

Cloud Computing and Big Data

Cloud overview and introduction

Oxford University
Software Engineering
Programme
Sept 2017

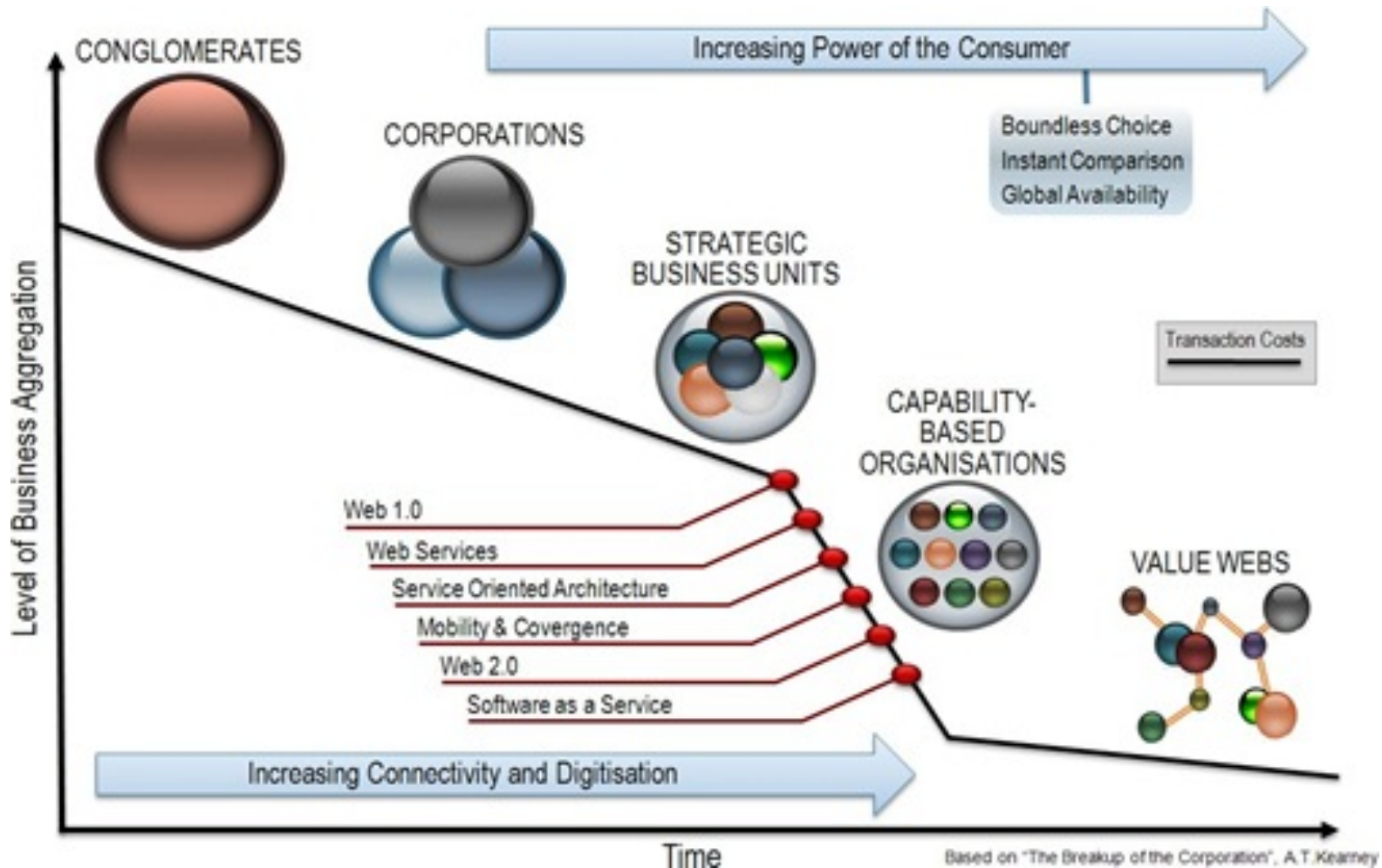


Contents

- Definitions
- Origins of Cloud Computing
- Case Studies and Motivations



Drivers for a new IT model



What is Cloud?

