

# PaaSterns for Java developers in the cloud

*(aka The Cloud with no pain in the PaaS)*

**Sacha Labourey**  
**CEO, CloudBees, Inc.**

**September 18/19, 2012 - Zurich/Berne**

# Introduction – Sacha Labourey



- Born in 1975 in Switzerland
- JBoss
  - Clustering lead – 2001
  - GM Europe - 2003
  - CTO - 2005
- Red Hat
  - JBoss acquisition in June 2006
  - co-GM of Red Hat's middleware division
  - Left Red Hat in April 2009
- CloudBees
  - Started in April 2010
  - About 30 bees in 6 countries



# Agenda

- What does “Cloud” mean to developers?
  - IaaS vs. PaaS vs. SaaS
- A few words on CloudBees
- A few PaaSterns
- Closing comments



# What does “Cloud” mean to developers?

---

IaaS vs. PaaS vs. SaaS



Easy: « Just use a XaaS! »

**SaaS**



**PaaS**



**IaaS**



• Click to edit Master text styles  
Traditional software stack  
We have done this for 20 years!

Application  
Second level  
Third level

LB • Fourth level  
– Fifth level

AS

JVM

VM

Hypervisor

Server

You

# Cloud Computing: How to do it? Who does what?

– Second level

– Third level

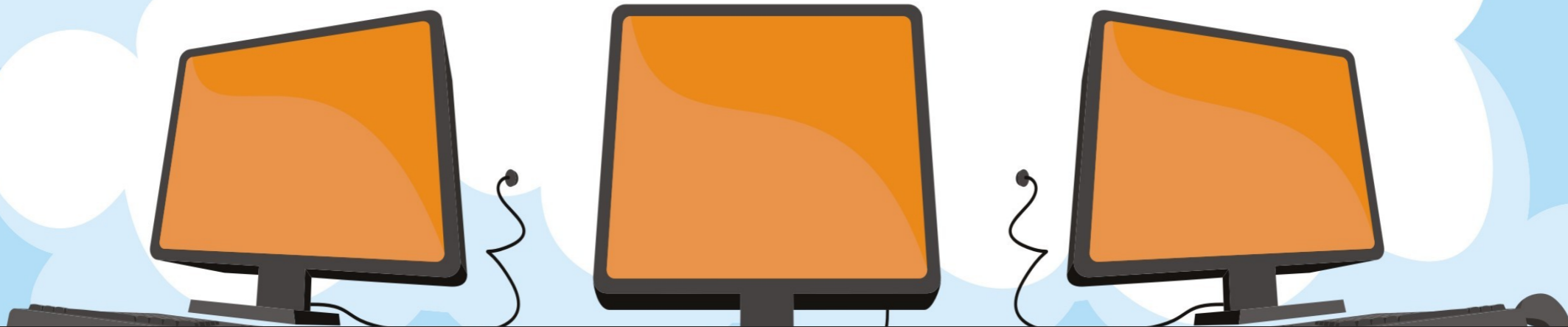
- Fourth level

- Fifth level

JVM

VM

**Cloud  
Provid  
er?**

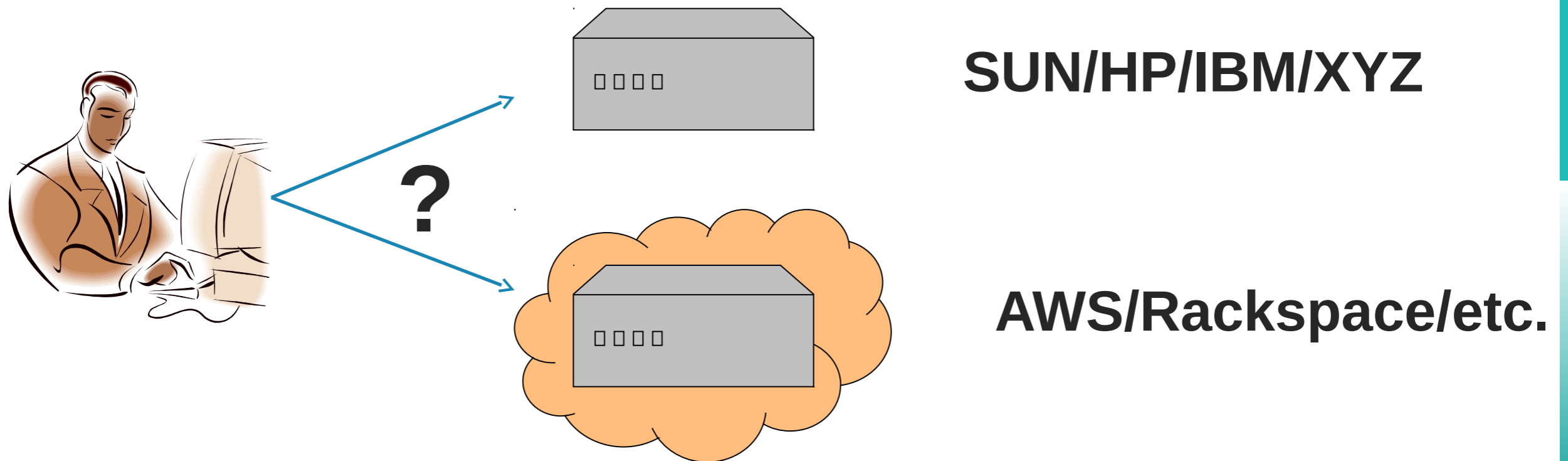


# IaaS

- AWS – the most popular example
- Server Lego blocks – VM, storage, IP, etc.
- Development environment:
  - “Give me a server, an OS, a virtualization layer, an application server, a firewall, a database, I’ll deal with it! And patch it. And monitor it. And...”
  - Flexible but c0mpLiCaTed
- User point of view
  - Custom systems and environments



# IaaS – Let's try not to change too much...



**Must probably be the same...**

# IaaS - Congratulations!



```
ubuntu@domU-12-31-39-00-ED-A3 - byobu
File Edit View Terminal Help
Linux domU-12-31-39-00-ED-A3 2.6.31-300-ec2 #3-Ubuntu SMP Sat Sep 26 10:31:44 UTC
C 2009 i686

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/

System information as of Mon Nov  2 23:09:43 UTC 2009

System load: 0.0          Memory usage: 5%    Processes:    59
Usage of /:  9.4% of 9.92GB  Swap usage:  0%    Users logged in: 0

Graph this data and manage this system at https://landscape.canonical.com/

-----
At the moment, only the core of the system is installed. To tune the
system to your needs, you can choose to install one or more
predefined collections of software by running the following
command:

sudo tasksel --section server
-----
ubuntu@domU-12-31-39-00-ED-A3:~$
```

ec2-terminate-instances  
ec2-purchase-reserved-instance-of  
ec2-reboot-instances  
ec2-detach-volume  
ec2-stop-instances  
ec2-create-image  
ec2-revoke  
ec2-allocate-address  
ec2-migrate-image  
chef  
ec2-attach-volume  
puppet  
ec2-associate-address  
ec2-delete-snapshot  
ec2-release-address  
ec2-delete-volume  
ec2-get-params  
ec2-register  
ec2-start-instances  
Puppet

**“Great” news: you have now become responsible for the data-center!**

# IaaS – Consequences

- ✓ Directly managing your IaaS resources provides you with resource elasticity (CAPEX-free), but...
- The amount of soft-IT typically required to do so is ... higher!
  - All of the traditional IT activities remain (maintain/patch/monitor OS+JVM+AS+DB++, etc.)
  - + cloud-specific items: elasticity/security/automation
- I can read your mind: this **is** where you will start
- My advice? Move on...

