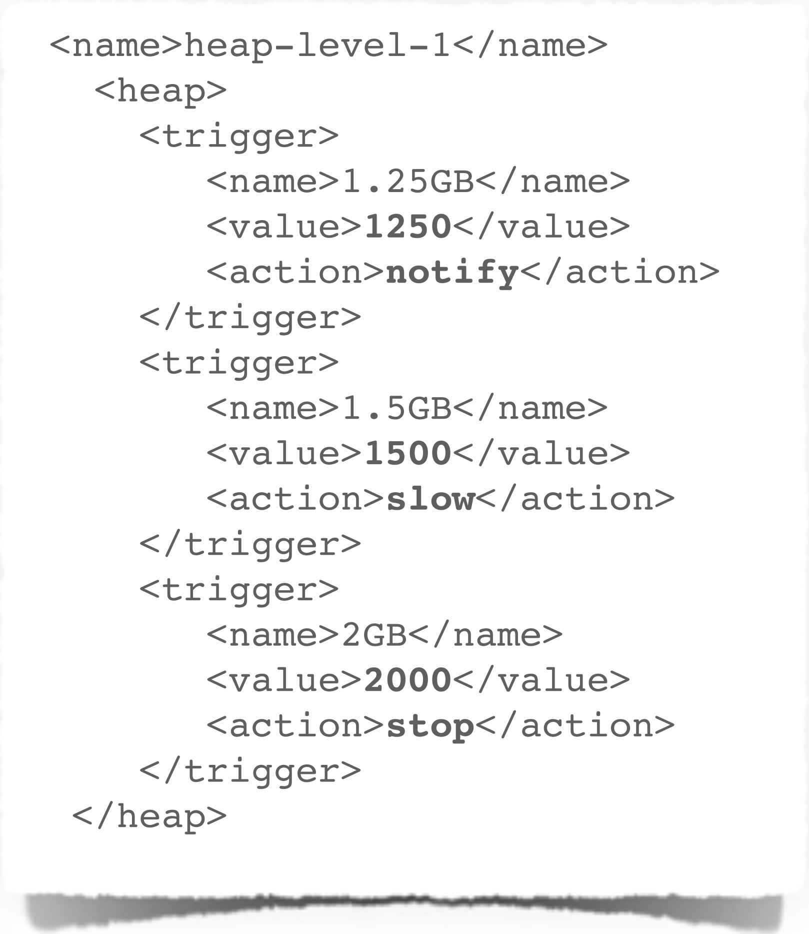
Runtime Isolation Within a JVM

- Deep integration between WebLogic Server and Oracle JDK
- Prevents resource hogging, protects applications in a shared JVM
  - Retained heap, CPU time, open file descriptors
- “Boundaries” and Fair Share usage patterns
- Triggerable actions
  - **Notify** – Inform administrator that a threshold has been crossed
  - **Slow** – Reduce partition’s ability to consume resources
  - **Fail** – Reject requests for the resource (file descriptors only)
  - **Stop** – Initiate the shut down sequence for the offending partition

Resource Manager Policy

Retained Heap Example





```
<name>heap-level-1</name>
  <heap>
    <trigger>
       <name>1.25GB</name>
       <value>1250</value>
       <action>notify</action>
    </trigger>
    <trigger>
       <name>1.5GB</name>
       <value>1500</value>
       <action>slow</action>
    </trigger>
    <trigger>
       <name>2GB</name>
       <value>2000</value>
       <action>stop</action>
    </trigger>
 </heap>
```

