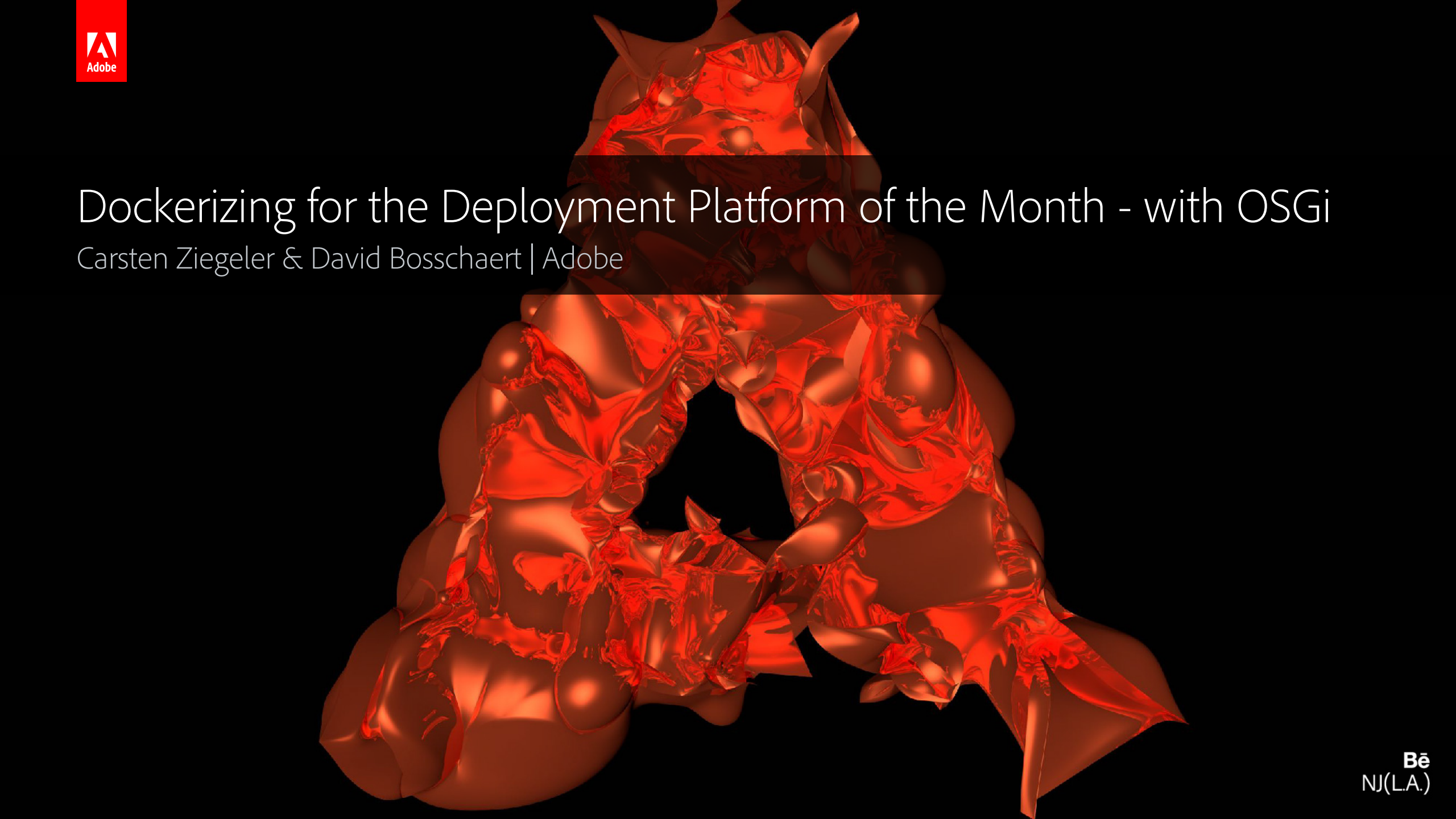


Dockerizing for the Deployment Platform of the Month - with OSGi

Carsten Ziegeler & David Bosschaert | Adobe

Bē
NJ(L.A.)

