

DevOps in action

Yuta Imai

Solutions Architect, Amazon Data Services Japan, K.K.

今日のテーマ

- いままで見てきたお客様や海外事例のなかで「これはおもしろいなー、すごいなー」と感じたDevOpsっぽい話を紹介していきます。

Agenda: Case Studies!

- NetFlix 1/2 : Service Oriented Architecture and Organization
- A Global Gaming : Application integrated deploy
- MMO RPG : Application integrated deploy
- NetFlix 2/2 : Monitor everything

Case study: Netflix 1/2

Service Oriented Architecture and Organization

NetflixMeetup in Kyoto

The screenshot shows a web browser displaying the Connpass event page for 'NetflixMeetup in Kyoto'. The URL in the address bar is connpass.com/event/9837/. The page features the Connpass logo, a search bar, and navigation links for 'カテゴリ一覧' (Category List) and 'Recent Events'. Below the header, there are social media sharing buttons for Facebook (10), Google+ (0), Like (40), and Tweet (64). The event title 'NetflixMeetup in Kyoto' is displayed with a date selector set to '12月 1' (December 1st). The main content area features the word 'NETFLIX' in large, bold, red capital letters. Below this, the hashtag '#netflix_kyoto' is shown. At the bottom, there is a registration information section with a button labeled '参加枠' (Participation Slot) and the text 'Free'. A counter indicates 'FCFS 13/15'.

connpass.com/event/9837/

connpass
produced by Be PROUD

Search カテゴリ一覧 Recent Events

10 0 40 64

12月 1 NetflixMeetup in Kyoto

NETFLIX

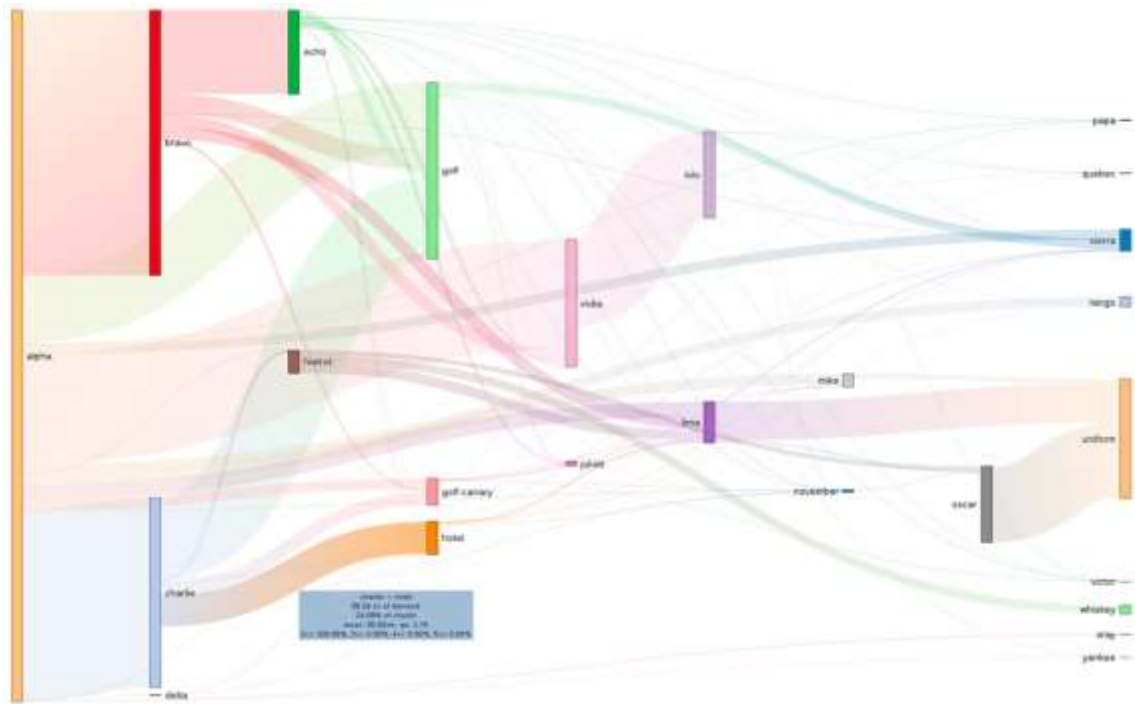
Hashtag : #netflix_kyoto

Registration info 参加枠 Free FCFS 13/15

<http://connpass.com/event/9837/>

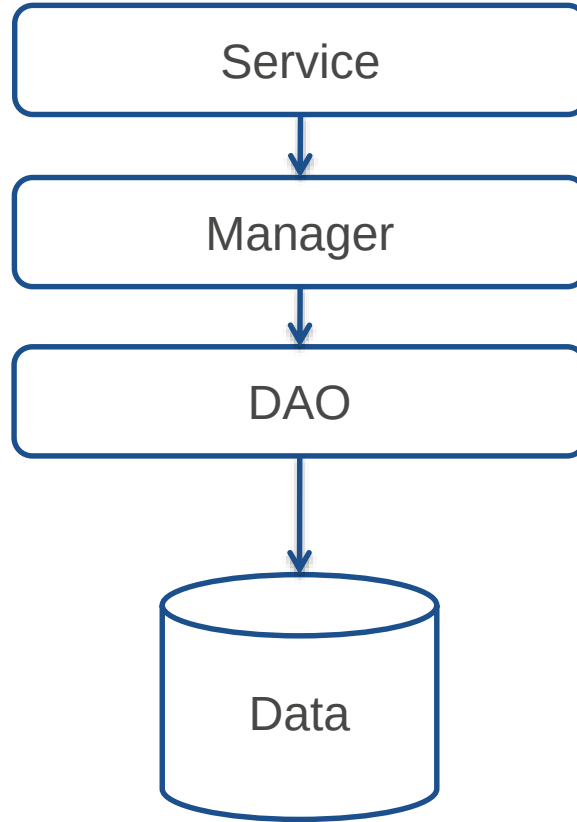
Netflix's way

- Service Oriented Architecture
- Appropriate delegation
- No dependency among teams

