eBay Architecture

Tony Ng

Director, Systems Architecture

October 2011

About Me

DRAFT

- eBay – Systems Architecture and Engineering
- Yahoo! – Social, Developer Platforms, YQL
- Sun Microsystems – J2EE, GlassFish, JSRs
- Author of books on J2EE, SOA

eBay Stats

DRAFT

- 94 million active users
- 200 million items for sale in 50,000 categories
- A cell phone is sold every 5 seconds in US
- An iPad sold every 2.2 minutes in US
- A pair of shoes sold every 9 seconds in US
- A passenger vehicle sold every 2 minutes
- A motorcycle sold every 6 minutes

http://www.ebayinc.com/factsheets

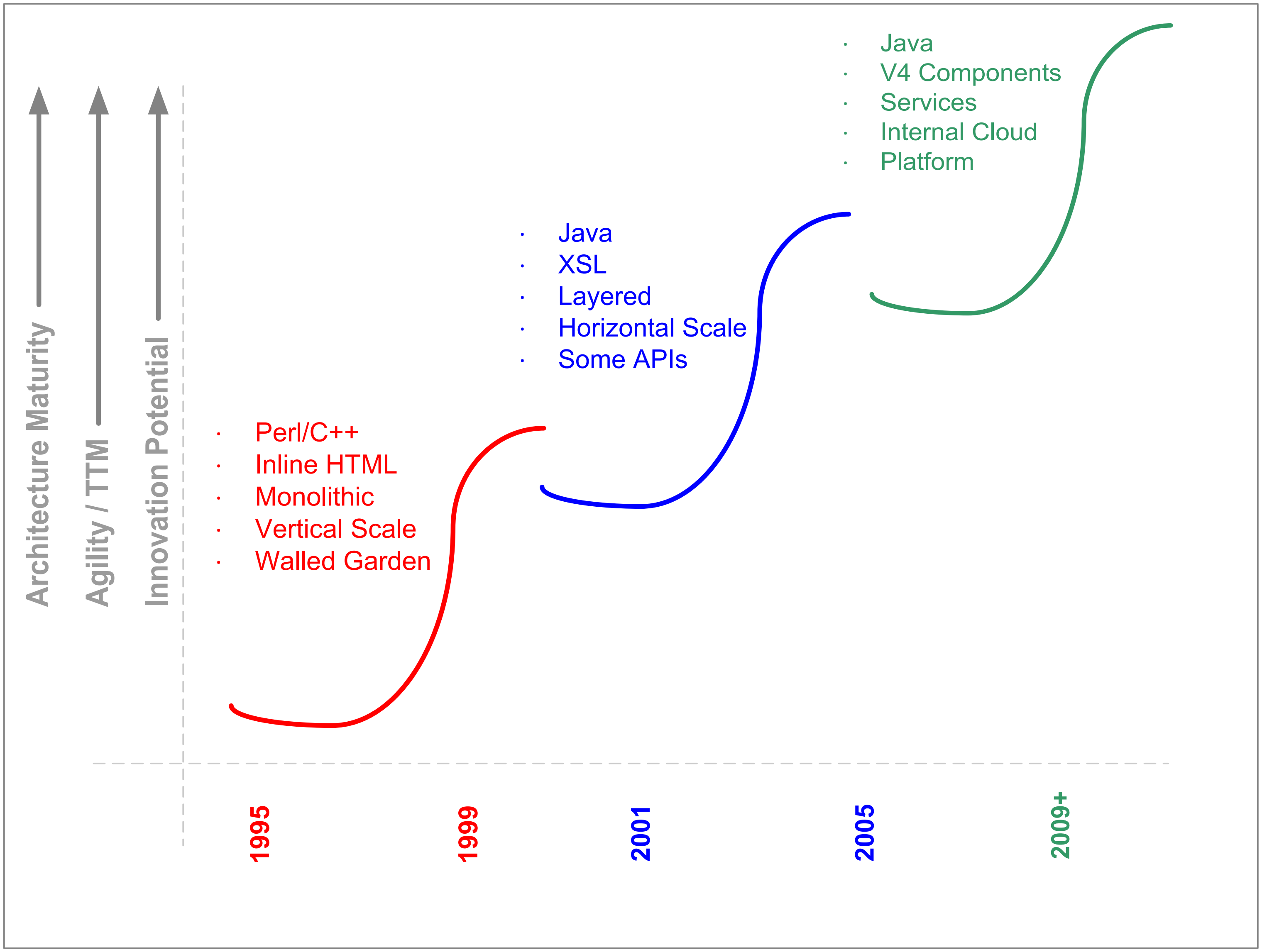
eBay Scale

DRAFT

- 9 Petabytes of data storage
- 10,000 application servers
- 44 million lines of code
- 2 billion pictures
- 99.94% site availability
- A typical day
  - 75B database calls
  - 4B page views
  - 250B search queries
  - Billions of service calls
  - 100s of millions of internal asynchronous events

History of Technology

DRAFT



Architecture Maturity

Agility / TTM

Innovation Potential

- Perl/C++
- Inline HTML
- Monolithic
- Vertical Scale
- Walled Garden

- Java
- XSL
- Layered
- Horizontal Scale
- Some APIs

- Java
- V4 Components
- Services
- Internal Cloud
- Platform

1995

1999

2001

2005

2009+

Qualities Attributes Concerns

DRAFT

- Scalability
- Availability
- Latency
- Security
- Manageability
- Cost

eBay Scalable Architecture

DRAFT

- Partition everything
  - Databases, application tier, search engine
- Stateless preference
  - No session state in app tier
- Asynchronous processing
  - Event streams, batch
- Manage failures
  - Central application logging
  - Mark downs

Next Challenges

DRAFT

- Maintain site stability but deliver quality features and innovations at accelerating paces
- Complexity as our codebase grows
- Build on our architecture maturity to enable faster time-to-market
- Developer productivity

Scalability with Agility

DRAFT

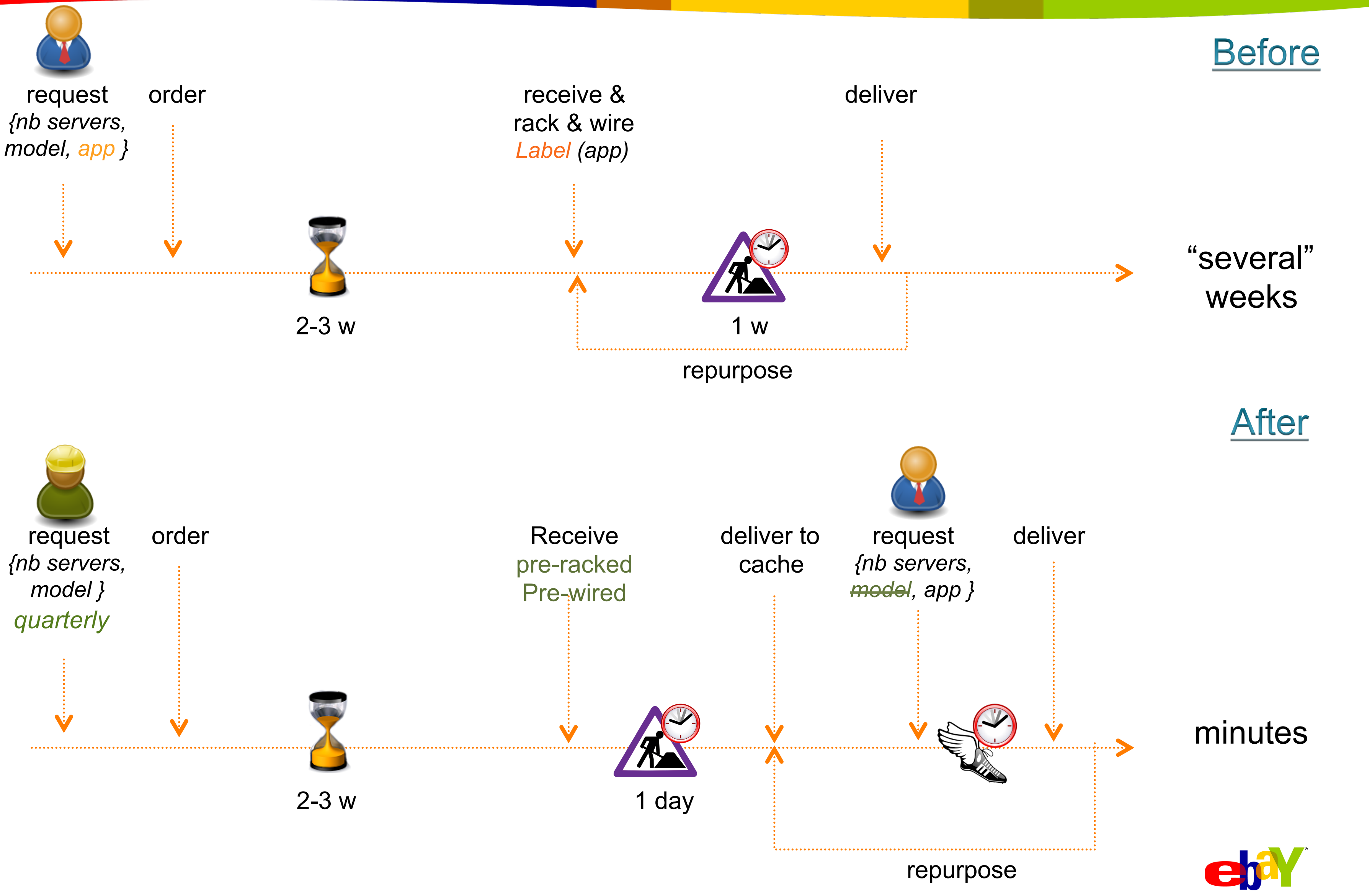
- Strategy 1: Automation with Cloud
- Strategy 2: Next Gen Service orientation
- Strategy 3: Modularity
- … and more …