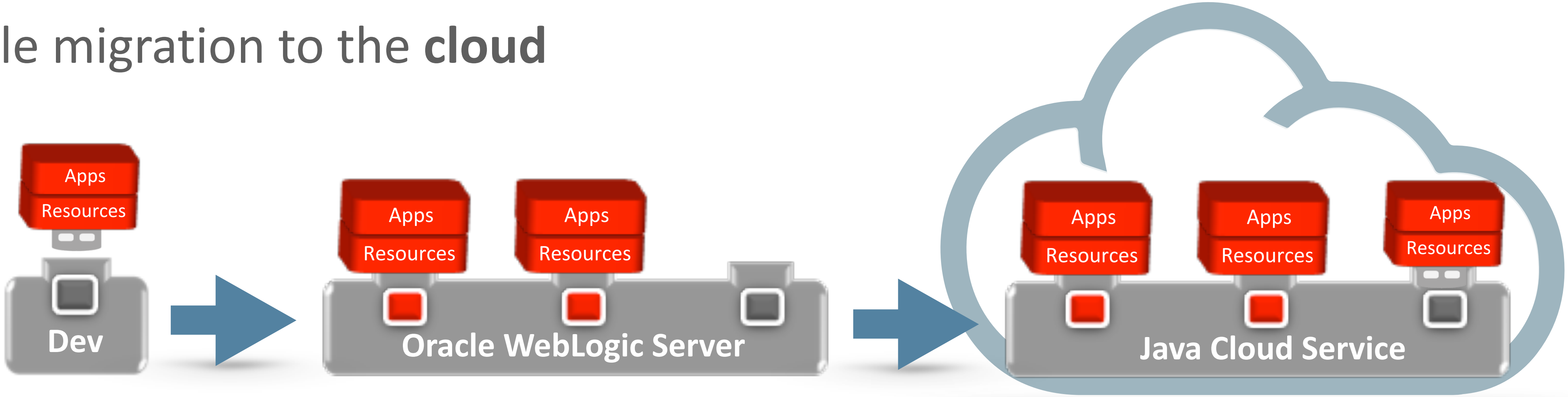
WebLogic Server Multitenant

Domain Partition

- Partitions are **isolated**
- Partitions can **span clusters**
- Partitions can be **started/stopped independently**
- Partitions can be **“exported”** and **“imported”**
- Partitions support **live migration**

Microcontainers in WebLogic Server 12.2.1

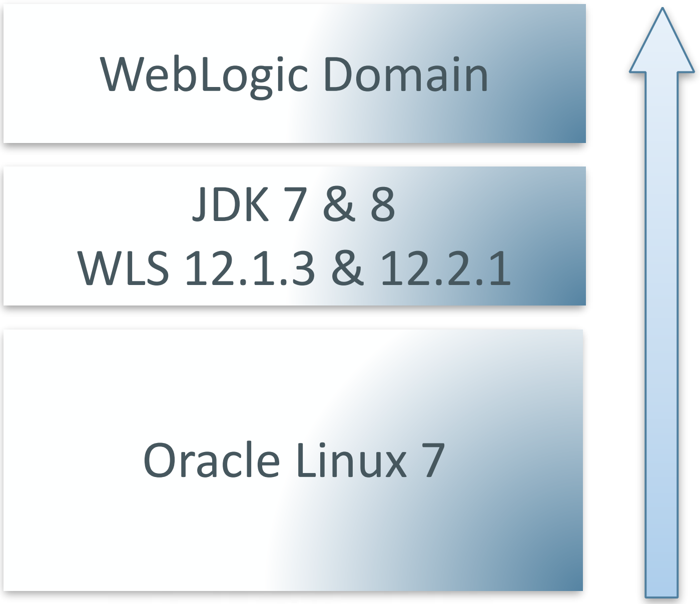
- Maximum **portability** between environments
- **Parity** between dev and production
- **Fast** startup/shutdown – disposability
- Easy **scale up**
- Enable migration to the **cloud**



WebLogic Server & Docker

Containerization

- Base Image
  - Oracle Linux 7 or RedHat 7
  - Pull from Docker Hub
- WebLogic Install Image
  - Download WebLogic Servers installers & JDK
  - Docker files extend base image with JDK 8 and a WLS 12.2.1 installation
- WebLogic Domain Image
  - Extend install image to create domains
  - Dockerfile on GH to create a domain configuration