- Depends who **you** are
 - **My daughter:** iCloud (her music in the cloud)
 - **My mum:** gmail (her email in the cloud)
 - **My VP sales:** Salesforce (his prospects in the cloud)
 - **Sysadmin:** Amazon/Rackspace/etc (his infrastructure in the cloud)
 - *****: what *you* care about, self-provisioned, managed, metered and paid per use, in the cloud

Cloud Computing Definition (NIST)

- On-demand self-service
 - Users can provision resources without human intervention
- Broad network access
 - Heterogeneous access to resources
- Resource pooling
 - Multi-tenant shared capabilities
- Rapid elasticity
 - Services can scale up and down automatically
- Measured service
 - Resources can be metered and charged for based on real-world measures



Cloud Native

<http://pzf.fremantle.org/2010/05/cloud-native.html>

- **Distributed/Dynamically Wired** (works properly in the cloud)
 - Supports deploying in a dynamically sized cluster
 - Finds services across applications even when they move
- **Elastic** (Uses the cloud efficiently)
 - Scales up and down as needed
 - Works with the underlying IaaS
- **Multi-tenant** (Only costs when you use it)
 - Virtual isolated instances with near zero incremental cost
 - Implies you have a proper identity model
- **Self-service** (in the hands of users)
 - De-centralized creation and management of tenants
 - Automated Governance across tenants
- **Granularly Billed and Metered** (pay for just what you use)
 - Allocate costs to exactly who uses them
- **Incrementally Deployed and Tested** (seamless live upgrades)
 - Supports continuous update, side-by-side operation, in-place testing and incremental production



New definition of Cloud Native

- From the Cloud Native Computing Foundation
 - Container based
 - Dynamic
 - Microservice oriented

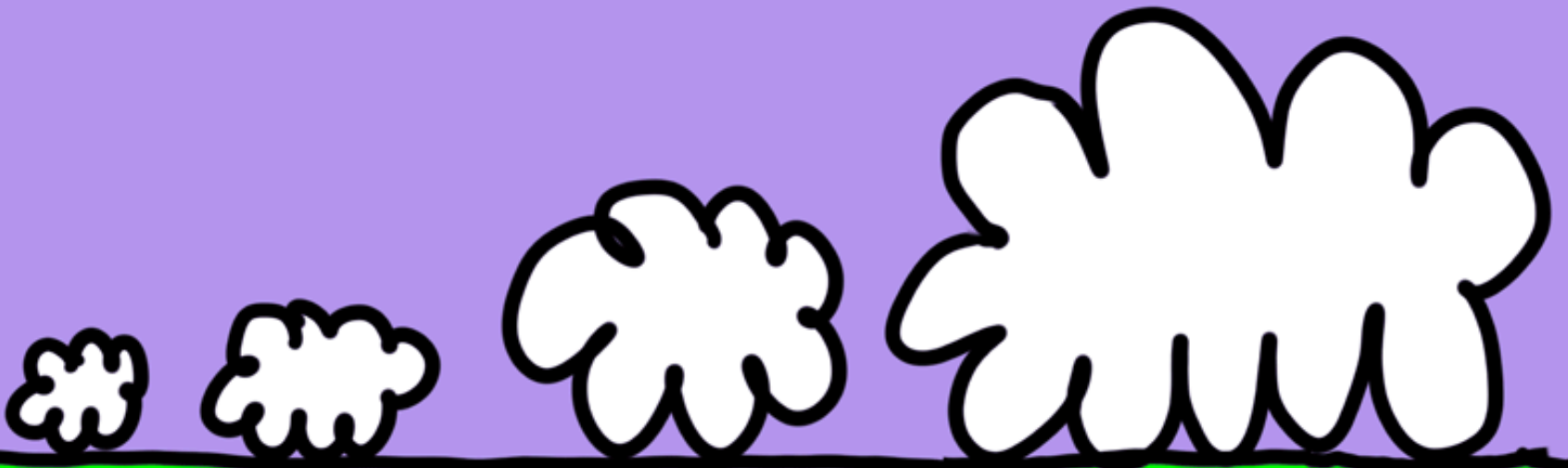


Origins of Cloud Computing

- Virtual Machines on Mainframes
 - VM/370 – 1972
- Grid Computing
 - Grid computing is the collection of computer resources from multiple locations to reach a common goal.
- Software-as-a-Service
 - Salesforce.com 1999
- Amazon AWS
 - 2002