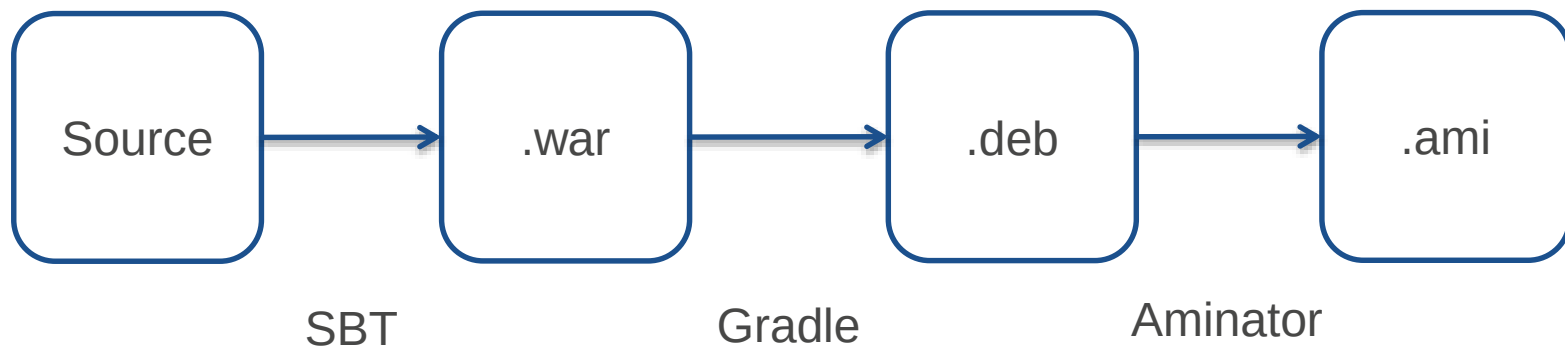


<http://techblog.netflix.com/2015/02/a-microscope-on-microservices.html>

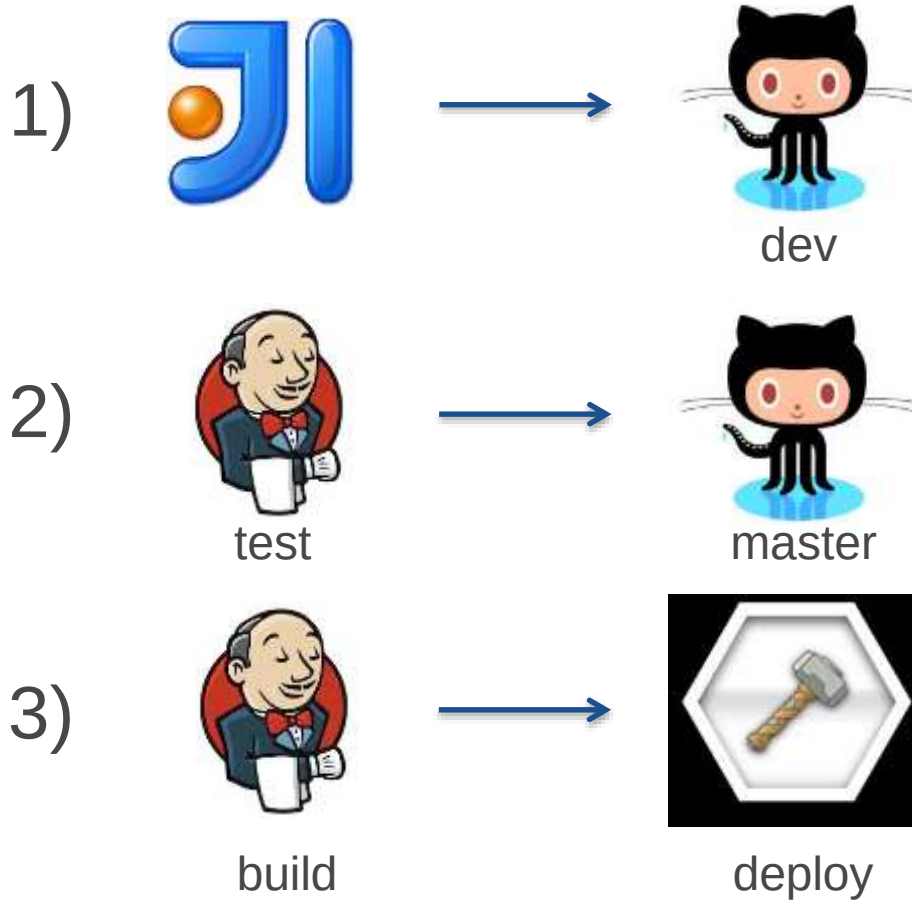
Components in each Services



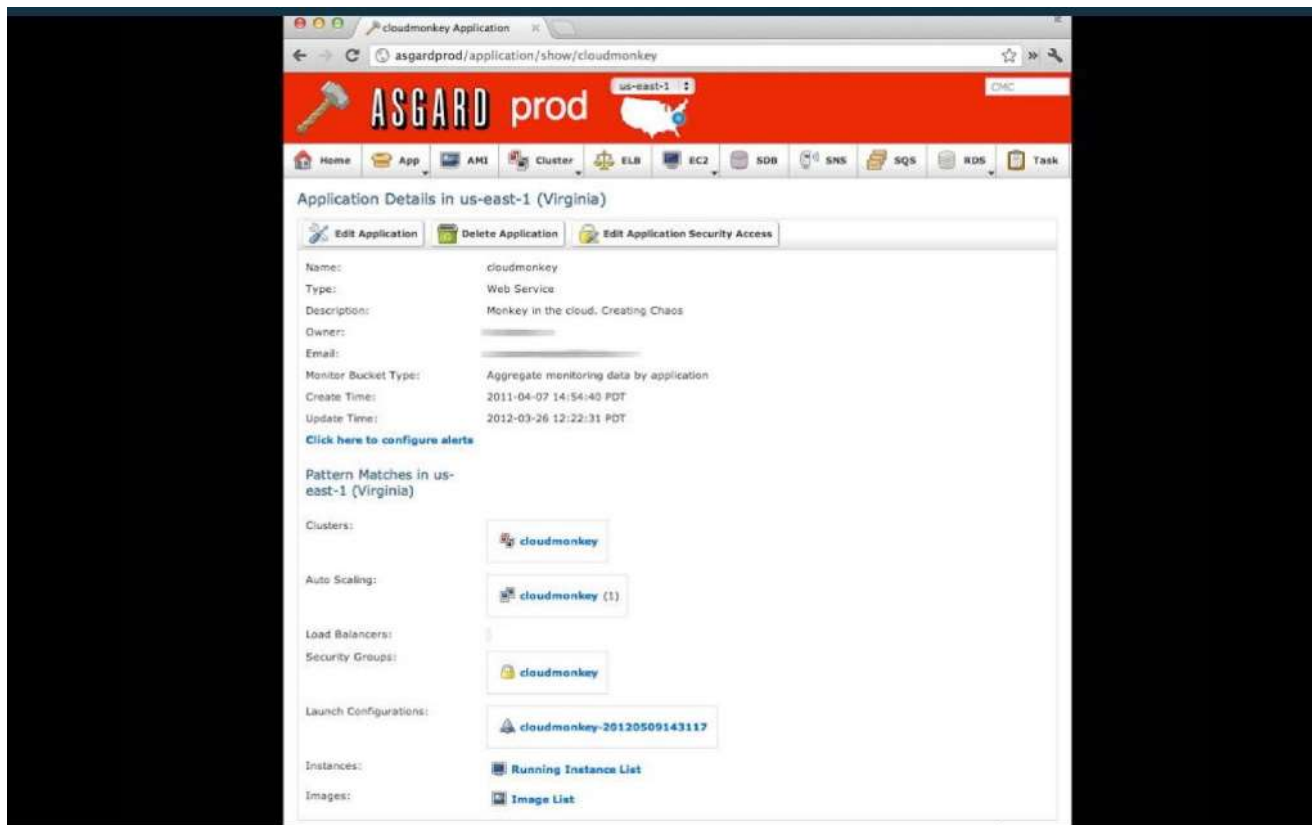
Build



Deploy



Deploy with ASGARD



The screenshot displays the ASGARD prod web interface. The browser address bar shows the URL `asgardprod/application/show/cloudmonkey`. The interface features a red header with the ASGARD logo and a navigation bar with icons for Home, App, AMI, Cluster, ELB, EC2, SDB, SNS, SQS, RDS, and Task. The main content area is titled "Application Details in us-east-1 (Virginia)" and includes tabs for "Edit Application", "Delete Application", and "Edit Application Security Access".

Application Details:

- Name: cloudmonkey
- Type: Web Service
- Description: Monkey in the cloud. Creating Chaos
- Owner: [redacted]
- Email: [redacted]
- Monitor Bucket Type: Aggregate monitoring data by application
- Create Time: 2011-04-07 14:54:40 PDT
- Update Time: 2012-03-26 12:22:31 PDT

[Click here to configure alerts](#)

Pattern Matches in us-east-1 (Virginia)

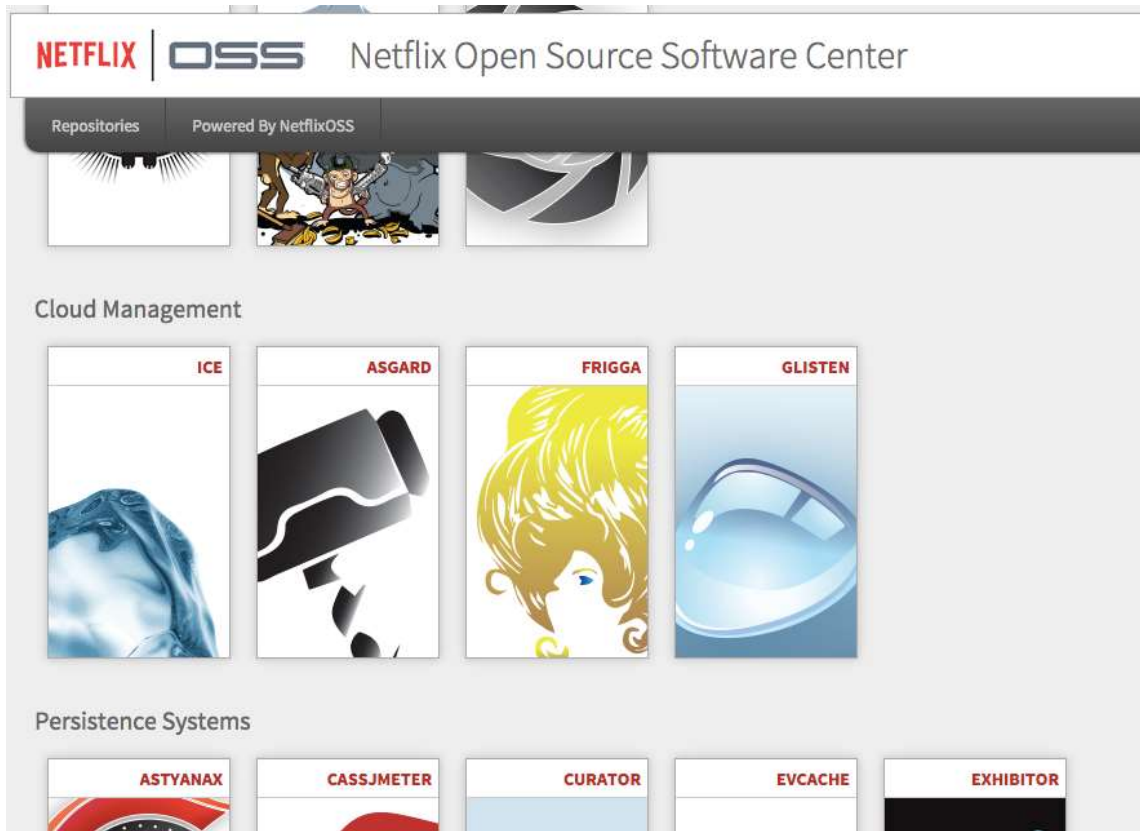
- Clusters: cloudmonkey
- Auto Scaling: cloudmonkey (1)
- Load Balancers: [redacted]
- Security Groups: cloudmonkey
- Launch Configurations: cloudmonkey-20120509143117
- Instances: Running Instance List
- Images: Image List

Deploy with ASGARD

The screenshot shows the AWS Management Console for the ASG 'obiwan-v063'. The console displays the group's configuration, including the AMI ID 'ami-0f8a321f' and the instance type 'm1.large'. The 'Create Next Group' panel is visible on the right, showing the configuration for the new group 'obiwan-v065'. The 'obiwan-v063' group has 9 instances, all in the 'OUT_OF_SERVICE' state. The 'Create Next Group' panel shows the configuration for the new group, including the AMI ID 'ami-0f8a321f' and the instance type 'm1.large'. Red annotations highlight the 'obiwan-v063' group and the 'Create Next Group' panel, with text indicating that traffic is still on the old version and that the new version is being created.

More Netflix OSS..

<http://netflix.github.io/>



You can try with ready made docker containers.

NETFLIX | OSS



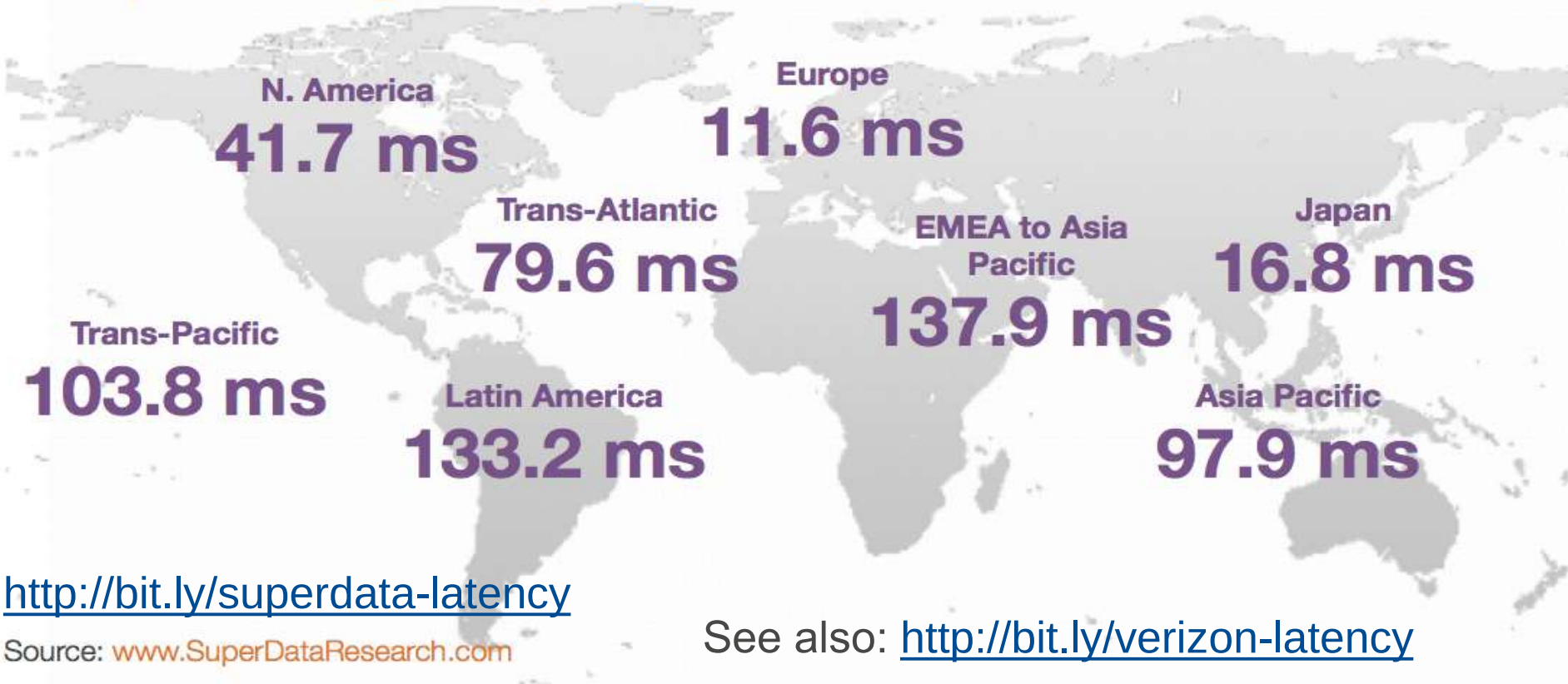
zeroto**docker**

Case study: A Global Gaming

Application integrated deploy

Domestic latency in selected regions:

"higher latency decreases game response time, user performance and perceived game quality."

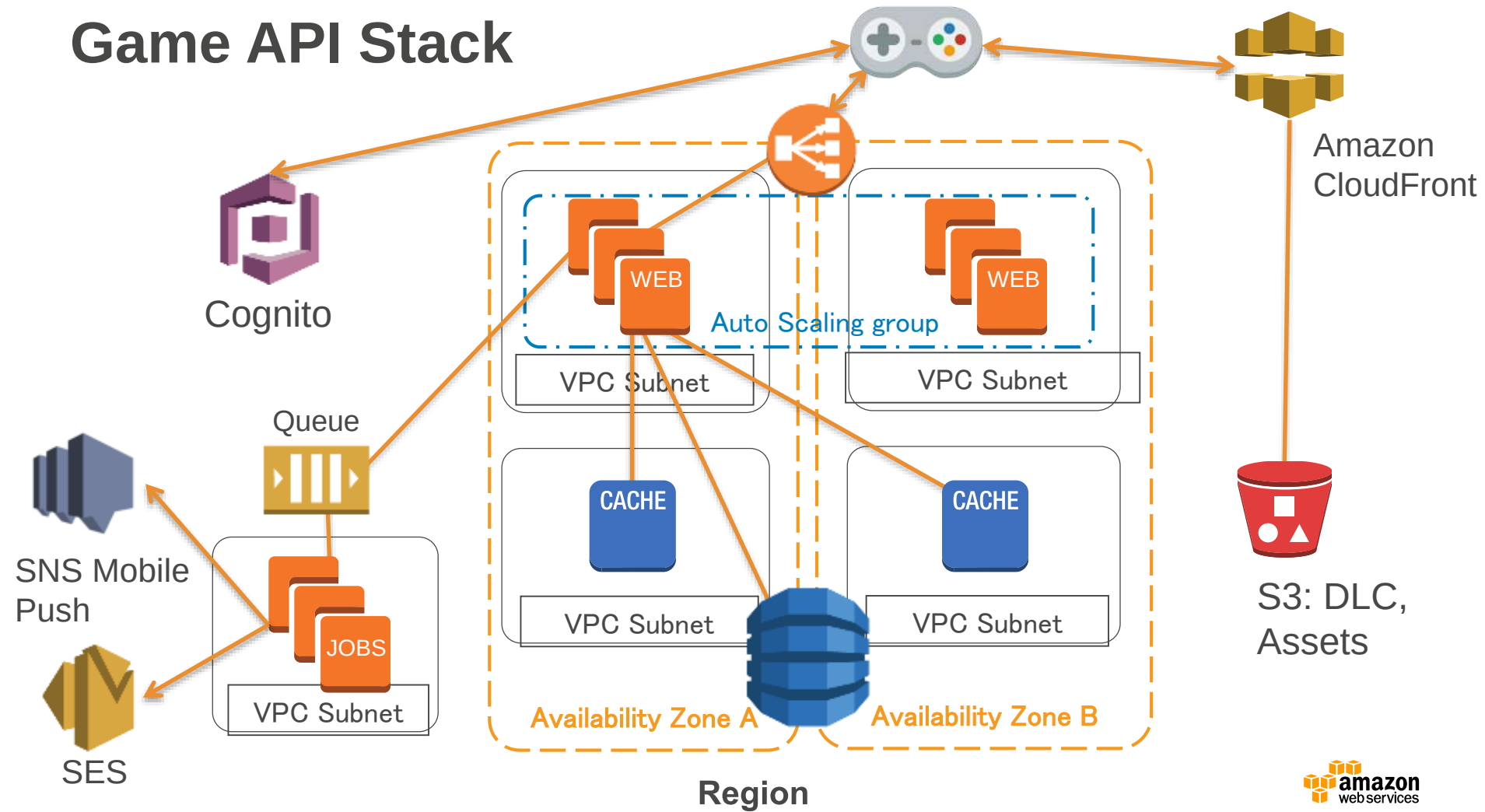


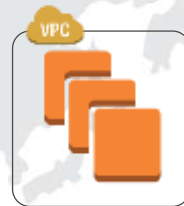
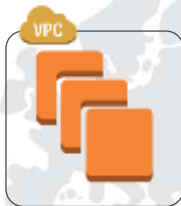
<http://bit.ly/superdata-latency>