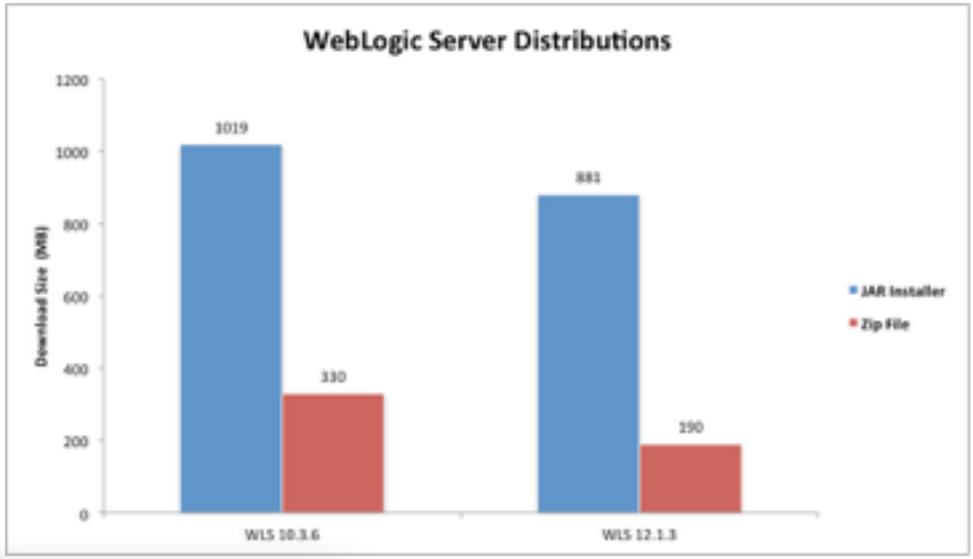
https://github.com/oracle/docker-images/tree/master/OracleWebLogic

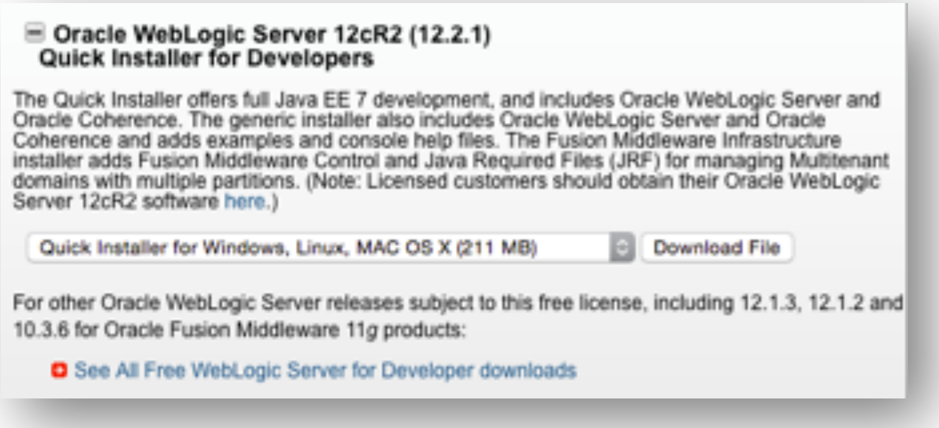
DevOps

WebLogic Server

Simplify installation

- Quick Installer
  - Unzip, execute configure script, ready to use
- Core WebLogic Server
  - Full console, WLST & Maven support





⊟ **Oracle WebLogic Server 12cR2 (12.2.1)**
Quick Installer for Developers

The Quick Installer offers full Java EE 7 development, and includes Oracle WebLogic Server and Oracle Coherence. The generic installer also includes Oracle WebLogic Server and Oracle Coherence and adds examples and console help files. The Fusion Middleware Infrastructure installer adds Fusion Middleware Control and Java Required Files (JRF) for managing Multitenant domains with multiple partitions. (Note: Licensed customers should obtain their Oracle WebLogic Server 12cR2 software here.)

For other Oracle WebLogic Server releases subject to this free license, including 12.1.3, 12.1.2 and 10.3.6 for Oracle Fusion Middleware 11*g* products:

See All Free WebLogic Server for Developer downloads

WebLogic Server

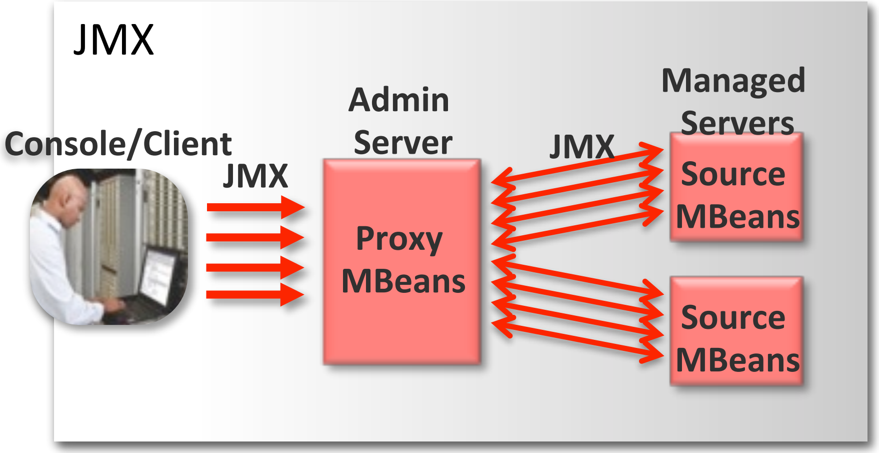
Simplify provisioning with automation, standardization, repeatability