# SaaS

- Salesforce.com, Zendesk, NetSuite, etc.
- Development environment
  - Rigid – mostly through CONFIGURATION
  - When available “development” takes place **within** the SaaS itself
- User point of view
  - Standard applications
  - Very fast bootstrap
  - Most of the time, lock-in is very high





# PaaS

- Cloud concepts are applied to **Applications** and **Data**
  - On-demand, pay-as-you-go, elasticity, etc.
  - No need to handle updates, patches, scalability, failover, etc.
- Development environment
  - “Give me my typical development environment and manage everything else for me – servers, scalability, etc.”
- User point of view
  - Custom applications
  - Harder to “grasp” initially



This is a **Service**, not just some **Software!**

# One last warning...

- Could I please get a « Private PaaS »?
  - Sure! We can also sell you a private jet!
  - Very tempting! All of the advantages but no decision about the cloud is needed! And I'll be able to customize it!
- Yes, but...
  - A great part of the value from a PaaS comes from the « S »: SERVICE
  - With a public PaaS, you are outsourcing your Operations & DevOps
  - With a Private PaaS you'll get a better « interface » between DEV and IT compared to traditional middleware, but, overall, you remain in charge of all operations!
  - And remember: Customization is the root of all evil!
  - « Can I get a 160V plug just for this toaster? »

# Why am I telling you all of this?

---

Enter into the world of CloudBees!

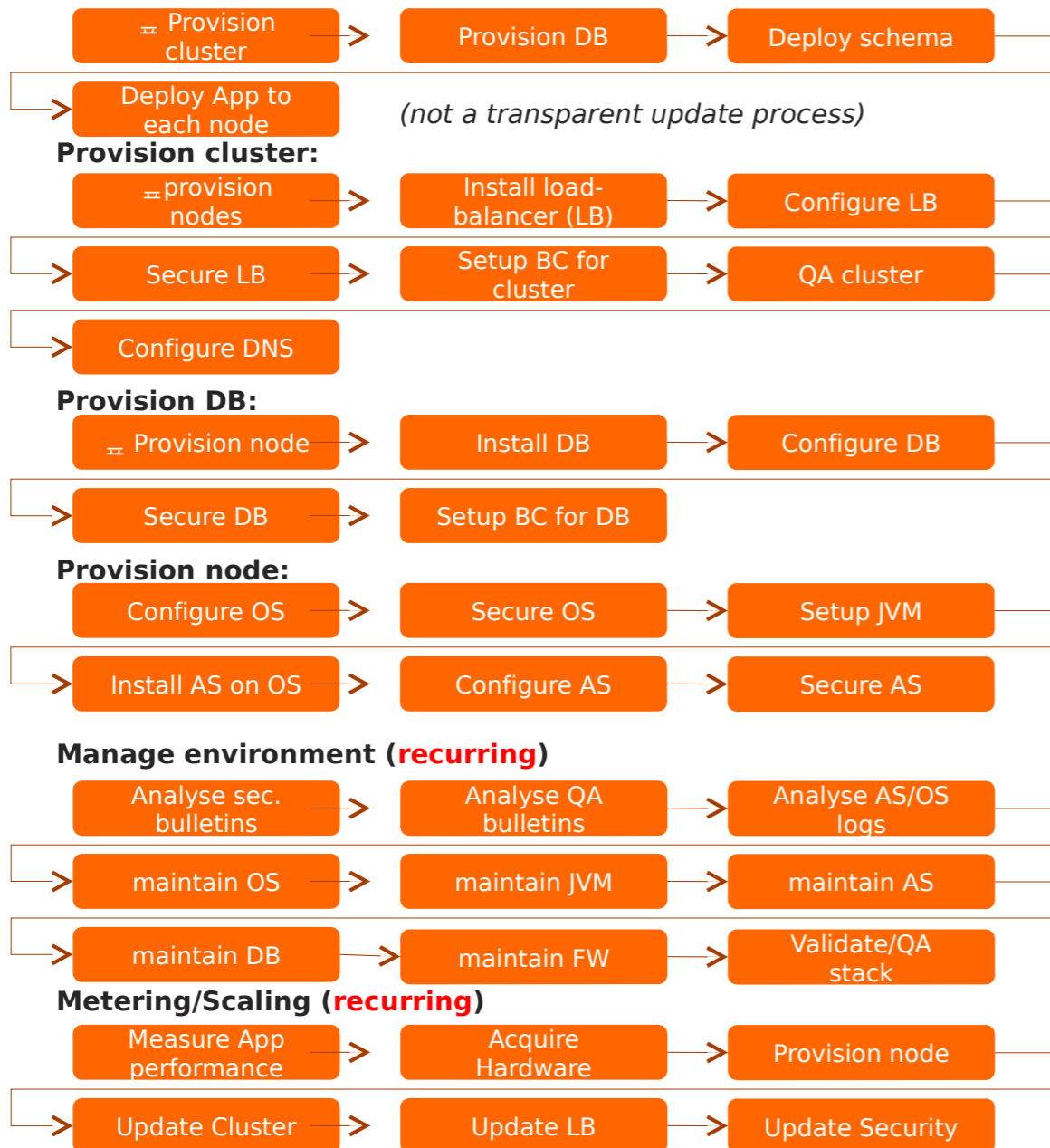


# Why does CloudBees rock?

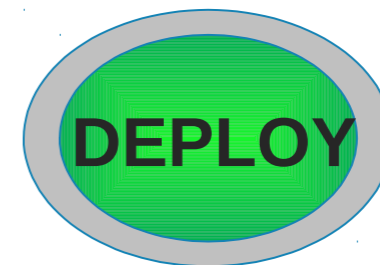
## Deploy app to traditional Java platform



### Deploy App:



## Deploy app to CloudBees

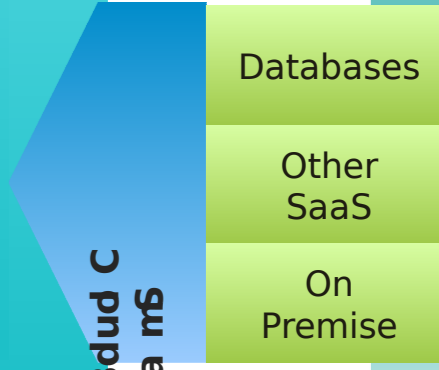
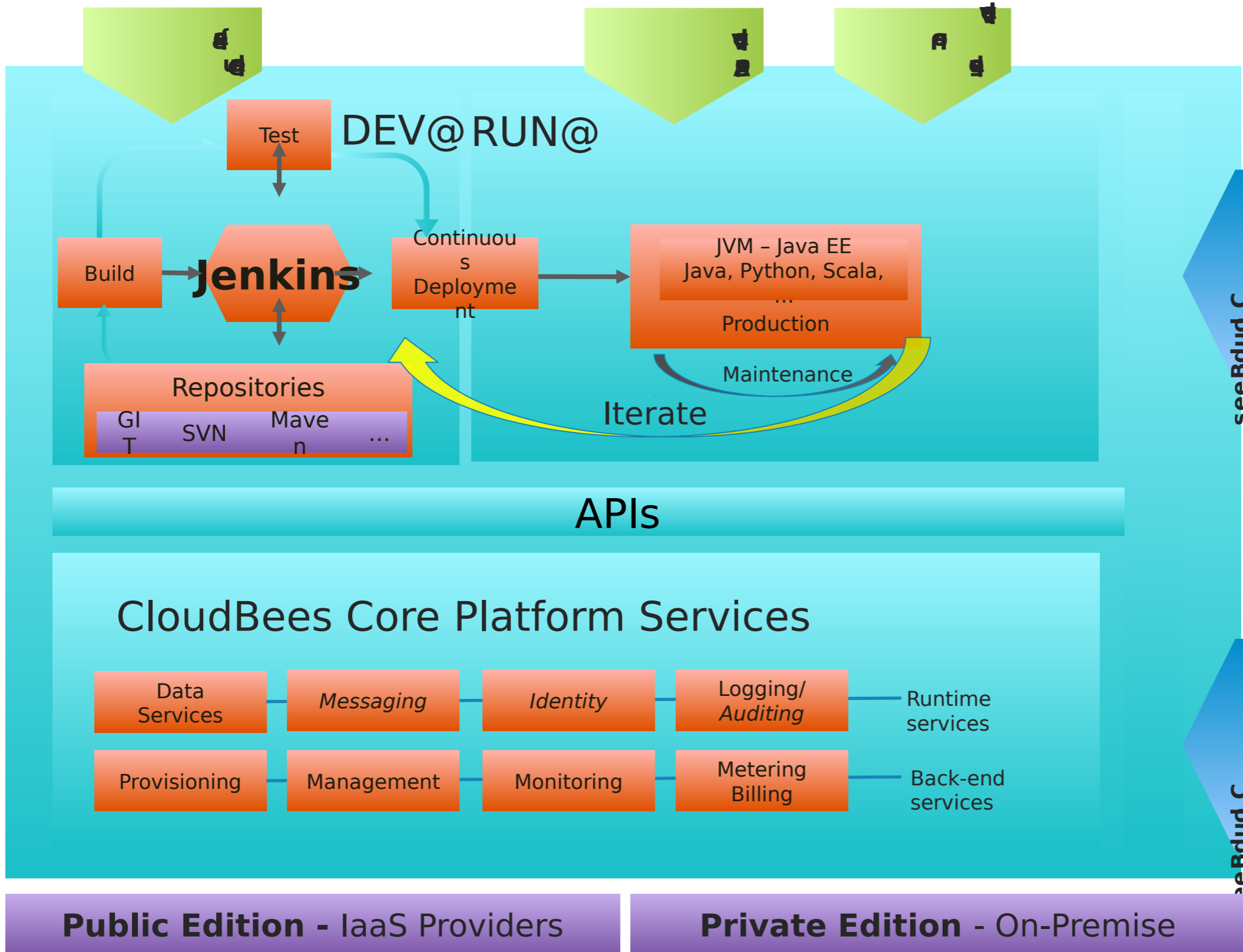