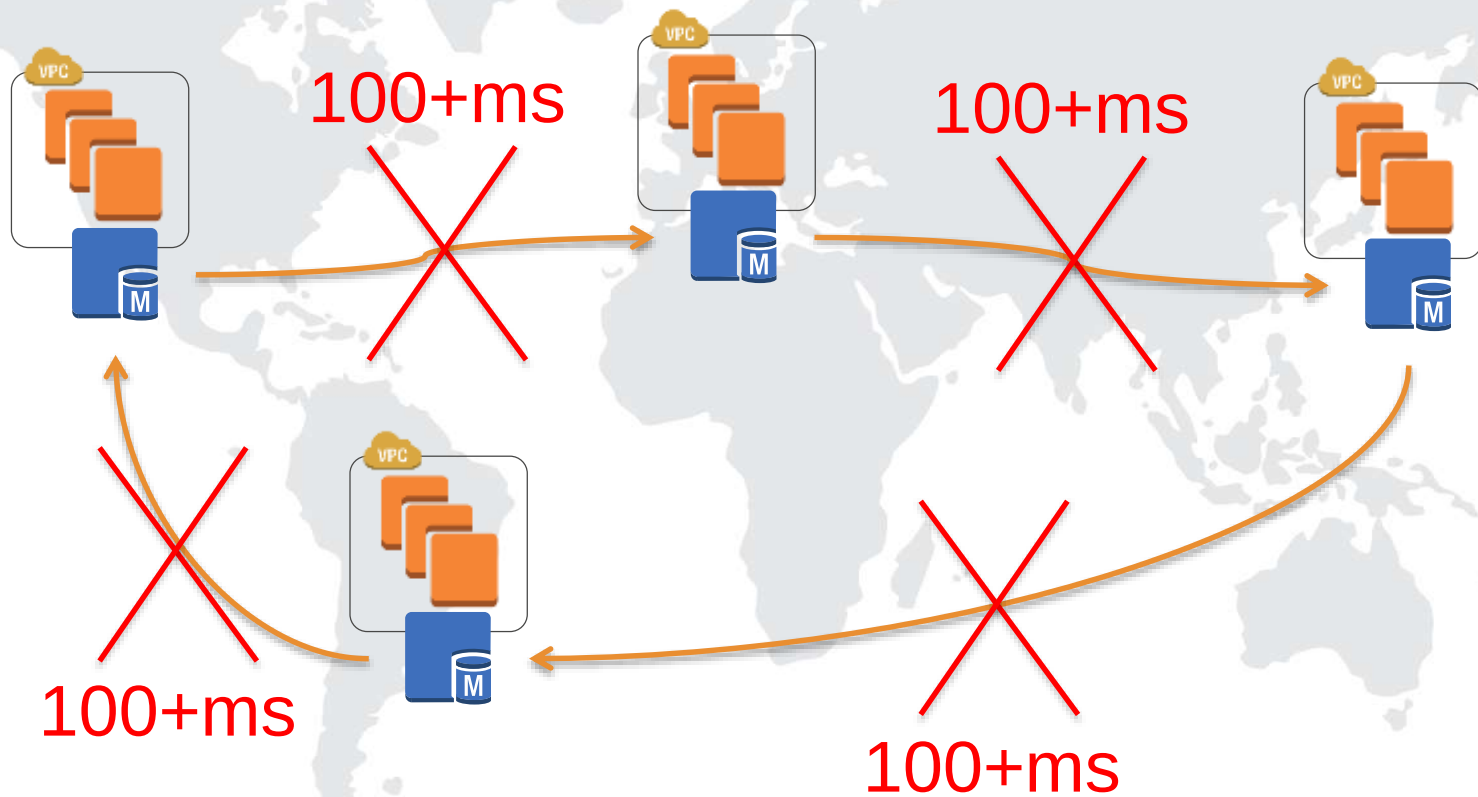
Source: www.SuperDataResearch.com

See also: <http://bit.ly/verizon-latency>

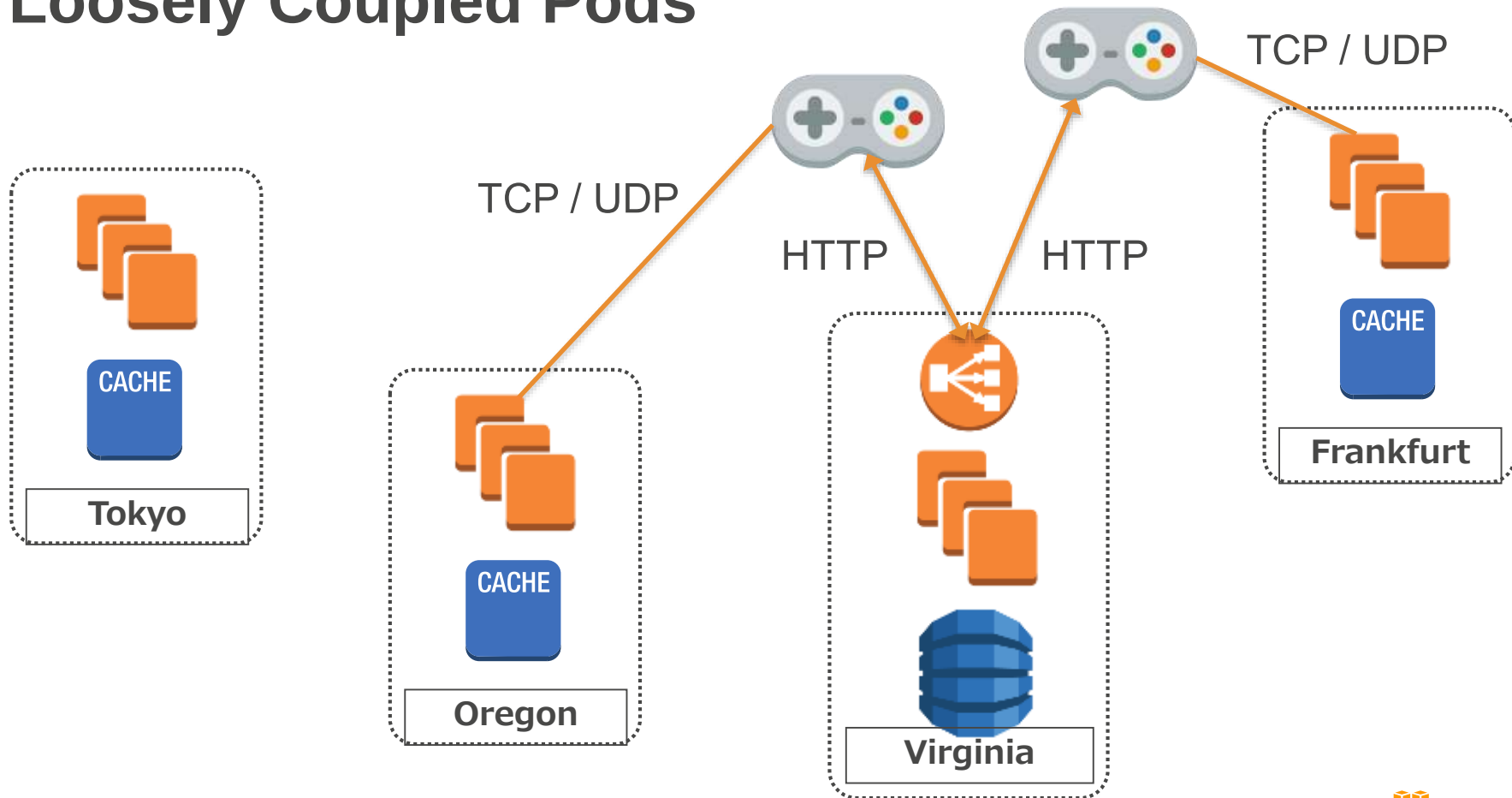
Game API Stack





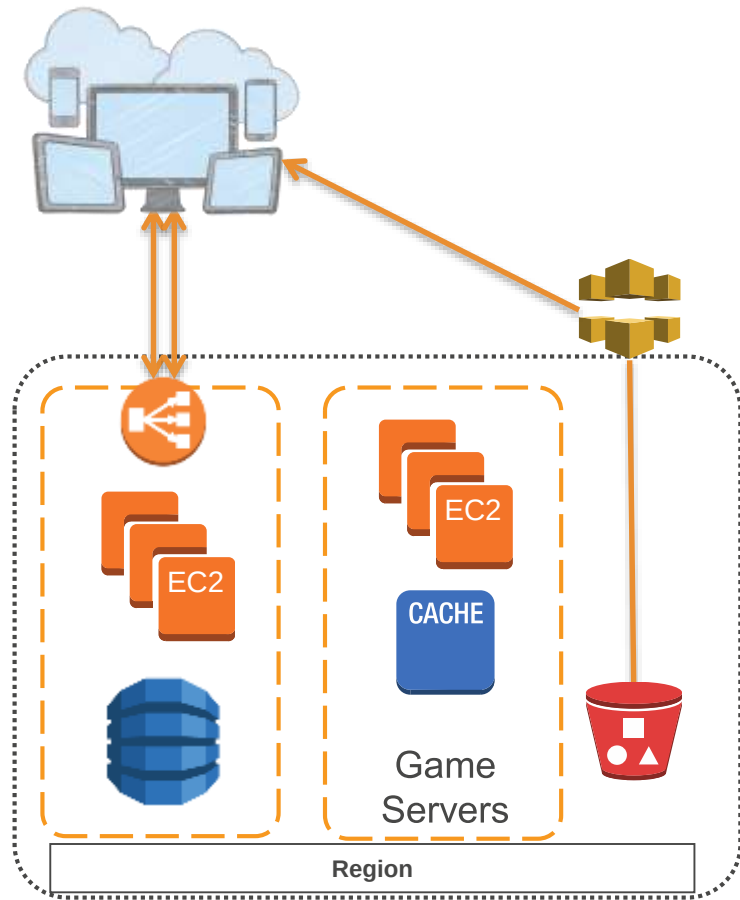


Loosely Coupled Pods



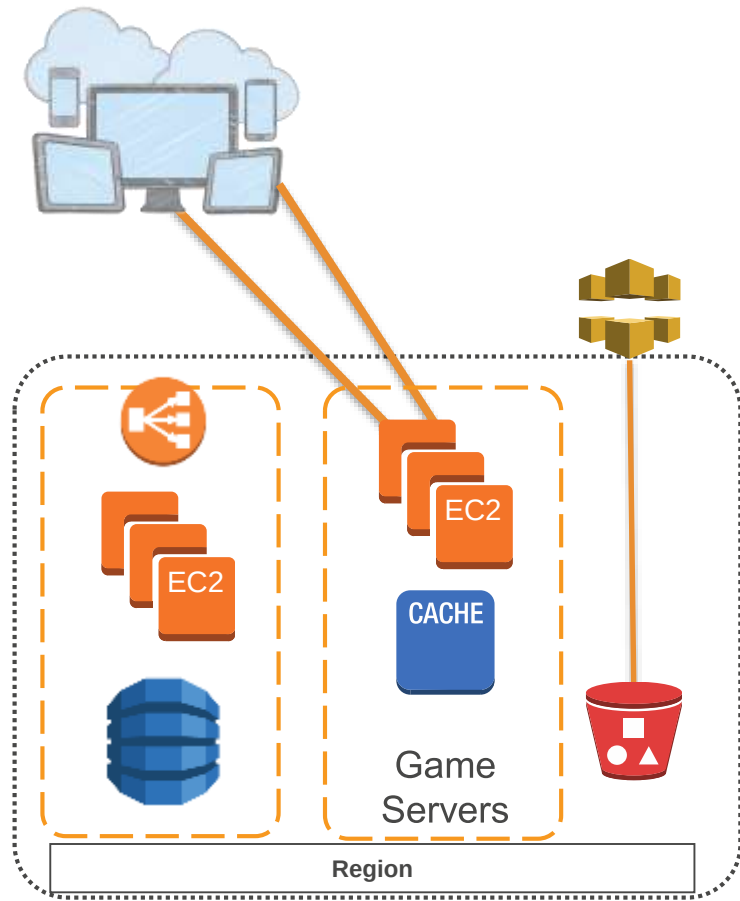
Game Flow

- ① Login via HTTP API
- ② Download Game Assets
- ③ Matchmaking to Game Server



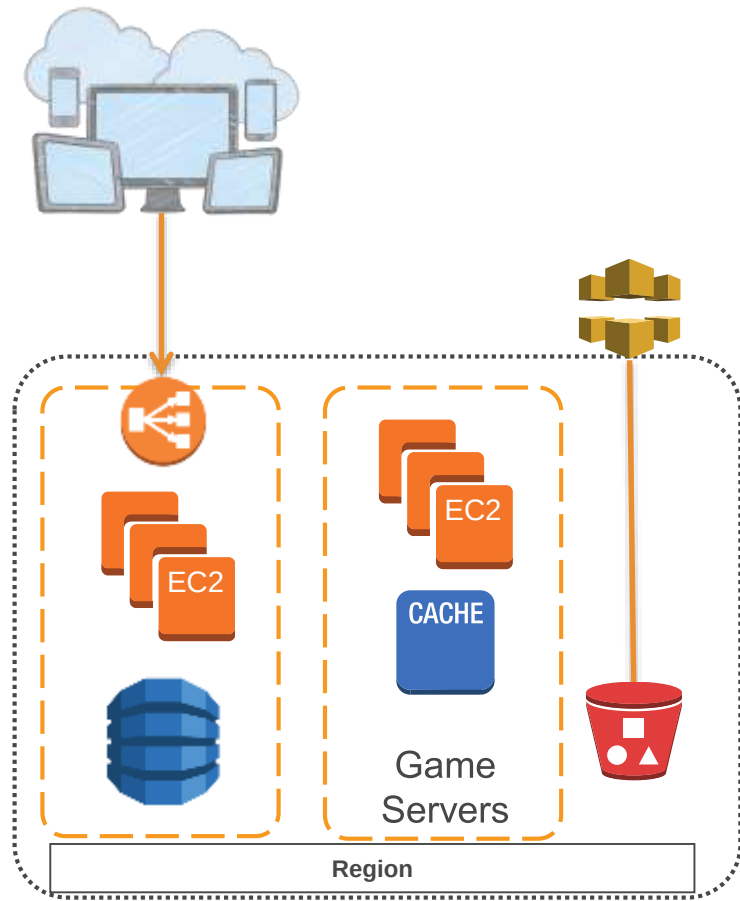