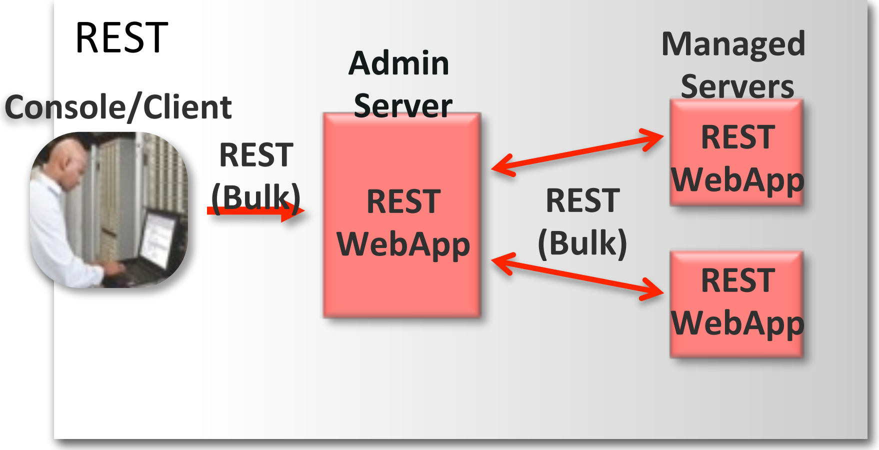
- WebLogic Scripting Tool
  - Configuration, deployment, lifecycle management
    - Record operations from console to bootstrap script development
  - Offline
    - Create domains based on templates
    - Read and modify domains
    - Create and modify templates
  - Online
    - Connect to Admin Server to interact with Mbeans

WebLogic Server

Simplify provisioning with automation, standardization, repeatability

- REST Management Interface
  - Dynamically generated interfaces
  - Asynchronous / Synchronous
  - Admin Server & Managed Servers
- Benefits
  - Comprehensive, simple & agnostic
  - Consolidated query & local processing
  - Faster response times (5x-10x)

