Netflix in the Cloud

Nov 6, 2010

Adrian Cockcroft

@adrianco #netflixcloud

http://www.linkedin.com/in/adriancockcroft

Combined Slides

For both Developer and Operations Audiences (2-3 hours of content)

Teaser intro slides also at slideshare.net/adrianco
Oct 14th Beta #devops subset - Cloud Computing Meetup
Nov 3rd GA – QConSF Developer Oriented subset
Nov 6th, 2010 – Combined slides at slideshare.net/adrianco

With more than 16 million subscribers in the United States and Canada, Netflix, Inc. is the world's leading Internet subscription service for enjoying movies and TV shows.

Source: http://ir.netflix.com

NETFLIX

Why Give This Talk?

Netflix is Path-finding

The Cloud ecosystem is evolving very fast

Share with and learn from the cloud community

adrianco adrian cockcroft
This week's choice: buy/sell cloud at #CloudExpo or build/use cloud at #QConSF, I'm a user, you?
15 hours ago

We want to use clouds, not build them

Cloud technology should be a commodity

Public cloud and open source for agility and scale

NETFLIX

We are looking for talent

Netflix wants to connect with the very best engineers

Why Use AWS?

We stopped building our own datacenters

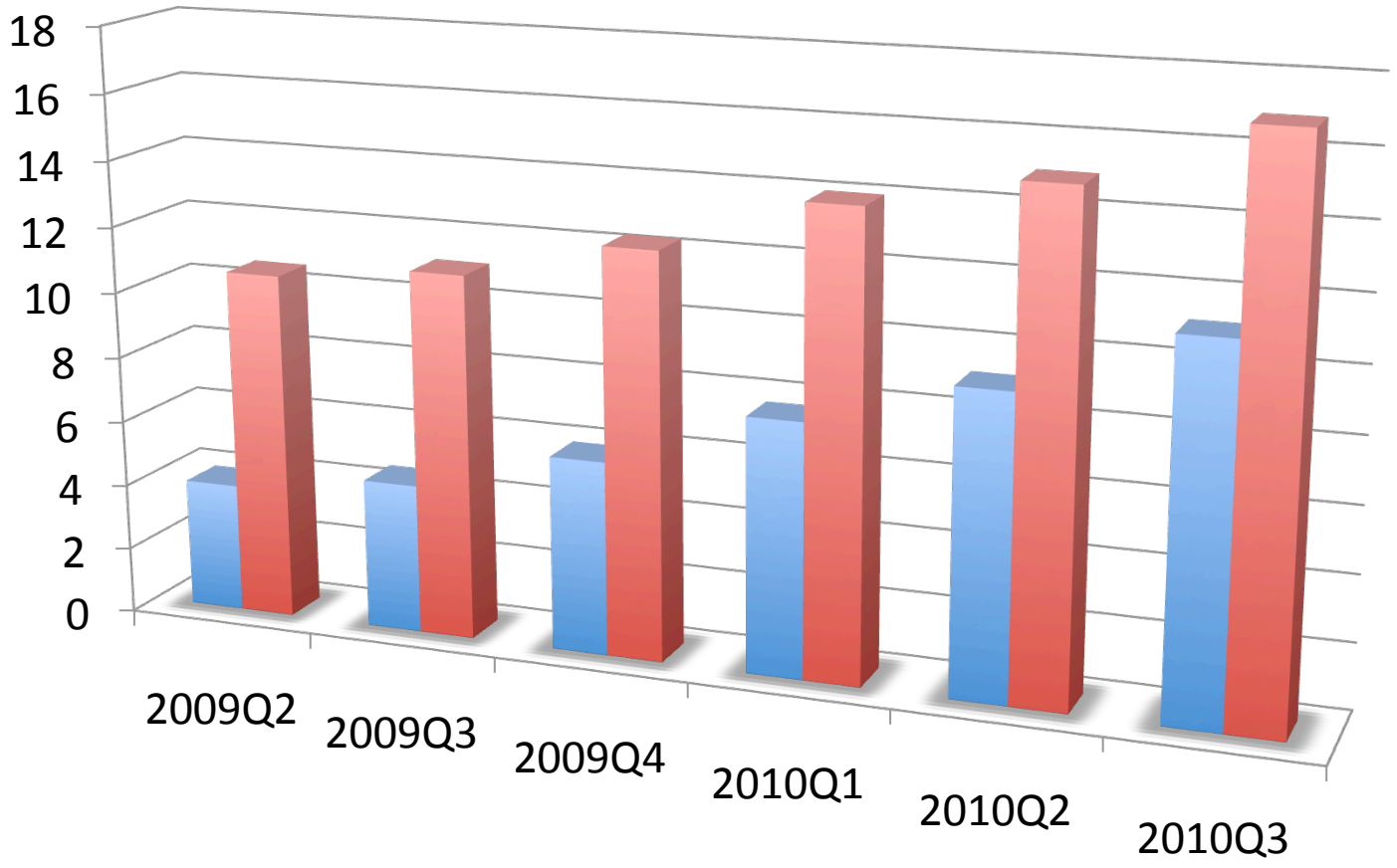
Capacity growth rate is accelerating, unpredictable

Product launch spikes - iPhone, Wii, PS3, XBox

Datacenter is large inflexible capital commitment

Customers

Q3 year/year +52% Total and +145% Streaming



Source: http://ir.netflix.com

NETFLIX

Leverage AWS Scale
“the biggest public cloud”

AWS investment in tooling and automation

AWS zones for high availability, scalability

AWS skills are common on resumes…

Leverage AWS Feature Set
“two years ahead of the others”

EC2, S3, SDB, SQS, EBS, EMR, ELB, ASG, IAM, RDB…

"The cloud lets its users focus on delivering differentiating business value instead of wasting valuable resources on the ***undifferentiated heavy lifting*** *that makes up most of IT infrastructure."*

Werner Vogels

Amazon CTO

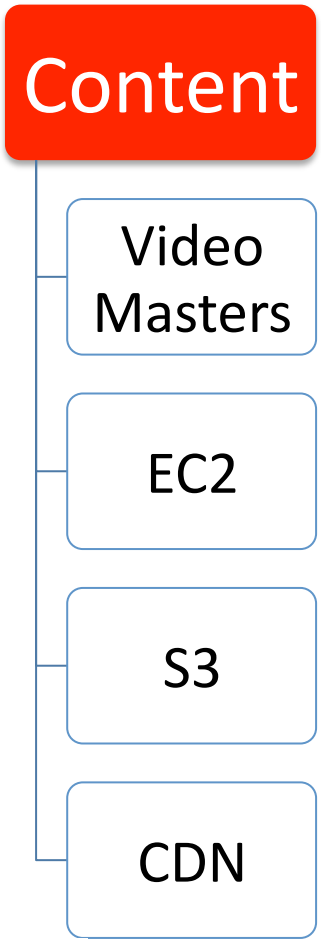
amazon web services™

NETFLIX

Netflix Deployed on AWS

Content	Logs	Play	WWW	API
Video Masters	S3	DRM	Search	Metadata
EC2	EMR Hadoop	CDN routing	Movie Choosing	Device Config
S3	Hive	Bookmarks	Ratings	TV Movie Choosing
CDN	Business Intelligence	Logging	Similars	Mobile iPhone

Movie Encoding farm (2009)

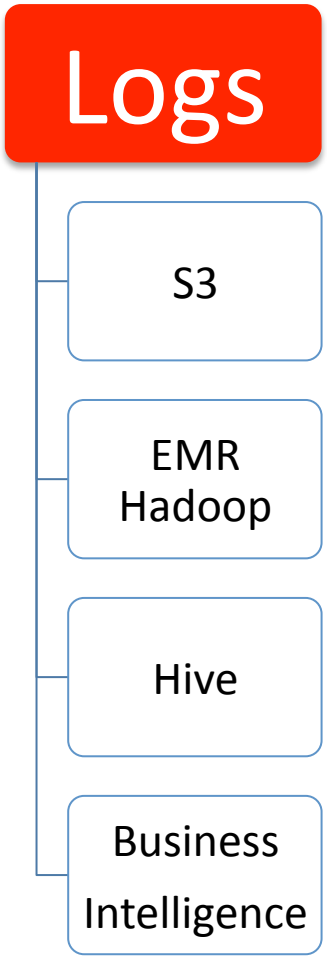


- Tens of thousands of videos
- Thousands of EC2 instances
- Encoding apps on MS Windows
- ~100 speed/format permutations
- Petabytes of S3
- Content Delivery Networks

"Netflix is one of the largest customers of the biggest CDNs Akamai and Limelight"

Akamai Limelight NETWORKS NETFLIX

Hadoop - Elastic Map-Reduce (2009)

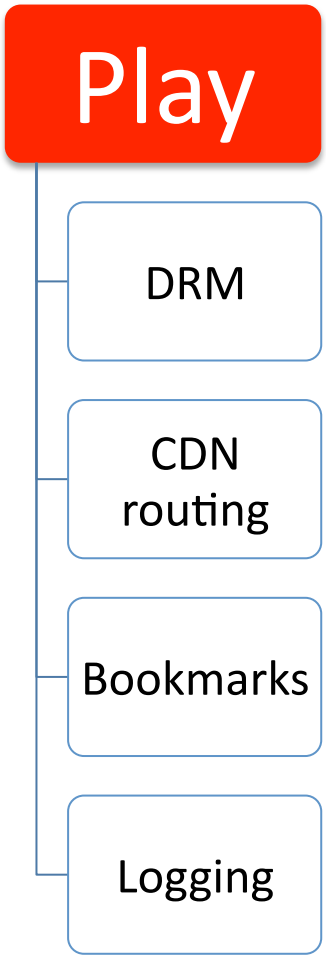


- Web Access Logs
- Streaming Service Logs
- Terabyte per day scale
- Easy Hadoop via Amazon EMR
- Hive SQL “Data Mart”
- Gateway to Datacenter BI