Game Flow

- ① Login via HTTP API
- ② Download Game Assets
- ③ Matchmaking to Game Server
- ④ Connect to Server
- ⑤ Hack Apart Your Friends
- ⑥ Game Over

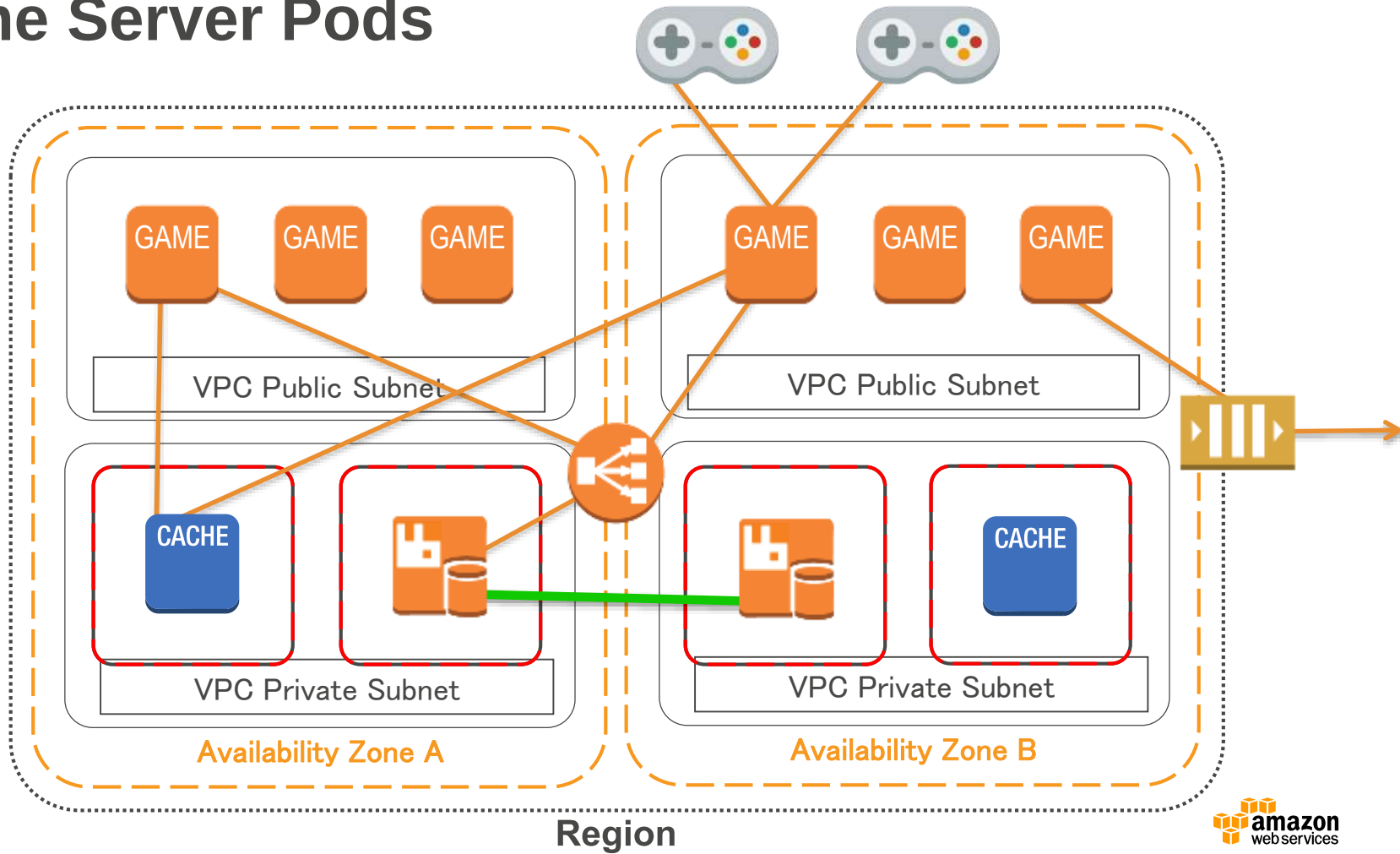


Game Flow

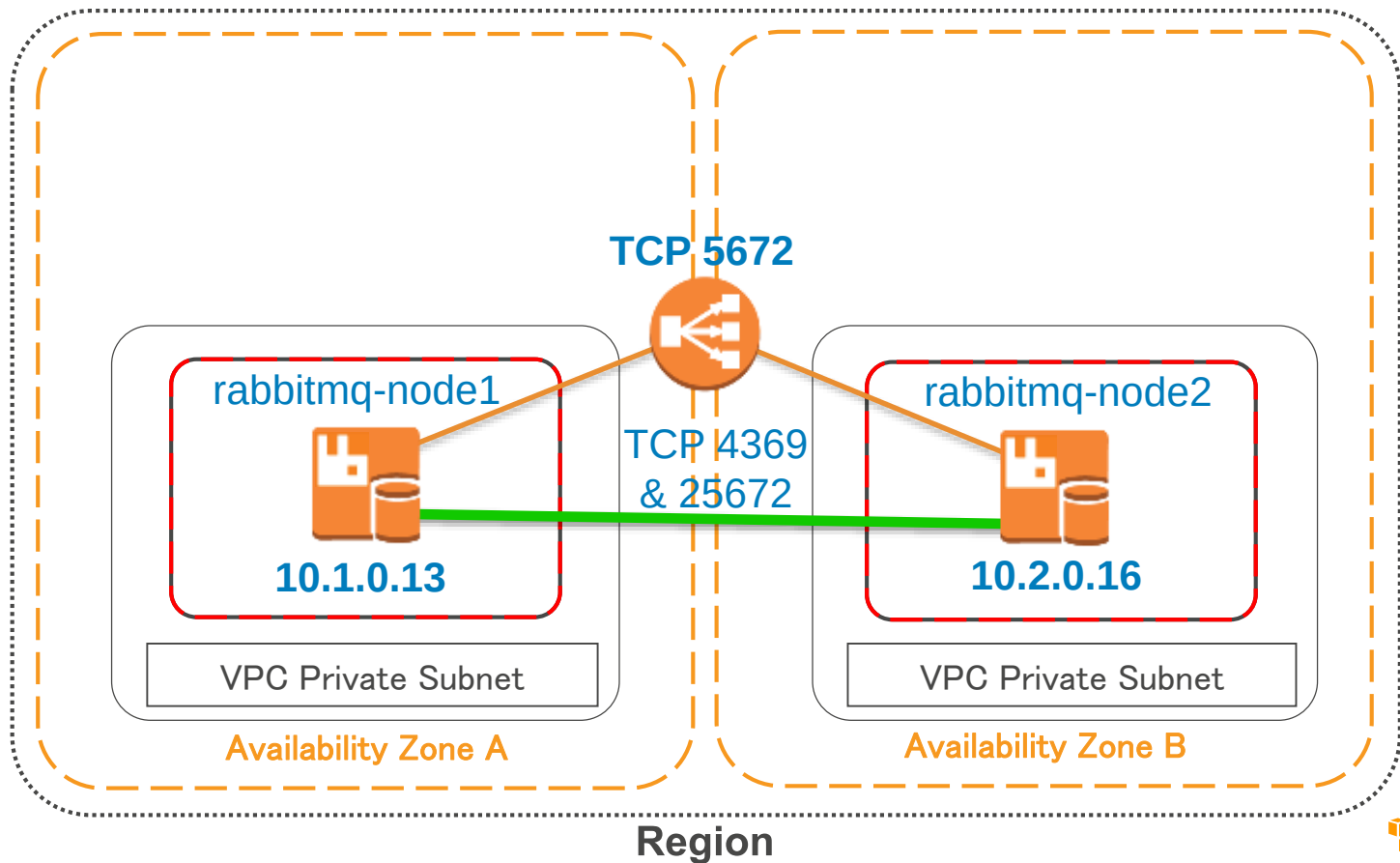
- ① Login via HTTP API
- ② Download Game Assets
- ③ Matchmaking to Game Server
- ④ Connect to Server
- ⑤ Hack Apart Your Friends
- ⑥ Game Over
- ⑦ Write via HTTP API