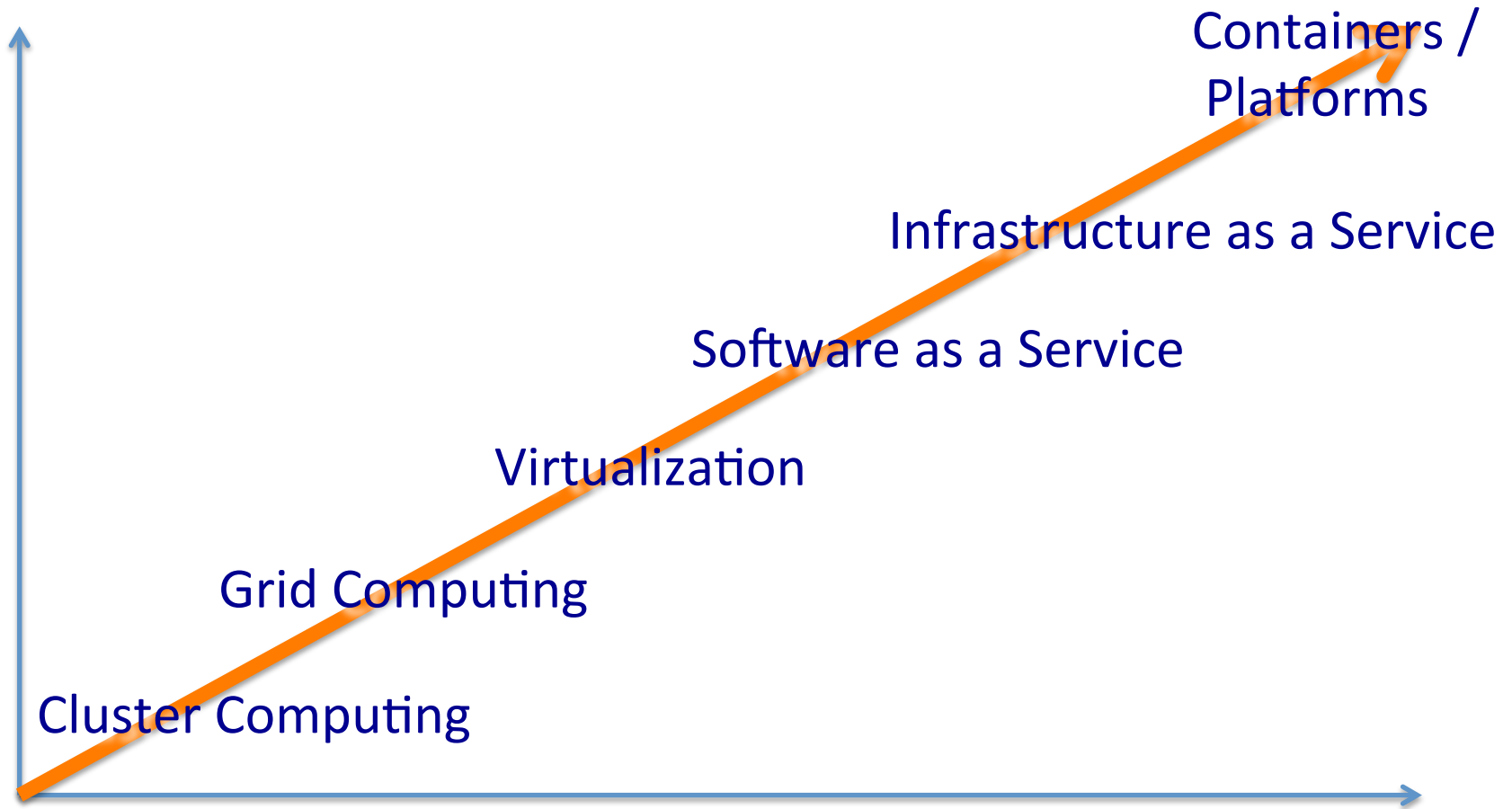


the evolution of the cloud:



@gapingvoid

Evolution of Cloud



CASE STUDIES



© Paul Fremantle 2015. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Netflix - Watch TV Program

https://signup.netflix.com

Offline Mail | Inbox (124,820) - pa | WSO2 | WSO2, Inc. - Calende | + bitmark | Shorten with bit.ly | Gmail - Inbox (720)

NETFLIX

Questions? Call 0800 096 6379 - 24 | Buy / Redeem Gift | Member

1 MONTH FREE TRIAL

Watch TV programmes & films anytime, anywhere.

Only £5.99 a month.

Start Your Free Month

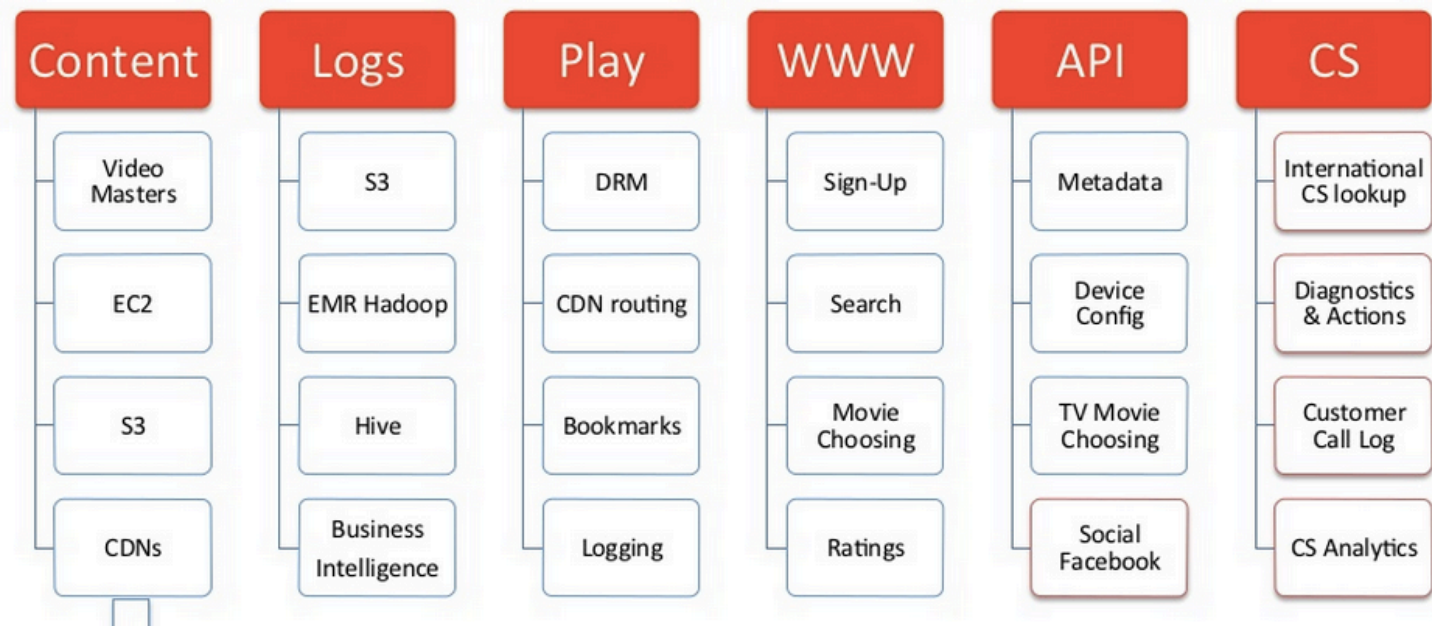
The banner features a large background image of a family (a man, a woman, and two children) sitting on a couch in a dimly lit room, looking at a large television. The television screen displays the Netflix logo at the top and a grid of movie and TV show thumbnails below it. Some visible titles include 'TAXI', 'WILDA BEAST', 'KATIE MAC', 'HOT SOULS', 'ZEN: ZACK', 'KENYA', 'CONTRACT VIOLENCE', 'AMERICAN', 'ROCKY HONOLULU', 'OPUM', 'JULIEZ', and 'drama yogo'.