DRAFT

Automation with Cloud

Improving Automation

DRAFT



Before

request
{nb servers, model, app }

order

receive &
rack & wire
Label (app)

deliver

2-3 w

1 w

repurpose

"several" weeks

After

request
{nb servers, model }
quarterly

order

Receive
pre-racked
Pre-wired

deliver to
cache

request
{nb servers, ~~model~~, app }

deliver

2-3 w

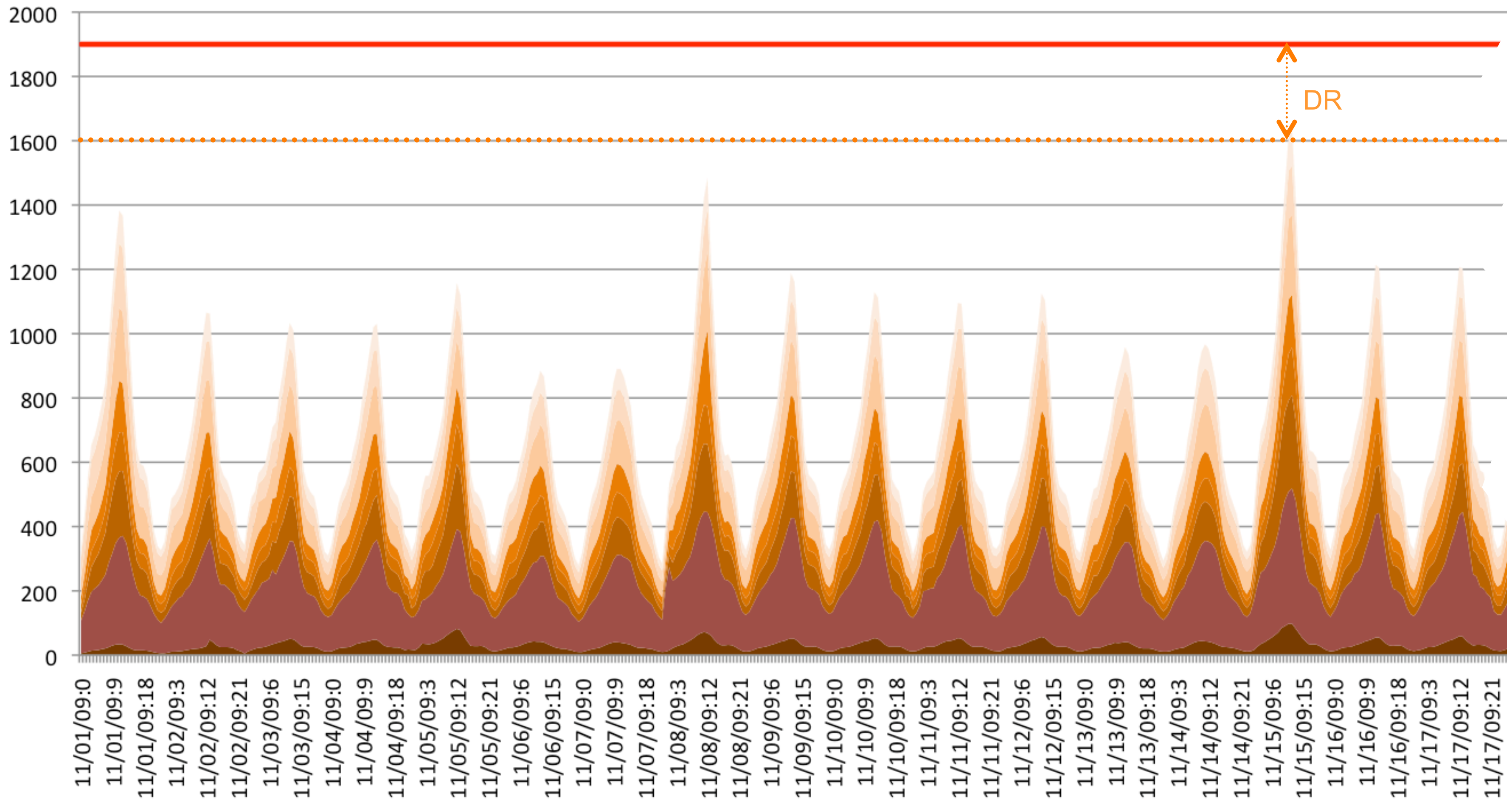
1 day

repurpose

minutes

Improving Utilization

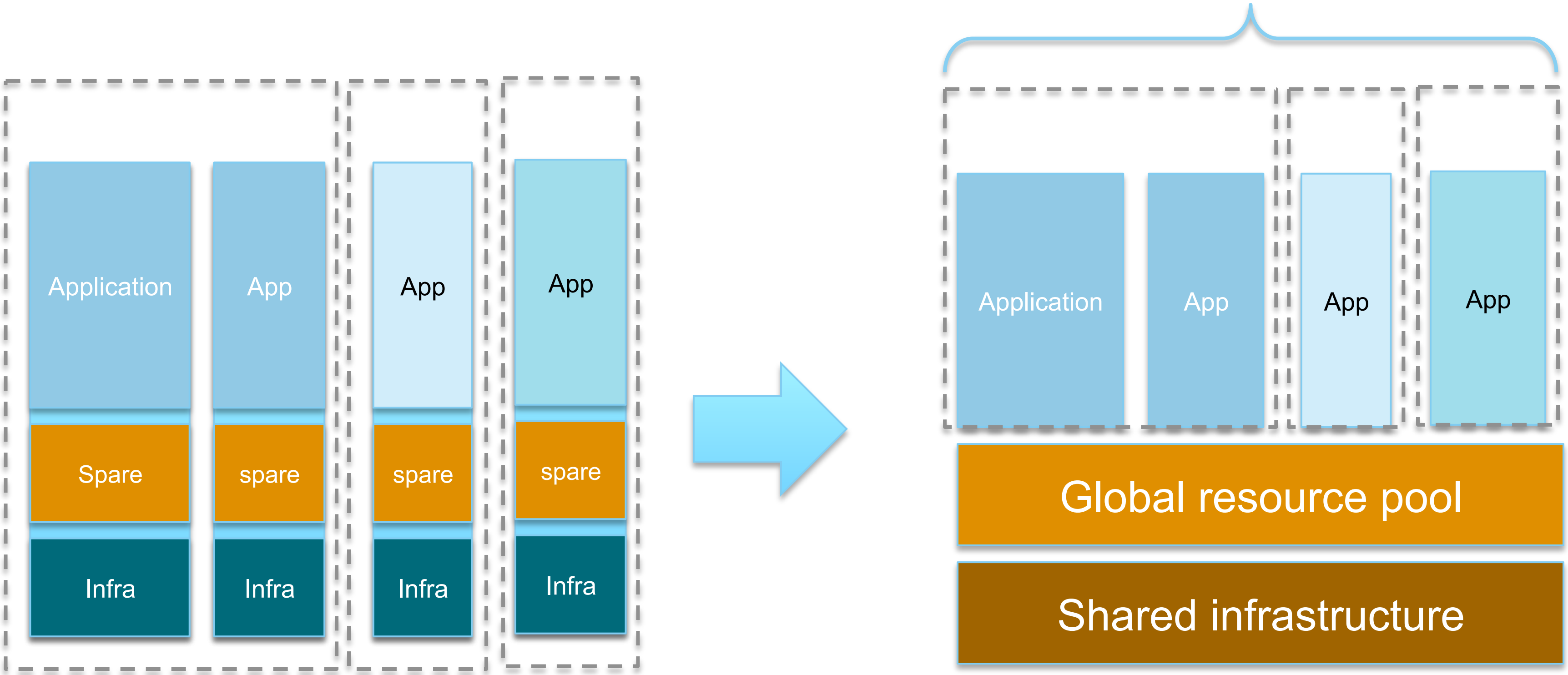
DRAFT



Number of servers required based on utilization for 8 pools

Infrastructure Virtualization

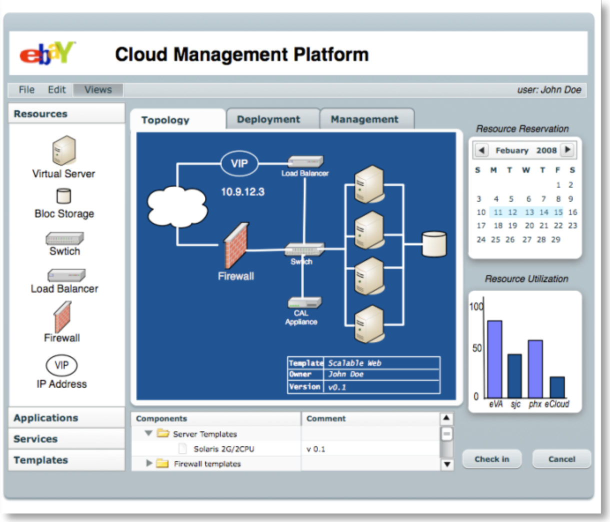
DRAFT



eBay Cloud

DRAFT

Self Service Portal



Cloud Management Platform

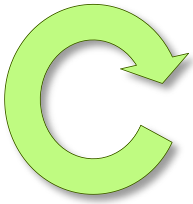
S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	