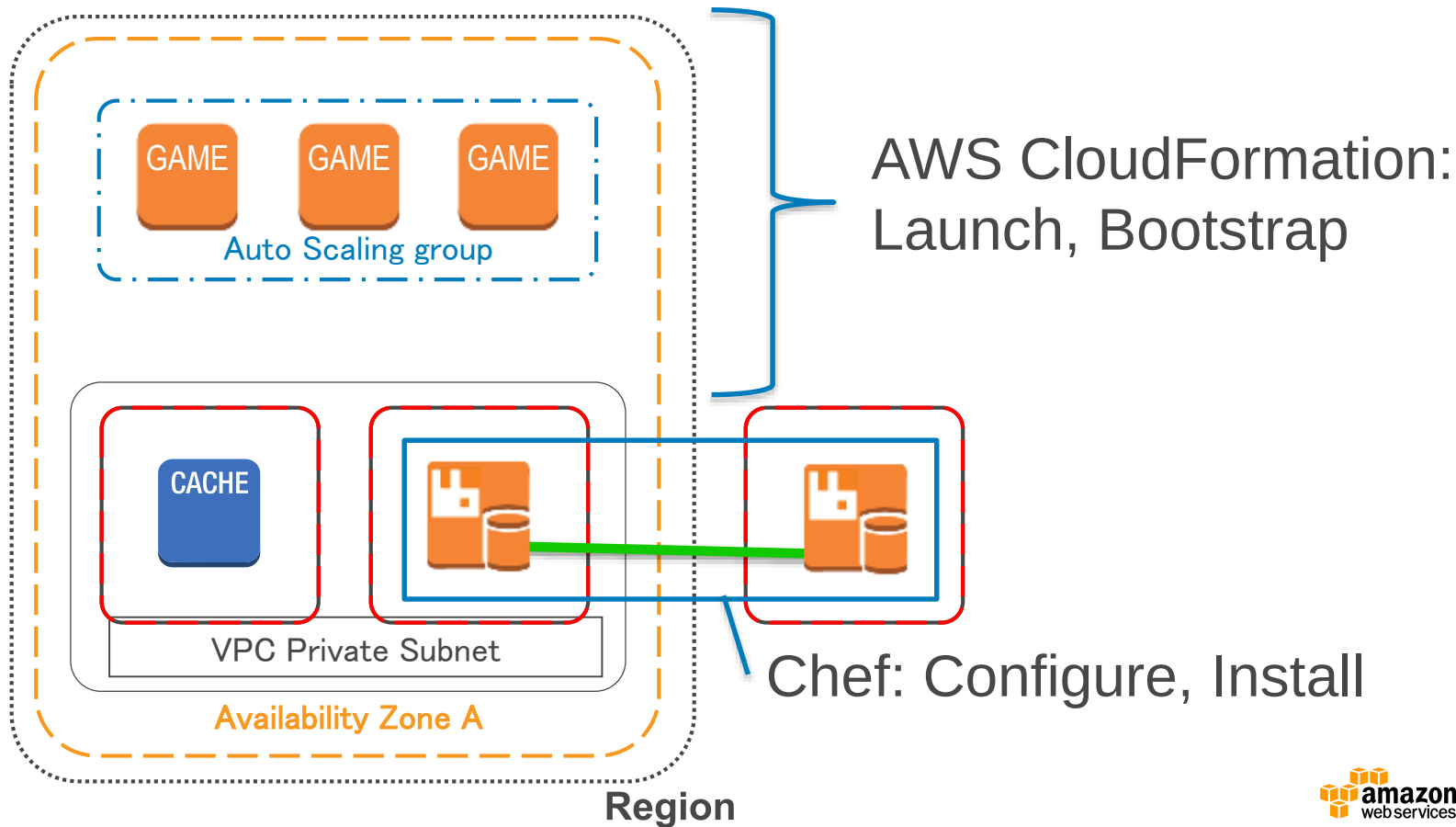
Game Server Pods



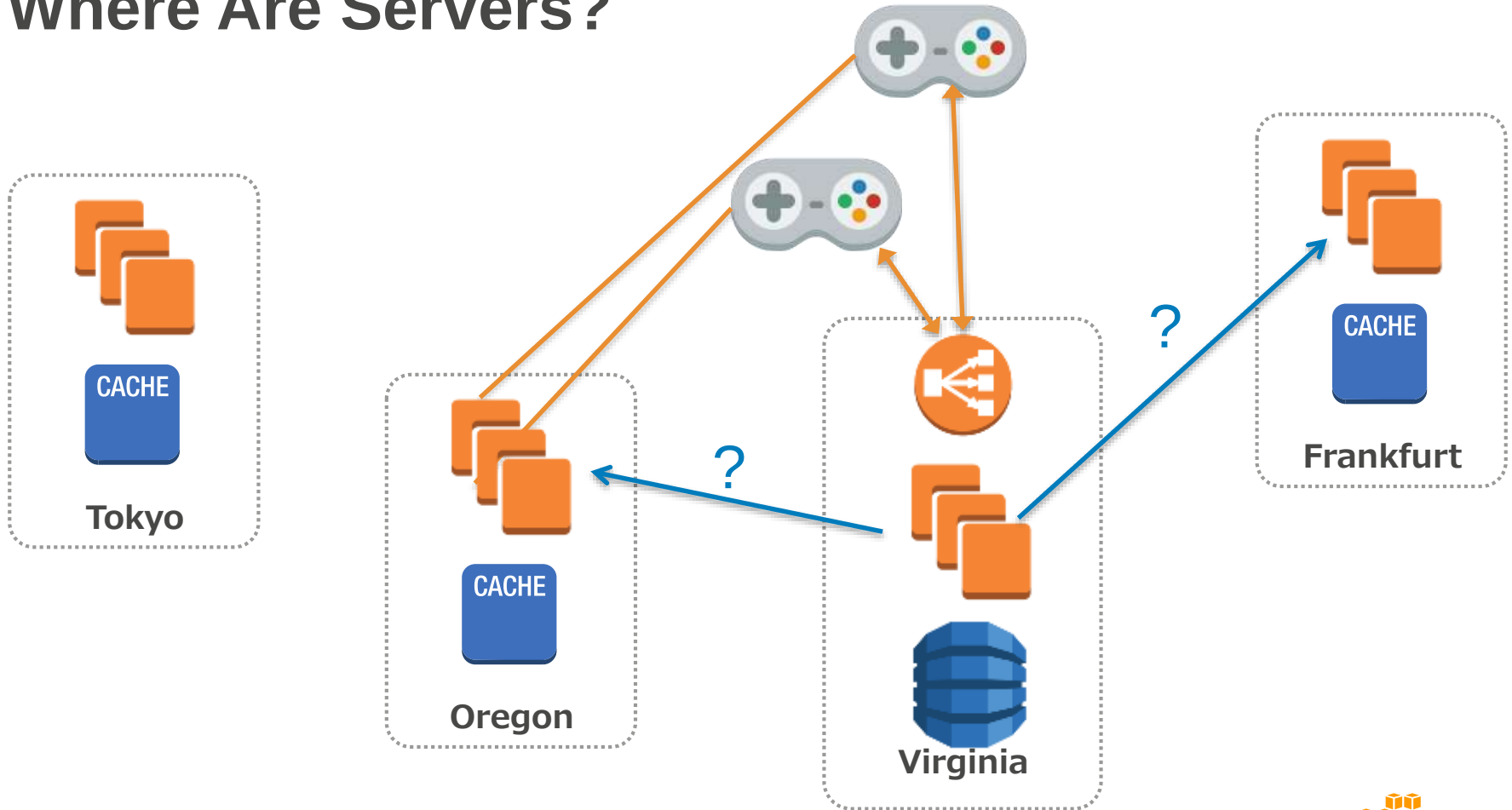
RabbitMQ + Elastic Load Balancing



CloudFormation + Chef

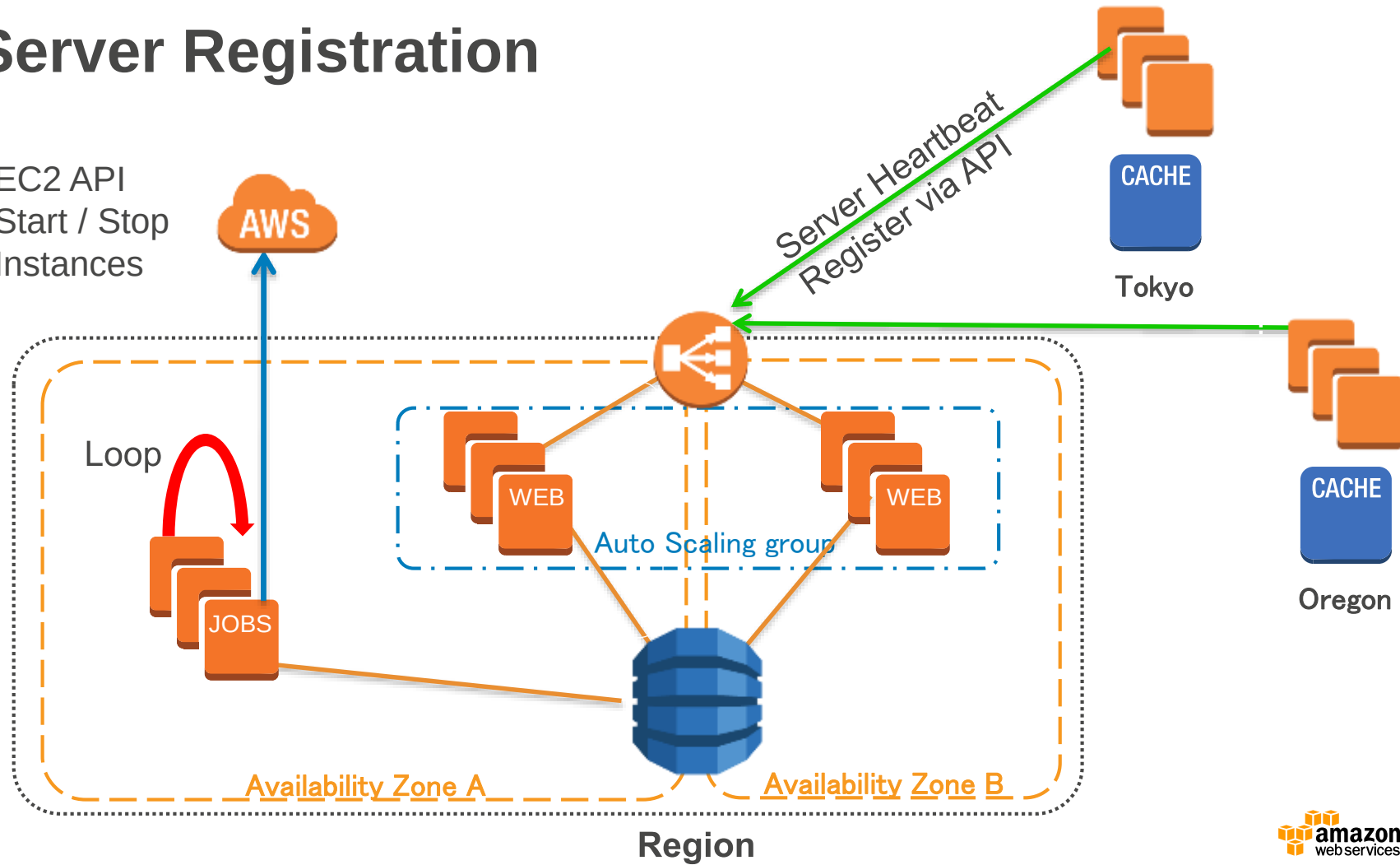


Where Are Servers?