(transparent, multi-version, HA, scalable)

- No need to care about servers, load-balancers, firewalls, backups, etc.
- The environment is constantly managed and monitored
- Scalability happens in realtime
- Integrated failover/HA
- We do « Ops », you do « Dev »



# CloudBees Platform as a Service



# Welcome to CloudBees - Click a service to get started!

[Browse Ecosystem »](#)

## CloudBees Platform



Repositories



Jenkins Builds



Applications







Databases



More Services

## Community Links

-  [Grails](#)
-  [Java EE](#)
-  [Play!](#)
-  [more...](#)

## Downloads

-  [CloudBees Eclipse Tools](#)
-  [CloudBees SDK](#)

## Quick Reference

- [DEV@cloud Wiki](#)
- [GIT/SVN repositories](#)
- [Application monitoring](#)
- [Selenium browser testing](#)
- [RUN@cloud Wiki](#)
- [Maven Integration](#)
- [MySQL databases](#)
- [Sonar code coverage](#)

## Getting Started

- [DEV@cloud - Jenkins as a service](#)
- [RUN@cloud - Deploying Java apps](#)
- [Continuous deployment](#)
- [CloudBees for Eclipse](#)

## Recent Blog Entries

- [Continuous Integration to Continuous Deployment with Jenkins and Deployit](#)
- [Q&A: "Continuous Integration to Continuous Deployment with Jenkins and Deployit"](#)
- [Scripted Integration Tests with Jenkins and CloudBees](#)
- [2012, the Year of the Postmortem?](#)
- [CAMP, Standards, and Relevance](#)
- [Continuous Deployment for Mobile Apps with Jenkins: Back-end Services and Integration Testing](#)
- [Continuous Deployment for Mobile Apps with Jenkins: Xcode Builds with MacOS Slaves](#)

# All of you assets in a click!

The screenshot displays the CloudBees dashboard interface. At the top, there is a navigation bar with the following items: CloudBees.com, Home, ClickStart, Apps, DBs, Builds (selected), Repositories, Services, and Help. The main content area is divided into several sections:

- RUN@cloud Services:** A list of services including:
  - RUN@cloud:** Deploy your Java app on cloud.
  - Cloudbant:** Data Layer as a Service. Store your data, globally. Compatible with PostgreSQL.
  - SendGrid:** A fast and easy way to send emails into your applications.
- DEV@cloud Services:** A list of services including:
  - Sauce Labs:** Manual and Selenium browser testing on the cloud.
  - xWiki:** The next generation of Wiki. It is easy to use and easy to integrate.
- Jenkins Build Jobs:** A grid of 20 build jobs, each with a status icon (yellow, blue, or red), a title, and details on its last build status, health, and date.

Job Name	Status	Health	Last Build Date
cloudbees-security-groups	Yellow	0%	Fri Aug 24 2012
devops-dns-report	Yellow	76%	Fri Aug 24 2012
Internal Maven Repository Reindex	Blue	100%	Fri Aug 24 2012
Nectar Update Center Metadata Generator	Red	0%	Fri Aug 24 2012
GAV-repo mapper	Blue	100%	Fri Aug 24 2012
slave-security-check	Red	0%	Fri Aug 24 2012
CloudBees-repositories-nexus-index-generation	Red	0%	Fri Aug 24 2012
stax-console	Red	0%	Fri Aug 24 2012
bees-cli-plugins	Blue	100%	Fri Aug 24 2012
bees-cli-db-plugin	Blue	100%	Fri Aug 24 2012
bees-cli-config-plugin	Blue	100%	Fri Aug 24 2012
bees-cli-app-plugin	Blue	100%	Fri Aug 24 2012
bees-cli-ant-plugin	Blue	100%	Fri Aug 24 2012
bees-cli	Blue	100%	Fri Aug 24 2012
Sent RUN stats by e-mail	Blue	100%	Fri Aug 24 2012
Services-Platform	Blue	80%	Fri Aug 24 2012
ci-addon-postgresql	Blue	100%	Thu Aug 23 2012
glassfish-jruby2	Blue	100%	Wed Aug 22 2012
nectar-docs	Blue	100%	Fri Aug 24 2012
stax-boss	Blue	100%	Fri Aug 24 2012





Let's move onto some  
“PaaSterns” now...

(mostly “good design” principles)



# Patterns...

Starting...

Portability in the cloud

Tiers vs Tears

Deployment Environments

Dev & Prod Parity

Data & App cloning

Networking

CDNs

Cloud Automation

Continuous Deployment

SOA

...



# Getting started

**Lots of new things to learn!**  
*(delegation of trust, access to legacy data, latency, elasticity, iterativity, etc.)*

**The cloud is not perfect**  
**Might not solve all of your problems today**

**Put a team together**  
**Start small**  
**Non-critical**  
**Learn and Extend**  
**(and enjoy!)**



# Portability



## Follow the right patterns

Avoid lock in to any cloud!