Template	Scalable Web
Owner	John Doe
Version	v0.1

Components	Comment
Server Templates	
Solaris 2G/2CPU	v 0.1
Firewall templates	

+

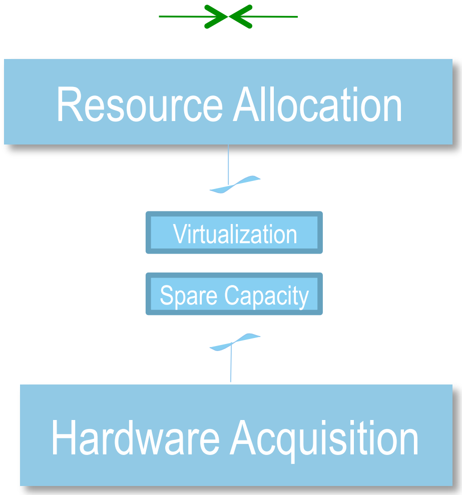
Automation



pool
provisioning
in minutes

+

Capacity Management



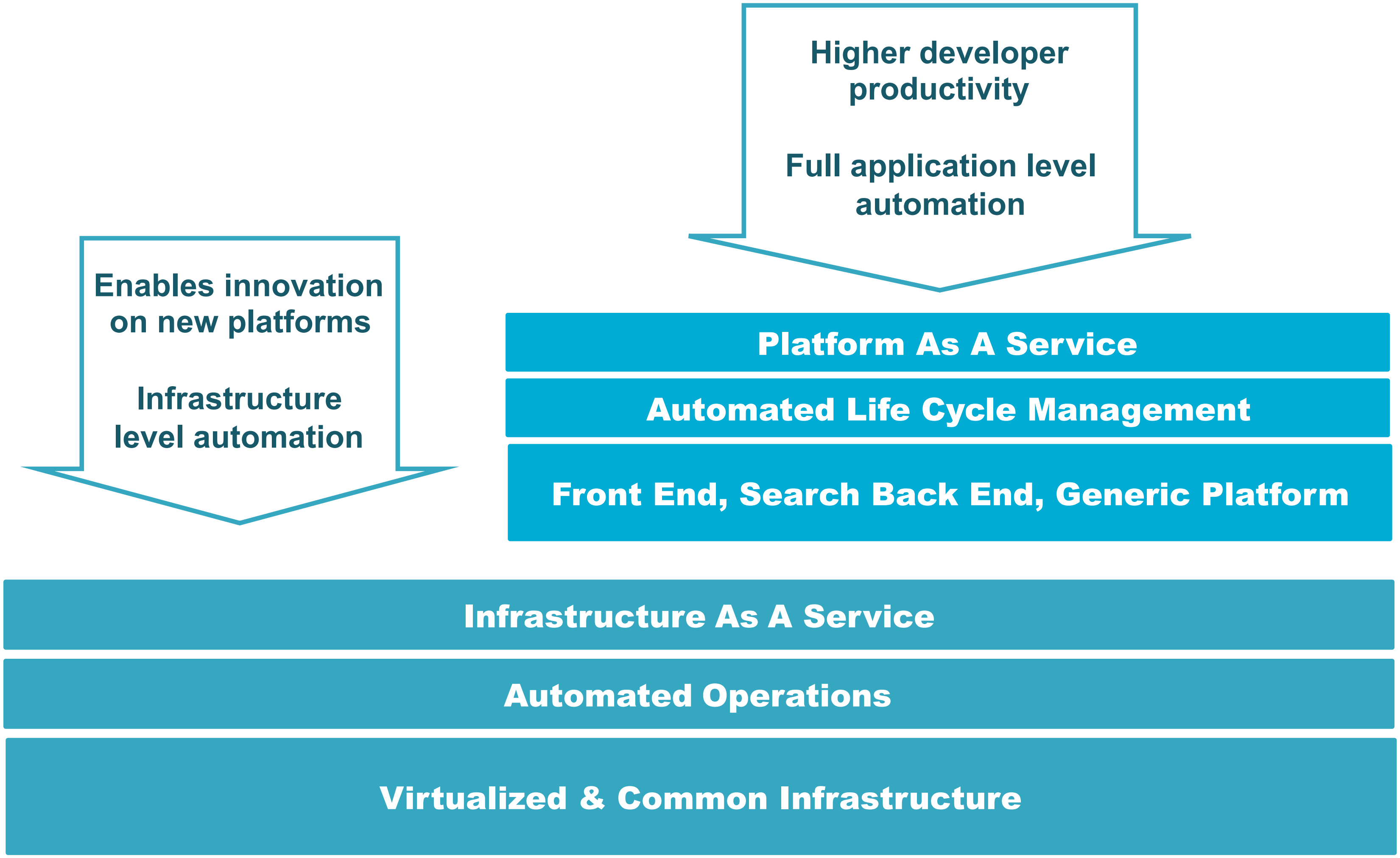
= Improved Time to Market

Design Principles

DRAFT

- Network isolation to enable mobility and isolation at scale
- Capability to automate reliably
- Standardization
- Private vs. Public
  - Start with Private, option to go Hybrid
- Buy vs. Build
  - Build + OSS

Infrastructure & Platform as a service



Higher developer productivity

Full application level automation

Enables innovation on new platforms

Infrastructure level automation

Platform As A Service

Automated Life Cycle Management

Front End, Search Back End, Generic Platform

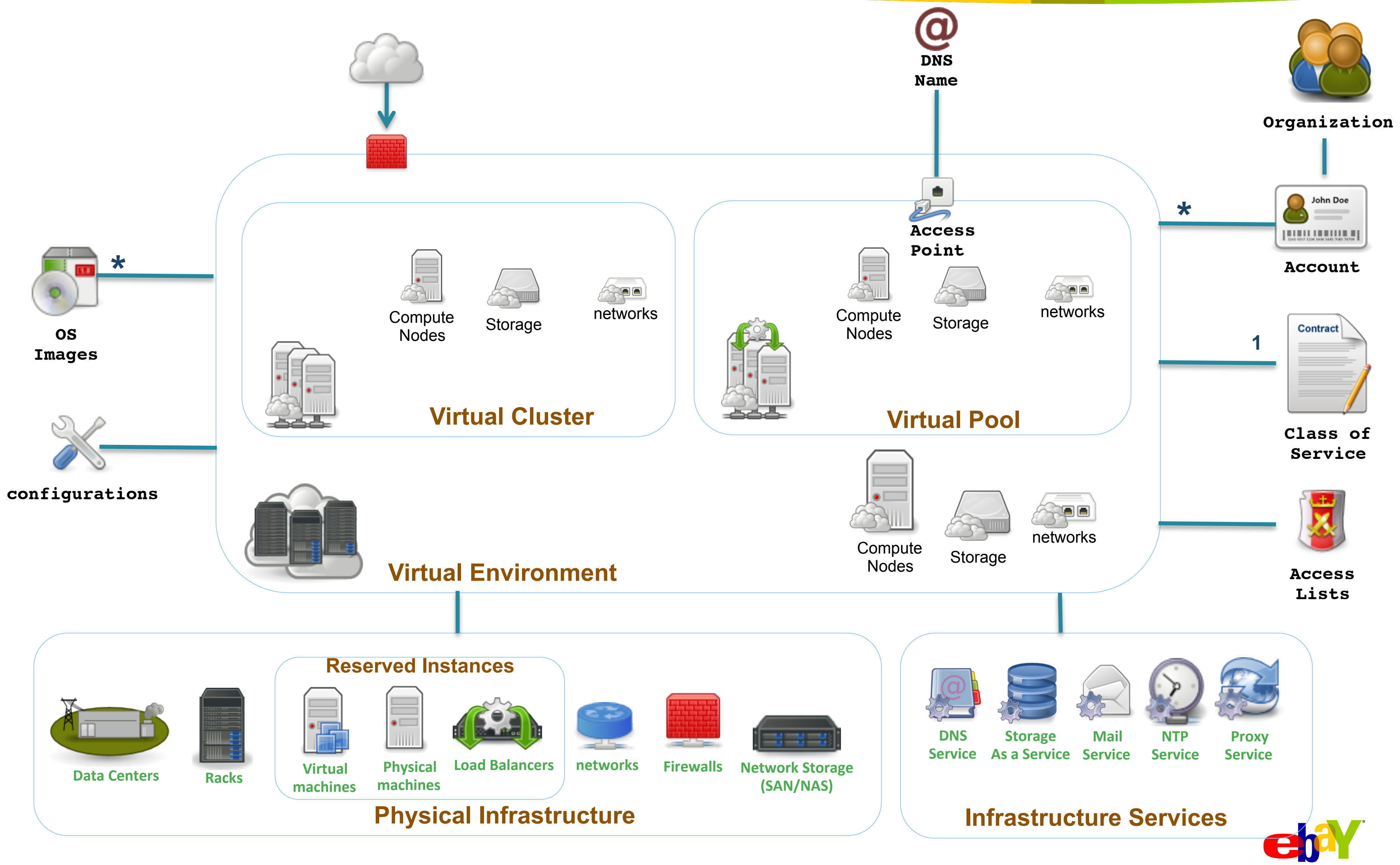
Infrastructure As A Service

Automated Operations

Virtualized & Common Infrastructure

IaaS

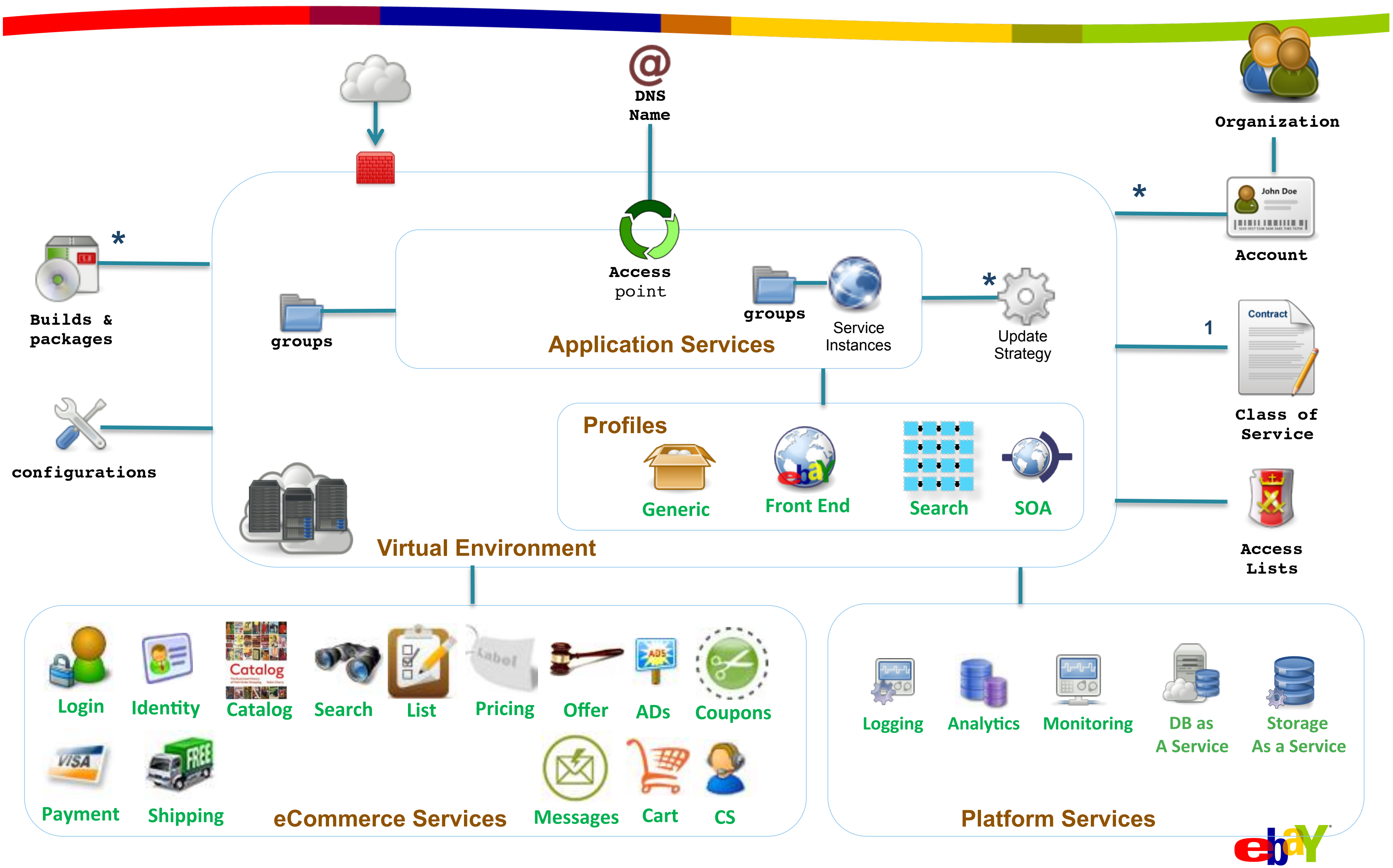
DRAFT



DNS
Name
Organization
Access
Point
*
Account
OS
Images
*
Compute
Nodes
Storage
networks
Compute
Nodes
Storage
networks
1
Virtual Cluster
Virtual Pool
Class of
Service
configurations
Compute
Nodes
Storage
networks
Virtual Environment
Access
Lists
Reserved Instances
Data Centers
Racks
Virtual
machines
Physical
machines
Load Balancers
networks
Firewalls
Network Storage
(SAN/NAS)
Physical Infrastructure
DNS
Service
Storage
As a Service
Mail
Service
NTP
Service
Proxy
Service
Infrastructure Services