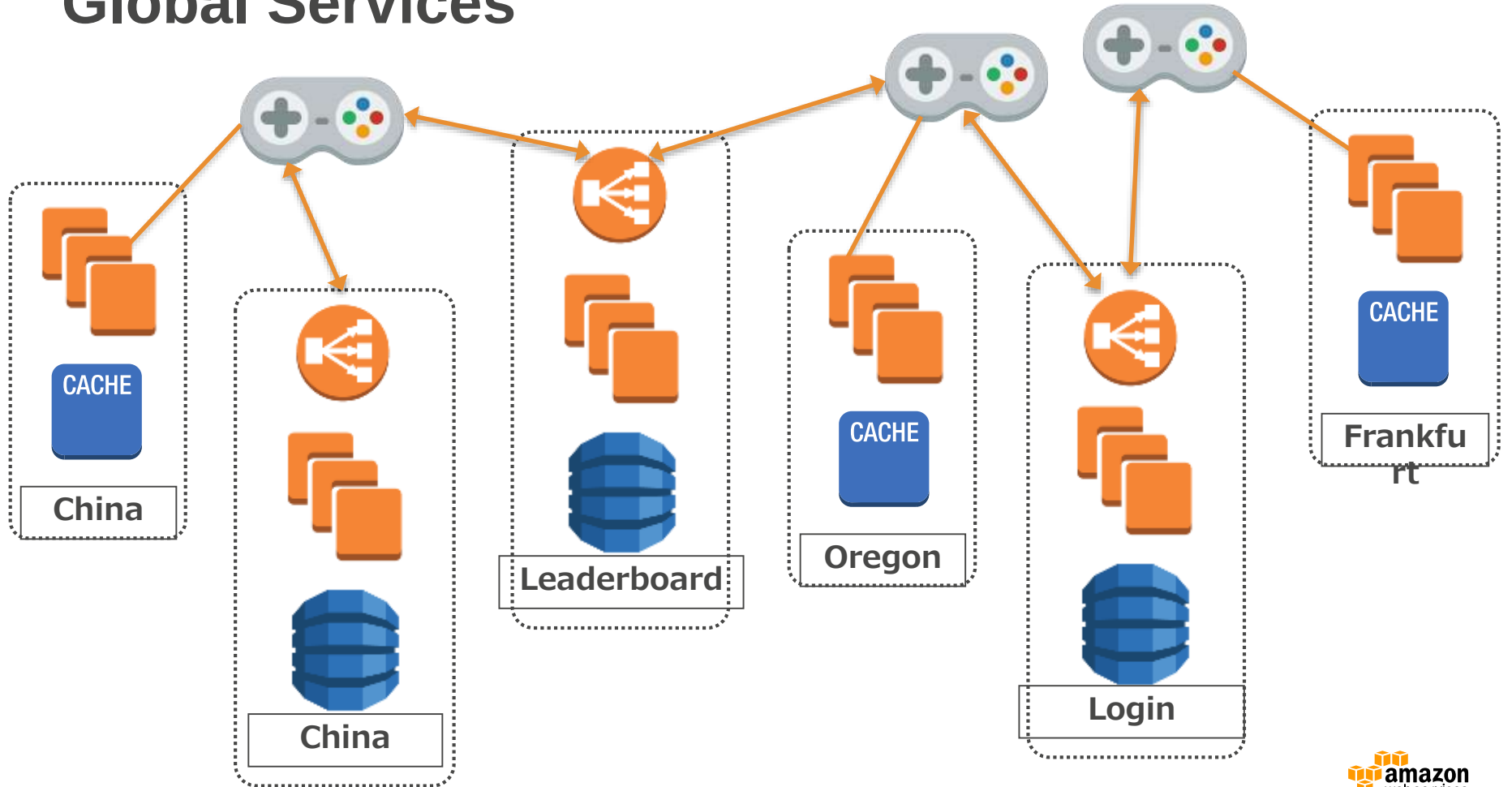


Server Registration

EC2 API
Start / Stop
Instances



Global Services

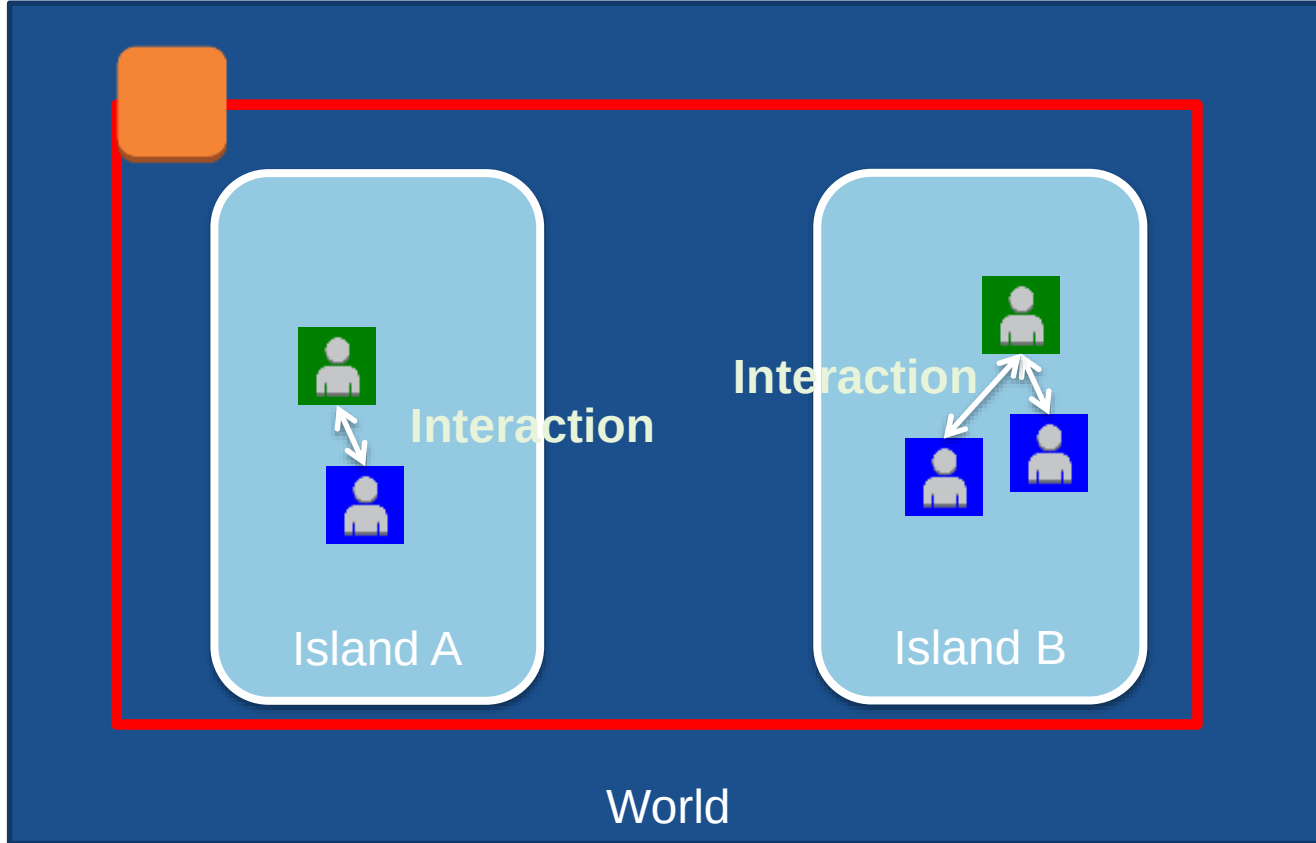


Case study: MMO RPG

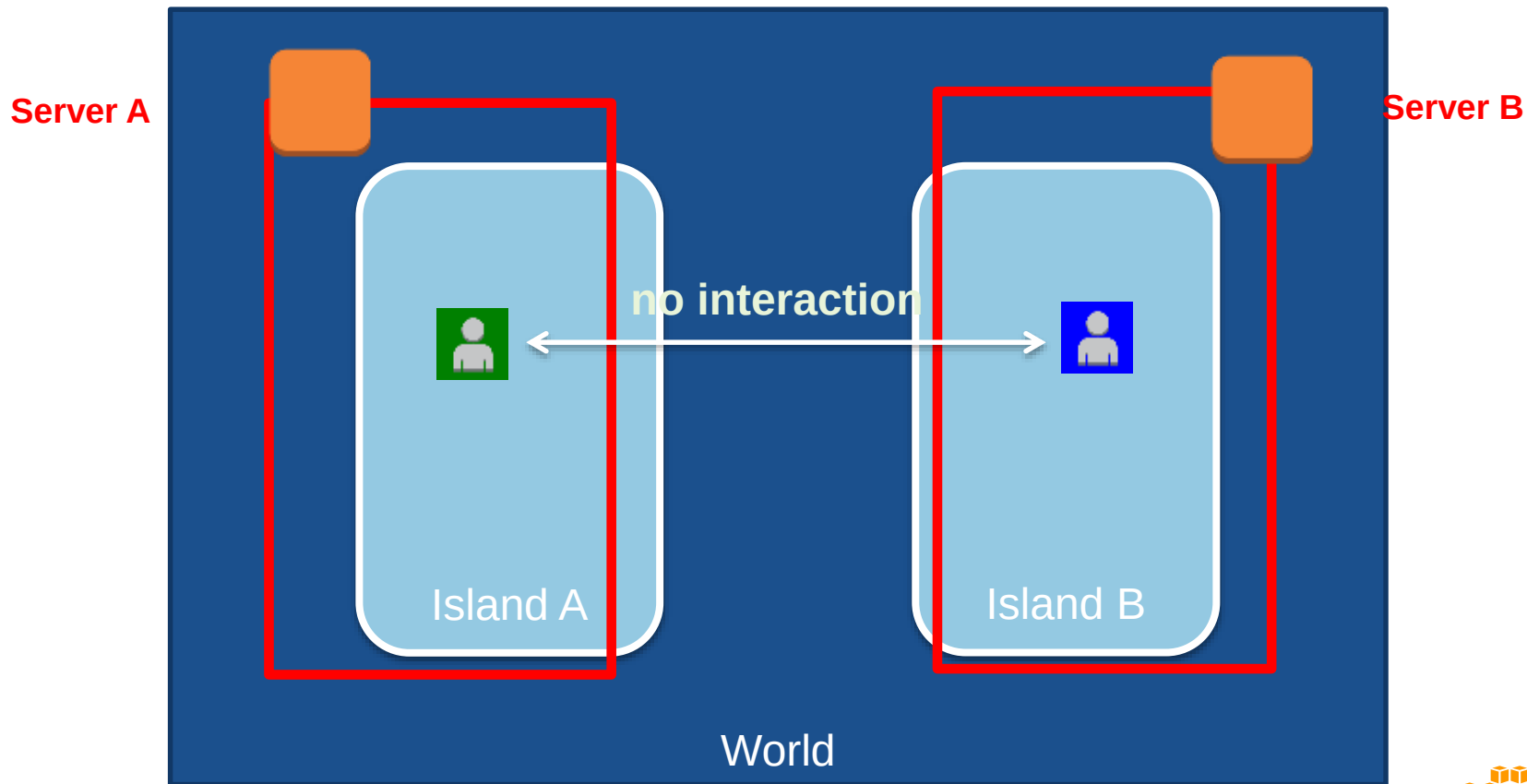
Application integrated deploy

Usual way of MMO implementation: Single big server

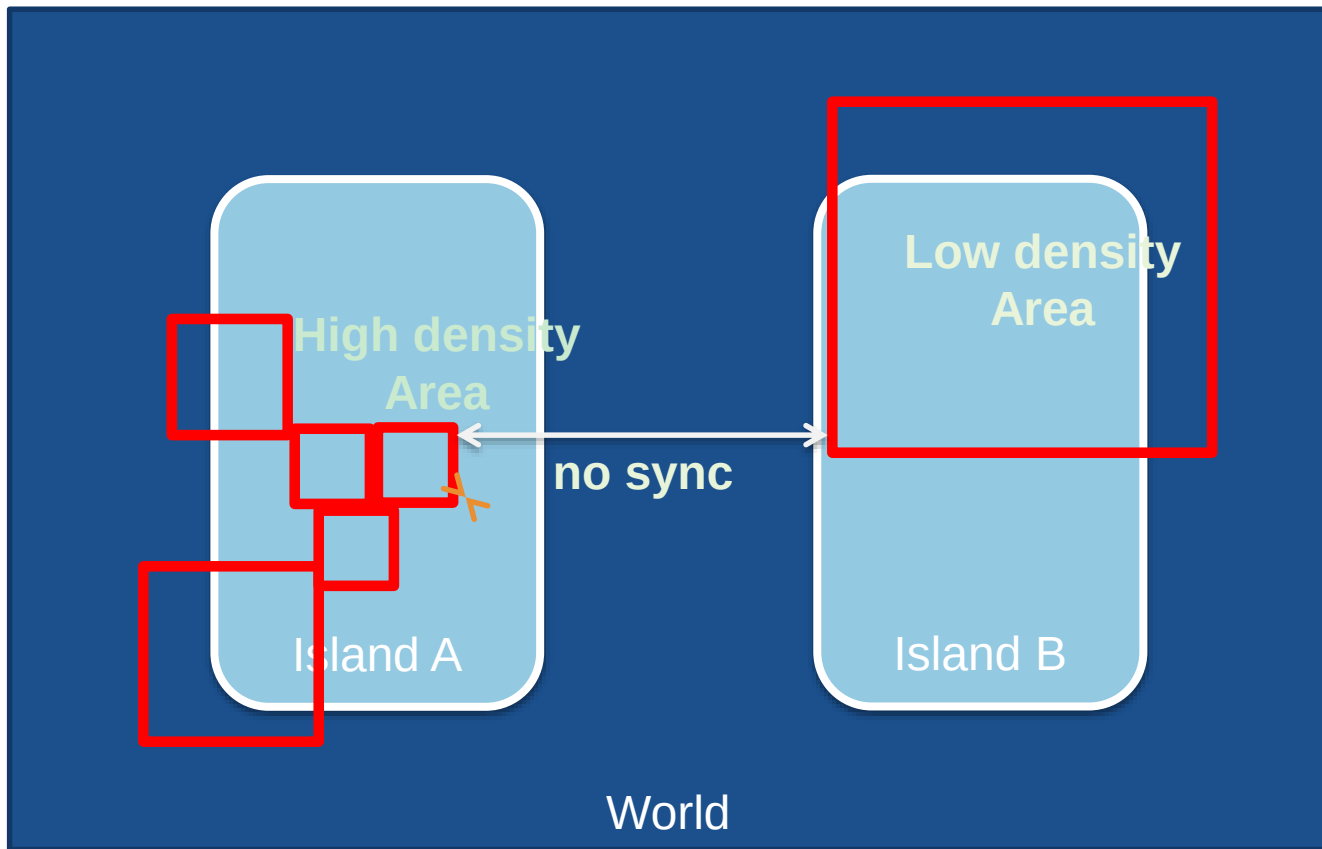
Server A



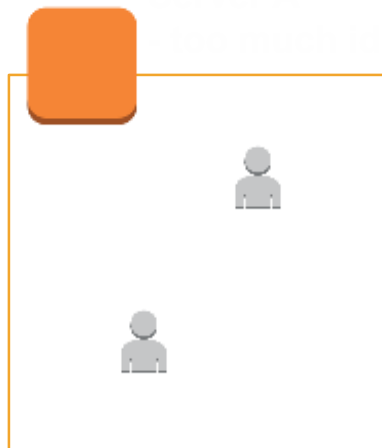
Distribute users by geo location in game map