Slideshare.net talks
evamtse “Netflix: Hive User Group” http://slidesha.re/aqJLAC
adrianco “Crunch Your Data In The Cloud” http://slidesha.re/dx4oCK

NETFLIX

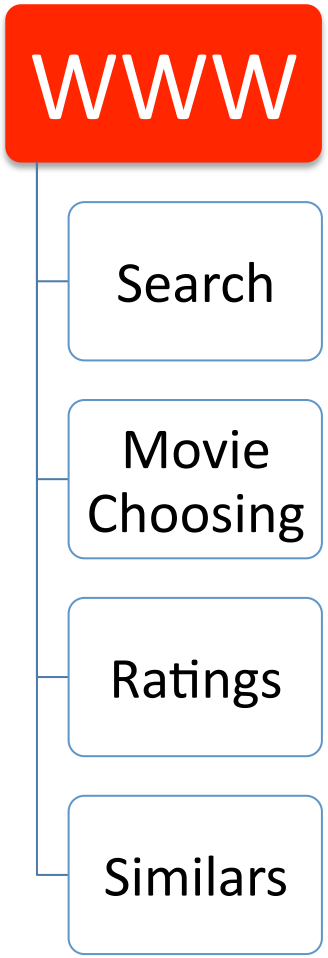
Streaming Service Back-end (early 2010)



- PC/Mac Silverlight Player Support
- Highly available “play button”
- DRM Key Management
- Generate route to stream on CDN
- Lookup bookmark for user/movie
- Update bookmark for user/movie
- Log quality of service

NETFLIX

Web site, a page at a time (through 2010)

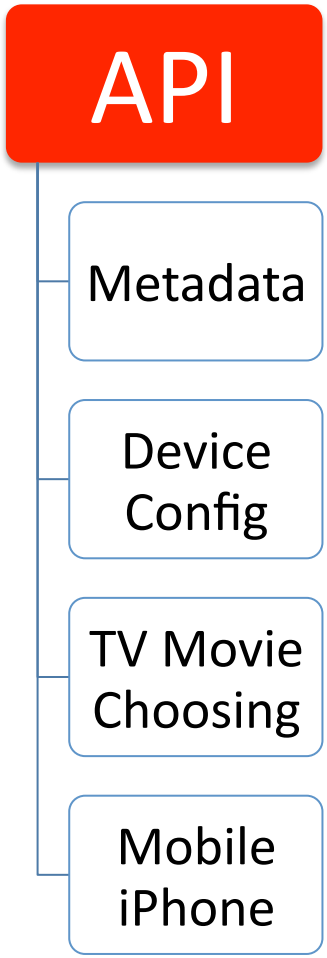


- Clean presentation layer rewrite
- Search auto-complete
- Search backend and landing page
- Movie and genre choosing
- Star ratings and recommendations
- Similar movies
- Page by page to 80% of views

(leave account signup in Datacenter for now)

NETFLIX

API for TV devices and iPhone etc. (2010)



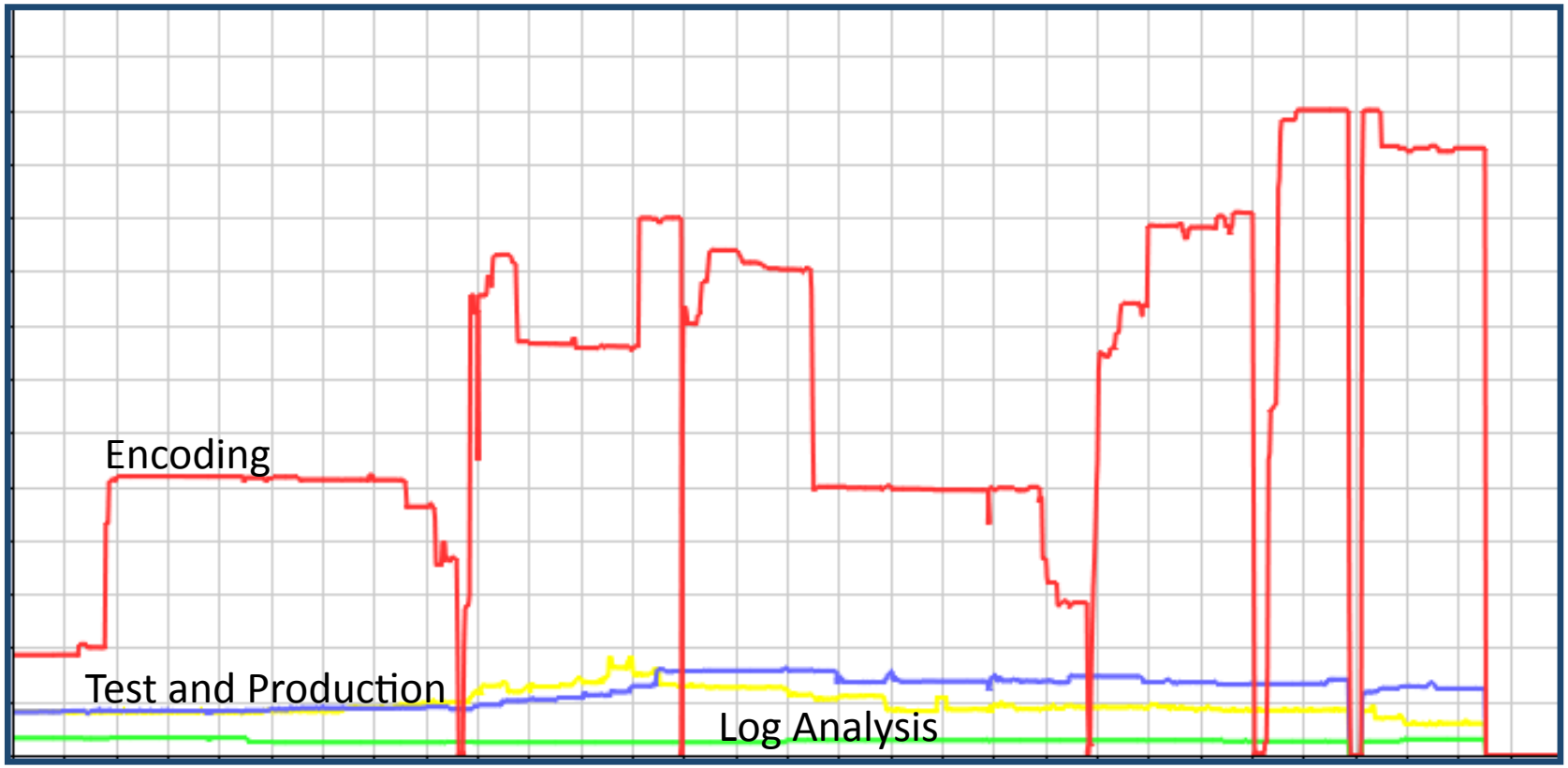
- REST API: developer.netflix.com
- Interfaces to everything else
- TV Device Configuration
- Personalized movie choosing
- iPhone Launch in the cloud only

"Netflix is an API for streaming to TVs

(we also do DVD's and a web site)"

NETFLIX

Netflix EC2 Instances per Account



NETFLIX

Learnings…

Datacenter oriented tools don’t work

Ephemeral instances

High rate of change

Cloud Tools Don’t Scale for Enterprise

Too many are “Startup” oriented

Built our own tools

Drove vendors hard

“fork-lifted” apps don’t work well

Fragile

Too many datacenter oriented assumptions

Faster to re-code from scratch

- Opportunity to pay down technical debt
- Re-architected and re-wrote most of the code
- Fine grain web services
- Leveraged many open source Java projects
- Systematically instrumented
- “NoSQL” SimpleDB backend

NETFLIX

“In the datacenter, robust code is best practice. In the cloud, it’s essential.”

Takeaway

Netflix is path-finding the use of public AWS cloud to replace in-house IT for non-trivial applications with hundreds of developers and thousands of systems.

(Pause for questions before we dive into details)

amazon web services™

NETFLIX

What, Why and How?

The details…

Synopsis

- The Goals
  - Faster, Scalable, Available and Productive
- Anti-patterns and Cloud Architecture
  - The things we wanted to change and why
- Cloud Bring-up Strategy
  - Developer Transitions and Tools
- Roadmap and Next Steps

Goals

- Faster
  - **Lower latency** than the equivalent datacenter web pages and API calls
  - Measured as mean and 99th percentile
  - For both first hit (e.g. home page) and in-session hits for the same user
- Scalable
  - **Avoid needing any more datacenter capacity** as subscriber count increases
  - No central vertically scaled databases
  - Leverage AWS elastic capacity effectively
- Available
  - Substantially **higher robustness and availability** than datacenter services
  - Leverage multiple AWS availability zones
  - No scheduled down time, no central database schema to change
- Productive
  - Optimize **agility** of a large development team with automation and tools
  - Leave behind complex tangled datacenter code base (~8 year old architecture)
  - Enforce clean layered interfaces and re-usable components

NETFLIX

Cloud Architecture Patterns

Where do we start?