Netflix

- A REST and Cloud based SOA approach
- Continuous Delivery
- 100% Based in the cloud
- See excellent presentations from Adrian Cockcroft
 - e.g.
<http://www.slideshare.net/adrianco/global-netflix-platform>



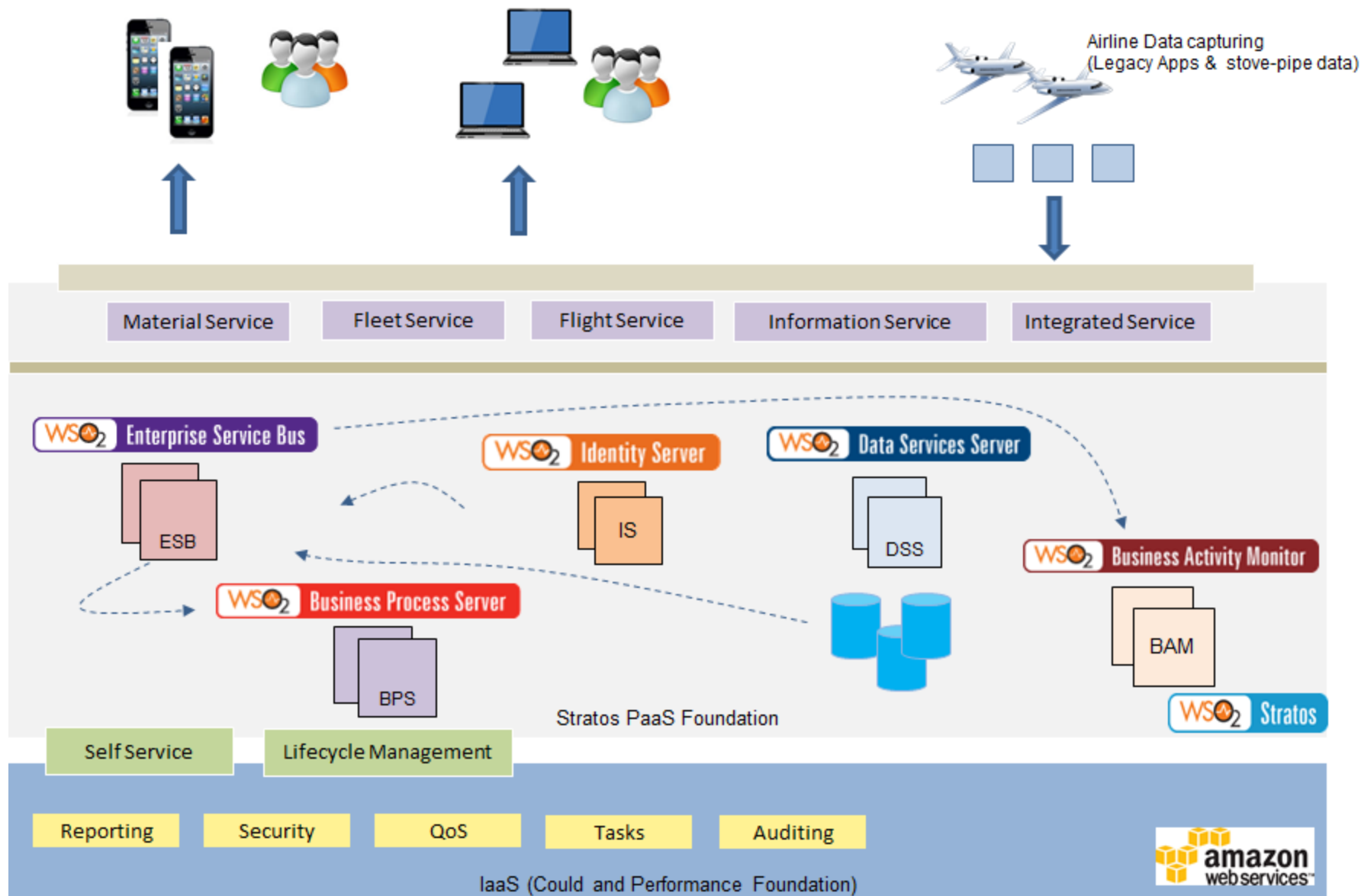
Netflix Deployed on AWS



Boeing Digital Airline



Case Study : Boeing - A PaaS based Integration and API ecosystem



Case Study : Multi-tenanted Mobile Orchestration Gateway Platform

Customer

One of the largest global networking solutions providers required to build a mobile services orchestration gateway platform, enabling