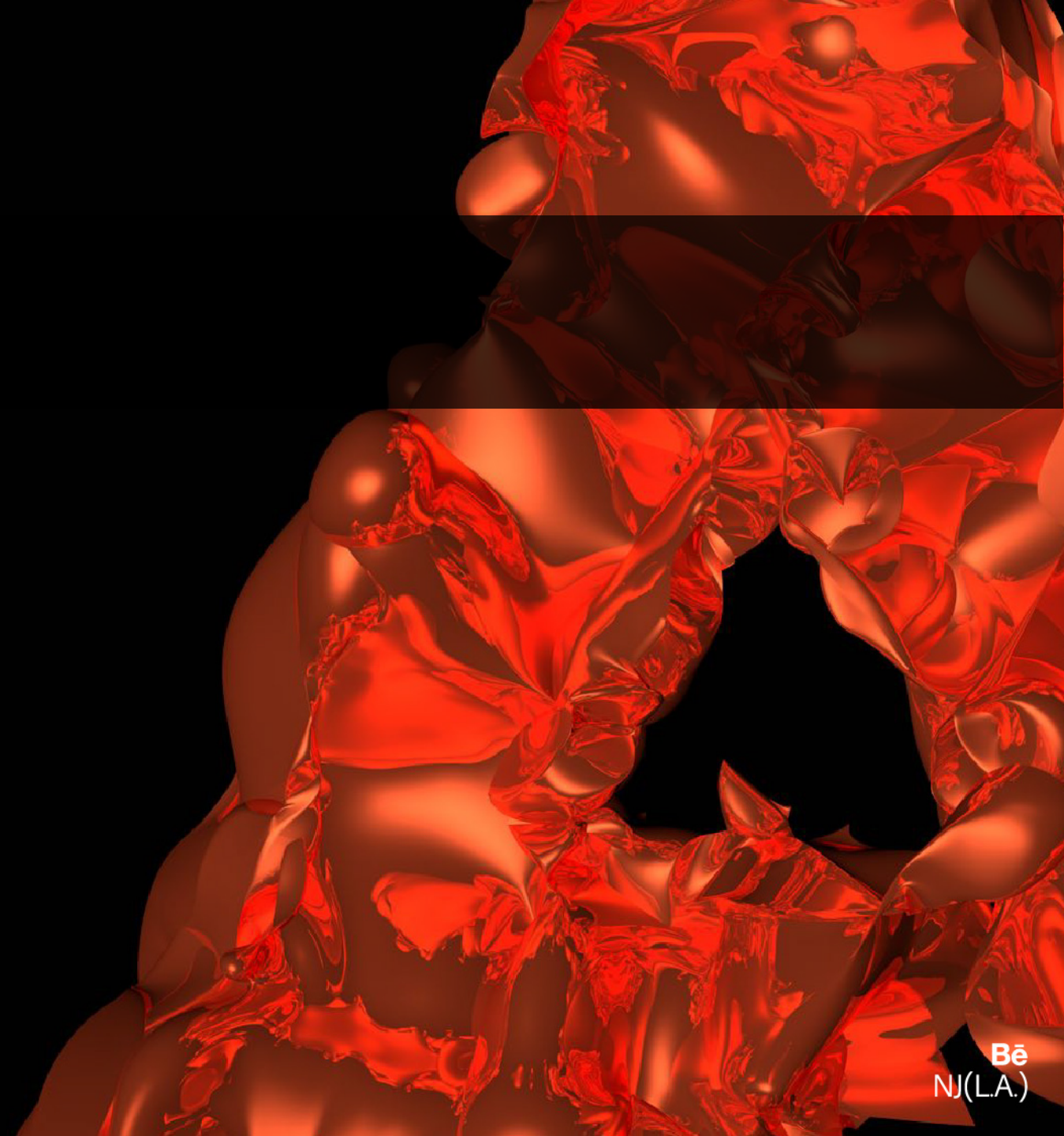
David Bosschaert

- Adobe R&D Ireland
- OSGi EEG co-chair
- Apache member and committer
- ISO JTC1 SC38 (Cloud Computing) Ireland committee member

Carsten Ziegeler

- RnD Adobe Research Switzerland
- Member of the Apache Software Foundation
- VP of Apache Felix and Sling
- OSGi Expert Groups and Board member

Create...



Bē
NJ(L.A.)

Containers to the Rescue

- Rapid / Simplified application deployment
- Lightweight, reduced overhead
- Portability across machines
- Sharing

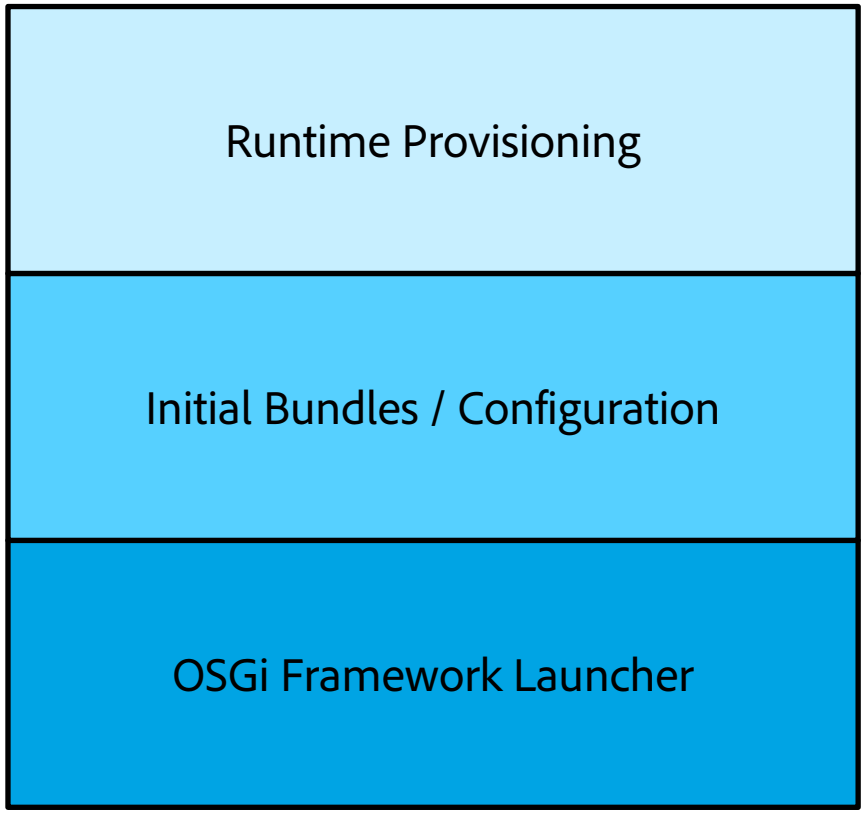
C

Application Canning

- OSGi app == Java App
- Various solutions
- Apache Karaf
  - **Distribution**
- Apache Sling
  - **Launchpad**
  - **Provisioning Model**