Datacenter Anti-Patterns

What do we currently do in the datacenter that prevents us from meeting our goals?

Architecture

- Software Architecture
  - The abstractions and interfaces that developers build against
- Systems Architecture
  - The service instances that define availability, scalability
- Compose-ability
  - software architecture that is independent of the systems architecture
  - decoupled flexible building block components

NETFLIX

Rewrite from Scratch

Not everything is cloud specific

Pay down technical debt

Robust patterns

Old Datacenter vs. New Cloud Arch

Central SQL Database	Distributed Key/Value NoSQL
Sticky In-Memory Session	Shared Memcached Session
Chatty Protocols	Latency Tolerant Protocols
Tangled Service Interfaces	Layered Service Interfaces
Instrumented Code	Instrumented Service Patterns
Fat Complex Objects	Lightweight Serializable Objects
Components as Jar Files	Components as Services

The Central SQL Database

- Datacenter has a central database
  - Everything in one place is convenient until it fails
  - Customers, movies, history, configuration
- Schema changes require downtime

Anti-pattern impacts scalability, availability

NETFLIX

The Distributed Key-Value Store

- Cloud has many key-value data stores
  - More complex to keep track of, do backups etc.
  - Each store is much simpler to administer
  - Joins take place in java code

- No schema to change, no scheduled downtime
- Latency for Memcached vs. Oracle vs. SimpleDB
  - Memcached is dominated by network latency <1ms
  - Oracle for simple queries is a few milliseconds
  - SimpleDB has replication and REST overheads >10ms

NETFLIX

The Sticky Session

- Datacenter Sticky Load Balancing
  - Efficient caching for low latency
  - Tricky session handling code
  - Middle tier load balancer has issues in practice
- Encourages concentrated functionality
  - one service that does everything

Anti-pattern impacts productivity, availability

The Shared Session

- Cloud Uses Round-Robin Load Balancing
  - Simple request-based code
  - External shared caching with memcached
- More flexible fine grain services
  - Works better with auto-scaled instance counts

NETFLIX

Chatty Opaque and Brittle Protocols

- Datacenter service protocols
  - Assumed low latency for many simple requests
- Based on serializing existing java objects
  - Inefficient formats
  - Incompatible when definitions change

Anti-pattern causes productivity, latency and availability issues

NETFLIX

Robust and Flexible Protocols

- Cloud service protocols
  - JSR311/Jersey is used for REST/HTTP service calls
  - Custom client code includes service discovery
  - Support complex data types in a single request
- Apache Avro
  - Evolved from Protocol Buffers and Thrift
  - Includes JSON header defining key/value protocol
  - Avro serialization is **half the size** and several times faster than Java serialization, more work to code

Persisted Protocols

- Persist Avro in Memcached
  - Save space/latency (zigzag encoding, half the size)
  - Less brittle across versions
  - New keys are ignored
  - Missing keys are handled cleanly
- Avro protocol definitions
  - Can be written in JSON or generated from POJOs
  - It's hard, needs better tooling

NETFLIX

Tangled Service Interfaces

- Datacenter implementation is exposed
  - Oracle SQL queries mixed into business logic
- Tangled code
  - Deep dependencies, false sharing
- Data providers with sideways dependencies
  - Everything depends on everything else

Anti-pattern affects productivity, availability

Untangled Service Interfaces

- New Cloud Code With Strict Layering
  - Compile against interface jar
  - Can use spring runtime binding to enforce
- Service interface **is** the service
  - Implementation is completely hidden
  - Can be implemented locally or remotely
  - Implementation can evolve independently

NETFLIX

Untangled Service Interfaces

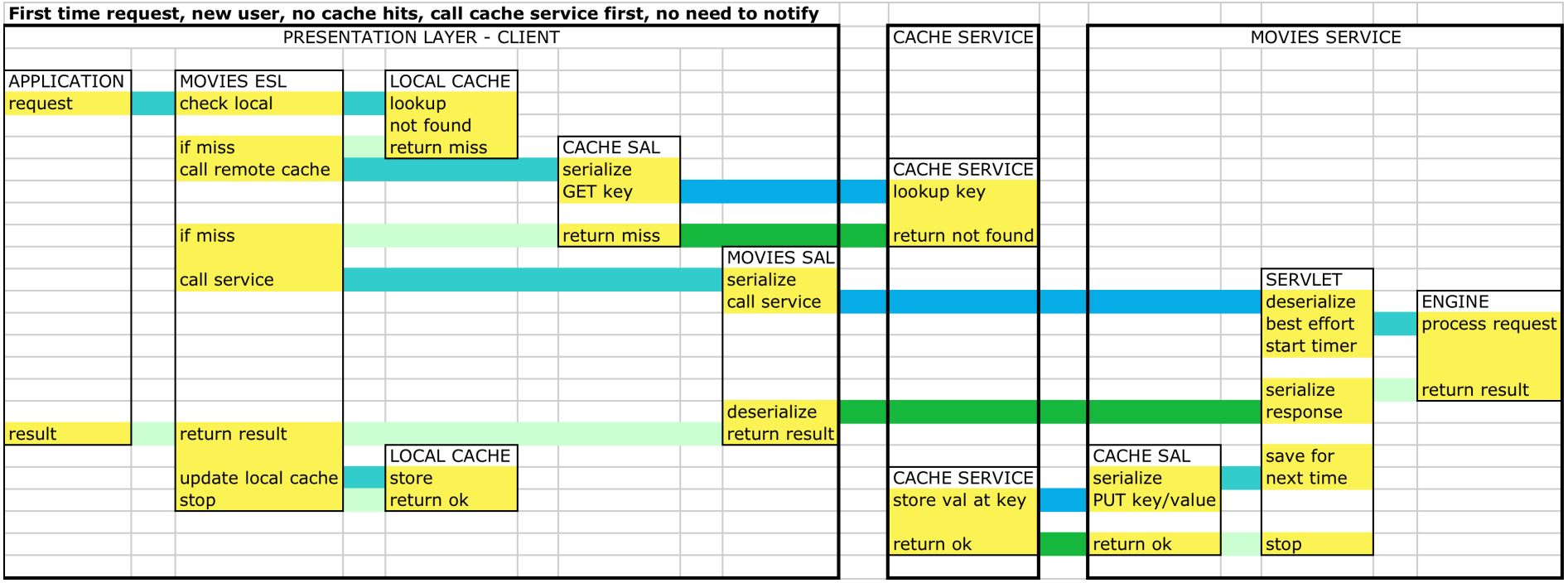
Two layers:

- SAL - Service Access Library
  - Basic serialization and error handling
  - REST or POJO's defined by data provider
- ESL - Extended Service Library
  - Caching, conveniences
  - Can combine several SALs
  - Exposes faceted type system (described later)
  - Interface defined by data consumer in many cases