Java is your friend here

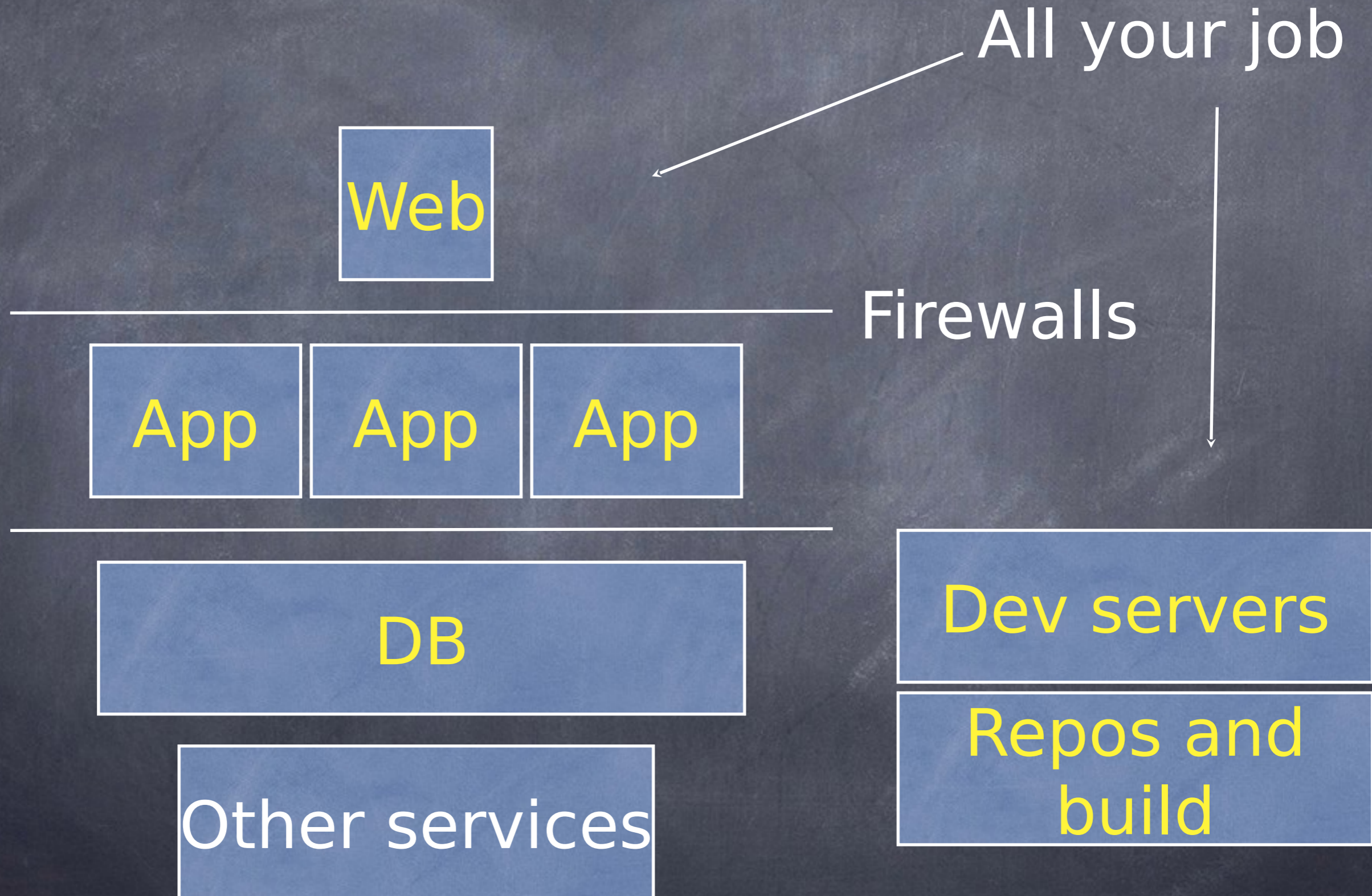
Lock-in mostly with PaaS API (automation)  
not an issue initially

In fact - easier to go “cloud  $\leftrightarrow$  on premises”  
than other way around (cloud keeps you “clean”)

Data is “sticky” so be careful with DBs  
(any DB, cloud or not cloud)

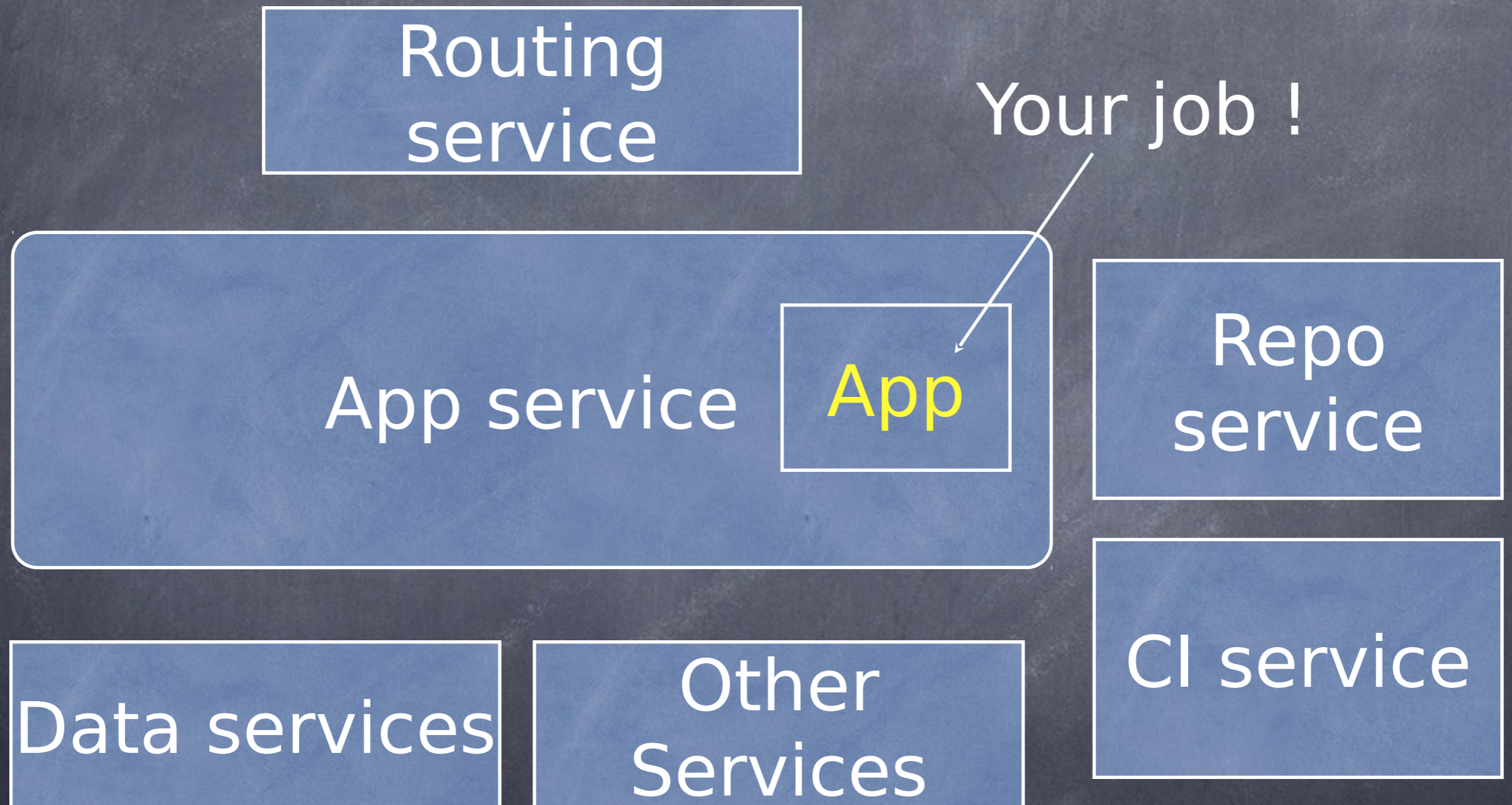


# tiers





# PaaS cloud “tiers”





# problem

All services are someone else's problem  
Note transition from server -> service

*Similar, but nuanced*



# servers



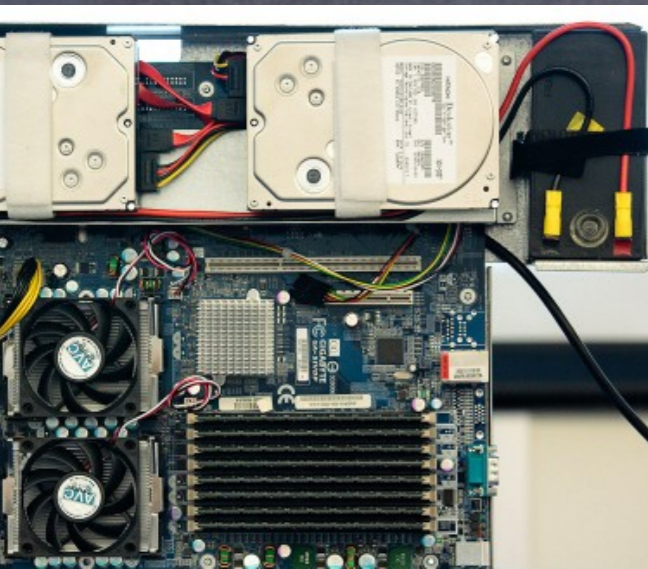
Large, expensive  
Vertically scalable  
Intrinsic redundancy  
CAPEX not OPEX  
outages rare\*\*, but long

**Cost of provisioning: high in time and money**

\*\* Really?



# Common cloud servers



Numerous, cheap  
Horizontally scalable \*\*

Outages more common, but short  
PaaS job is to hide all this !  
Not your problem !