C

Common Setup




```
<feature name="my-project" version="1.0.0">
   <bundle>mvn:com.mycompany.myproject/myproject-dao</bundle>
   <bundle>mvn:com.mycompany.myproject/myproject-service</bundle>
   <config name="com.foo.bar"/>
 </feature>
```

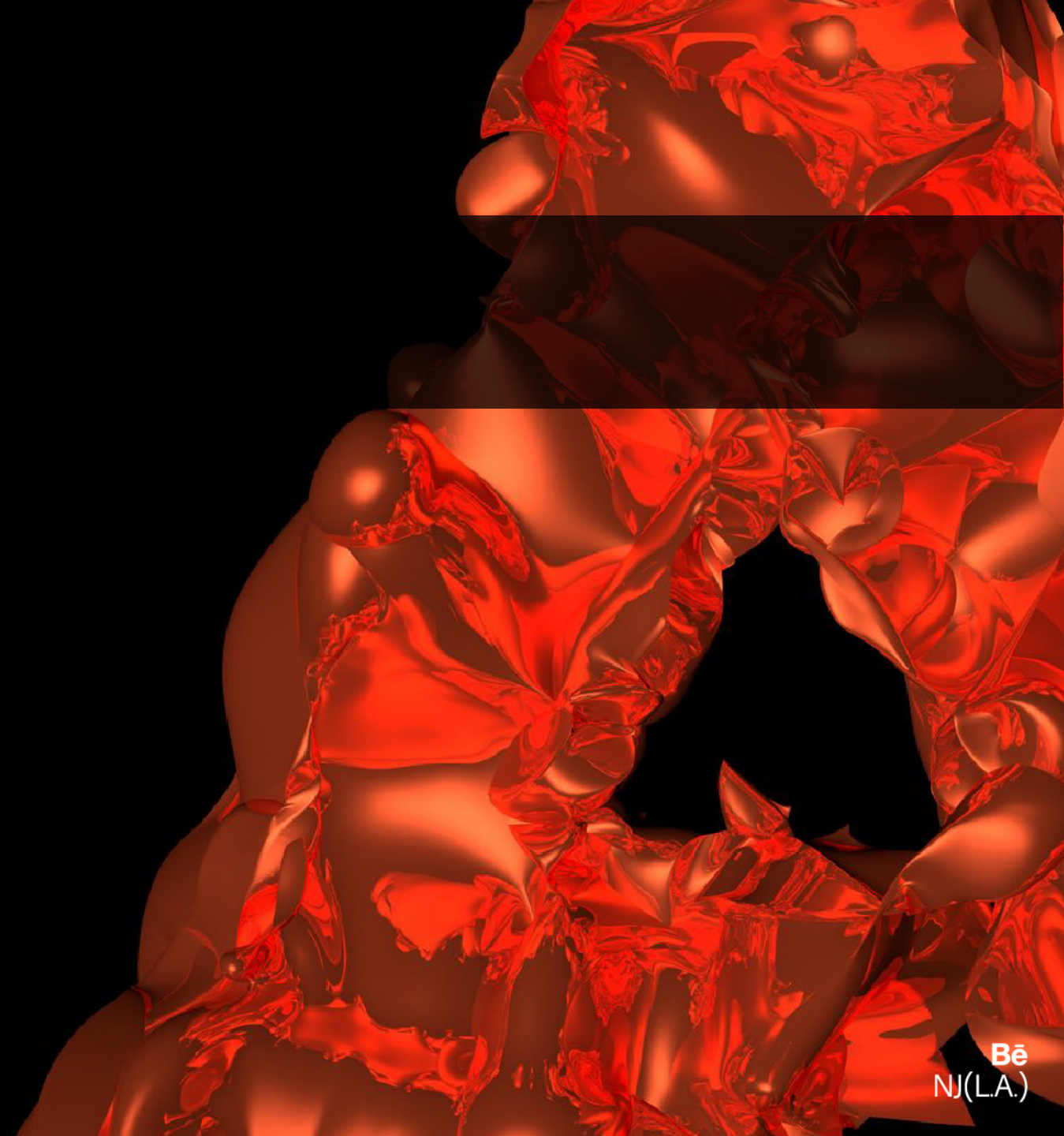
```
[feature name="my-project"]
   com.mycompany.myproject/myproject-dao/1.0.0
   mvn:com.mycompany.myproject/myproject-service/1.0.0
[configurations]
   com.foo.bar
```