NETFLIX

Service Interaction Pattern

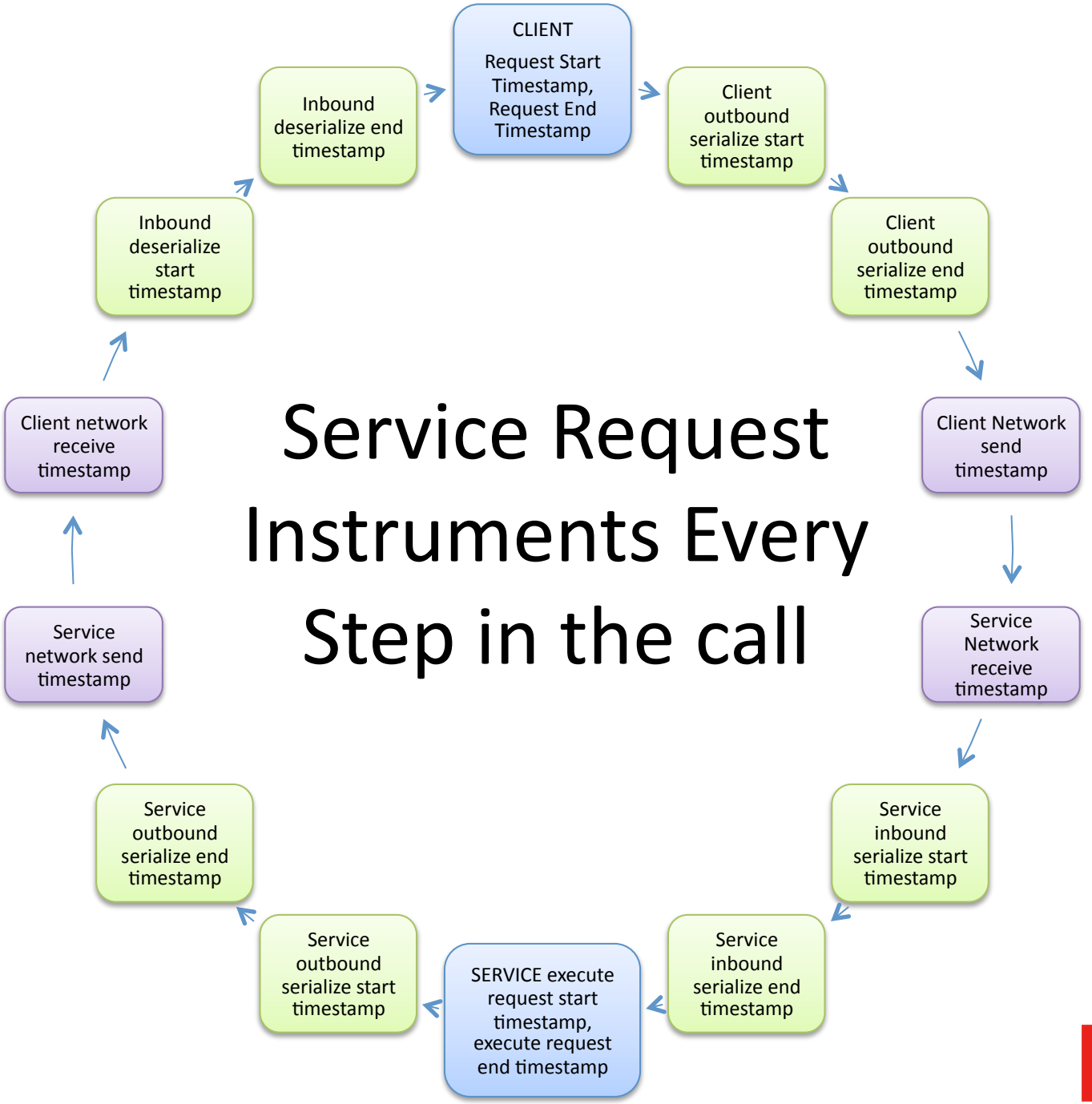
Sample Swimlane Diagram



Service Architecture Patterns

- Internal Interfaces Between Services
  - Common patterns as templates
  - Highly instrumented, observable, analytics
  - Service Level Agreements – SLAs
- Library templates for generic features
  - Instrumented Netflix Base Servlet template
  - Instrumented generic client interface template
  - Instrumented S3, SimpleDB, Memcached clients

NETFLIX


CLIENT
Request Start Timestamp, Request End Timestamp
Client outbound serialize start timestamp
Client outbound serialize end timestamp
Client Network send timestamp
Service Network receive timestamp
Service inbound serialize start timestamp
Service inbound serialize end timestamp
SERVICE execute request start timestamp, execute request end timestamp
Service outbound serialize start timestamp
Service outbound serialize end timestamp
Service network send timestamp
Client network receive timestamp
Inbound deserialize start timestamp
Inbound deserialize end timestamp
Service Request Instruments Every Step in the call

NETFLIX

Boundary Interfaces

- Isolate teams from external dependencies
  - Fake SAL built by cloud team
  - Real SAL provided by data provider team later
  - ESL built by cloud team using faceted objects
- Fake data sources allow development to start
  - e.g. Fake Identity SAL for a test set of customers
  - Development solidifies dependencies early
  - Helps external team provide the right interface

NETFLIX

One Object That Does Everything

- Datacenter uses a few big complex objects
  - Movie and Customer objects are the foundation
  - Good choice for a small team and one instance
  - Problematic for large teams and many instances
- False sharing causes tangled dependencies
  - Unproductive re-integration work

Anti-pattern impacting productivity and availability

NETFLIX

An Interface For Each Component

- Cloud uses faceted Video and Visitor
  - Basic types hold only the identifier
  - Facets scope the interface you actually need
  - Each component can define its own facets
- No false-sharing and dependency chains
  - Type manager converts between facets as needed
  - video.asA(PresentationVideo) for www
  - video.asA(MerchableVideo) for middle tier

Software Architecture Patterns

- Object Models
  - Basic and derived types, facets, serializable
  - Pass by reference within a service
  - Pass by value between services
- Computation and I/O Models
  - Service Execution using Best Effort
  - Common thread pool management

Netflix Systems Architecture

API

AWS EC2

Front End ELB

Discovery Service

API Proxy

API etc.

API ELB

Component Services

API

SQS

memcached

memcached

Replication

Oracle

EBS

S3

SimpleDB

AWS Storage

Netflix Data Center

NETFLIX

Netflix Undifferentiated Lifting

- Middle Tier Load Balancing
- Discovery (local DNS)
- Encryption Services
- Caching
- Distributed App Management

We want cloud vendors to do all this for us as well!

NETFLIX

Load Balancing in AWS

- Middle tier currently not supported in AWS
  - ELB are public-facing only
  - Cannot apply security group settings
- ELB vertical scalability for concentrated clients
  - Too few proxy IP addresses leads to hot spots
- ELB needs support for balancing heuristics
  - Proportional balance across Availability Zones
  - Weighted Least connections, Weighted Round Robin
- Zone aware routing
  - Default to instances in the same Availability Zone
  - Falls back to cross-zone on failure

Discovery

- Discovery Service (Redundant instances per zone)
  - Simple REST interface
  - Cloud apps register with Discovery
- Apps send heartbeats every 30 sec to renew lease
  - App evicted after 3 missed heartbeats
  - Can re-register if the problem was transient
- Apps can store custom metadata
  - Version number, AMI id, Availability Zone, etc.
- Software Round-robin Load Balancer
  - Query Discovery for instances of specific application
  - Baked into Netflix REST client (JSR311/Jersey based)

AWS Middle-tier ELB would eliminate most use cases

NETFLIX

Database Migration

- Why SimpleDB?
  - No DBA's in the cloud, Amazon hosted service
  - Work started two years ago, fewer viable options
  - Worked with Amazon to speed up and scale SimpleDB
- Alternatives?
  - Investigating adding Cassandra and Membase to the mix
  - Need several options to match use cases well
- Detailed SimpleDB Advice
  - Sid Anand - QConSF Nov 5th – Netflix' Transition to High Availability Storage Systems
  - Blog - http://practicalcloudcomputing.com/
  - Download Paper PDF - http://bit.ly/bhOTLu

NETFLIX

Oracle to SimpleDB

(See Sid's paper for details)

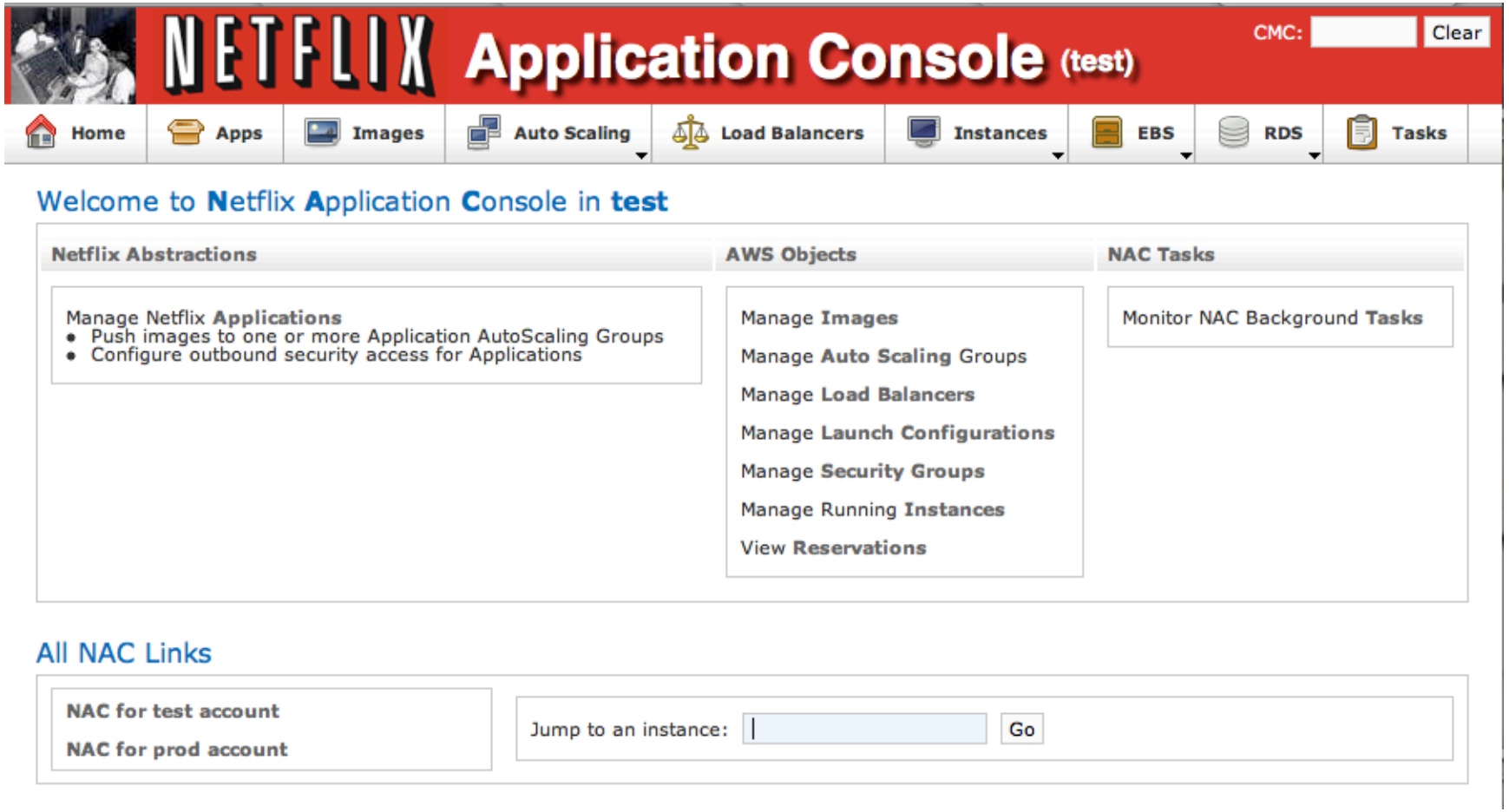
- SimpleDB Domains
  - De-normalize multiple tables into a single domain
  - Work around size limits (10GB per domain, 1KB per key)
  - Shard data across domains to scale
  - Key – Use distributed sequence generator, GUID or natural unique key such as customer-id
  - Implement a schema validator to catch bad attributes
- Application layer support
  - Do GROUP BY and JOIN operations in the application
  - Compose relations in the application layer
  - Check constraints on read, and repair data as a side effect
- Do without triggers, PL/SQL, clock operations