mobile providers to simplify QoS service access to their external business partners.

Challenge

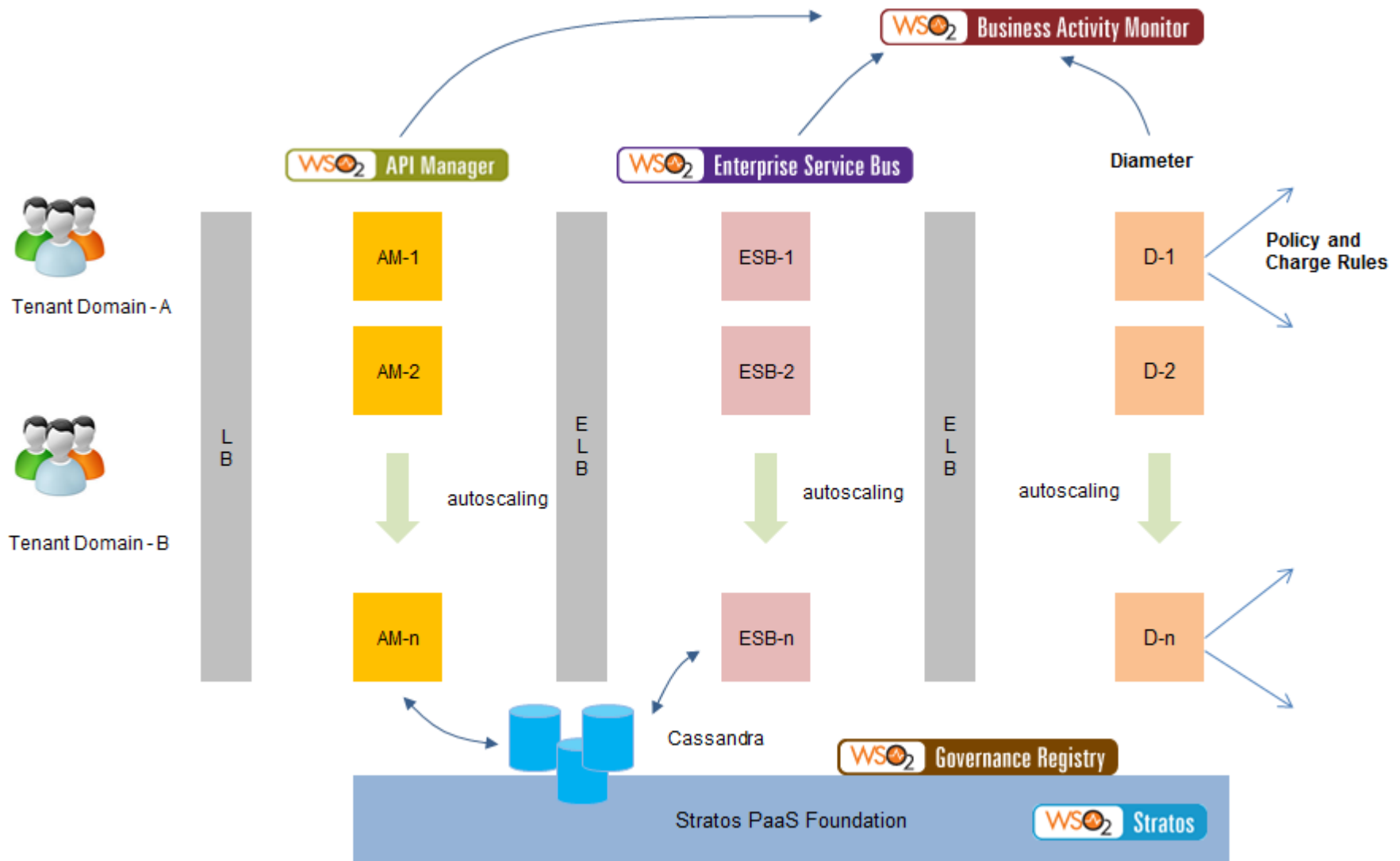
- Build a mobile services orchestration gateway than can scale upto 40,000 TPS with 99.999% service availability.
- Extensible architecture capable of interfacing with multiple protocols such as XMPP, Diameter whilst maintaining pre-defined SLAs and throughput.
- Integrating with ASR5000K, Third-party PCRF systems
- Multi-tenancy support for API lifecycle management.
- Multi-geographical deployment with autoscaling and failover compensation.