C

OSGi Goodies

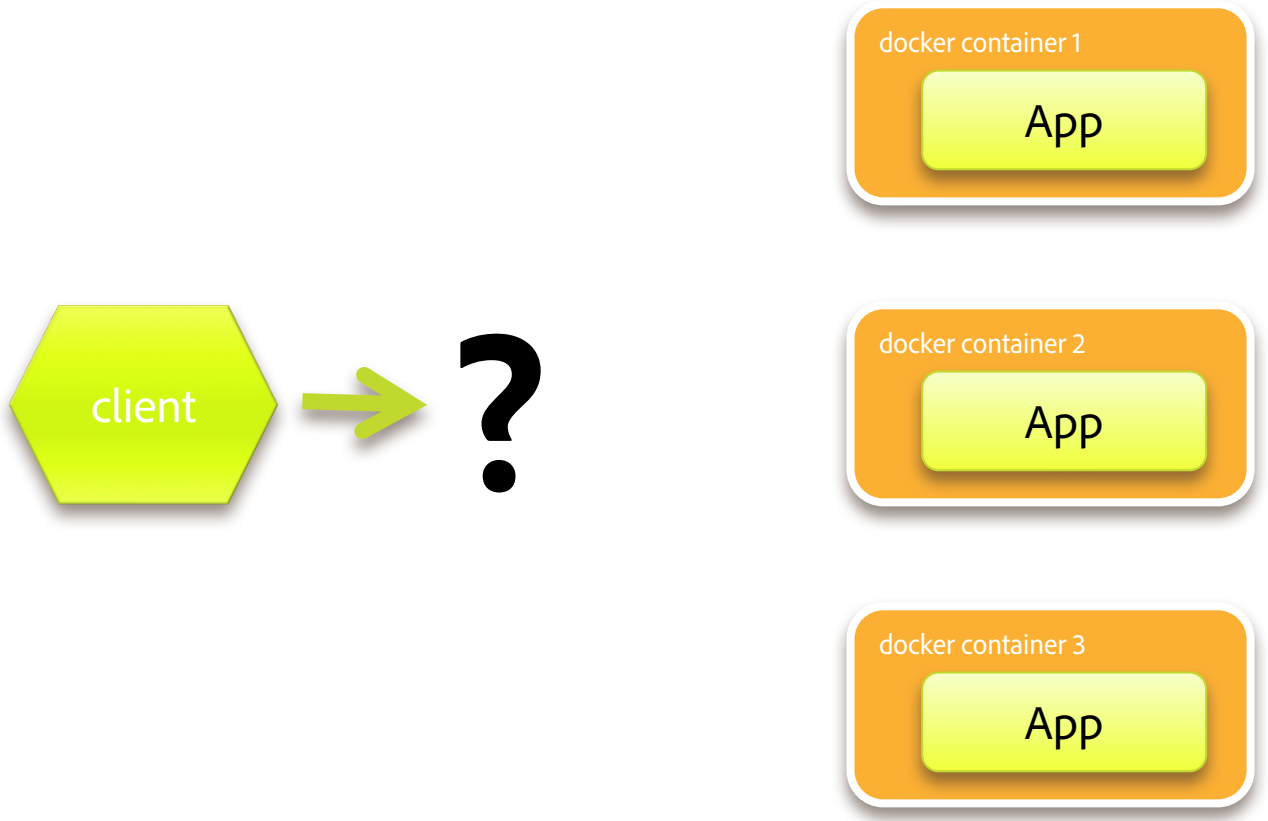
- Modularity
- Requirements and Capabilities
  - **Explicitly declare and isolate dependencies**

Run...

