Why Enterprise Management Is All About Relationships And Not Things

Paul Strong
Distinguished Engineer, eBay Research Labs
Acting Chair, Open Grid Forum (OGF)
Copyright Notice

© 2008 eBay Inc. All rights reserved.

- No part of these materials may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording, or otherwise) without the prior permission of eBay Inc.
- eBay and the eBay logo are registered trademarks of eBay Inc.
- PayPal and the PayPal logo are registered trademarks of PayPal, Inc.
- Other trademarks and brands are the property of their respective owners.
- Please do not take our picture or record the class/session without asking permission.
Why Infrastructure Matters

- Successful web applications/services demand scalability, agility, availability
- Service Level Management, i.e. setting and meeting expectations, is key
- Not fashionable *but* many interesting problems yet to be solved
eBay – The 30 Second Introduction!

eBay users trade about $2,039 worth of goods on the site every second

On an average day on eBay...

- A vehicle sells every minute
- A motors part or accessory sells every second
- Diamond jewelry sells every 2 minutes

1.3m people make all or part of their living selling on eBay.

*ACNielsen International Research, June 2006

©2008, eBay Inc.
Why eBay Is A Useful Example

New Challenges
Extreme Engineering

The Bleeding Edge
Everyday use

Technology trickle down/transfer

©2008, eBay Inc.
eBay’s Drivers

• Extreme Scale
  276m Registers Users, 113m+ Items, 6m+ New Items Per Day
• Extreme Growth
  Near exponential growth in listings for most of history – 12 years
• Extreme Agility
  Roll code to the site every 2 weeks
• Constant, predictable presence
  Must be 24x7x365
• Efficiency

Failure To Keep Up Is Not An Option!
Challenges

- Scaling The Database
- Scaling Services
- Managing At Scale
- Better Management Through Semantics
High Level Data/Storage Architecture
eBay Example #1
Making The Database Scale

- Second Database for failover
- CGI pools, Listings, Pages, and Search continued to scale horizontally

However …

By November 1999, the database servers approached their limits of physical growth.
eBay Example #1
Making The Database Scale

- Database "split" technology.
- Logically partition database into separate instances.
- Horizontal scalability through 2000, but not beyond.
eBay Example #1
Virtualizing the Database

• Separate Application notion of a database from physical implementation
• Databases may be combined and separated with no code changes
• Reduce cost of creating multiple environments (Dev, QA, …)
• Application can continue to function without non-critical data (markdown)
eBay Example #1
Virtualizing & Scaling the Database

November, 1999
eBay Example #1
Virtualizing & Scaling the Database

December, 2002
SAN
eBay Example #1
Virtualizing & Scaling the Database

- **Scales Out**
  276 million registered users
  113 million Items
  6+ million new items per day
  34 billion SQL transactions per day
  600+ production database instances (inc replicas)
  100+ clusters

- **Cheaper**
  Smaller, potentially commodity, servers

- **Highly Resilient**
  2-4 copies of everything
  Minimized impact of outage to [relatively] small sub-set of data

- **Flexible/Agile**
  Easy to change – database, schemas, partitioning etc.
  Minimal impact on architecture or code
eBay Example #2
Scaling Services

Stateless is good but how do you scale things that share dependencies?
**eBay Example #2**

**Scaling Services**

- **Partition code into functional areas**
  - Application is specific to a single area (Buying, Selling etc.)
  - Domain contains common business logic across applications
- **Restrict inter-dependencies**
  - Applications depend on Domains, not on other applications
  - No dependencies among shared domains

![Diagram showing applications and domains](image-url)
eBay Example #2
Scaling The Application

- Everything behaves as loosely coupled services
- Minimize inter-dependencies
- Infrastructure is like a giant FPGA
  - Potential to re-program by re-routing traffic
- Scales
  - Scale out means scaled throughput and resilience
  - 16000+ concurrent instances
  - 8000+ servers (mainly blades)
- Efficiency
  - Run traffic from different time zones on the same server but different instances
Scaling Search – Voyager

- **Real-time feeder infrastructure**
  Reliable multi-cast from primary database to search nodes

- **Real-time indexing**
  Search nodes update index in real time from messages

- **In memory search index**

- **Horizontal segmentation (scatter, gather)**
  Search index divided into N slices (“columns”)
  Each slice replicated to M instances (“rows”)
  Aggregator parallelizes query over all N slices, load balances over M instances

- **Caching**
  Cache results for highly expensive and frequently used queries
Architectural Lessons Learnt

• Scale Out, Not Up
  Horizontal scaling at every tier
  Functional decomposition

• Prefer Asynchronous Integration
  Minimize availability coupling
  Improve scaling options

• Virtualize Components
  Reduce physical dependencies
  Improve deployment flexibility

• Design For Failure
  Automated failure detection and notification
  “Limp mode” operation of business features
The Big Problem

Management complexity scales with this

©2008, eBay Inc.
Understanding Relationships

Service A is composed of
  Persistence Sub-Service B
  Business Logic Sub-Service C
  Presentation Sub-Service D
Understanding Relationships

Business Logic Sub-Service C is composed of
A Load Balancing Service
Several Application Instances
Understanding Relationships

The Application Instances are *hosted on* Operating System Instances

The Load Balancing Service is *hosted on* A Load Balancer Operating System
Understanding Relationships

The Operating System Instances are *hosted* on
Servers or Virtual Servers, which are in turn hosted on servers

The Load Balancer OS is *hosted* on
A Physical Load Balancer
Interaction/Traffic Relationships
Starting To Look Complicated!
Relationships Are Everything!

- Everything is interconnected
- Changing one thing causes ripples
- How you connect things together determines business functionality and business value
- Agility is the ability to change these relationships dynamically (easier with loosely coupled services)
- Virtualization is about standardizing relationships and interposing/isolating one end from the other
- Understanding these relationships allows you to
  - Tie business processes to the infrastructure they run on
  - Map value to cost
  - Understand and manage traffic flow
  - Understand and manage provisioning etc.
- It’s all about managing relationships, not things!
Patterns can aid efficiency in all dimensions
- Configuration
- Provisioning
- Diagnosis

eBay’s storage infrastructure is managed by 11 people

However, patterns can constrain agility in other dimensions
- Adoption of new vendors
- Adoption of new products
- Adoption of new application or architectural patterns

Especially if processes are heavily automated where the patterns are codified
The Future…

• **Datacenter is becoming non-deterministic or chaotic**
  – Legacy (>2 years old!) applications
  – Internal SOA
  – Service disaggregation, distribution
  – Outsourcing of non-core services: Partners, SaaS, ASPs etc.
  – Biz 2.0 Mashups

• **Emergent behavior of services**
  – Unexpected interactions and behaviors
The Future…
Better Management Through Semantics

- Capture relationships in Semantic Query Service (in memory, custom OWL/RDF based Ontology)
  - Extensible
  - Patterns in data and not in code!
- Feed from CMDB and from Run Time Telemetry
- Query-able by other management services
- Visualized as graphs (DAGs) through UI
The Future…
Better Management Through Semantics Example

- Capture relationships in Semantic Query Service (in memory, custom OWL/RDF based Ontology)
  Extensible
  Patterns in data and not in code!
- Feed from CMDB and from Run Time Telemetry
- Query-able by other management services
- Visualized as graphs (DAGs) through UI
- Fundamentally changes the focus!
Conclusions

• Clouds are cool!
• Social networks are cool!
• Infrastructure is cool to - honestly :o)
Thank You

Paul Strong
Paul.Strong@eBay.com
Distinguished Engineer
eBay Research Labs,
eBay Inc.