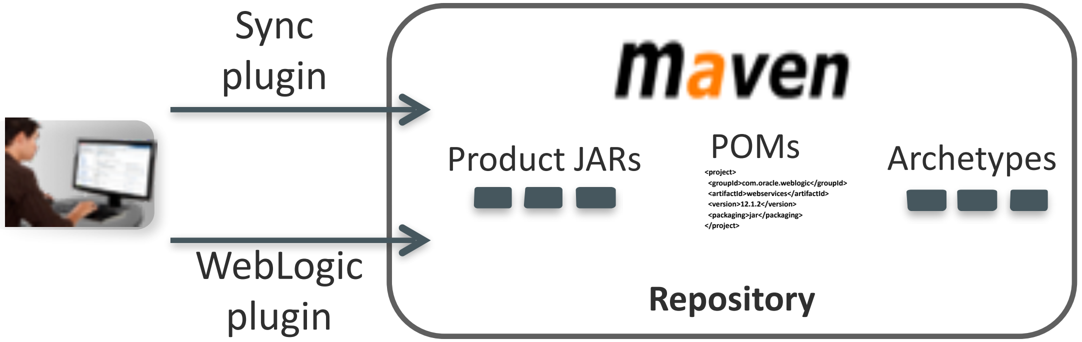




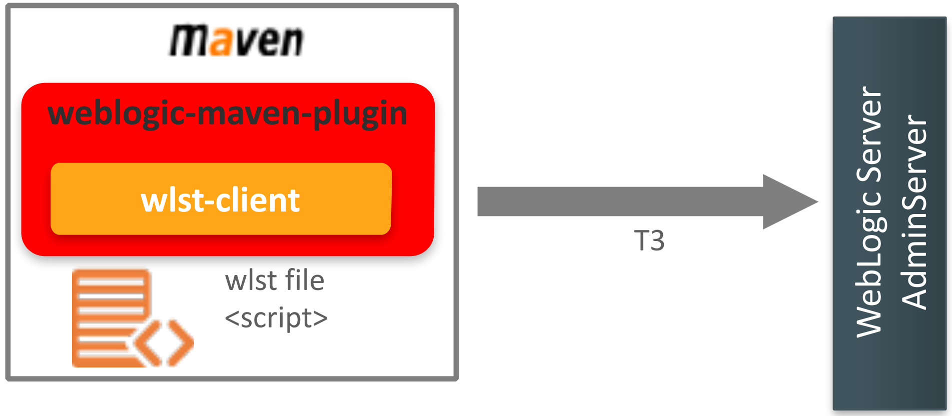
WebLogic Server and DevOps

WebLogic & Maven

- Maven Plug-In
  - Install, start, stop servers
  - Create domains, clusters
  - Configure and validate resources
  - Deploy, update, undeploy applications
  - WLST
- Public-facing Maven repository
  - Oracle artefacts (plugins, POMs, …)



```
<project>
  <groupId>com.oracle.weblogic</groupId>
  <artifactId>webservices</artifactId>
  <version>12.1.2</version>
  <packaging>jar</packaging>
</project>
```