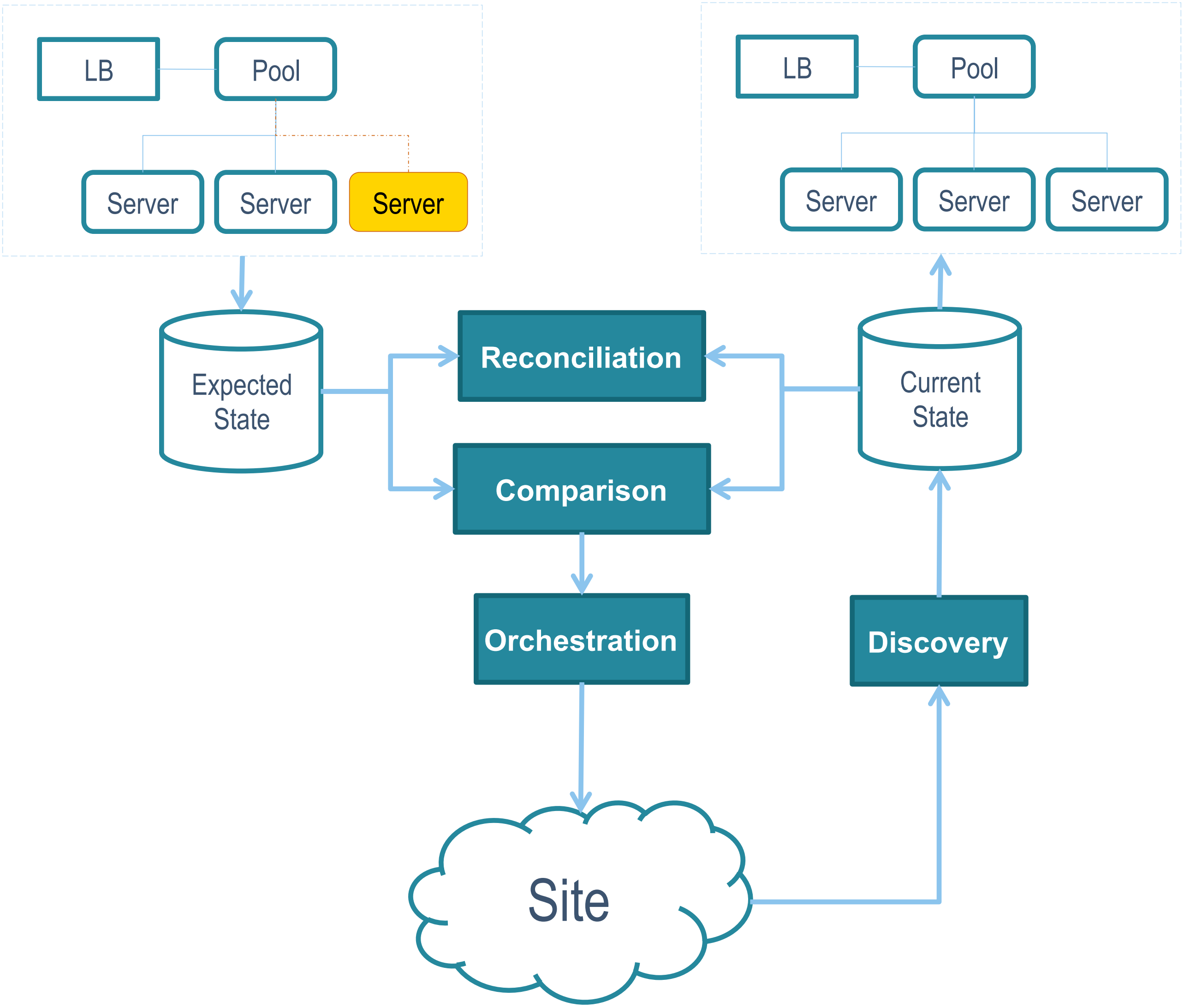
PaaS

DRAFT


DNS
Name
Organization
*
Account
*
Builds &
packages
groups
Access
point
groups
Service
Instances
Application Services
*
Update
Strategy
1
Class of
Service
configurations
Profiles
Generic
Front End
Search
SOA
Virtual Environment
Access
Lists
Login
Identity
Catalog
Search
List
Pricing
Offer
ADs
Coupons
Payment
Shipping
eCommerce Services
Messages
Cart
CS
Logging
Analytics
Monitoring
DB as
A Service
Storage
As a Service
Platform Services

Model Driven Automation for Reliability

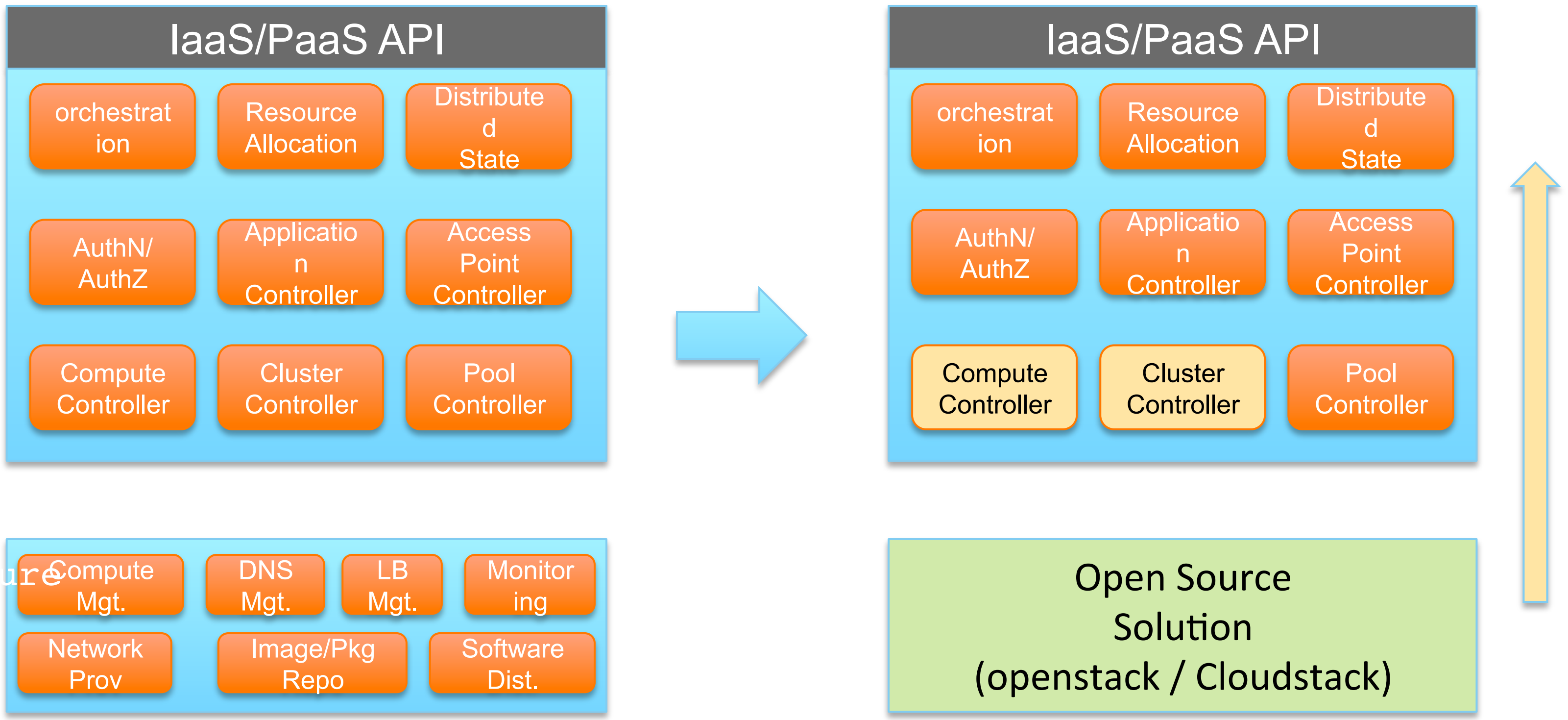
DRAFT



- Desired configuration is specified in the expected state and persisted in CMS
- Upon approval, the orchestration will configure the site to reflect the desired configuration.
- Updated site configuration is discovered based on detection of configuration events
- Reconciliation between the expected and current state allows to verify the proper configuration.
- On going validation allows the detection of out of band changes.

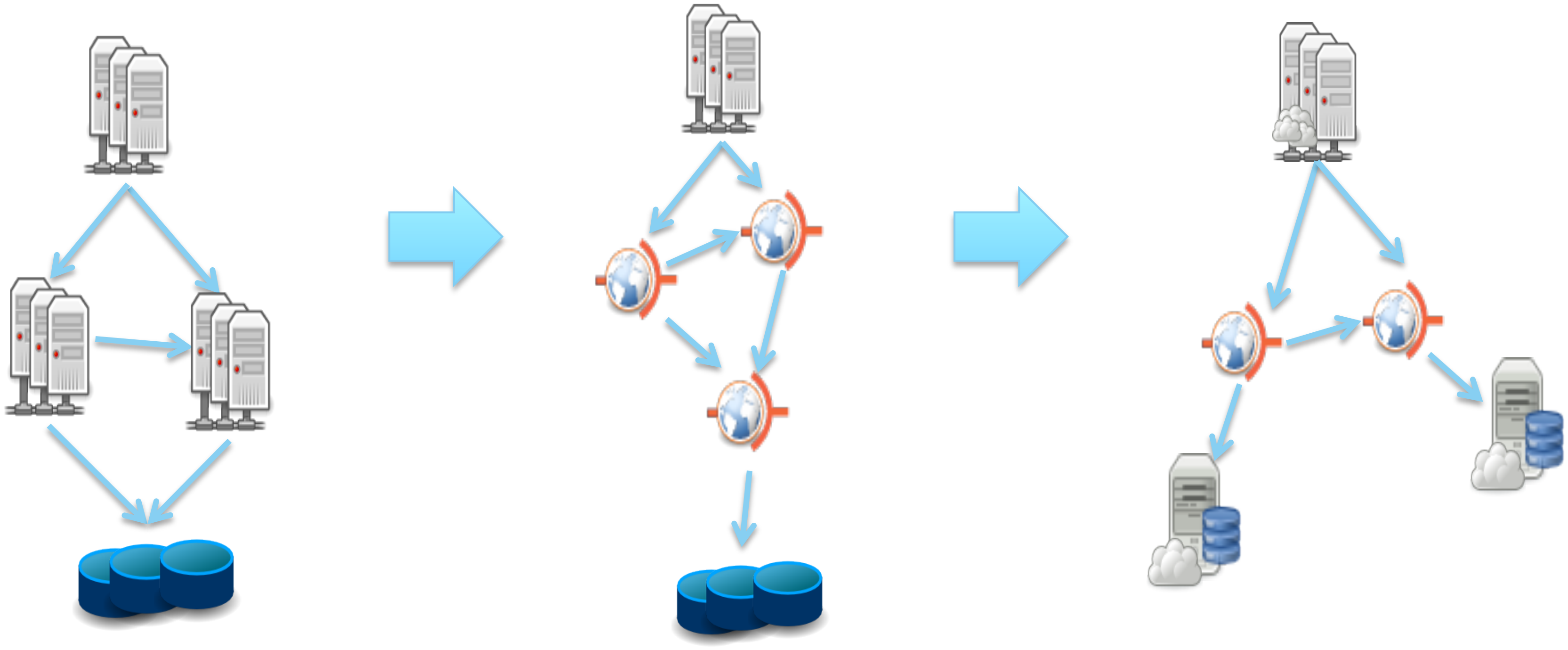
Open Source Integration

DRAFT



Application Architecture

DRAFT



Before

Ongoing
“Cloud
Friendly”

Future
‘Cloud
ready’