Tools and Automation

- Developer and Build Tools
  - Jira, Eclipse, Hudson, Ivy, Artifactory
  - Builds, creates .war file, .rpm, bakes AMI and launches
- Custom Netflix Application Console
  - AWS Features at Enterprise Scale (hide the keys!)
  - Auto Scaler Group is unit of deployment to production
- Open Source + Support
  - Apache, Tomcat, OpenJDK, CentOS
- Monitoring Tools
  - Keynote – service monitoring and alerting
  - AppDynamics – Developer focus for cloud
  - EpicNMS – flexible data collection and plots http://epicnms.com
  - Nimsoft NMS – ITOps focus for Datacenter + Cloud alerting

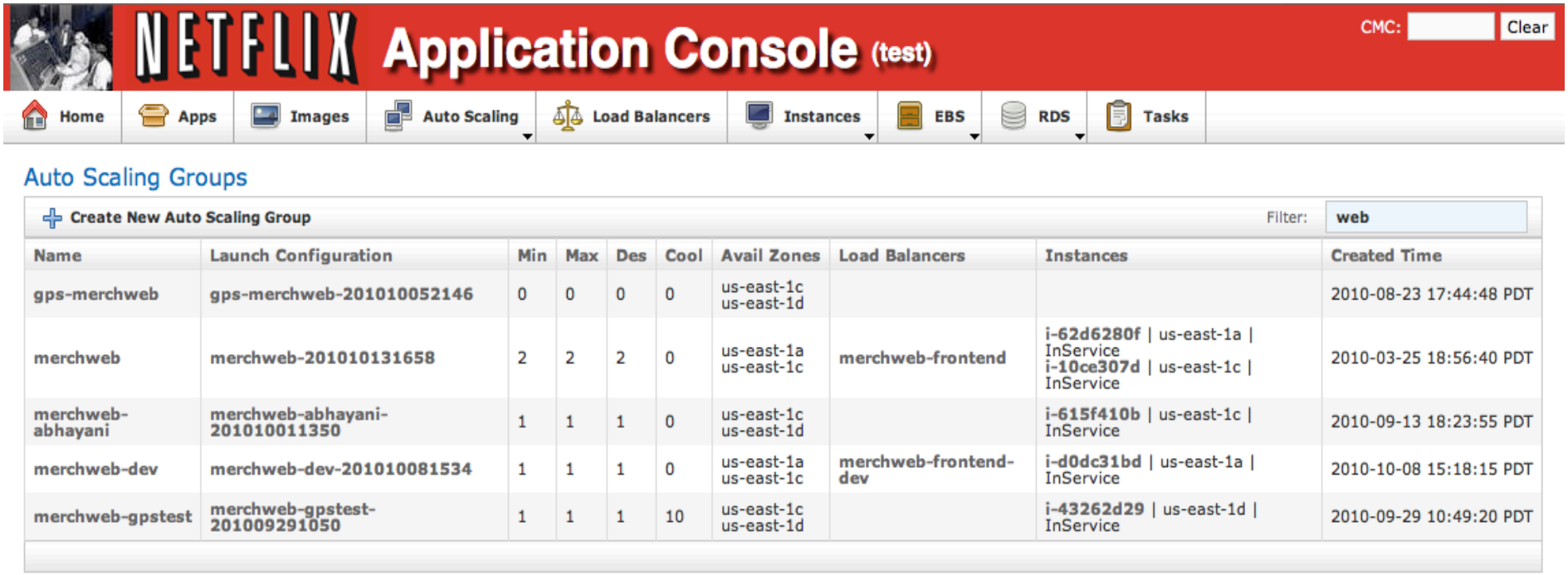
NETFLIX

Netflix App Console



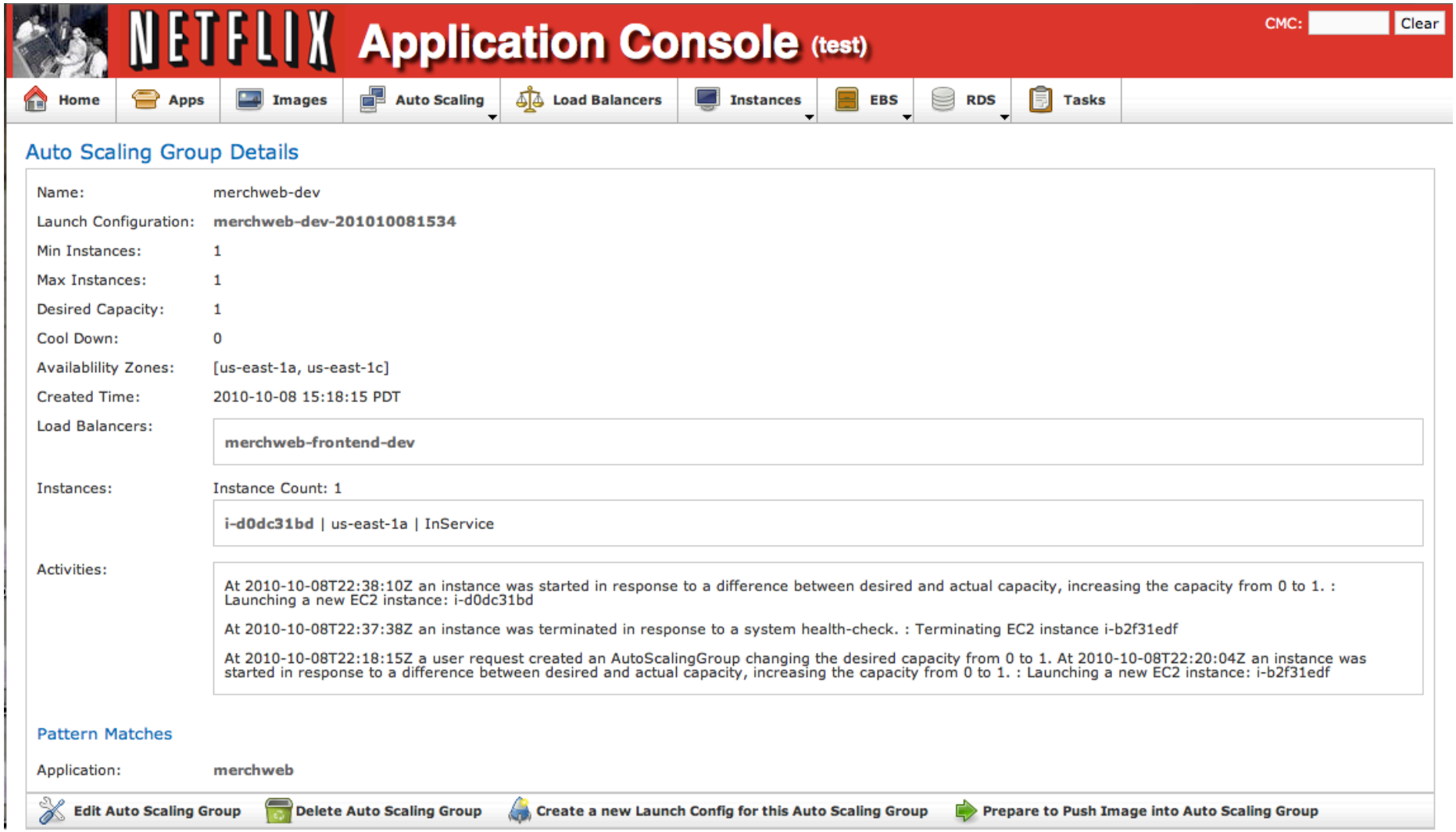
NETFLIX

Auto Scaling Groups



NETFLIX

ASG Configuration



NETFLIX

Monitoring Tools

Monitoring Vision

- Problem
  - Too many tools, each with a good reason to exist
  - Hard to get an integrated view of a problem
  - Too much manual work building dashboards
  - Tools are not discoverable, views are not filtered
- Solution
  - Get vendors to add deep linking and embedding
  - Integration “portal” ties everything together
  - Dynamic portal generation, relevant data, all tools