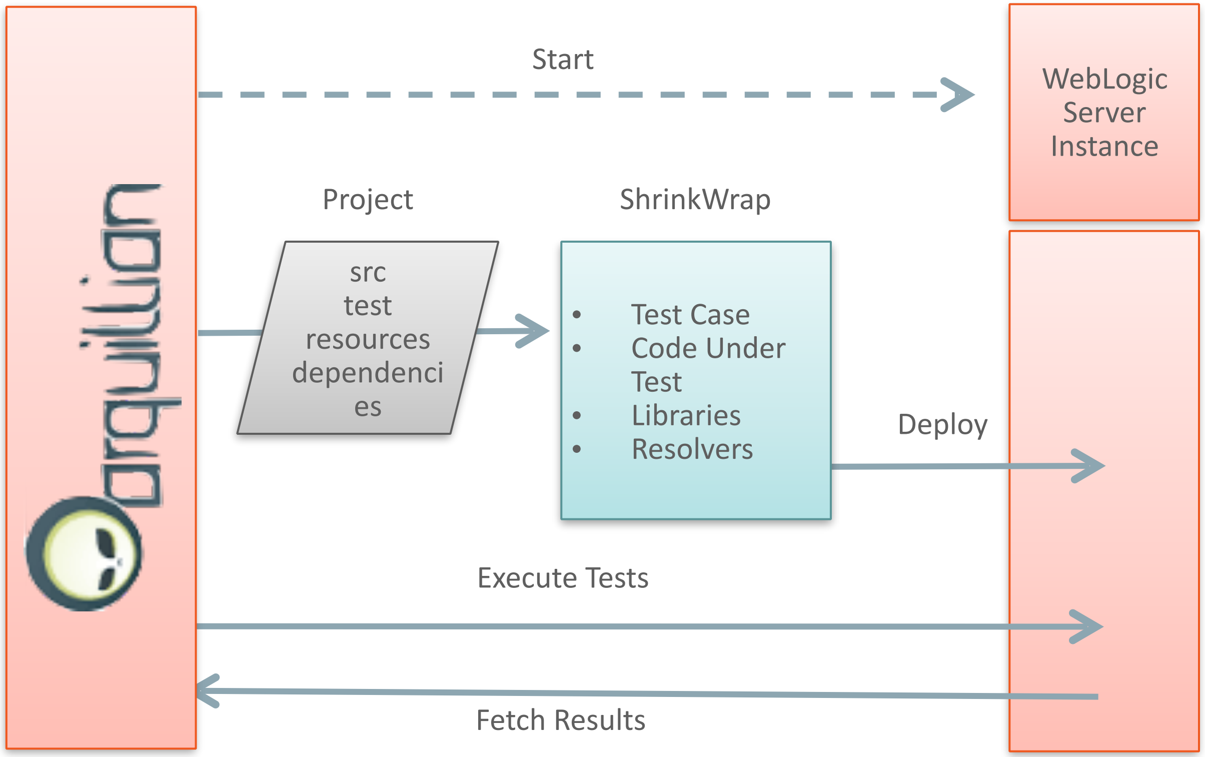


http://maven.oracle.com

WebLogic Server

Testing with Arquillian

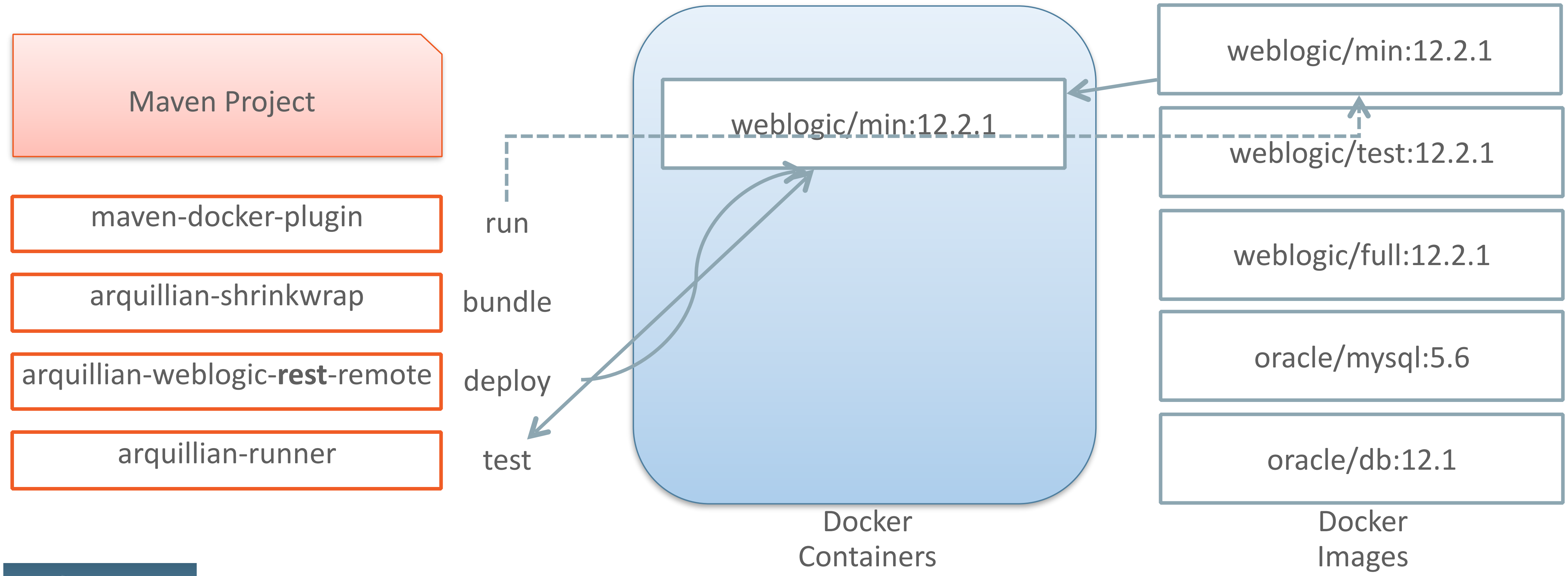
- Testing Framework
  - Server lifecycle management
  - Packages application + test code
  - Deploys, executes, reports test results
- Contributing to project – REST
- Adapters available now