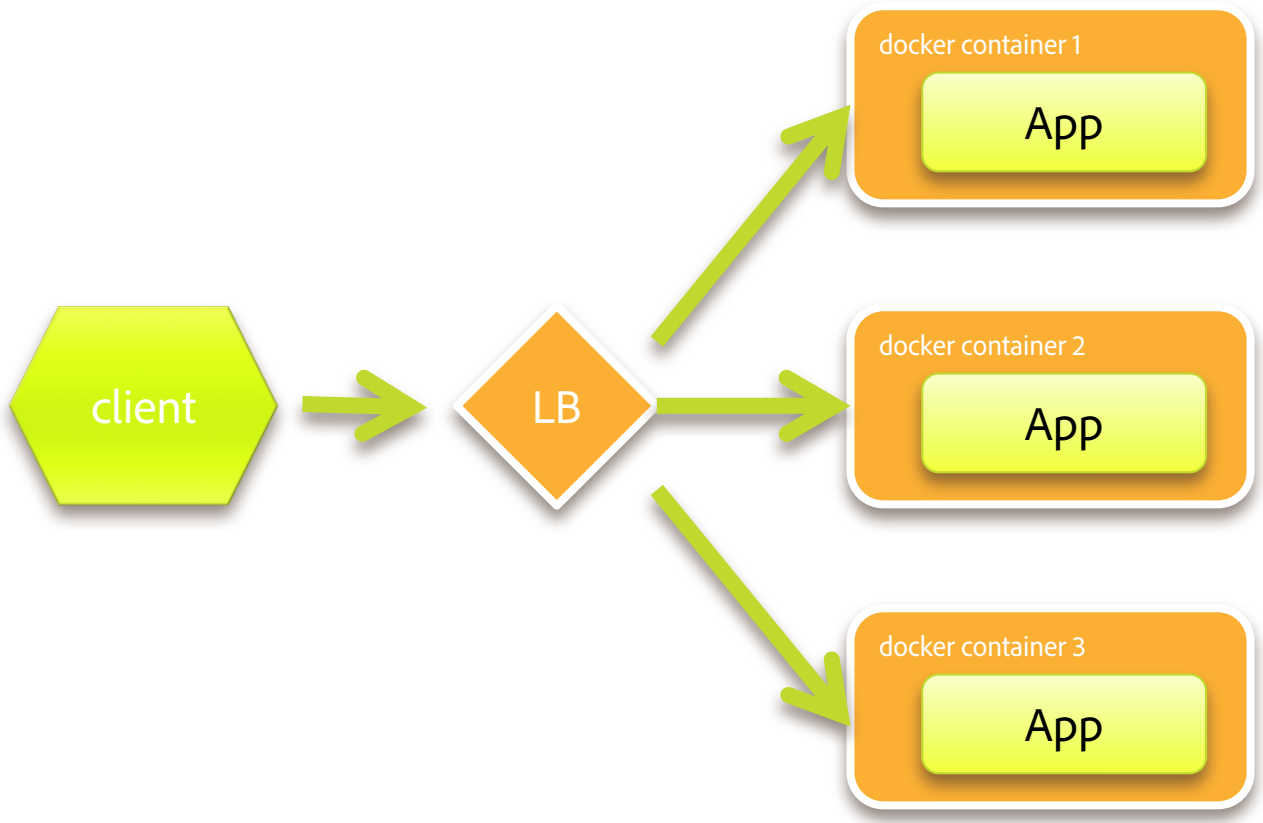


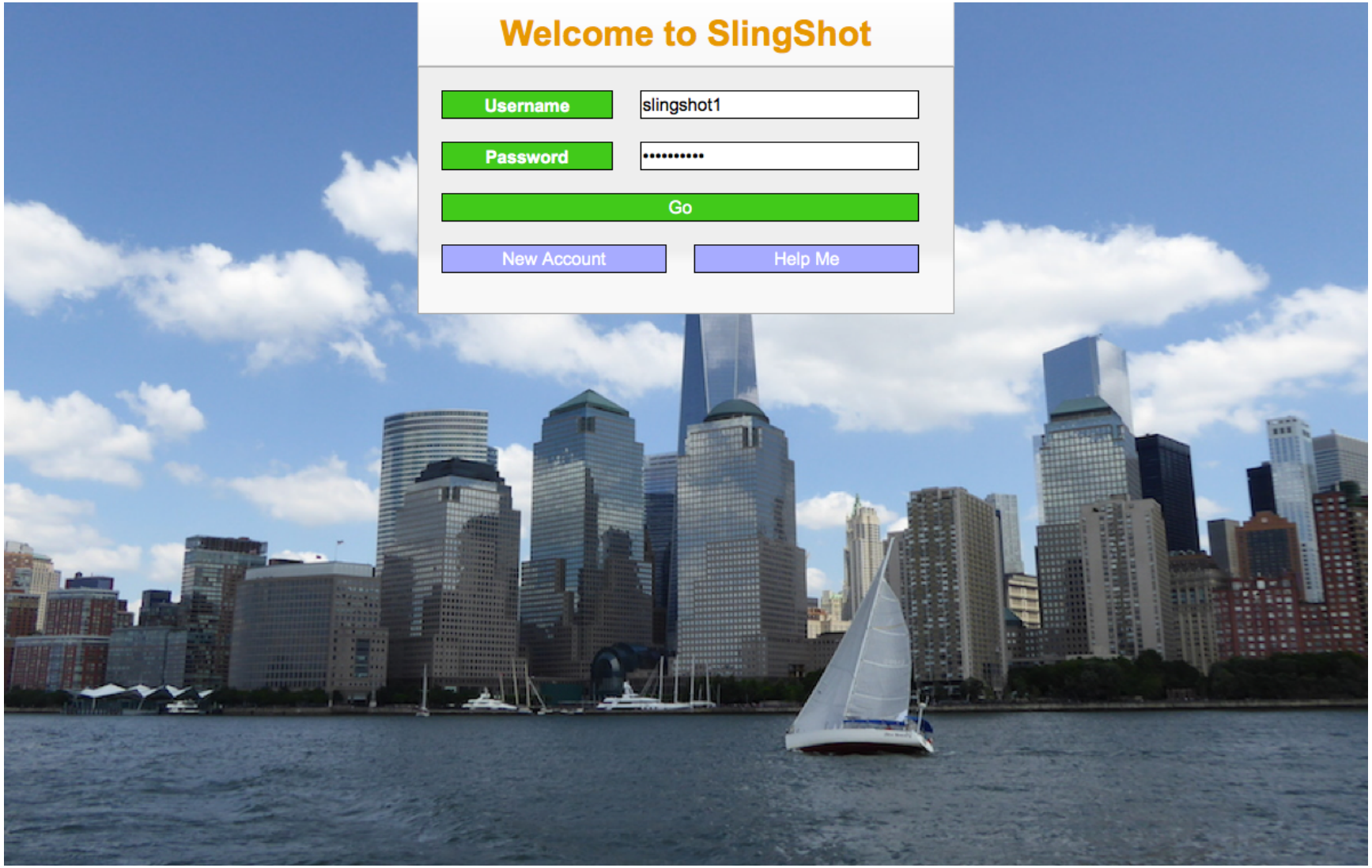
Bē
NJ(L.A.)