Solution

- Rebuilt an 18 month project in 4 weeks
- API Governance powered by multi-tenanted API Manager cluster with enforced security and lifecycle management.
- Business logic through ESB mediators exposed as REST APIs.
- Stateful caching using Cassandra
- Analytics and monetization of API usage using BAM integrated with enterprise licensing platform.
- Partner Onboarding interfaces and authorization workflows.
- Enterprise-grade cloud deployment based on Stratos PaaS foundation with native support for multi-tenancy, resource pooling and elastic scaling.



Case Study : Multi-tenanted Mobile Orchestration Gateway Platform

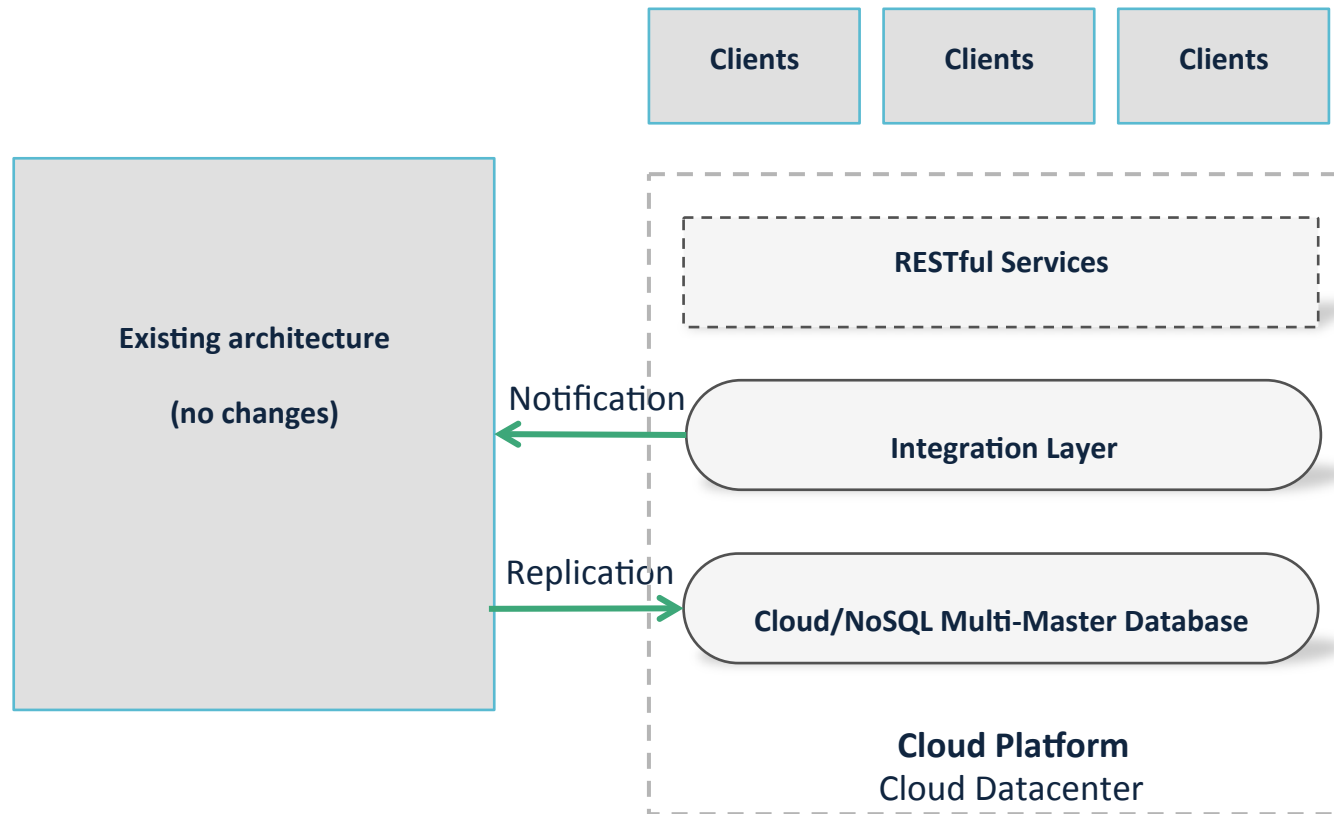


Pay TV company

- Needed to scale up to provide instant pay-as-you-go on mobile devices
- Support Disaster Recovery (DR)
- Elastic Scale e.g. during an important football match



Architecture



Large-scale cluster management at Google with Borg

Abhishek Verma[†] Luis Pedrosa[‡] Madhukar Korupolu

David Oppenheimer Eric Tune John Wilkes

Google Inc.

Abstract

Google's Borg system is a cluster manager that runs hundreds of thousands of jobs, from many thousands of different applications, across a number of clusters each with up to tens of thousands of machines.