DRAFT

Next Gen Service Orientation

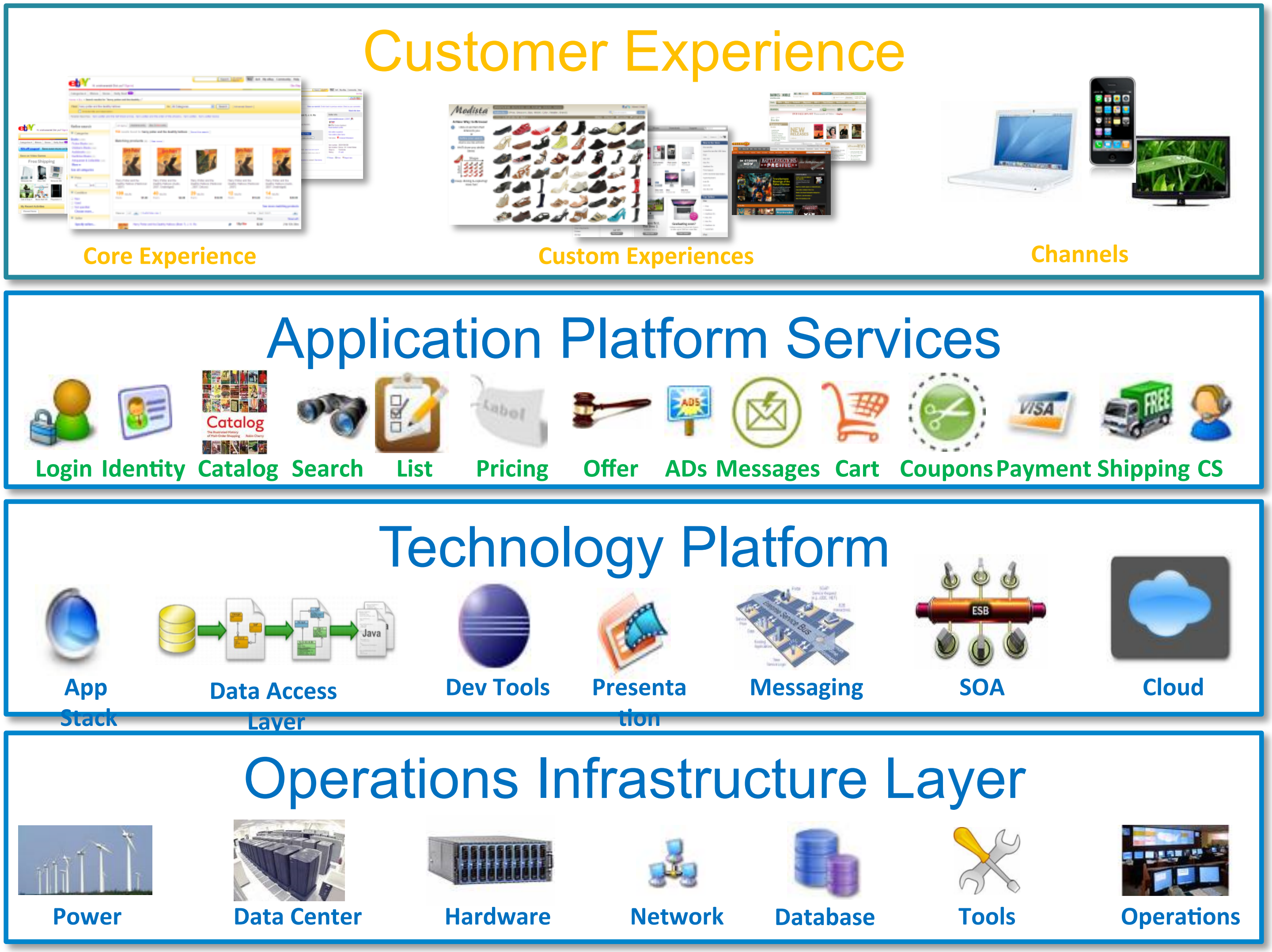
Services @ eBay

DRAFT

- It’s a journey !
- History
  - One of the first to expose APIs /Services
  - In early 2007, embarked on service orienting our entire ecommerce platform, whether the functionality is internal or external
  - Support REST style as well as SOA style
  - Have close to 300 services now and more on the way
  - Early adopters of SOA governance automation (Discovery focus rather than control)

Architecture Vision

DRAFT



Customer Experience

Core Experience

Custom Experiences

Channels

Application Platform Services

Login Identity Catalog Search List Pricing Offer ADs Messages Cart Coupons Payment Shipping CS

Technology Platform

App Stack

Data Access Layer

Dev Tools

Presentation

Messaging

SOA

Cloud

Operations Infrastructure Layer

Power

Data Center

Hardware

Network

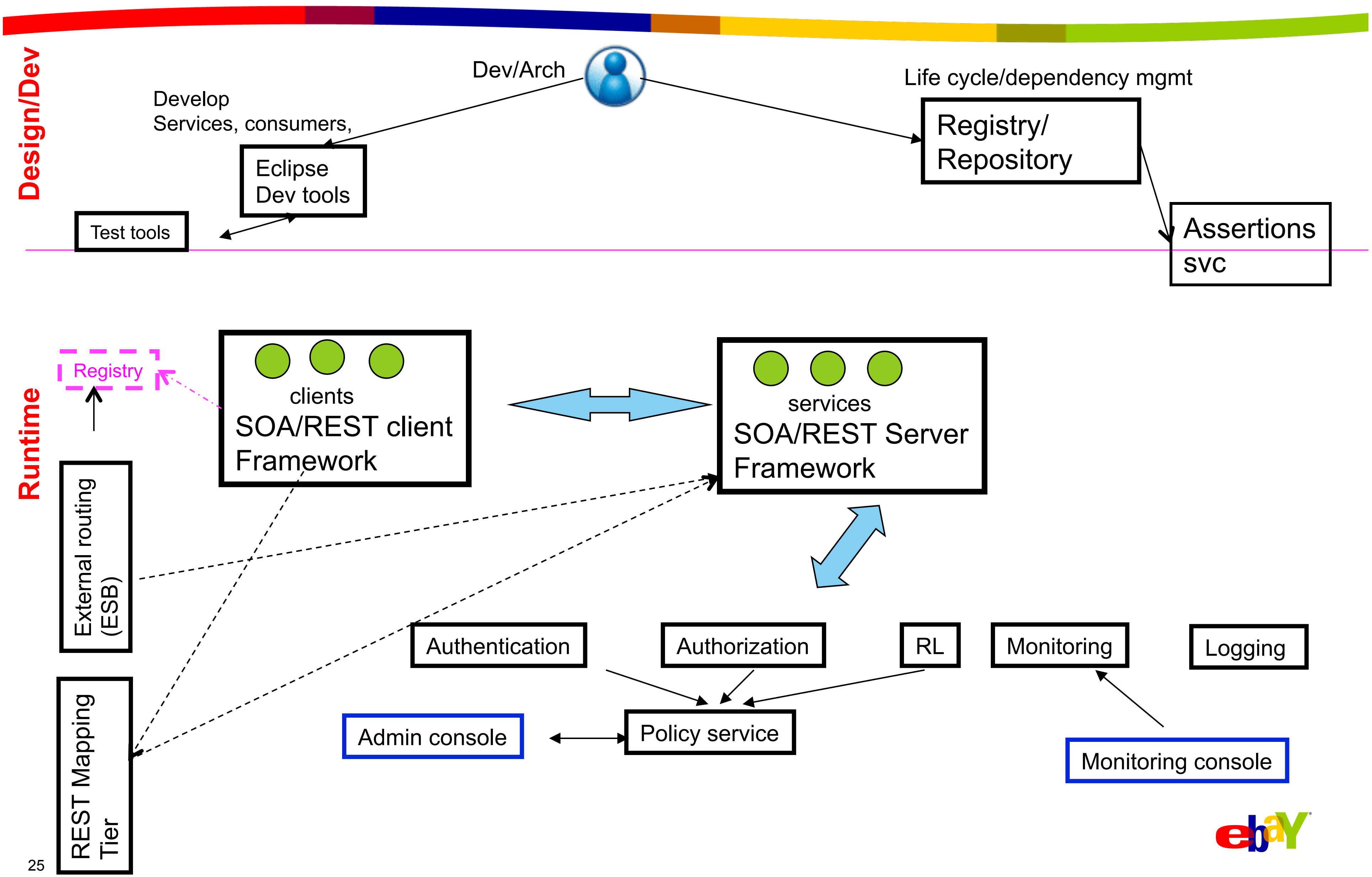
Database

Tools

Operations

eBay SOA Stack Overview

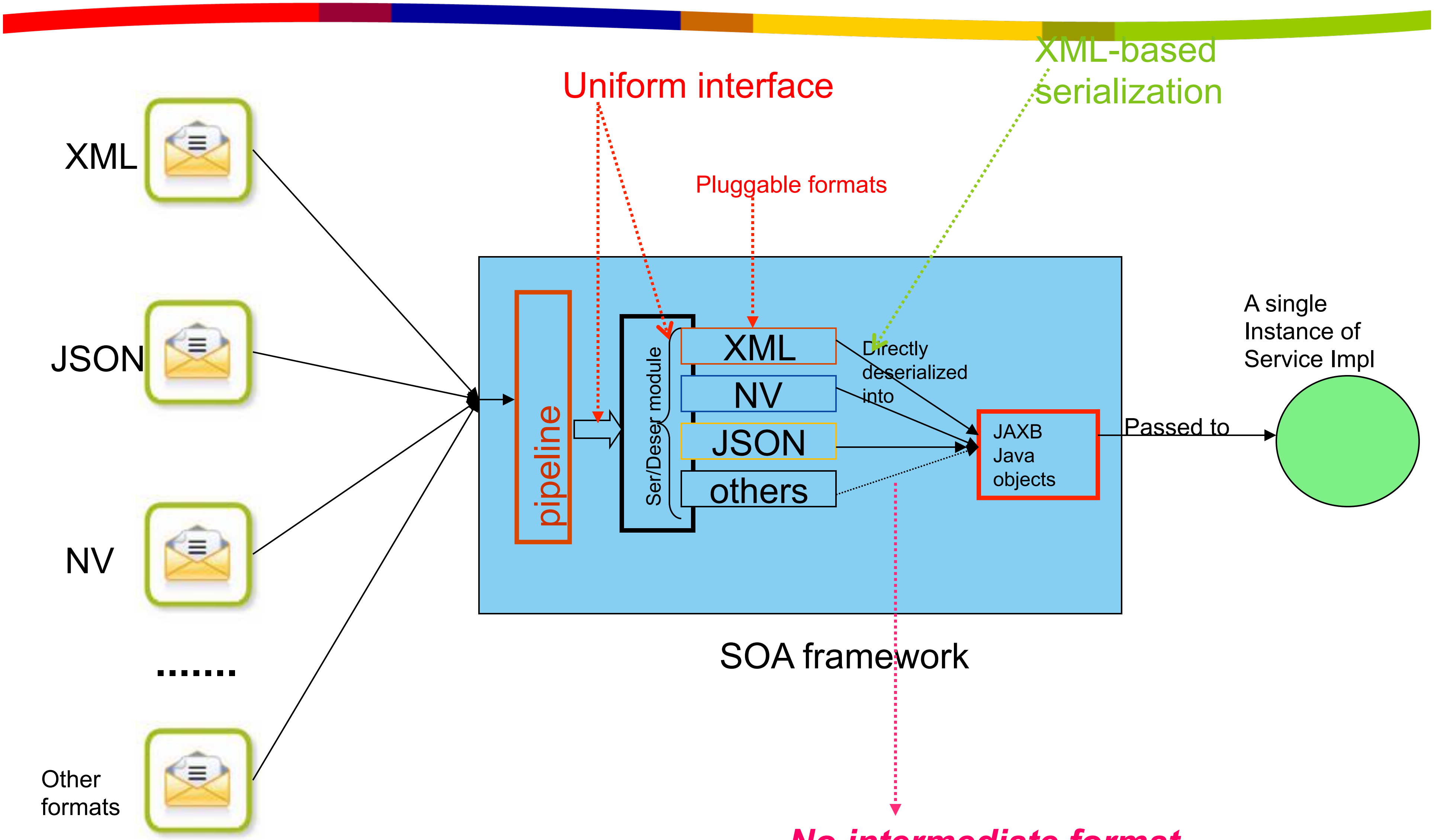
DRAFT



Challenge 1: Multiple Data Formats

- 2005: Mix of user preferences
  - SOAP
  - REST-like: HTTP GET with all request information in the URI
  - Plain Old XML (POX): HTTP POST with XML data but no SOAP envelope
  - JSON
- Shopping API was our first “XML Unified Field” web service
  - Input formats: Name-value encoded in URI; XML; JSON; SOAP
  - Output formats: XML, JSON, SOAP
- Key concepts:
  - Users ask for whatever data format they want.
  - When using our frameworks, developers don’t have to change **any** code to get a different data format.
  - **Anything you can express in XML, you can express in other formats**
  - **Complete mapping from XML structures to NV and JSON**
- Service developers **don’t want to write extra code to do conversions**; too much maintenance impact