**Cost of provisioning: low in time and  
money**

\*\* Actually a lot of choice nowadays

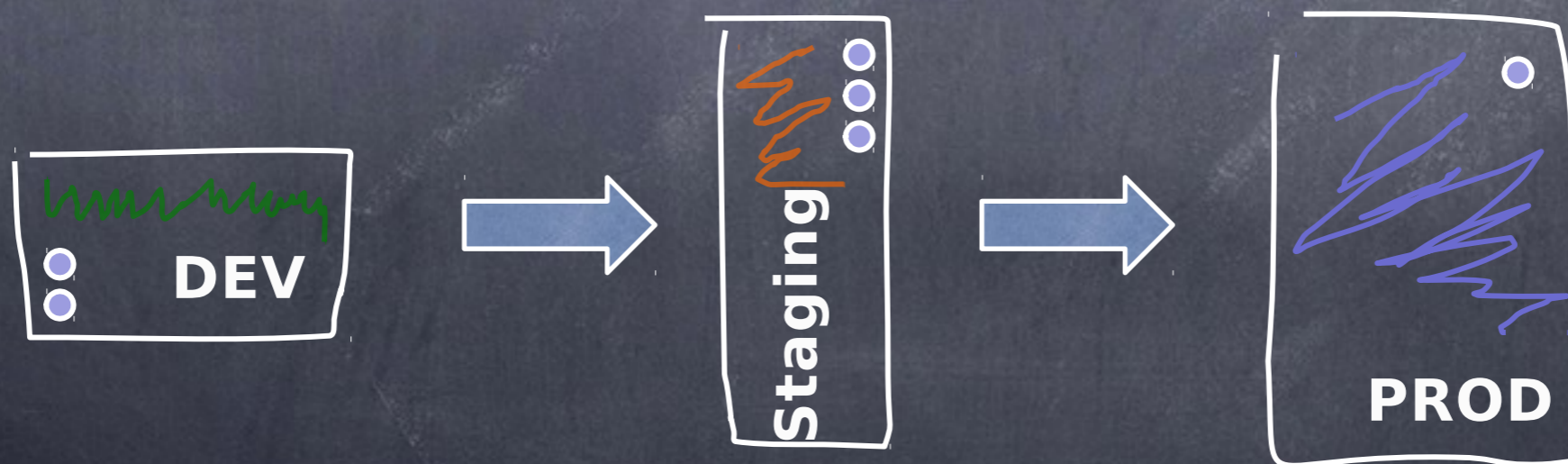


# Environments

Classic: Dev, Test, UAT, Staging, Production

Often anemic hardware for non prod

Work to move between  
Hardware/software not consistent





# Dev and Prod parity

Homogenous environments

Dev and prod “identical”  
differ in name only

No “server crunch” - so why compromise

No “surprises” when promoting

Only pay for what you use:

hi-fidelity environments - start/stop as needed



**THAT POINT**

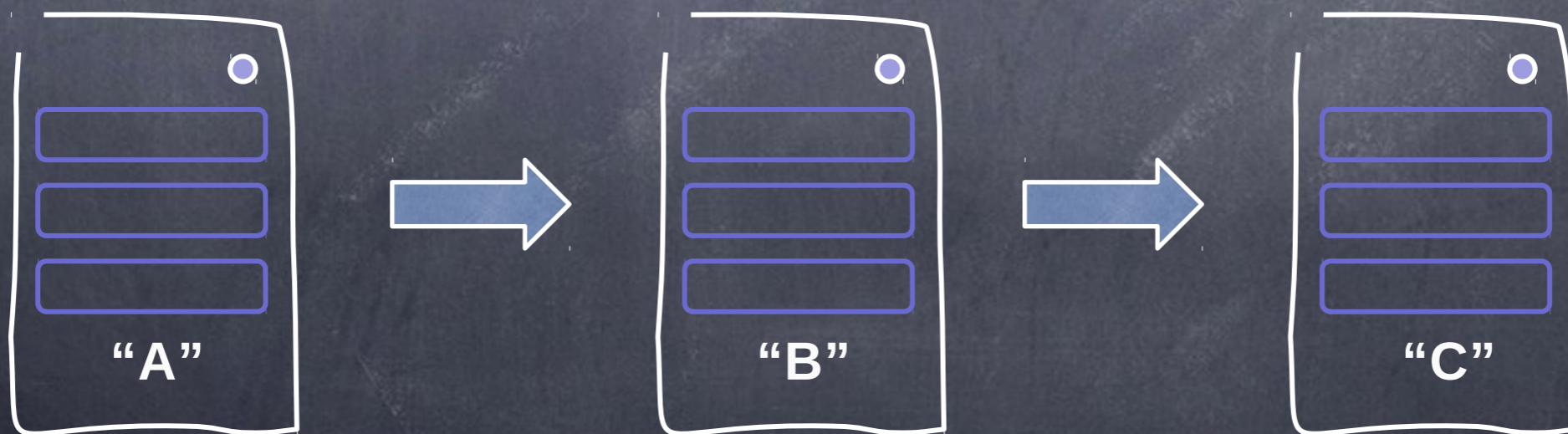


**BEARS REPEATING**



# Dev and Prod parity

Dev and prod  
differ in name only





# One App Many Deploys

Parameterize via Environment variables  
When build “promoted” - same bits as tested !!

“common sense” ??

*“Config varies substantially across deploys,  
code does not” \*\**

\*\* (<http://www.12factor.net/>)



### Applications

[View All](#)

<b>Opzilla</b> Status: active <a href="#">View</a>	<b>Run</b> Status: active <a href="#">View</a>
<b>API</b> Status: active <a href="#">View</a>	<b>run-staging</b> Status: active <a href="#">View</a>
<b>providers</b> Status: active <a href="#">View</a>	<b>mytestapp</b> Status: hibernate <a href="#">View</a>
<b>nectar-license-generator</b> Status: active <a href="#">View</a>	<b>security-test</b> Status: hibernate <a href="#">View</a>
<b>permission-test</b> Status: hibernate <a href="#">View</a>	<b>gc-executor</b> Status: active <a href="#">View</a>
<b>xwiki</b> Status: active <a href="#">View</a>	<b>services-platform</b> Status: active <a href="#">View</a>
<b>providers-dev</b> Status: active <a href="#">View</a>	<b>apptester</b> Status: hibernate <a href="#">View</a>
<b>services-dev</b> Status: active <a href="#">View</a>	<b>newrelic-dev</b> Status: hibernate <a href="#">View</a>
<b>services-ui</b> Status: active <a href="#">View</a>	<b>jenkins-proxy-dev</b> Status: active <a href="#">View</a>
<b>newrelic-staging</b> Status: hibernate <a href="#">View</a>	<b>providers-staging</b> Status: stopped <a href="#">View</a>
<b>newrelic</b> Status: hibernate <a href="#">View</a>	<b>services-staging</b> Status: hibernate <a href="#">View</a>

## Click a service to get started!



Jenkins Builds



Applications

### Quick Reference

- DEV@cloud Wiki
- GIT/SVN repositories
- Application monitoring
- Selenium browser testing
- RUN@cloud Wiki
- Maven Integration
- MySQL databases
- Sonar code coverage

### Getting Started

- DEV@cloud - Jenkins as a service
- RUN@cloud - Deploying Java apps
- Continuous deployment
- CloudBees for Eclipse