It achieves high utilization by combining admission control, efficient task-packing, over-commitment, and machine sharing with process-level performance isolation. It supports high-availability applications with runtime features that minimize fault-recovery time, and scheduling policies that reduce the probability of correlated failures. Borg simplifies life for its users by offering a declarative job specification language, name service integration, real-time job monitoring, and tools to analyze and simulate system behavior.

We present a summary of the Borg system architecture and features, important design decisions, a quantitative analysis of some of its policy decisions, and a qualitative examination of lessons learned from a decade of operational experience with it.

1. Introduction



Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

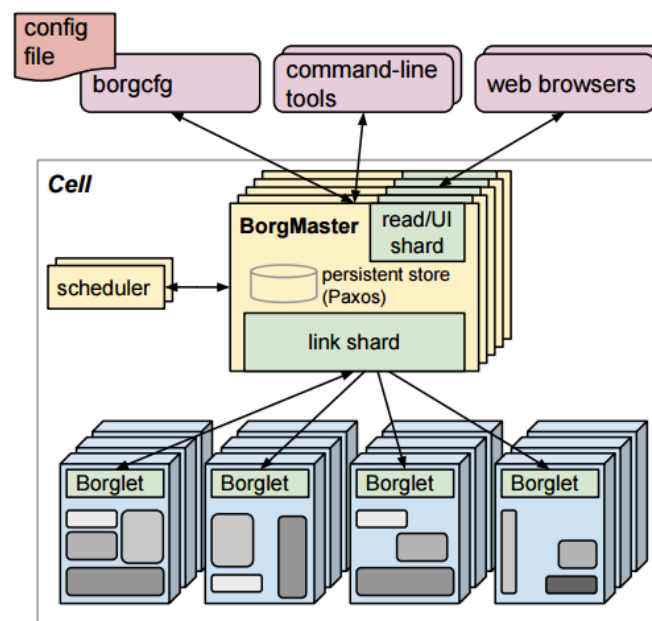


Figure 1: The high-level architecture of Borg. *Only a tiny fraction of the thousands of worker nodes are shown.*

cluding with a set of qualitative observations we have made from operating Borg in production for more than a decade.

Questions?



© Paul Fremantle 2015. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>