Solution: Pluggable Data Formats Using JAXB

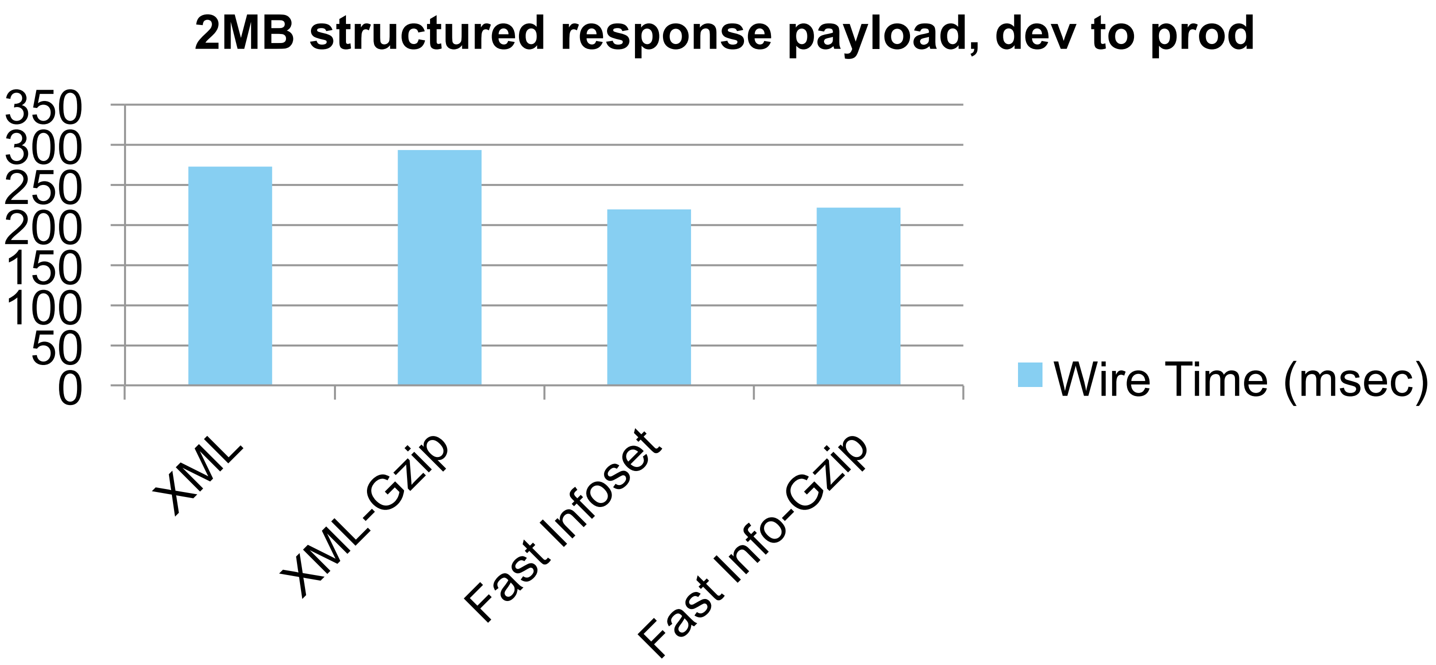


No intermediate format, Avoids extra conversion

Challenge 2: Latency

DRAFT

- JAXB unification of XML, JSON, Name-Value, and Fast Infoset works well!
- BUT: for large datasets, there can be nasty latencies.
  - **Not** fixed by compressing or using Fast Infoset



Solution: Binary Formats

DRAFT

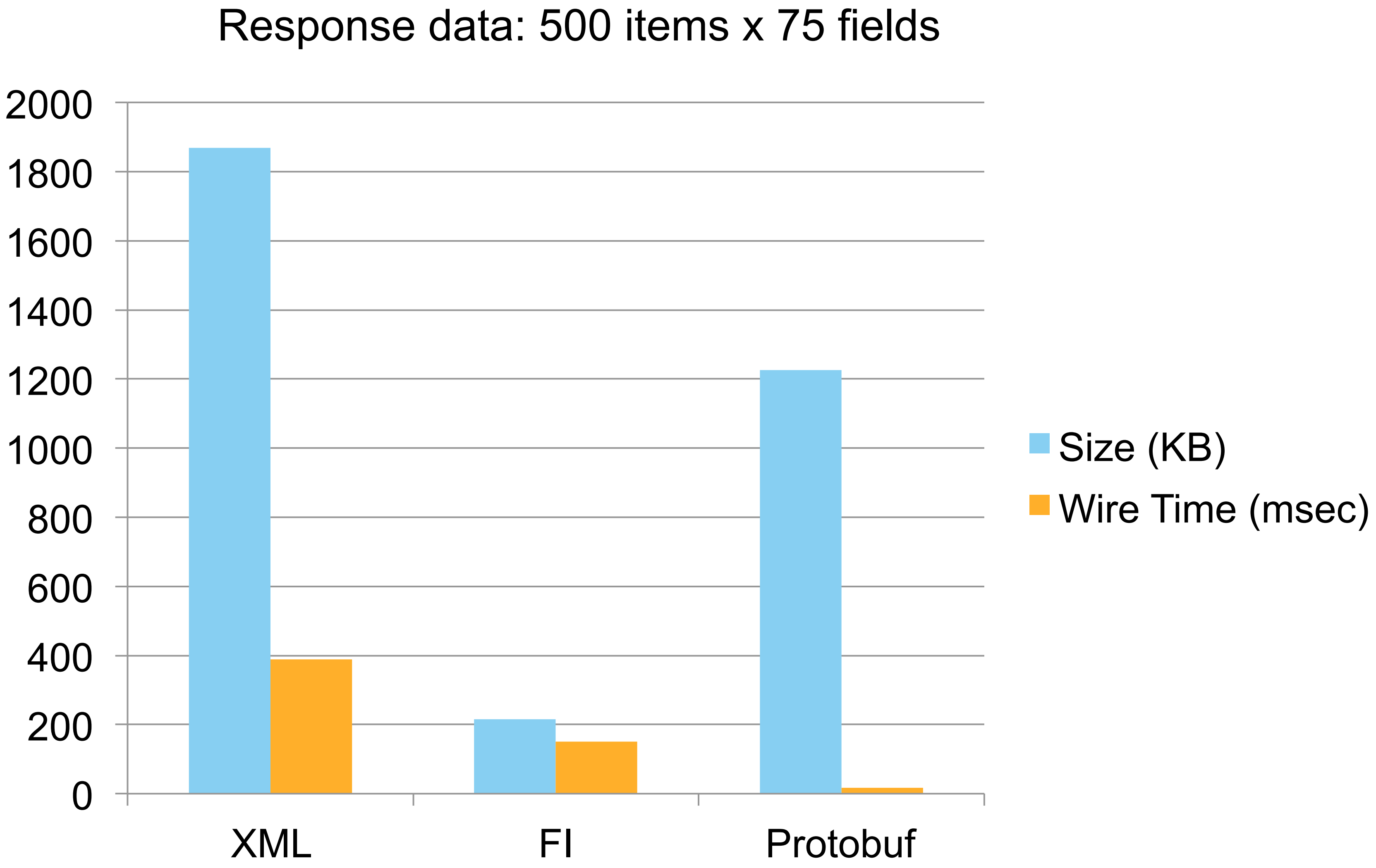
- What about using:
  - Binary formats: **Google Protocol Buffers, Avro,Thrift**, etc.?
- Numbers look promising (serialization, deserialization)
- New challenges with these:
  - Each has its own **schema** (type definition language) to model types and messages
  - Each has its own **code generation** for language bindings
    - NOT directly compatible with JAXB beans
  - Turmeric SOA platform uses **WSDL/XML Schema (XSD)** data modeling, and **JAXB** language bindings

Compare Popular non-XML Formats DRAFT

Protobuf	Avro	Thrift
• Own IDL/schema • **Sequence numbers** for each element • Compact **binary representation** on the wire • Most XML schema elements are **mappable** to equivalents, except polymorphic constructs • Versioning is similar to XML, a bit more complex in implementing due to sequence numbers	• **JSON based** Schema • Schema prepended to the message on the wire • Compact **binary representation** on the wire • Most XML schema elements are **mappable** to equivalent, except polymorphic constructs • Versioning is easier	• Own IDL/schema • **Sequence numbers** for each element • Compact **binary representation** on the wire • Most XML schema elements are **mappable** to equivalents, except polymorphic constructs • Versioning is similar to XML, a bit more complex in implementing due to sequence numbers