Yay dockerize your app!



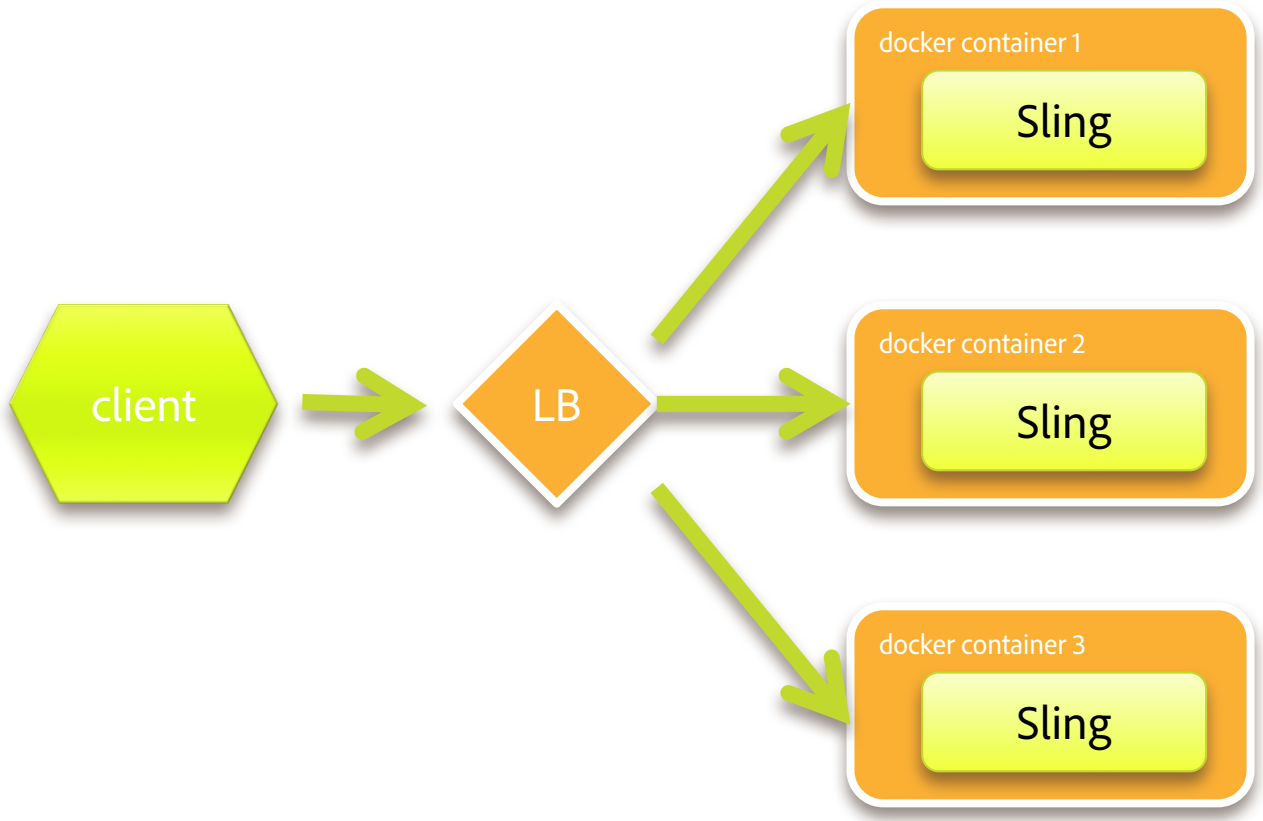
You need a load balancer



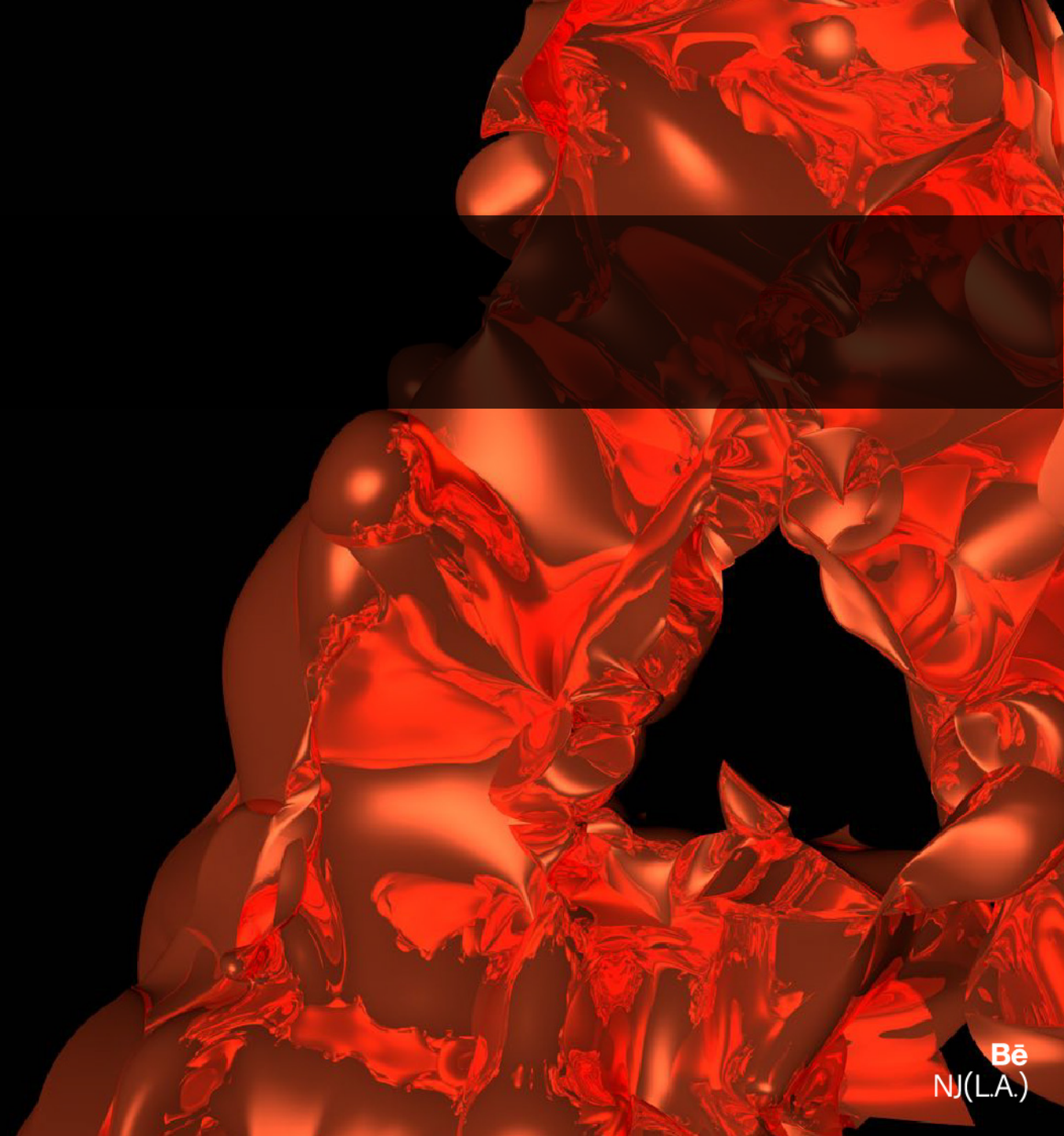


C

Example deployment: Sling Containers!



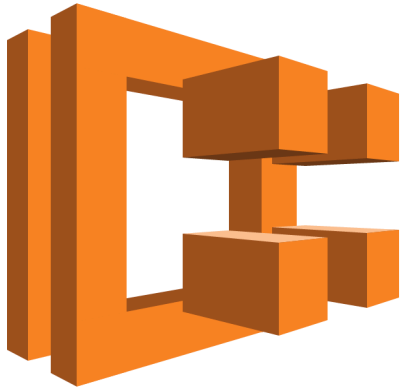
Manage...



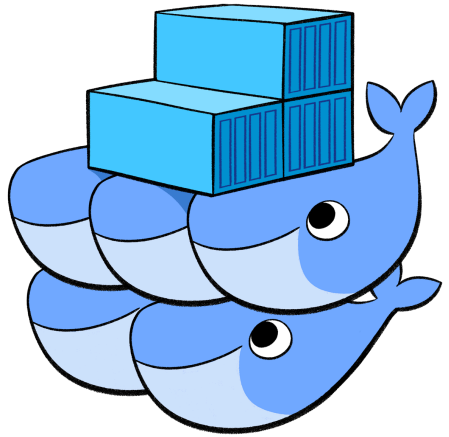
Solutions

MARATHON

MESOS



Amazon ECS



Docker Swarm

kubernetes

D

Adobe

Here's how…

- Docker Swarm/Compose

  - docker-compose up –d
  - docker-compose scale mysling 3
- DC/OS Marathon
  - curl -i -H 'Content-Type: application/json' -d @<mysling.json> http://mymarathon:8080/v2/apps

- Kubernetes
  - kubectl run mysling-node --image=**cziegeler/apacheslingdemo** --port:8080
    kubectl expose deployment mysling.node --type="LoadBalancer"
- Amazon ECS
  - aws ecs register-task-definition --family myslingfam --container-definitions '[{"name":"mysling","image":"**cziegeler/apacheslingdemo**","cpu":1,"memory":512,"essential":true}]'
  - aws ecs create-service --service-name myslingsvc --task-definition myslingfam --desired-count 3

D

Everybody takes Docker containers

- ... but applications are defined using various proprietary APIs
- makes it ***very*** labour intensive to switch
- can we have a common API please?