Cloud Monitoring Mechanisms

- Keynote
  - External URL monitoring
- Amazon CloudWatch
  - Metrics for ELB and Instances
- AppDynamics
  - End to end transaction view showing resources used
  - Powerful real time debug tools for latency, CPU and Memory
- Nimsoft NMS
  - Scalable and reliable monitoring and alerting, integration portal
- Epic
  - Flexible and easy to use to extend and embed plots
- Logs
  - High capacity logging and analysis framework
  - Hadoop (log4j -> chukwa -> EMR)

unified monitoring

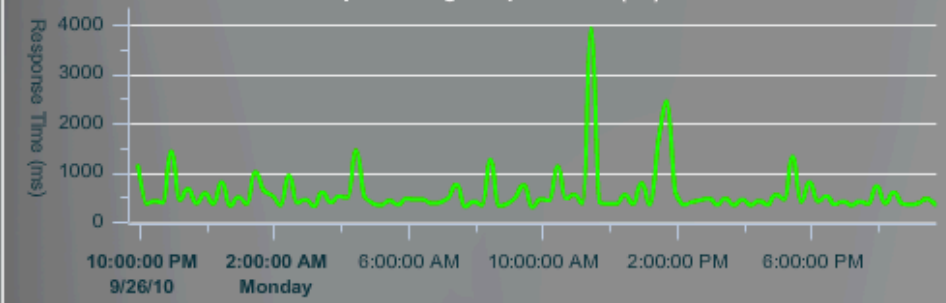
POWERED BY nimsoft

Learn More About Unified Monitoring

HEALTH STATUS DASHBOARD FOR SELECT REMOTE COMPUTING AND APPLICATION SERVICES

Website	Home Page Reponse (ms)	Home Page Response (ms)
Intacct		1110.00
Salesforce.com		483.00
Trinet HR Services		466.00
Rackspace Cloud		465.00
Google Apps Engine		272.00
Google Apps		214.00
Amazon Web Services		168.00

Top Home Page Response Time (ms)



Home Page Response - Sf.com (ms)

Google Apps
Overall Status
Response Time
Double Click for Details >>>

salesforce.com
Overall Status
Response Time
Double Click for Details >>>

SAAS SERVICES

the rackspace cloud
Overall Status
Response Time
Double Click for Details >>>

amazon web services
Overall Status
Response Time
Double Click for Details >>>

CLOUD/IAAS SERVICES

CISCO
Overall Status
Response Time
Double Click for Details >>>

nimsoft
Overall Status
Response Time
Double Click for Details >>>

vmware
Overall Status
Response Time
Double Click for Details >>>

NIMSOFT DATA CENTER - NETWORK, VOIP, VIRTUAL SERVERS

Nimsoft's Hosted MS Exchange
Overall Status
Response Time
Double Click for Details >>>

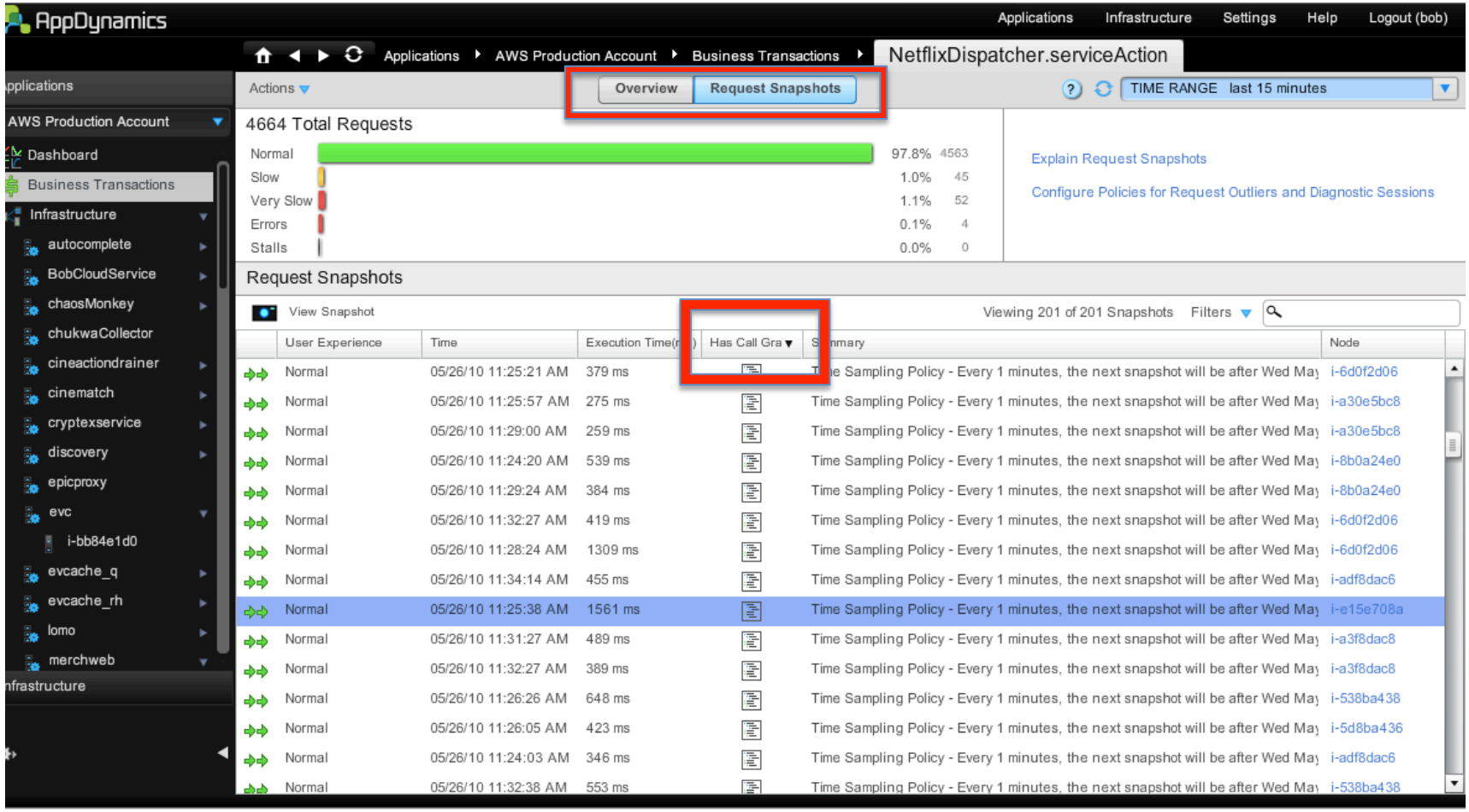
HOSTED SERVICES

Google App Engine
Overall Status
Response Time
Double Click for Details >>>

PAAS SERVICES

Snapshots for a Business Transaction

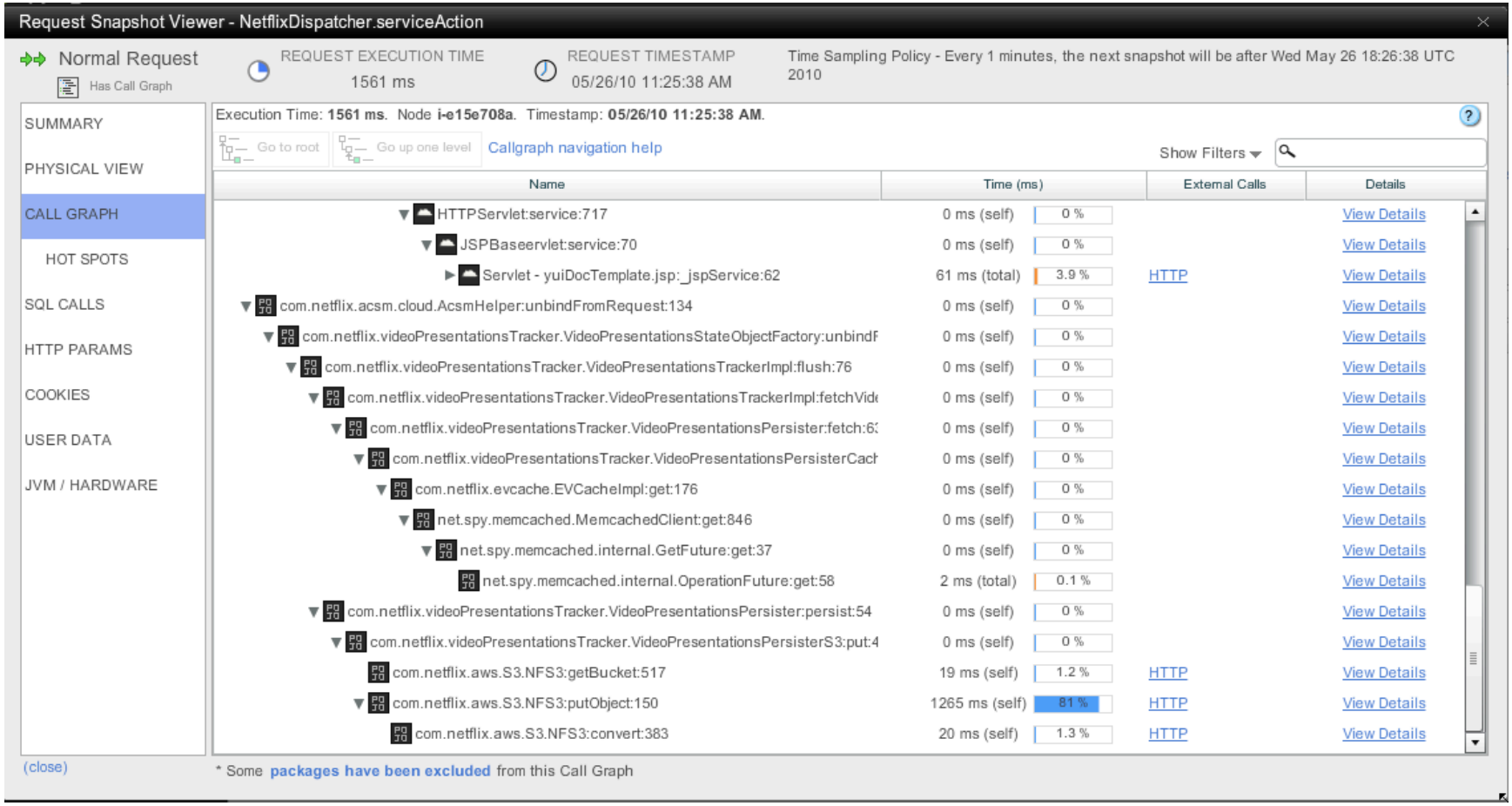
Sort Call Graphs to Top, pick a slow one