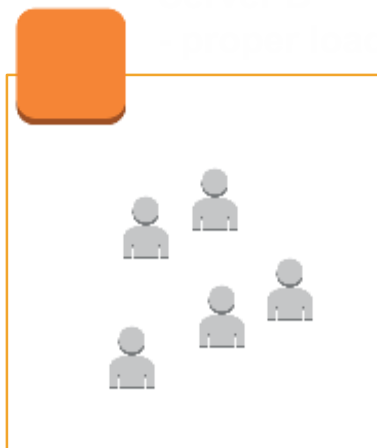
Distribute hot area with more game servers



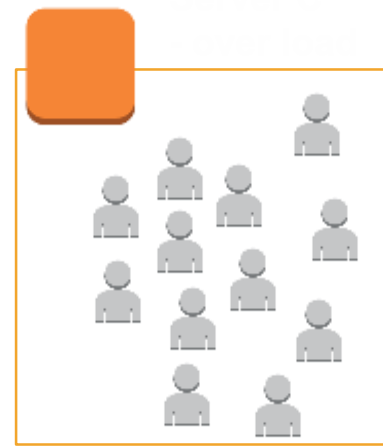
User density differentiated by map's popularity



Area A
less monsters
low experience points
poor item drop rate

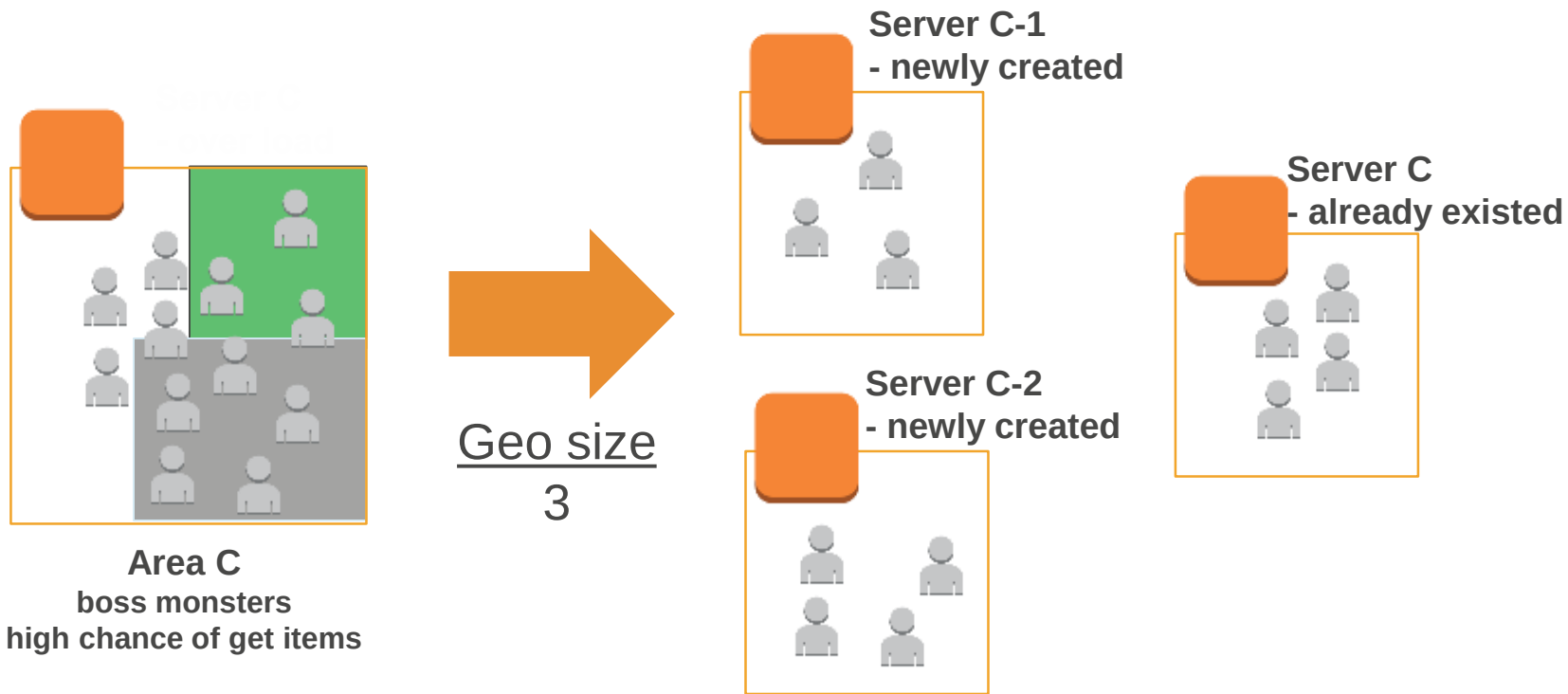


Area B
medium experience points
enough gold drops
avg. item drop rate



Area C
boss monsters
high chance of get items

When an area's population get higher



Case study: Netflix 2/2

Monitor everything

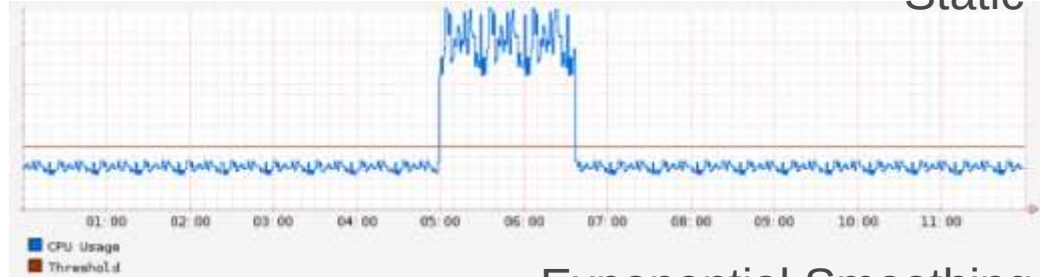
Monitor everything

- Let's monitor everything
 - User behavior
 - Application log
 - Access log
- Then make it available!

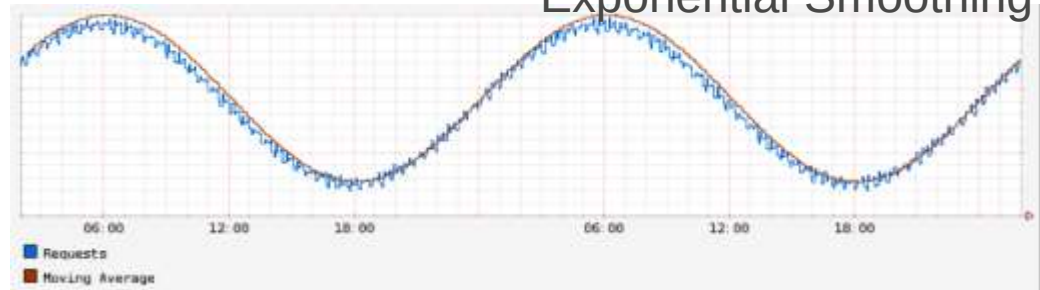
SPS : the Pulse of Netflix Streaming

- Starts Per Second
- They use SPS to detect issues
- Deviation modeling is important.