# One App Many Deploys

You can have multiple concurrent versions  
...as many as you need

Differ in URL

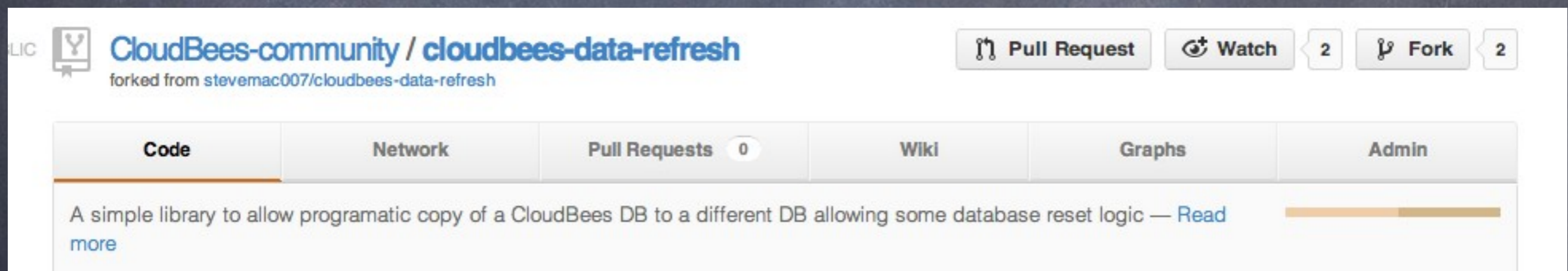
Slowly transition

Enables A/B testing



# Data cloning

For “modest” volumes of data  
Clone into each environment:



The screenshot shows a GitHub repository page for 'CloudBees-community / cloudbees-data-refresh'. The repository is forked from 'stevemac007/cloudbees-data-refresh'. The page features a navigation bar with tabs for 'Code', 'Network', 'Pull Requests' (0), 'Wiki', 'Graphs', and 'Admin'. The 'Code' tab is selected. Below the navigation bar, there is a description: 'A simple library to allow programatic copy of a CloudBees DB to a different DB allowing some database reset logic — [Read more](#)'. The page also includes action buttons for 'Pull Request', 'Watch' (2), and 'Fork' (2).

CloudBees-community / **cloudbees-data-refresh**  
forked from stevemac007/cloudbees-data-refresh

Pull Request Watch 2 Fork 2

Code Network Pull Requests 0 Wiki Graphs Admin

A simple library to allow programatic copy of a CloudBees DB to a different DB allowing some database reset logic — [Read more](#)



# Networking

Classic: layered, partitioned, ops controlled  
firewalls

Cloud: flatter networks, programmatic firewalls  
Firewalls also happen at server level

**A PaaS manages this for you**



# Hybrid Networking

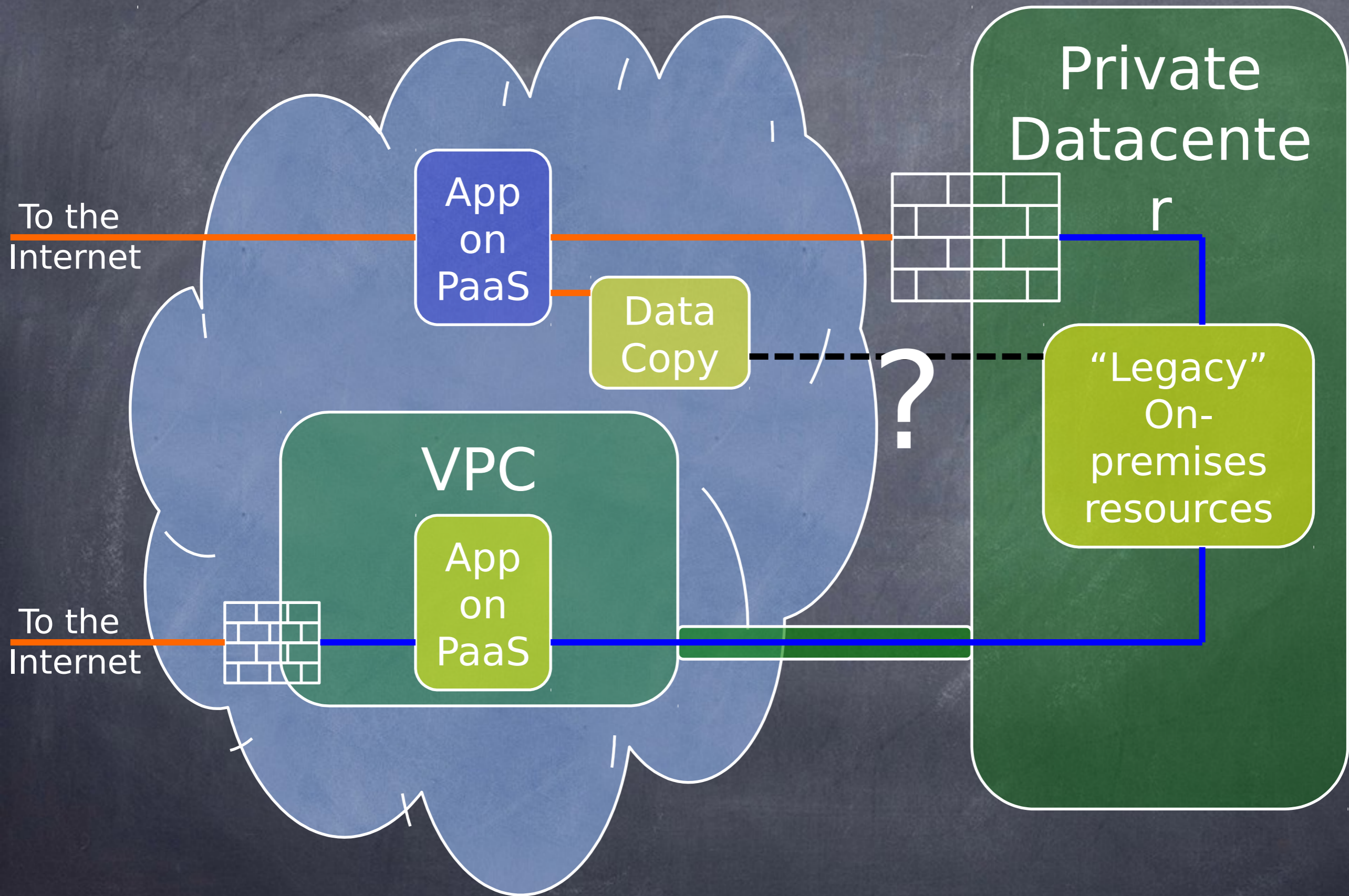
Need: cloud apps, but some “behind firewall data”

Very frequent within companies using Java

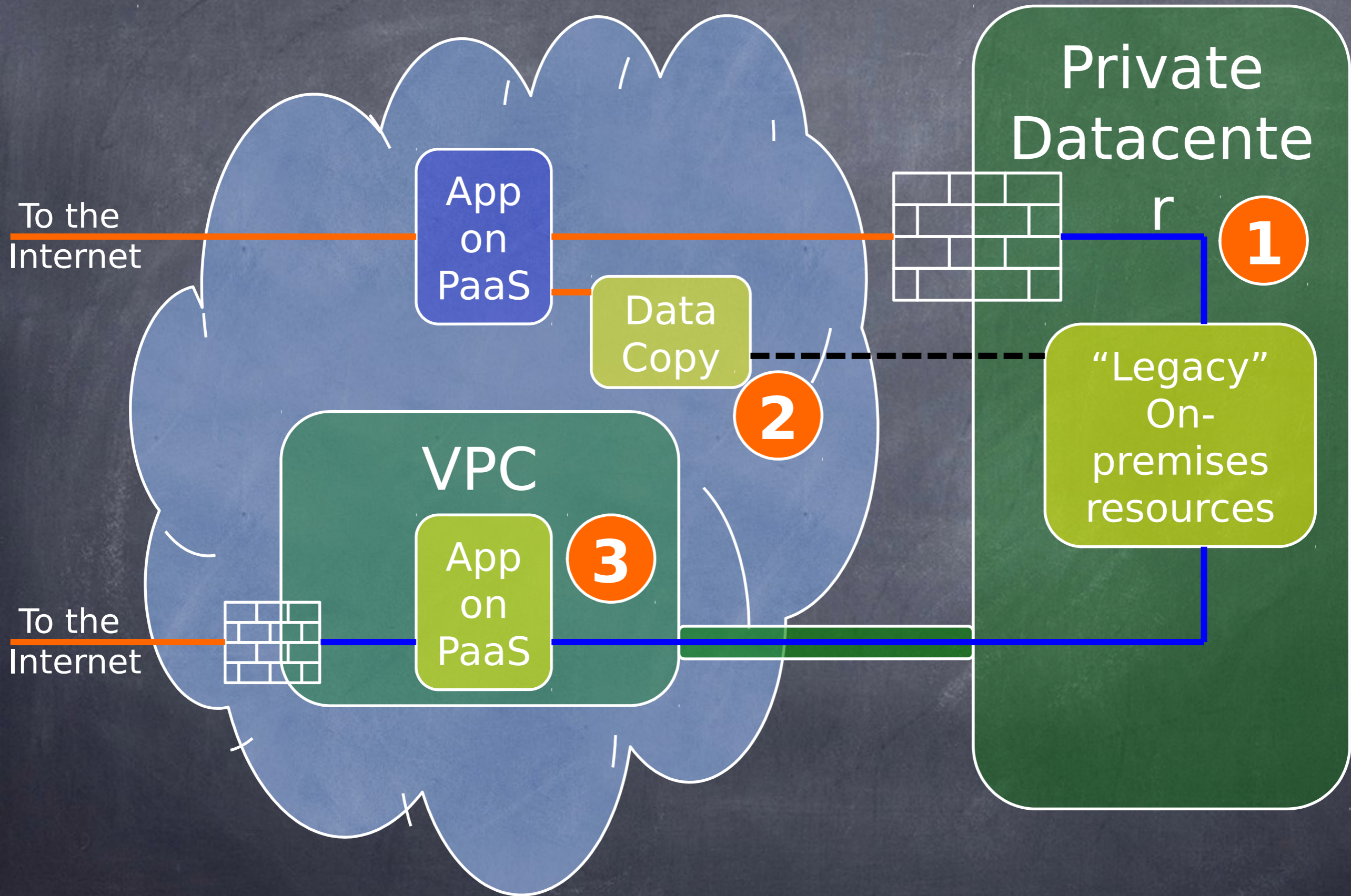
Typical Solutions:

- Replicate data
- VPN
- Firewall rules on private network (fixed IPs) that expose select services (optional tunneling)
- VPC (growing usage!)











# Use A CDN

CDN=Content Delivery Network

Fight latency

Move static/cacheable content to CDN edge nodes

Examples: Amazon Cloudfront, Cachefly

Now easily accessible by “anybody”





Shift to cloud -> shift to remote servers  
(sometimes)

**Great, let's take it to our advantage!**

Some users latency sensitive  
Use a CDN with geo-aware DNS (all of them)

Bulk of "data" delivered can be cached  
in some cases



# cache

Use hash based asset paths  
(framework + build tools help)

No expiry caching

==

Maximum chance of cache hit and low latency  
Massively improve experience of modern apps



# Latency Friendly GUIs

Web interfaces: lightweight js + background loading (eg backbone.js style)

Mobile apps: naturally latency friendly

Classic request/response web apps are latency sensitive.

Some latency inevitable, pick frameworks wisely



# APIs

All types of clouds have apis

Make use of them from builds:

Deploy time (easy roll back !)

Development and testing (multiple environments)

Cost saving: only run when needed

```
bees app:stop  
bees app:hibernate **
```

\*\* will wake when testers need it



# Deploy early and often

When it is easy to roll back, it is easy to deploy

Deliver smaller changes, more often

(data breaking changes, you can take your time !)

Risk in deployment is proportional to the time since last deployment

$$\text{Risk} = X * (\text{Now} - \text{LastDeployment})$$



# Deployment

CI == continuous integration

Continuous deployment takes the next step

Tests + Automation == Lower risk

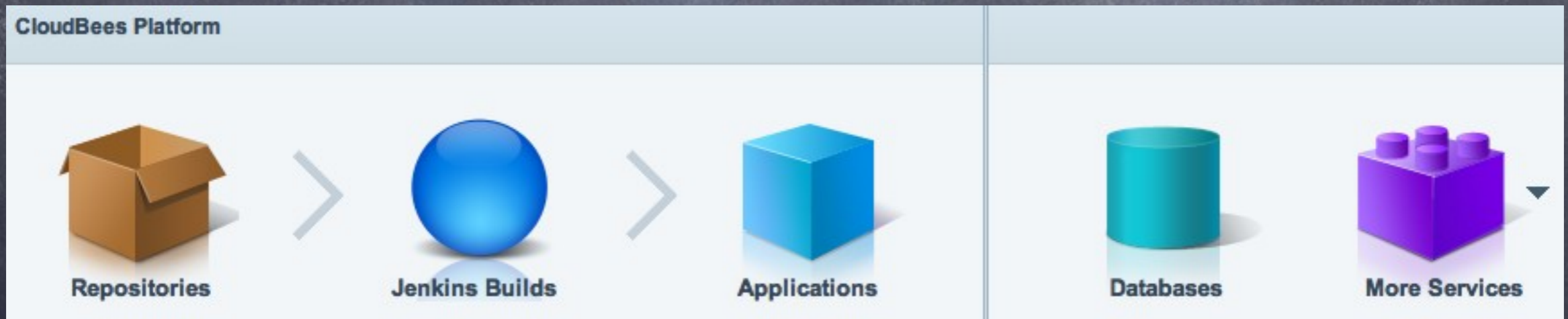
Deploy to “an environment” continuously  
(may not be production !)

Need a “build/test/dev” automation workflow  
server (eg Jenkins)



# Deployment

Change Deployment rules



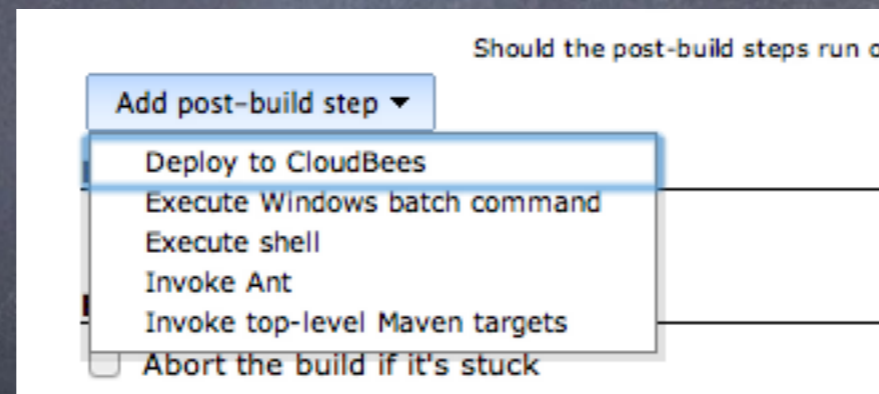
If only someone had built this !



# Build Workflow



# Jenkins



> 400 plugins



# Dog Feeding

Chrome File Edit View History Bookmarks Window Help

Cacoo - developer.cloudbees.com Final ClickStart Created D... Extensibility Roadmap v0... Click Start: Phase 1 Release Promoted Builds Plugin - CloudBees - GrandCentral

https://grandcentral.cloudbees.com

Apps DBs **Builds** Repositories Services Help Michael, w

### Jenkins Build Jobs [View All](#)

<b>GAV-repo mapper</b> Last: Stable Health: 80% Date: Sun Aug 05 2012	<b>devops-dns-report</b> Last: Stable Health: 79% Date: Sun Aug 05 2012
<b>slave-security-check</b> Last: Stable Health: 20% Date: Sun Aug 05 2012	<b>CloudBees-repositories-nexus-index-generation</b> Last: Stable Health: 20%
<b>stax-console</b> Last: Stable Health: 0% Date: Sun Aug 05 2012	<b>bees-cli-plugins</b> Last: Stable Health: 80% Date: Sun Aug 05 2012
<b>bees-cli-db-plugin</b> Last: Stable Health: 80% Date: Sun Aug 05 2012	<b>bees-cli-config-plugin</b> Last: Stable Health: 80% Date: Sun Aug 05 2012
<b>bees-cli-app-plugin</b> Last: Stable Health: 80% Date: Sun Aug 05 2012	<b>bees-cli-ant-plugin</b> Last: Stable Health: 80% Date: Sun Aug 05 2012
<b>bees-cli</b> Last: Stable Health: 80% Date: Sun Aug 05 2012	<b>Sent RUN stats by e-mail</b> Last: Stable Health: 100% Date: Sun Aug 05 2012
<b>Internal Maven Repository Reindex</b> Last: Stable Health: 100%	<b>Nectar Update Center Metadata Generator</b> Last: Stable Health: 0%
<b>build-image</b> Last: Stable Health: 80% Date: Sun Aug 05 2012	<b>Cloudbees gem</b> Last: Stable Health: 100% Date: Sun Aug 05 2012
<b>ci-addon-postgresql</b> Last: Stable Health: 100% Date: Sun Aug 05 2012	<b>nectar-docs</b> Last: Stable Health: 0% Date: Fri Aug 03 2012
<b>gcc-check</b>	<b>NewRelic provider</b>