D

Or how about a common API

- Started in Apache Aries Containers:
  http://aries.apache.org/modules/containers.html
- Currently implementation for Marathon / DCOS
- And implementation to locally execute docker
- Run the same cluster either on your local dev machine as well as in DCOS / Marathon
- Other implementations welcome
  - **e.g. Kubernetes**

D

Or how about a common API

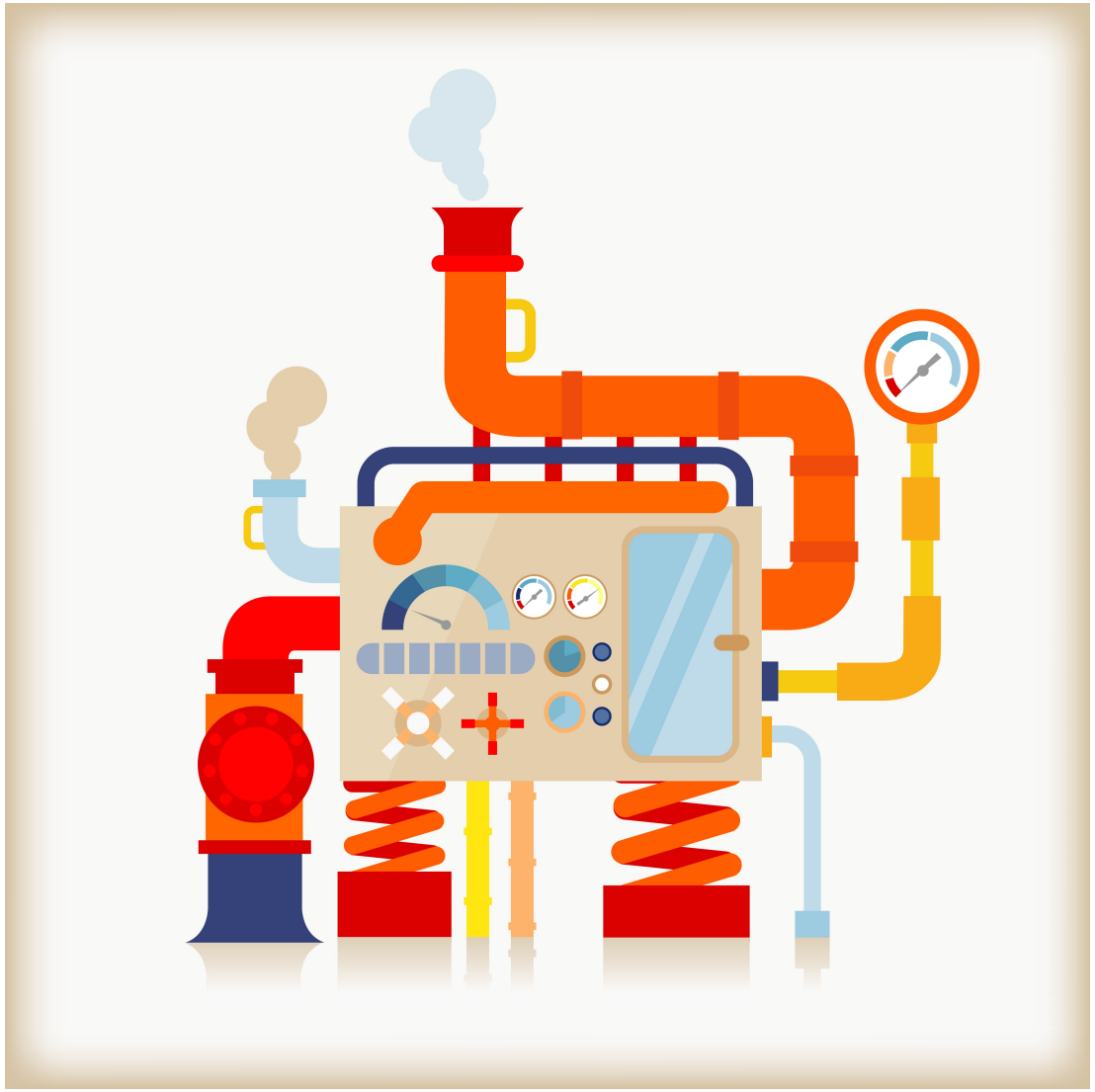
```
ServiceManager sm = … from service registry …

ServiceConfig sc = ServiceConfig.builder(" mysling ", "cziegeler/apacheslingdemo").
  cpu(1.0).
  memory(512).
  port(8080).
  instances(3).
  build();

Service svc = sm.getService(sc);          // The containers are now created
```

D

Demo!



OSGi RFP 179 – Compute Management Service

- Accepted OSGi RFP
- Independent API to manage compute nodes
- Can work with RFC 183
  - Cluster Information Specification
  - (previously Cloud Ecosystems)
  - Node discovery, metadata and provisioning

5 Requirements

CM0010 – The solution must provide a mechanism to create and destroy compute nodes.

CM0020 – It must be possible to implement the solution for existing compute platforms, including, but not limited to, cloud-based platforms, grid computing-based platforms and/or container-based platforms.

CM0030 – The solution must provide a mechanism to specify the root image to run on the compute node.

CM0040 – The solution must provide a mechanism to specify compute node parameters such as the amount of memory, cpu and exposed network ports.

CM0050 – The solution must provide a means to specify environment variables to be set in the compute node.

CM0060 – The solution must provide a mechanism to run an executable in the compute node once available.

CM0070 – The solution must support listing and querying of compute nodes.

CM0080 – The solution must support specifying that more than one node of a given definition be created.

CM0090 – The solution must allow the number of requested nodes of a given definition to be changed after the initial launch of the associated compute node(s).

CM0100 – The solution must support assigning logical names to nodes that can be used to address them.

CM0110 – The solution must allow integration with security features offered by the compute platform.

CM0120 – The solution must provide support for load balancers across a cluster of nodes.

D

Discussion


Adobe