NETFLIX

Drill in to Slow Call

Slow Asynchronous S3 Write – no big deal…



NETFLIX

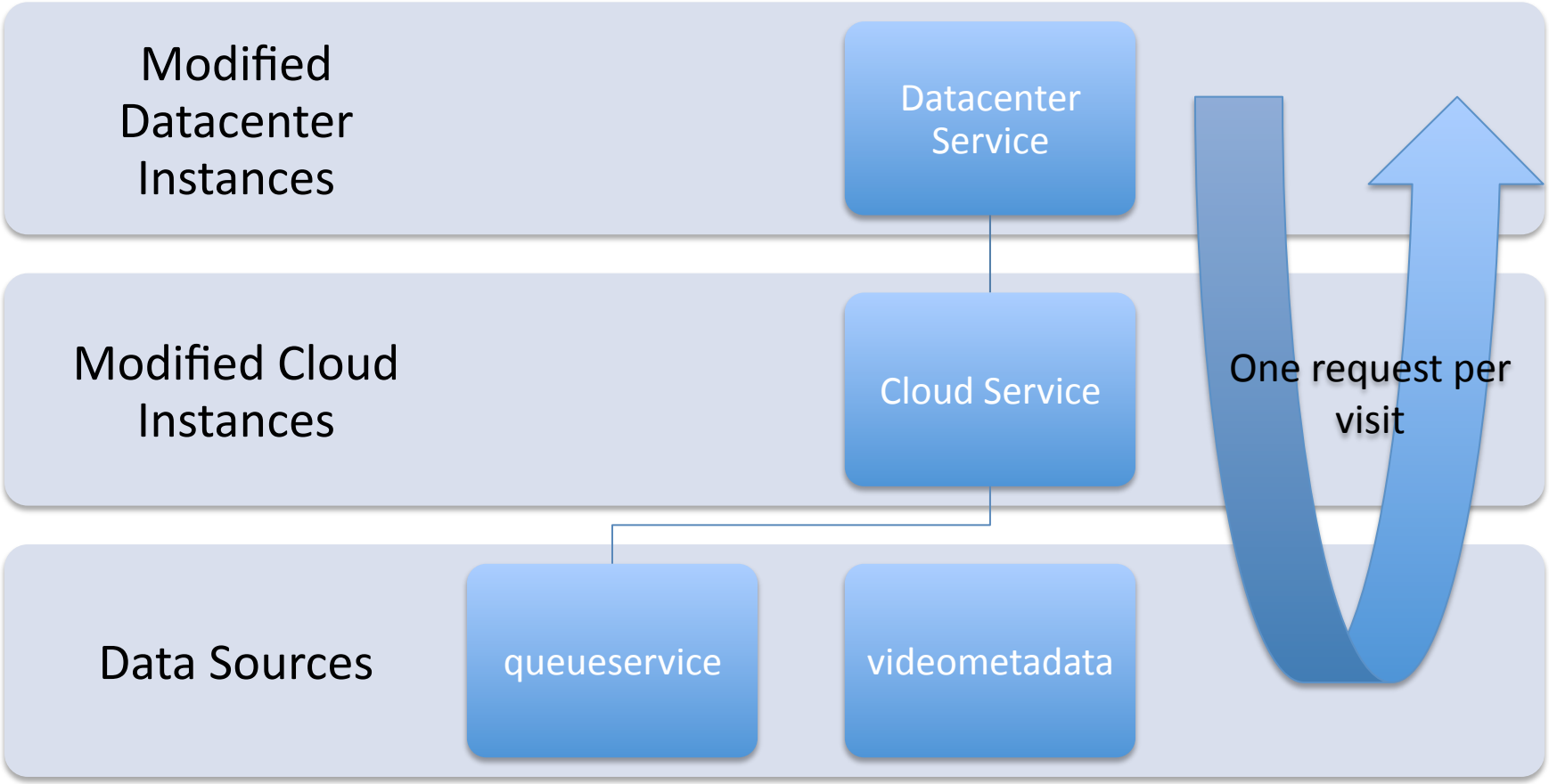
Cloud Bring-Up Strategy

Simplest and Soonest

Shadow Traffic Redirection

- Early attempt to send traffic to cloud
  - Real traffic stream to validate cloud back end
  - Uncovered lots of process and tools issues
  - Uncovered Service latency issues
- TV Device calls Datacenter API
  - Returns Genre/movie list for a customer
  - Asynchronously duplicates request to cloud
  - Start with send-and-forget mode, ignore response