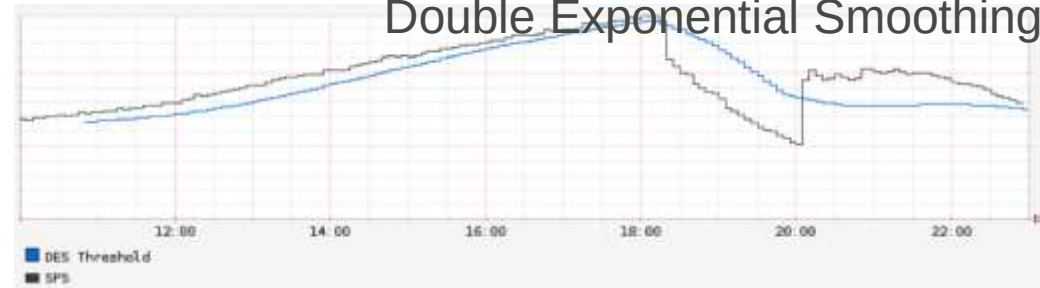
Static



Exponential Smoothing



Double Exponential Smoothing

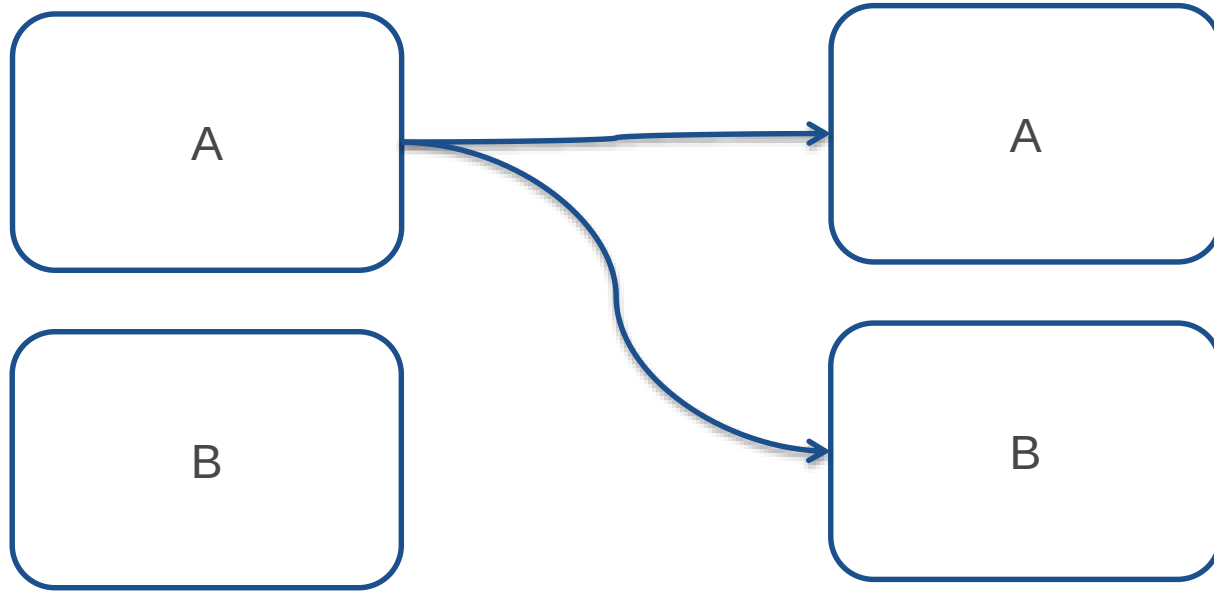


A/B Testing

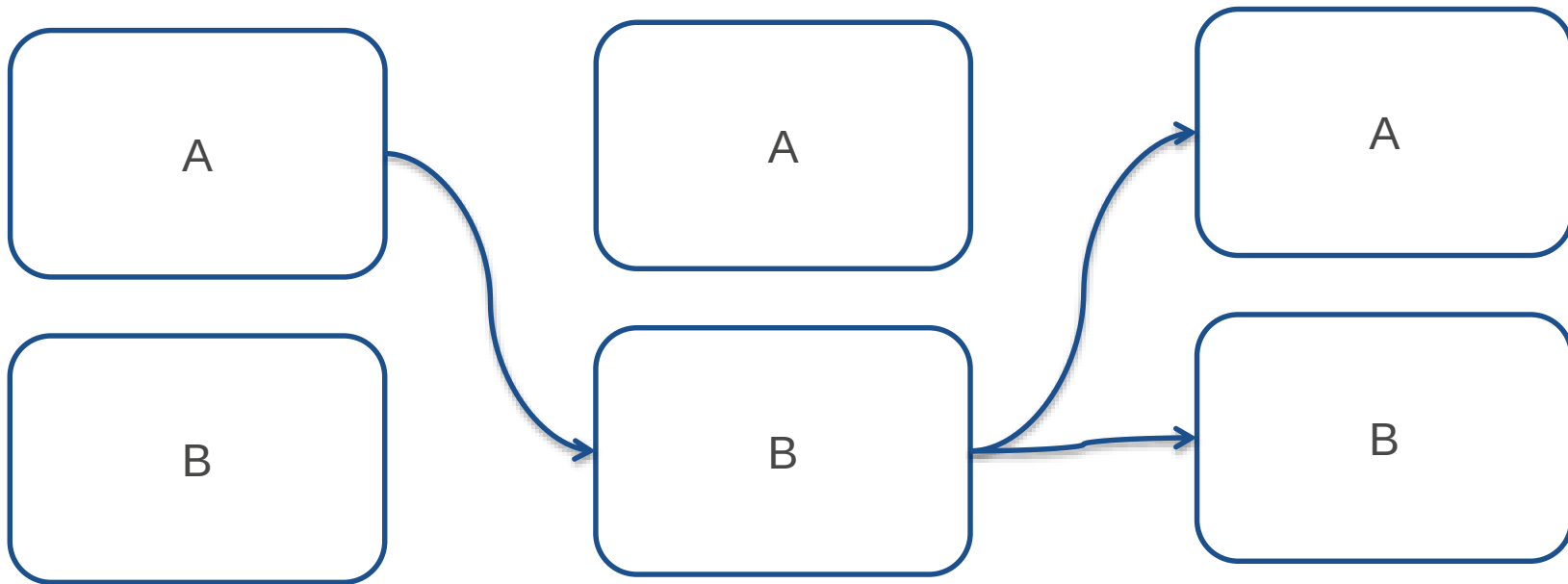
A

B

A/B Testing



A/B Testing



A/B Testing

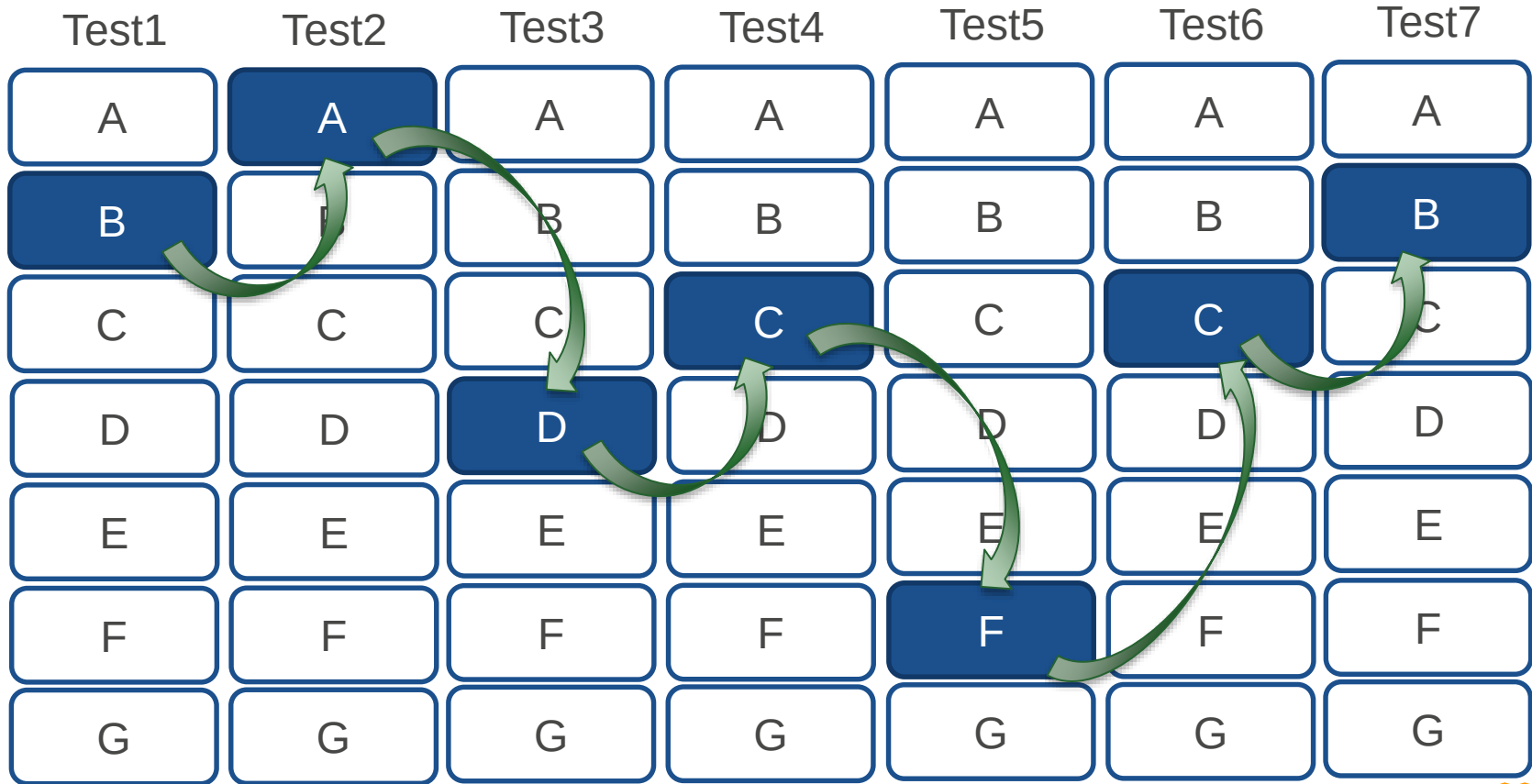
Test1	Test2	Test3	Test4	Test5	Test6	Test7
A	A	A	A	A	A	A
B	B	B	B	B	B	B
C	C	C	C	C	C	C
D	D	D	D	D	D	D
E	E	E	E	E	E	E
F	F	F	F	F	F	F
G	G	G	G	G	G	G

Huge numbers in A/B testing

- 2,097,152 unique experience across 7 tests
- Hundreds of new A/B tests per year
- 2,105 Templates, 566 CSS, 685 JS
- 2.5 million unique css/js packages per push cycle

This cannot be handled manually!

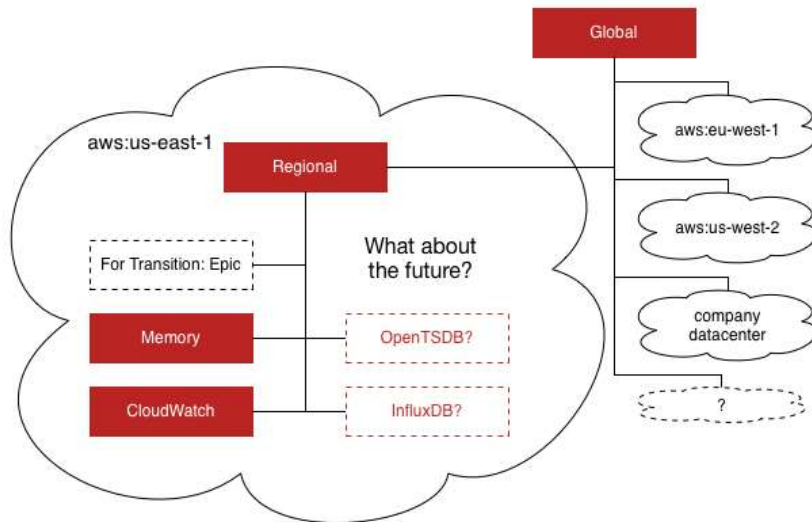
A/B tests should be automatically convergent



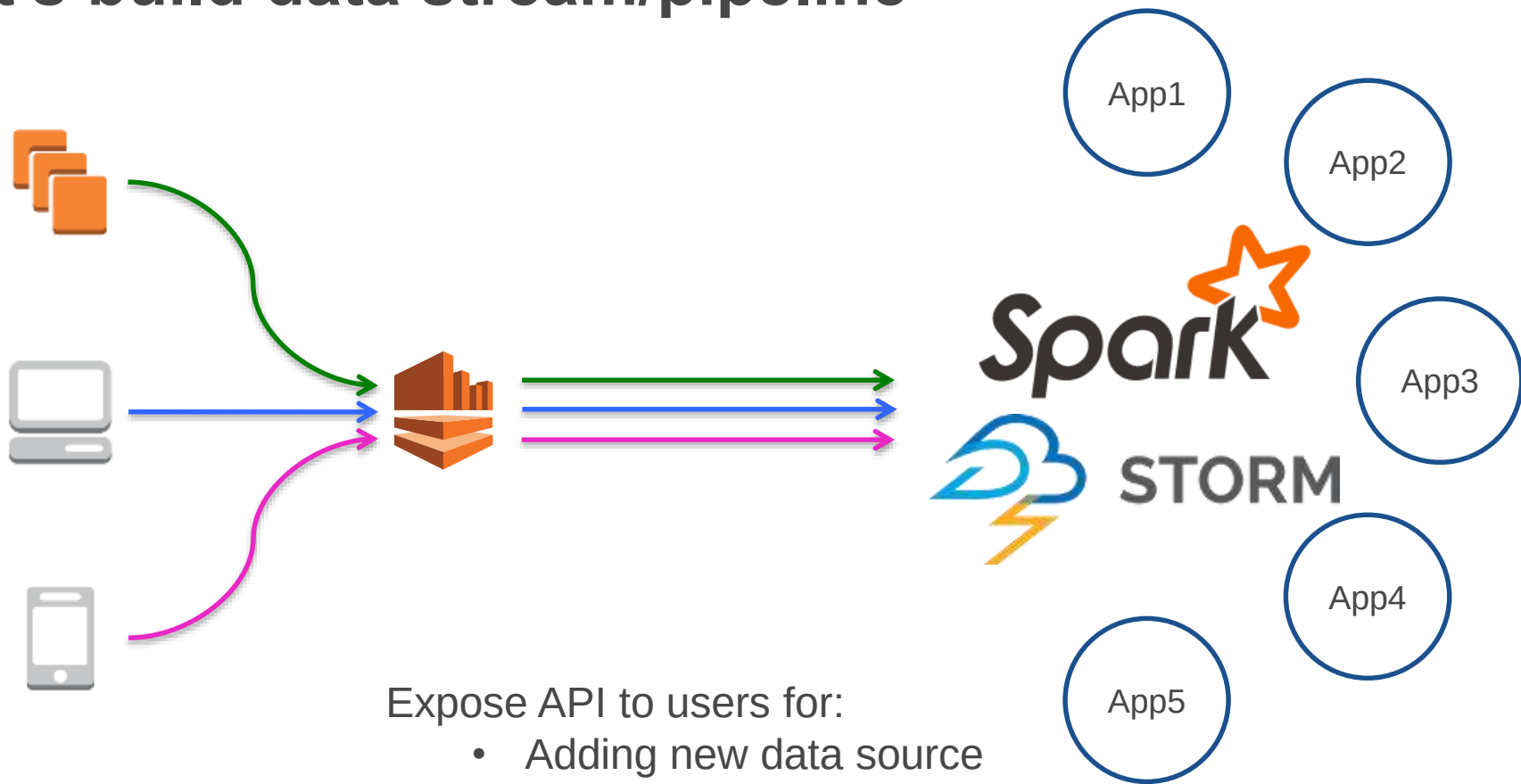
A/B tests should be automatically convergent: **How?**

- Monitor metrics
- Then
 - Deploy multiple versions of application
 - Configuration

Atlas: Telemetry Platform



Let's build data stream/pipeline



Expose API to users for:

- Adding new data source
- Adding new application

Conclusion

Conclusion

- Infra team should become 'Tool team'
- Many things to go other than Test, Build, Deploy
- DevOps affect software architecture