http://arquillian.org/blog/tags/wls/

WebLogic Server

WebLogic, Maven, Arquillian & Docker



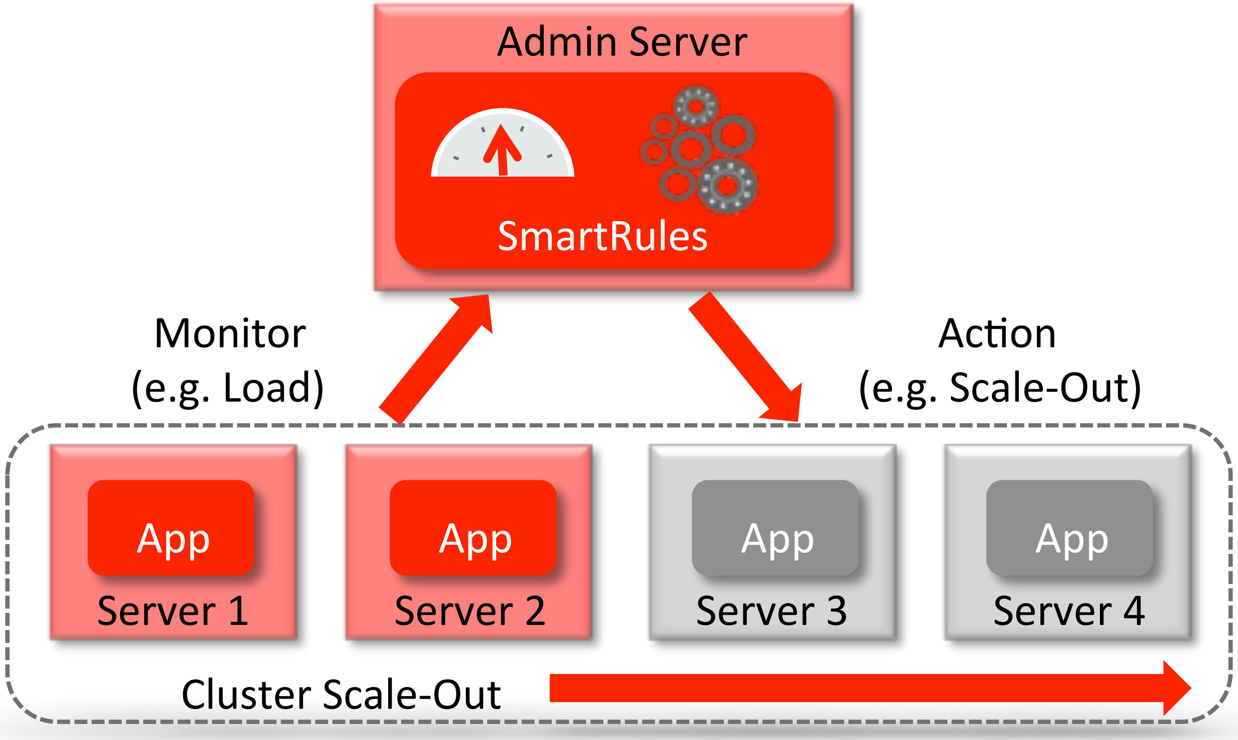
WebLogic Diagnostic Framework

Monitoring & Diagnostic Framework

- Gathering, analyze and persist diagnostic data
  - WLS and deployed applications
  - Ex. Monitor SLA violations, CPU Load; find bottleneck methods, …
- Archive & Harvester
- Monitoring Dashboard & Diagnostics Request Performance Page
- Policies & Actions
  - Observe specific diagnostic states
  - Send notifications based on configured rules
    - REST, SNMP, JMX, SMTP, JMS & Scaling

Automated Elasticity for Dynamic Clusters

Scaling



Automated Elasticity for Dynamic Clusters

Scaling

- Administration APIs for Dynamic Clusters
  - Start/stop a specified number of servers
  - Expand/shrink the size of the cluster
- Simple/automated scale up/down or tune
- Rules-based decisions based on capacity demand or schedule
- Watches, Notifications become Policies, Actions
  - **Policies**: SmartRules, Calendar-based policies
  - **Actions**: scaleUp, scaleDown, REST, script

Domain Partition Export/Import

Multi-tenancy

- Move partitions from one domain to another
  - Including deployed services
- Provides clean boundary around a deployment unit
  - Application bits + its resources (e.g. datasources, JMS destinations, etc.)
- Useful for replicating/moving partitions across domains
  - E.g. move from Dev to QA environment

Zero Downtime Patching

- Automatically orchestrates the rollout of patches and updates, without incurring any downtime or session loss
  - OTD & OHS integration
- Update roll-out - Initiate, Revert or Resume (failed) update
- Patch 'OracleHome' directory
  - WebLogic, JVM (and applications)
  - Production redeployment

Wrap-up

- Componentization
  - Docker
  - Micro-containers, pluggable partitions, isolation, …
- Infrastructure Automation / DevOps
  - WLST, REST, ZDT, WLDF, Elastic Scaling of Dynamic Cluster…
- Is it enough?
  - E.g. Circuit Breakers, Service Discovery, …
  - Culture!

https://www.oracle.com/weblogic

Hvala!

Java™

ORACLE®