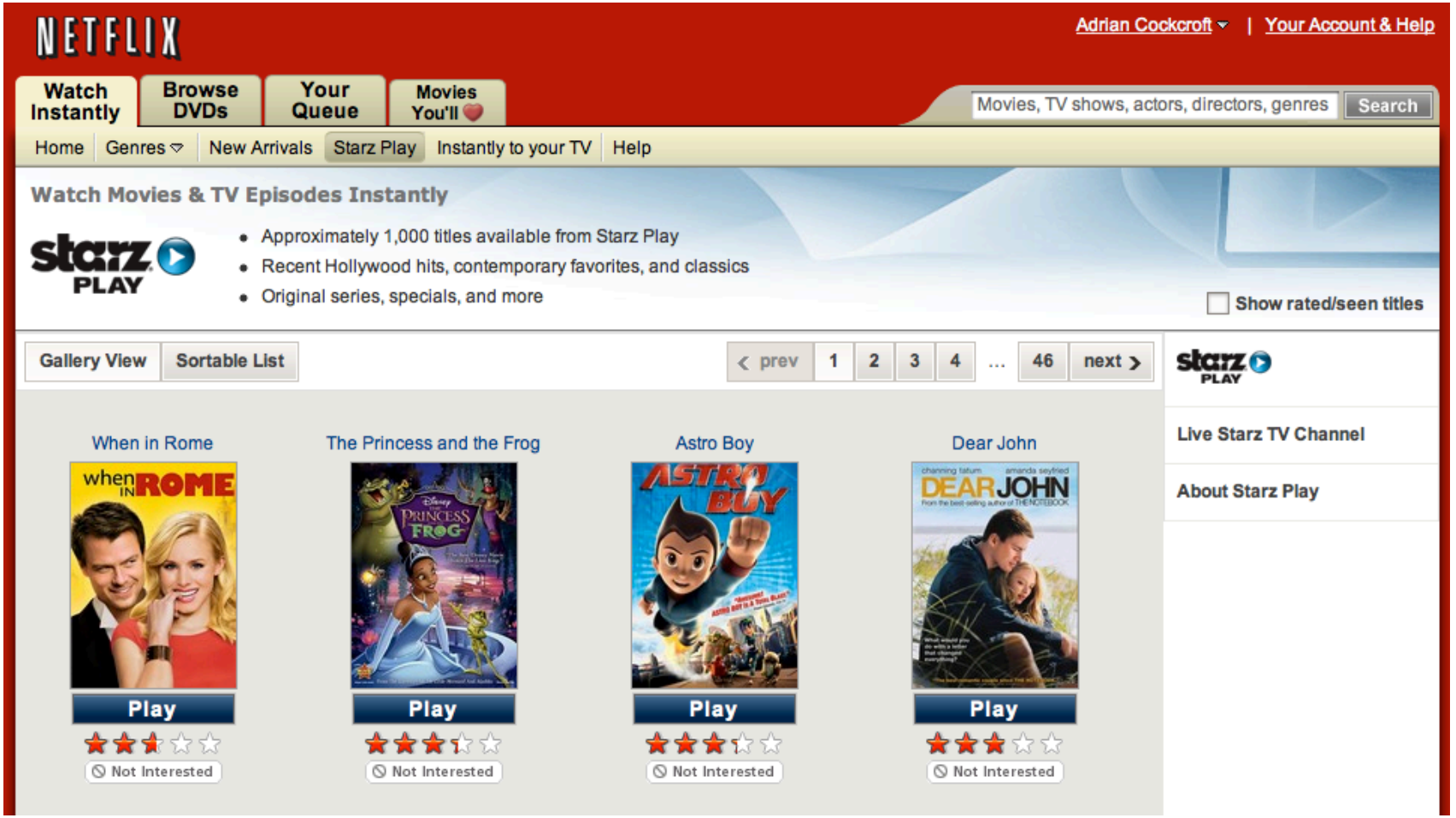
Shadow Redirect Instances



NETFLIX

First Web Pages in the Cloud

Starz Page

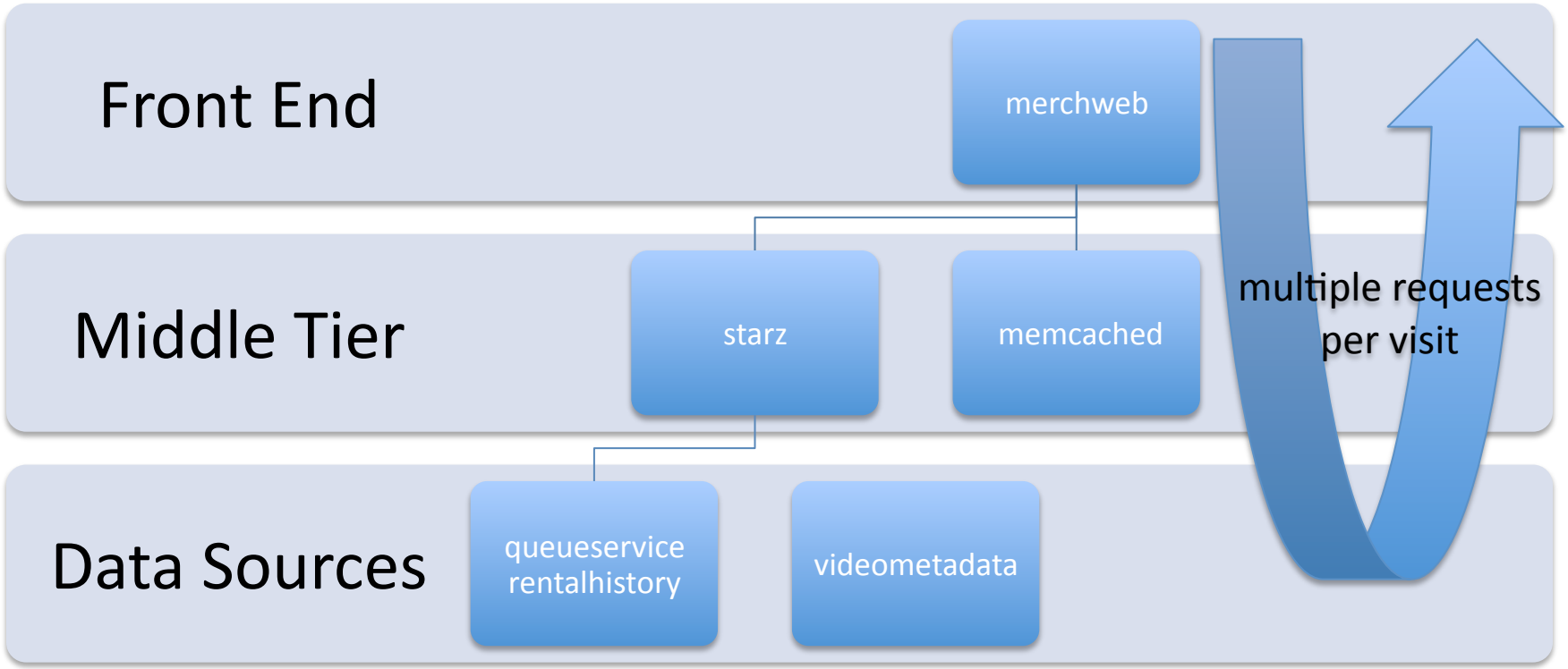

NETFLIX
Adrian Cockcroft | Your Account & Help
Watch Instantly
Browse DVDs
Your Queue
Movies You'll
Movies, TV shows, actors, directors, genres
Search
Home
Genres
New Arrivals
Starz Play
Instantly to your TV
Help
Watch Movies & TV Episodes Instantly
starz PLAY
Approximately 1,000 titles available from Starz Play
Recent Hollywood hits, contemporary favorites, and classics
Original series, specials, and more
Show rated/seen titles
Gallery View
Sortable List
< prev
1
2
3
4
...
46
next >
When in Rome
The Princess and the Frog
Astro Boy
Dear John
Play
Not Interested
starz PLAY
Live Starz TV Channel
About Starz Play

First Page

- First full page – Starz Channel Genre
  - Simplest page, no sub-genres, minimal personalization
  - Lots of investment in new Struts based page design
  - Uses identity cookie to lookup in member info svc
- New “merchweb” front end instance
  - movies.netflix.com points to merchweb instance
- Uncovered lots of latency issues
  - Used memcached to hide S3 and SimpleDB latency
  - Improved from slower to faster than Datacenter

NETFLIX

Starz Page Cloud Instances



NETFLIX

Controlled Cloud Transition

- WWW calling code chooses who goes to cloud
  - Filter out corner cases, send percentage of users
  - The URL that customers see is http://movies.netflix.com/WiContentPage?csid=1
  - If problem, redirect to old Datacenter page http://www.netflix.com/WiContentPage?csid=1
- Play Button and Star Rating Action redirect
  - Point URLs for actions that create/modify data back to datacenter to start with

NETFLIX

Developers

Cloud Developer Setup

- Cloud Boot Camp
  - Room full of engineers sharing the pain for 1-2 days
  - Built a very rough prototype working web site
  - Get everyone hands-on in the cloud with a new code base
  - Debug lots of tooling and conceptual issues very fast
  - Member info in SimpleDB with developer's accounts only
- Cloud Specific Key Setup
  - It's a pain, need to configure your IDE's JVM
  - Needed to integrate with AWS security model
- Startup Guide Wiki Pages
  - What object facets already exist, how to make your own
  - What components already exist or are work in progress

NETFLIX

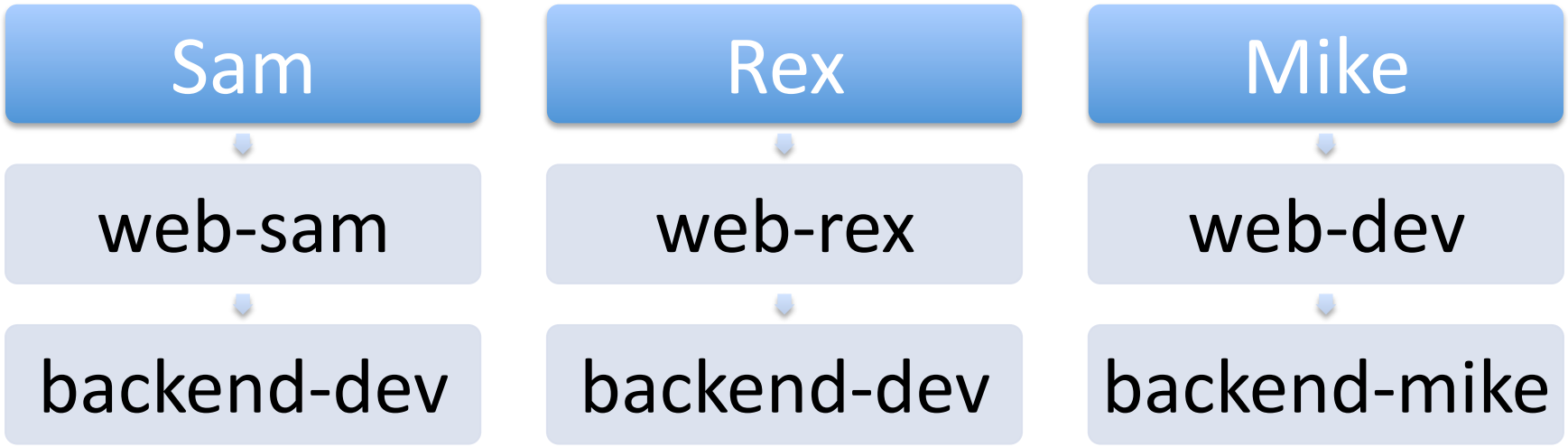
Developer Instances Collision

Sam and Rex both want to deploy web front end for development

NETFLIX

Per-Service Namespace Routing

Developers choose what to share



NETFLIX

Developer Service Namespaces

- Developer specific service instances
  - Configured via Java properties at runtime
  - Routing implemented by REST client library
- Server Configuration
  - Configure discovery service version string
  - Registers as <appname>-<namespace>
- Client Configuration
  - Route traffic on per-service basis including namespace

Current Status

WWW Page by Page during Q2/Q3/Q4

- Simplest possible page first
  - Minimal dependencies
- Add pages as dependent services come online
- Home page – most complex and highest traffic
- Leave low traffic pages for later cleanup

gradual migration from Datacenter pages

NETFLIX

Big-Bang Transition

- iPhone Launch (August/Sept)
  - No capacity in the datacenter, cloud only
  - App Store gates release, not gradual, can't back out
  - Market is huge (existing and new customers)
  - Has to work at large scale on day one
- Datacenter Shadow Redirect Technique
  - Used to stress back-end and data sources
- SOASTA Cloud Based Load Generation
  - Used to stress test API and end-to-end functionality

CloudTest® by SOASTA

NETFLIX

Current Work for Cloud Platform

- Drive latency and availability goals
  - More Aggressive caching
  - Improving Fault and latency robustness
- Logging and monitoring portal/dashboards
  - Working to integrate tools and data sources
  - Need better observability and automation
- Evaluating a range of NoSQL choices
  - Broad set of use cases, no single winner
  - Good topic for another talk…

NETFLIX

Wrap Up

Next Few Years…

- "System of Record" moves to Cloud
  - Master copies of data live only in the cloud, with backups etc.
  - Cut the datacenter to cloud replication link
- International Expansion – Global Clouds
  - Rapid deployments to new markets
- GPU Clouds optimized for video encoding
- Cloud Standardization
  - Cloud features and APIs should be a commodity not a differentiator
  - Differentiate on scale and quality of service
  - Competition also drives cost down
  - Higher resilience
  - Higher scalability

We would prefer to be an insignificant customer in a giant cloud

NETFLIX

Remember the Goals

Faster

Scalable

Available

Productive

Track progress against these goals

Takeaway

Netflix is path-finding the use of public AWS cloud to replace in-house IT for non-trivial applications with hundreds of developers and thousands of systems.

http://www.linkedin.com/in/adriancockcroft

@adrianco #netflixcloud

amazon web services

NETFLIX