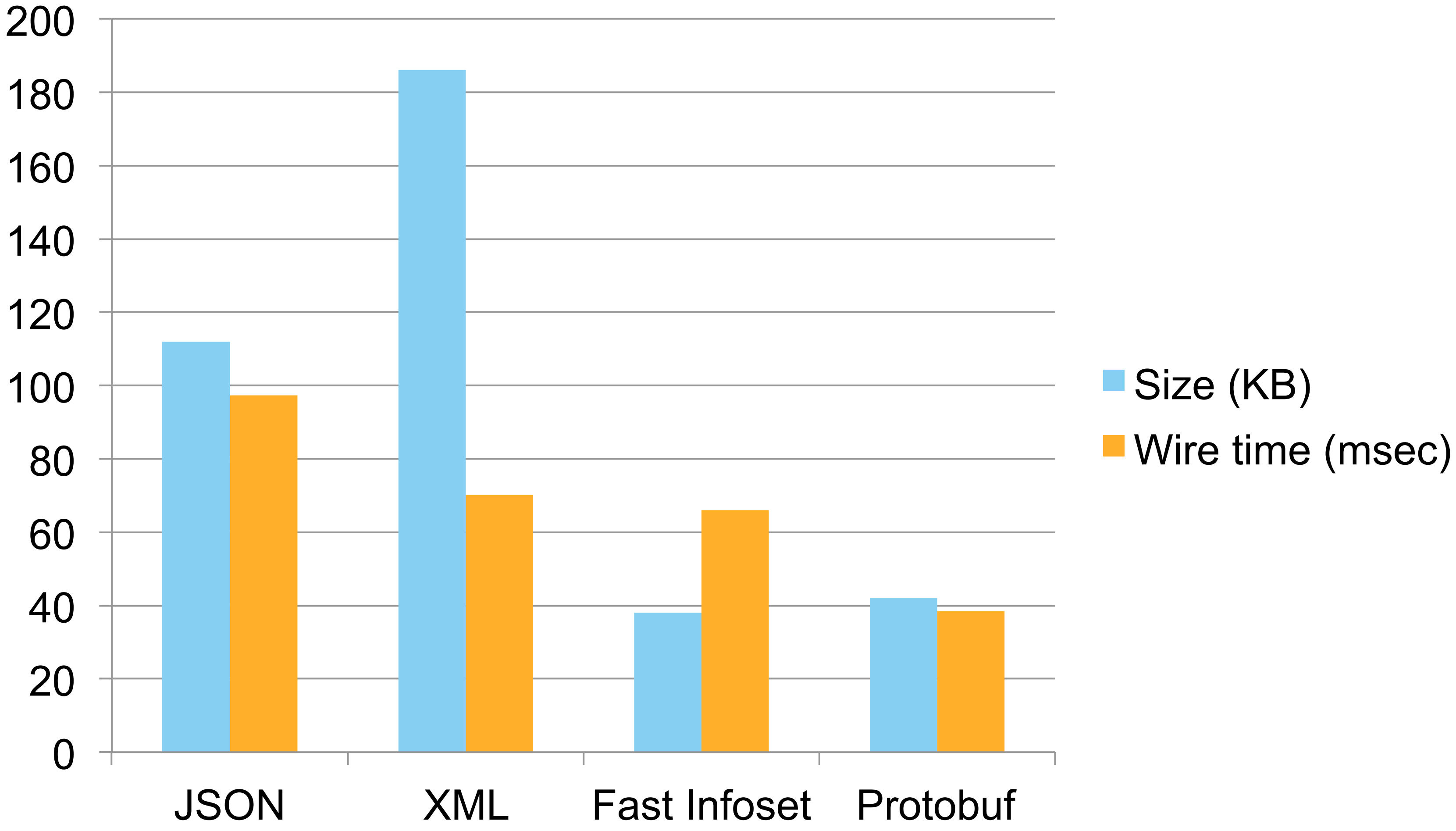
	Complex Types	Unions (Choice Type)	Self-References (Trees)	Enums	Inheritance / Polymorphism	Inline Attachment
Protobuf	Yes	No	Yes	Yes	No	No
Avro	Yes	Yes	Yes (with workaround)	Yes	No	No
Thrift	Yes	No	No	No	No	No
XML	Yes	Yes	Yes	Yes	Yes	Yes (MIME-TYPE)

Early In-JVM Test, 80 percentile DRAFT



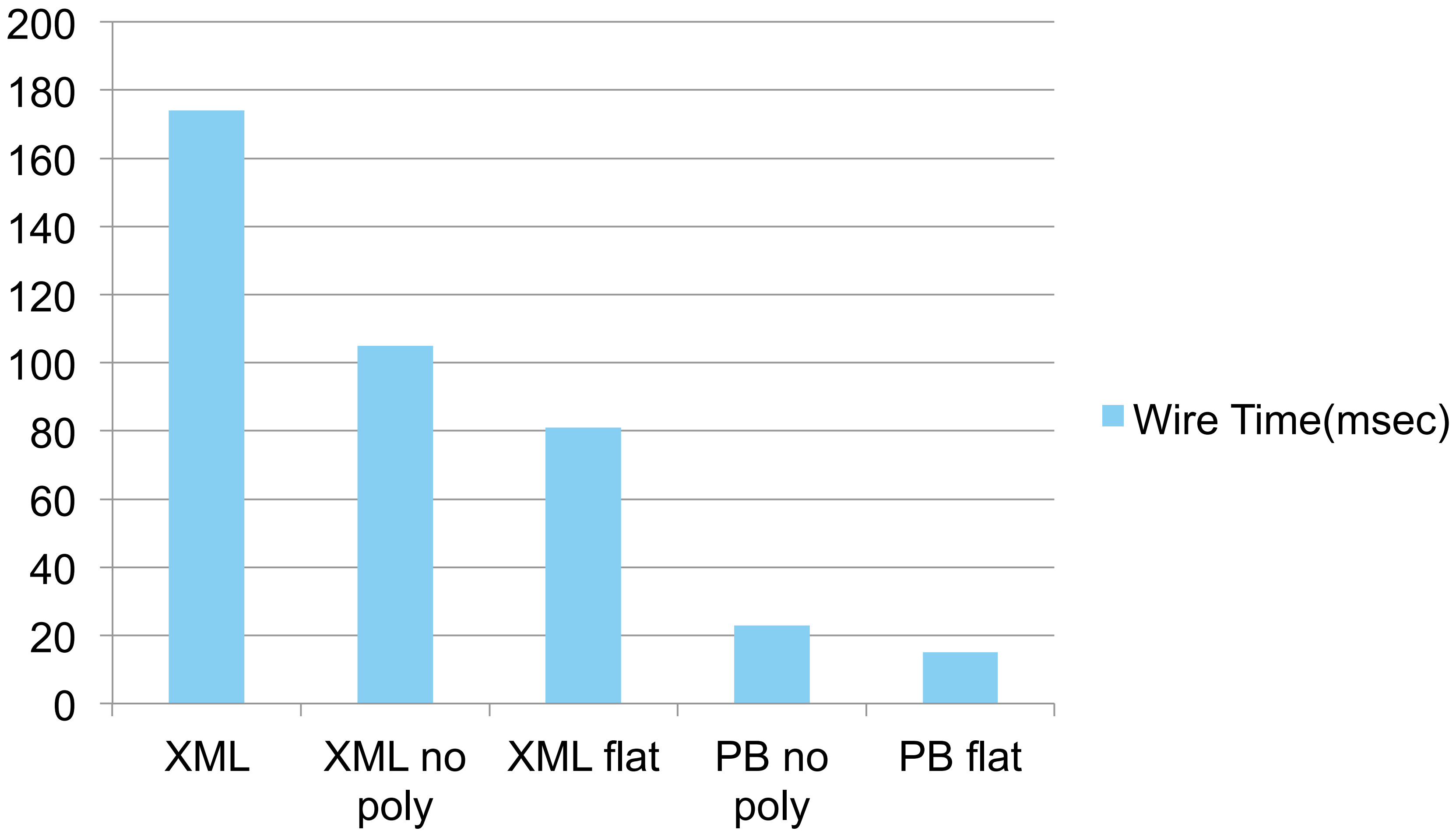
QA network test, 90% timings DRAFT

Response data: 50 items x 75 fields (about 8000 objects)



Production tests – progressive improvements

DRAFT



Challenge 3: Service Consumer Productivity

DRAFT

- Large, complex requests and responses
- Get exactly what they want in data returned from services
- Lack of consistency in service interface conventions and data access patterns
- Real client applications make calls to multiple services at a time
  - Serial calls increase latency. Managing parallel calls is complex
- Impedance mismatch between service interface and client needs
  - Too much data is returned
  - 1 + n calls to get detailed data

Sneak Preview: ql.io

DRAFT

- New technology from eBay
- Plan to open source soon
- SQL + JSON based scripting language for aggregation and orchestration of service calls
- Filtering and projections of responses
- Async orchestration engine
  - Automatic parallelization, fork / join

What ql.io Enables

DRAFT

- **Create consumer-controlled interfaces**
  - fix/patch APIs on the fly
- **Filter and project responses**
  - use a declarative language
- **Bring in consistency**
  - offer RESTful shims with simpler syntax
- **Aggregate multiple APIs**
  - such as batching
- **Orchestrate requests**
  - without worrying about async forks and joins

ql.io Demo

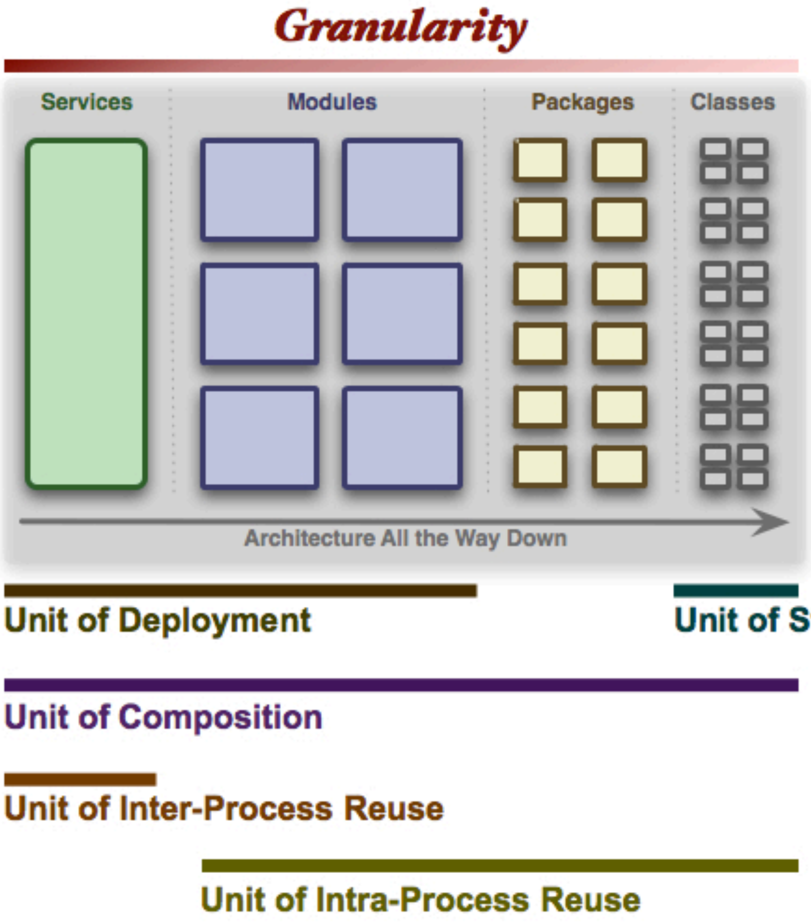
DRAFT

DRAFT

Modularity

Key modularity concepts for software

- Building blocks
- Re-use
- Granularity
- Dependencies
- Encapsulation
- Composition
- Versioning



Source: http://techdistrict.kirkk.com/2010/04/22/granularity-architectures-nemesis/
Author: Kirk Knoernschild