### Service to get started!

[Browse Ecosystem](#)

Applications Databases More Services

### Recent Blog Entries

- Using the CloudBees Maven Plug-in with Eclipse
- Getting Started with CloudBees and Eclipse
- Continuous Information Vol. 3: Jenkins Growth Explodes
- Denial of Service Attack on DNS Event
- Conclusion: The Cloud is IT's Most Important Tectonic Shift
- Jenkins User Conference Israel
- Cloud Beers this Friday in Our West Coast Office

- RUN@cloud Wiki
- Maven Integration
- MySQL databases
- Sonar code coverage



# SOA

Same Old Architecture?

Got a bad name due to WS-\* and Enterprise Vendors

Is a very useful approach

In use by all modern web companies

Just not WS-\* !

Build apps as composeable, re-useable services

On paas cloud: http + rest



# SOA

Build multi tiered applications  
Separate out tiers  
Separate out background vs online  
GUI vs backoffice functionality



All are just apps on a PaaS



# SOA

Services are still just apps !  
They are consumed by other apps

Different apps/services can have different  
release cadences

Use CI automation to help keep things in sync

HTTP interfaces for your own use (health,  
business stats) - not just user facing



# Use Natural Sharding

Many apps have a “natural” way to split up data if needed

(users, accounts etc)

As a way to scale out, split apps into multiple instances

Databases in to shards

Cheap and easy to clone environments if needed.



# REST-api first design

Increasingly common to build REST apis first

Separate apps for Web, Mobile clients

Server side becomes less GUI centric

Shift in frameworks - choose wisely

Services, like databases, live on, are reused,  
and composed into new services



# Prefer Stateless

Easy to say - hard to do

Stateless apps delegate state to:  
Databases, cookies, cache servers, external  
services

Places state can “hide”:

User sessions, caches, frameworks



# Prefer Stateless

Session data:

Use session service (cloudbees service) –  
cluster friendly (and transparent)

cookies (modulo security)  
database



# Maybe Stateful

Stateful is actually ok:

A cache that can be replaced  
eg Lift Web Framework  
not required to be replicated

In this case - use “sticky sessions” - session affinity to a server (just a flag on cloudbees)

Scale up vs out

State is short lived



# Prefer Stateless

Why stateless?

Allows fine grained scaling

Allows scaling out (as well as up +1 dimension)

High Availability

Zero downtime deploys

Client side UI state (eg mobile, HTML5) makes this easier than ever

*Why does a server care what tab a user has selected?*



# scaling

Smaller state-light apps  
Scale out quickly

Autoscale  
(track stats, ensure SLA being met  
automatically)  
(no work for you !)

Only pay for what you need



# Use Add-ons

Why host your own email service  
Why host your own monitoring/analysis

Some PaaS providers

(CloudBees, Heroku, etc.)

Offer a rich ecosystem to choose from  
(send grid, new relic, for example)



# Use Add-ons

The screenshot shows the CloudBees website interface. The browser window displays the URL <https://grandcentral.cloudbees.com>. The navigation menu includes **Apps**, **DBs**, **Builds**, **Repositories**, **Services**, and **Help**. The **Services** section is expanded, showing two main categories: **RUN@cloud Services** and **DEV@cloud Services**.

**RUN@cloud Services** (View All):

- RUN@cloud**: Deploy your Java applications in the cloud.
- Database**: Shared and dedicated MySQL databases.
- New Relic**: Real-time web monitoring and analytics, delivered as a service.
- Cloudant**: Data Layer as a Service - Scale your data, globally. CouchDB-compatible.
- MongoHQ**: Fast and dependable MongoDB in the cloud.
- Papertrail**: Hosted log management for apps, servers, and cloud services.
- SendGrid**: A fast and easy way to integrate emails into your apps.
- Websolr**: Index and search your content with the enterprise-grade Apache Solr search engine.
- AppDynamics**: Application performance management for modern application architectures.

**DEV@cloud Services** (View All):

- Sauce Labs**: Manual and Selenium-driven cross-browser testing on the cloud.
- JFrog**: Your Binary Repository in the cloud.
- Sonar**: Continuously inspect your source code.
- xWiki**: The next generation cloud wiki that is easy to use and easy to organize.
- Codesion**: Cloud hosting platform for SCM (Subversion & Git), development & Agile ALM tools.

On the right side of the page, there is a "get started!" section with icons for **Applications** and **Databases**. Below this is a "Recent Blog E" section with a list of articles:

- RUN@cloud Wiki
- Maven Integration
- MySQL databases
- Sonar code coverage
- Using the CloudBees with Eclipse
- Getting Started with Eclipse
- Continuous Inform Jenkins Growth E
- Denial of Service Event
- Conclusion: The Important Tectoni
- Jenkins User Cor
- Cloud Beers this Coast Office

At the bottom of the page, there are links for **CloudBees SDK** and **CloudBees for Eclipse**.



# Use Add-ons

This is the “re-use” we were “promised” many years ago

Finally delivered  
Compose services together



# Use Add-ons

Worried about lock in?

Data services are built on open source, you can always export your data and host it yourself

Other services are less “latency sensitive” (eg newrelic is popular for on-prem systems too)



# Skunkworks

“It is easier to ask forgiveness than it is to ask permission”  
-- Grace Hopper

Cost of trying/building is cheap, so you can  
“just build it”

Find a part of your application that could be  
split off into a cloud service on a paas

“ask forgiveness”



# Example: Lose it!

- Lose it!
  - Mobile application
  - >10 millions customers
  - >20,000 transactions per minute, at peak time
- And... only 4 employees and ...2 developers!
  - No IT, no servers, no DevOps
  - Complete focus on SOFTWARE DEVELOPMENT
  - Anything else is handled by CloudBees (AS + DB)
- Unmatched productivity level!
  - This is possible TODAY
  - Will you wait for your competitors to shoot first?



# Weather Forecast

By 2020, IT will either be about...

SaaS

or

PaaS

↓  
*“Can I find a pre-built solution that matches my needs?”*

☐ SaaS

↓  
*“If not, then I need to build a custom application”*

☐ PaaS



# Join the right side of the Force

You can't really get what is a PaaS until you try  
it

You need to feel this "ah ah!!" moment

Register on CloudBees.com (free and complete)

- *(Jenkins as a Service, Git, SVN, Maven, MySQL as a Service, Eclipse Plugin, etc.)*

[www.cloudbees.com/signup](http://www.cloudbees.com/signup)



# Get started in a click

The screenshot shows the ClickStart wizard interface for creating a Java EE 6 Web Profile application. The window title is "ClickStart" and the main heading is "Java EE 6 Web Profile". Below the heading is a tab labeled "Application Components". There are four component cards arranged in a 2x2 grid:

- Source repository**: Management URL <https://sgh.forge.cloudbees.com>, URL `ssh://git@git.cloudbees.com/sgh/mytest.git`
- Jenkins build**: Management URL <https://sgh.ci.cloudbees.com/job/mytest/>
- Web Application mytest**: Management URL <https://run.cloudbees.com/a/sgh#app-manage/development:sgh/mytest>, URL `http://mytest.sgh.cloudbees.net`
- Database mytest--1**: Management URL <https://run.cloudbees.com/a/sgh#db-manage:mytest-1>

At the bottom of the wizard, there are two buttons: "Cancel" and "Create App >".

i.e. you have no excuse!



Thank you



**CloudBees<sup>®</sup>**

@SachaLabourey