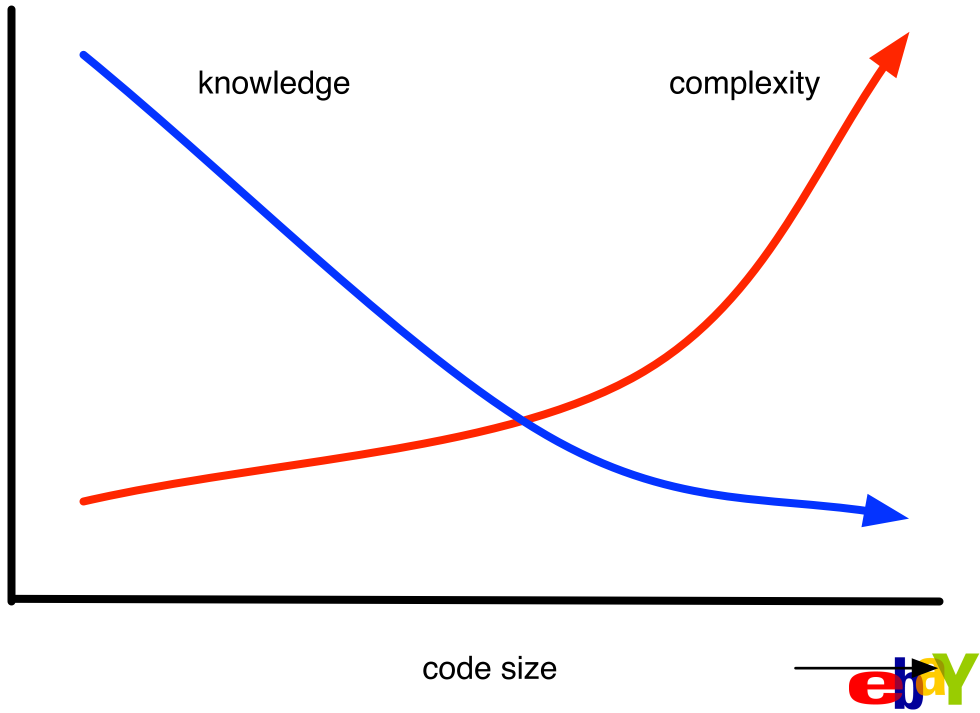
Challenges for Large Enterprises DRAFT

- Some stats on the eBay code base
  - ~ 44 million of lines of code and growing
  - Hundreds of thousands of classes
  - Tens of thousands of packages
  - ~ 4,000+ jars
- We have too many dependencies and tight coupling in our code
  - Everyone sees everyone else
  - Everyone affects everyone else

Challenges for Large Enterprises

DRAFT

- Developer productivity/agility suffers as the knowledge goes down
  - Changes ripple throughout the system
  - Fallouts from changes/features are difficult to resolve
  - Developers slow down and become risk averse



Our Goals with Modularity Efforts

- Tame complexity
- Organize our code base in loose coupling fashion
  - Coarse-grained modules: number matters!
  - Declarative coupling contract
  - Ability to hide internals
- Establish clear code ownership, boundaries and dependencies
- Allow different components (and teams) evolve at different speeds
- Increase development agility

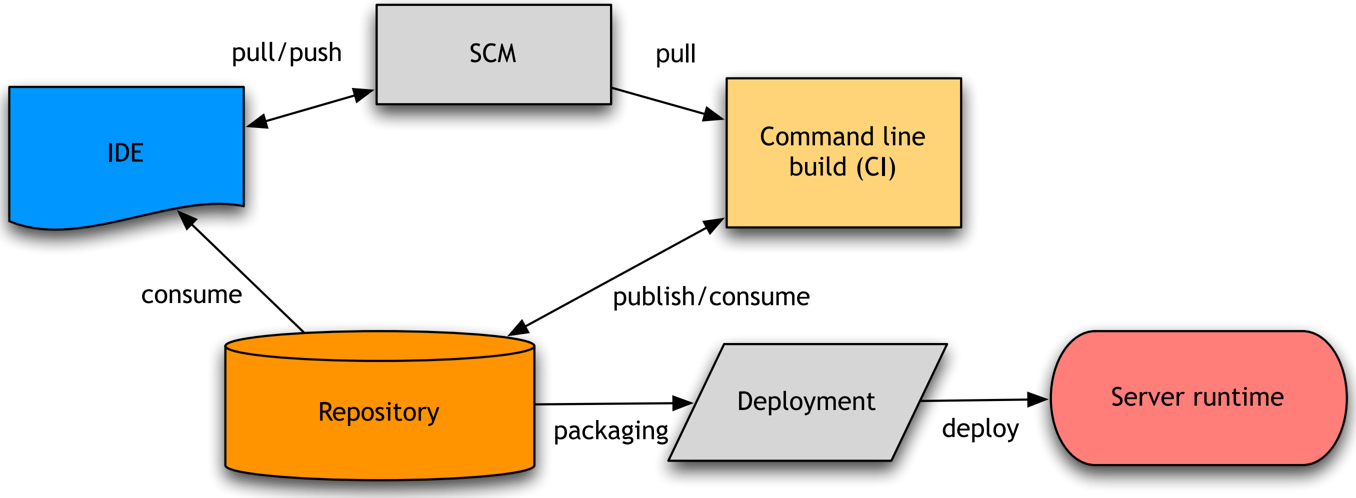
Considerations on Modularity Solutions DRAFT

- Scalability: enterprise software tends to be large scale
- We need to consider a large group of developers with varying skill levels
- End-to-end development lifecycle is crucial
- Conversion/migration of existing code base is crucial
  - We rarely start from vacuum
  - We want to move over and modularize bulk of existing code
  - It is imperative that we chart a realistic migration course that can be achieved within a reasonable amount of time
  - We cannot afford disruption to business meanwhile: "change parts while the car is running"

End-to-End Development

DRAFT

- IDE, command line build, repository, server runtime, etc.
  - Complete and mature tooling
  - Integration and fidelity of tools across phases



Modularity Solution Evaluation DRAFT

- Evaluated OSGi, Maven, Jigsaw and JBoss Module
- Criteria include:
  - Modularity enforcement
  - End-to-end development
  - Migration concerns
  - Adoption
  - Maturity
- Selected OSGi

OSGi

DRAFT

META-INF/MANIFEST.MF:

```
Bundle-ManifestVersion: 2
Bundle-SymbolicName: org.foo.bar
Bundle-Version: 1.2.1
Import-Package: org.foo.other;version=“[1.1,2.0)”,
 javax.xml.parsers
Export-Package: org.foo.bar;version=“1.2.3”,
 org.foo.bar.sub;uses=“org.foo.bar”;version=“1.2.1”
```

OSGi Pros

DRAFT

- Enforces modularity strongly: it will let you know if you violate it
- Mature and comprehensive: covers pretty much all use cases regarding modularity
- Open standard
- Services: the ultimate decoupling force
- Can run two versions of the same class easily

OSGi Cons

DRAFT

- Can run two versions of the same class easily, and run into trouble
- Some problems are nasty to troubleshoot (uses conflict anyone?)
- Still not many well-integrated tools across all phases: impedance mismatches
- Compared to strong runtime model, build side story is weak
- Migration can be quite painful
- Learning curve is still fairly steep

eBay Open Source Initiative

- eBay has been a strong supporter of Open Source model and community

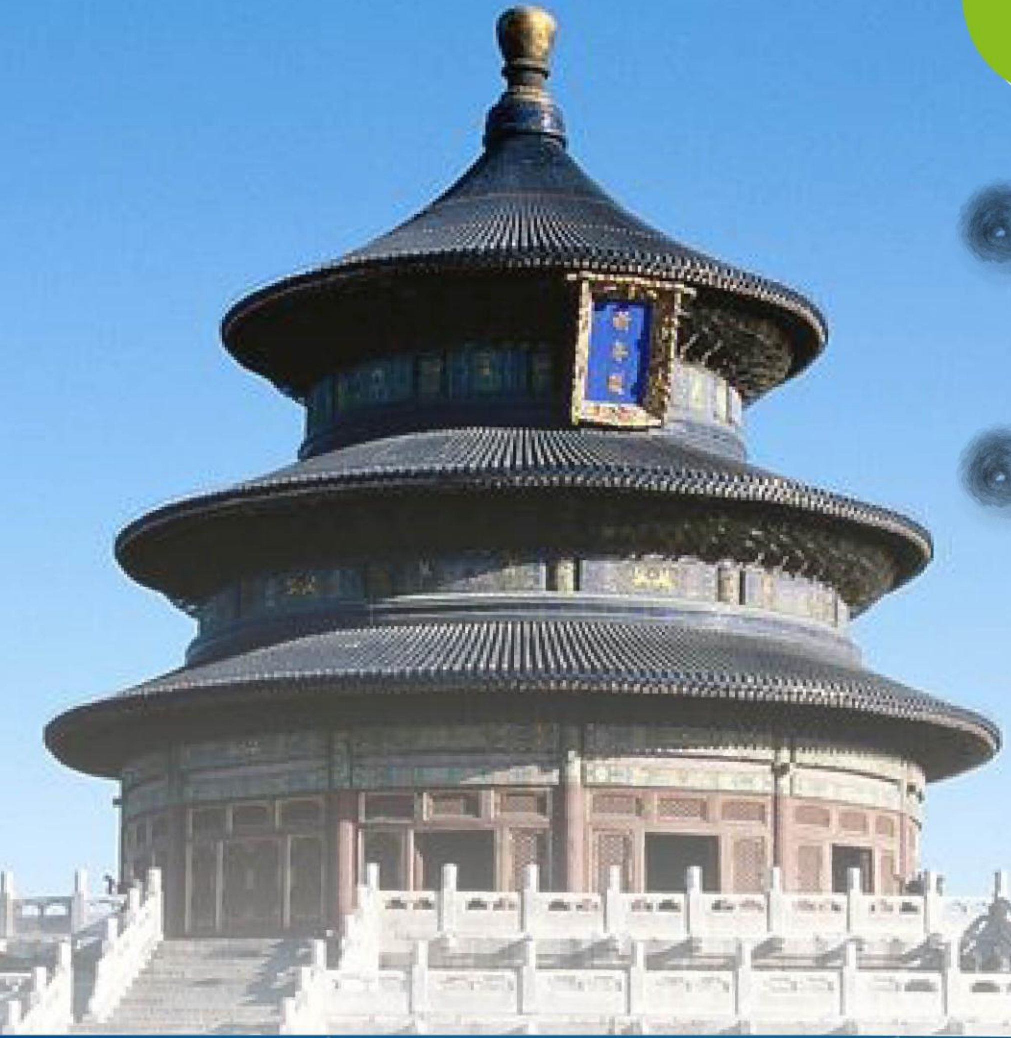
- Check out http://eBayOpenSource.org
  - Mission is to open source some of the best of breed technologies that were developed originally within eBay Inc.
  - For the benefit of the community
  - Under a liberal open source license.
  - These projects are generic technology projects and several years of development effort has gone into them to mature them.
  - Most parts of our services platform, code named Turmeric, is open sourced on this site.

Summary

DRAFT

- Systems quality & architecture as key foundation
- Complexity management becomes important over time
- Strike balance between agility and stability

ebay



QCon

北京站 · 2012年4月18~20日

www.qconbeijing.com（11月启动）

QCon杭州站官网和资料

www.qconhangzhou.com

全球企业开发大会

INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE