eBay's Scaling Odyssey

Growing and Evolving a Large eCommerce Site

Randy Shoup and Franco Travostino
eBay Distinguished Architects

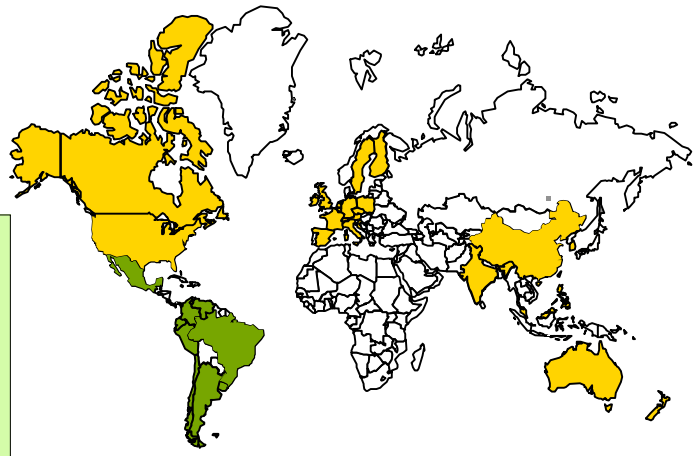
The 2nd Workshop on Large-Scale Distributed Systems and Middleware (LADIS '08)

IBM T.J. Watson Research Center, Yorktown, NY, USA
September 15-17, 2008

Challenges at Internet Scale

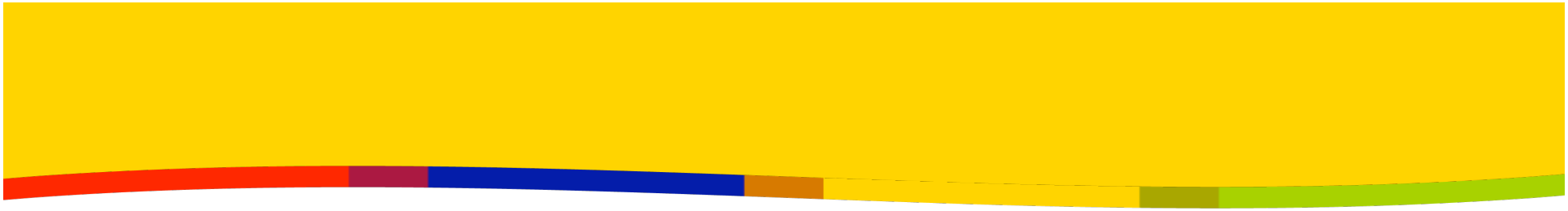
- eBay manages …
  - Over 276,000,000 registered users
  - Over 2 Billion photos
  - eBay users trade $2040 in goods every second -- $60 billion per year
  - eBay averages over 2 billion page views per day
  - eBay has roughly 120 million items for sale in over 50,000 categories
  - eBay site stores over 2 Petabytes of data
  - eBay Data Warehouse processes 25 Petabytes of data per day
- In a dynamic environment
  - 300+ features per quarter
  - We roll 100,000+ lines of code every two weeks
- In 39 countries, in 8 languages, 24x7x365

>48 Billion SQL executions/day!



Scaling dimensions

- Data
- Transactions
- SLAs
- Operations
- Deployment
- Productivity
- Feature velocity
- Adaptability



Principles and Patterns that we live by

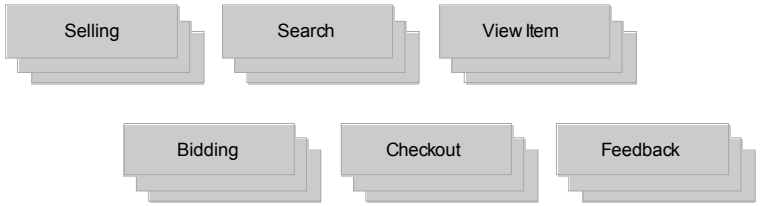
Principles for Scaling

1. Partition Everything
2. Asynchrony Everywhere
3. Automate Everything
4. Remember Everything Fails
5. Embrace Inconsistency

Principle 1: Partition Everything

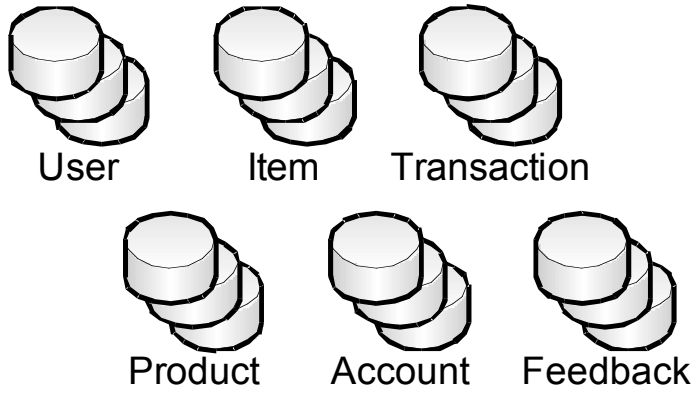
Pattern: Functional Segmentation

- Segment processing into pools, services, and stages
- Segment data along usage boundaries



Pattern: Horizontal Split

- Load-balance processing
  - Within a pool, all servers are created equal
- Split (or "*shard*") data along primary access path
  - Partition by range, modulo of a key, lookup, etc.



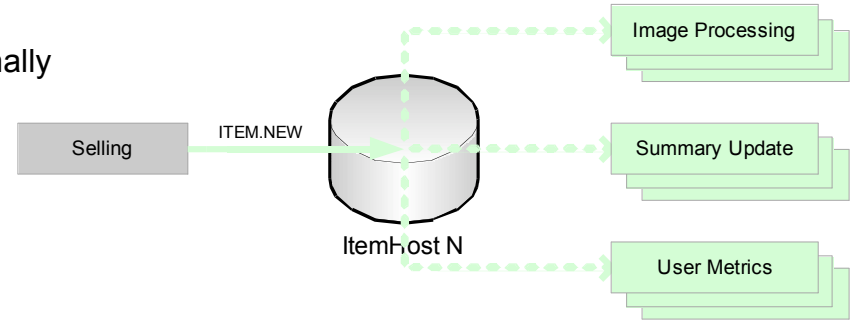
Corollary: No Session State

- User session flow moves through multiple application pools
- Absolutely no session state in application tier

Principle 2: Asynchrony Everywhere

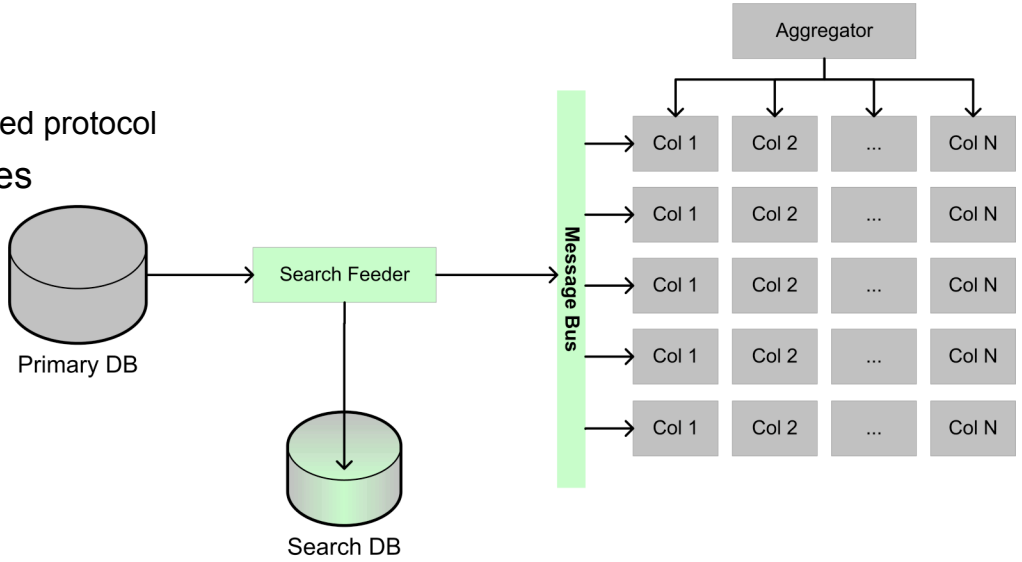
Pattern: Event Queue

- Primary use-case produces event
  - Create event (*ITEM.NEW*, *ITEM.SOLD*) transactionally with primary insert/update
- Consumers subscribe to event
  - At least once delivery
  - No guaranteed order
  - Idempotency and readback



Pattern: Message Multicast

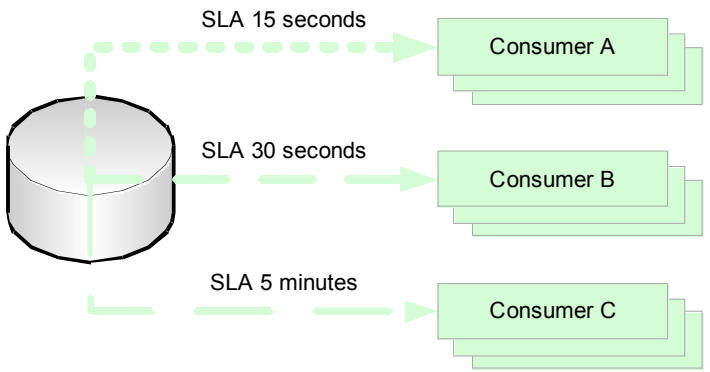
- Search Feeder publishes item updates
  - Reads item updates from primary database
  - Publishes sequenced updates via SRM-inspired protocol
- Nodes listen to assigned subset of messages
  - Update in-memory index in real time
  - Request recovery (NAK) when messages are missed



Principle 3: Automate Everything

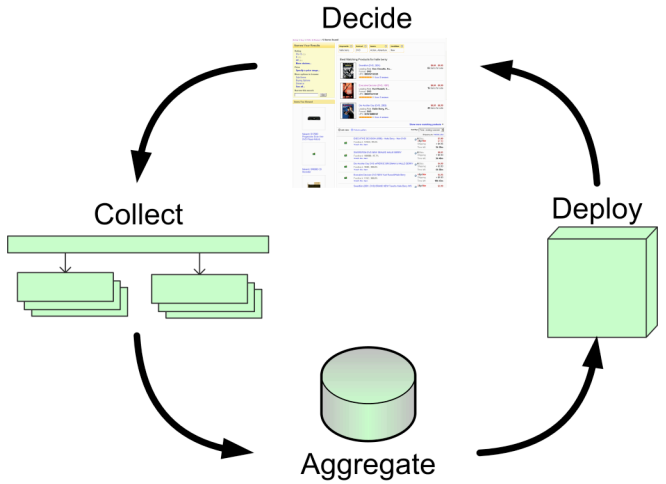
Pattern: Adaptive Configuration

- Define SLA for a given logical consumer
  - E.g., 99% of events processed in 15 seconds
- Dynamically adjust config to meet defined SLA



Pattern: Machine Learning

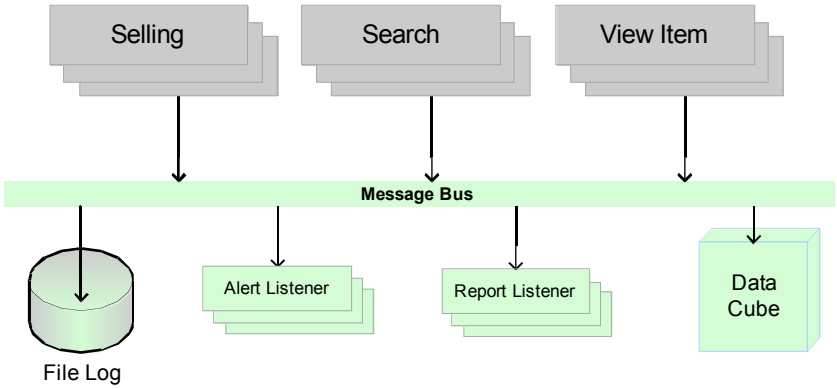
- Dynamically adapt search experience
  - Determine best inventory and assemble optimal page for that user and context
- Feedback loop enables system to learn and improve over time
  - Collect user behavior
  - Aggregate and analyze offline
  - Deploy updated metadata
  - Decide and serve appropriate experience
- Perturbation and dampening



Principle 4: Everything Fails

Pattern: Failure Detection

- Servers log all requests
  - Log all application activity, database and service calls on multicast message bus
  - ***Over 2TB of log messages per day***
- Listeners automate failure detection and notification



Pattern: Rollback

- *Absolutely no changes to the site which cannot be undone (!)*
- Every feature has on / off state driven by central configuration
  - Feature can be immediately turned off for operational or business reasons
  - Features can be deployed "wired-off" to unroll dependencies

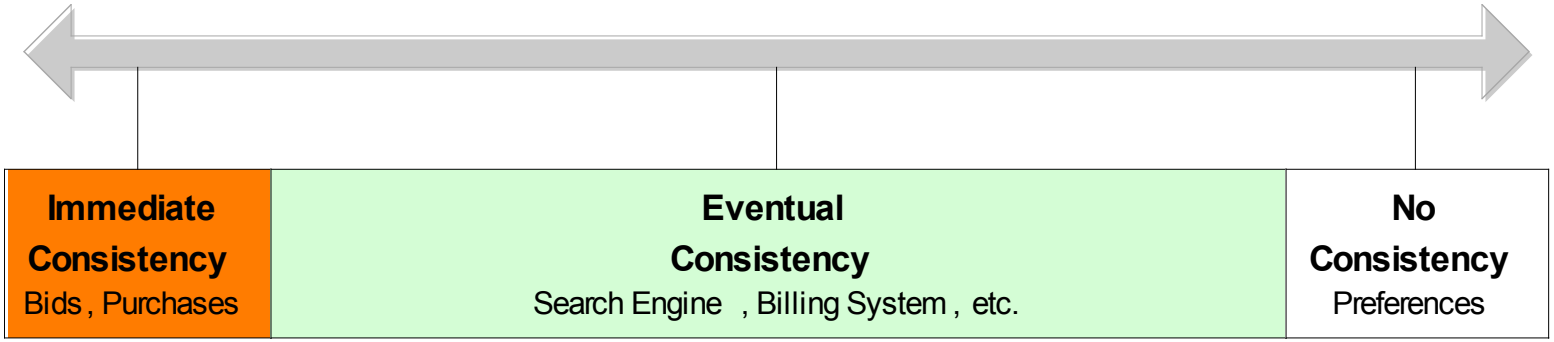
Pattern: Graceful Degradation

- Application "marks down" an unavailable or distressed resource
- Non-critical functionality is removed or ignored
- Critical functionality is retried or deferred

Principle 5: Embrace Inconsistency

Choose Appropriate Consistency Guarantees

- Brewer's CAP Theorem
  - To guarantee availability and partition-tolerance, we trade off immediate consistency
- Consistency is a spectrum, not binary
- Prefer eventual consistency to immediate consistency



Avoid Distributed Transactions

- eBay does absolutely no distributed transactions – no two-phase commit
- Minimize inconsistency through state machines and careful ordering of operations
- Eventual consistency through asynchronous event or reconciliation batch

The journey ahead

What sets the new course

Lower cost of operation

- ***power, power, power***
- ***manage complexity***

Business enablers

- ***Flexibility***
- ***Feature velocity***

TPS/Watt

TTM

Parallel Efficiency

Gap from model

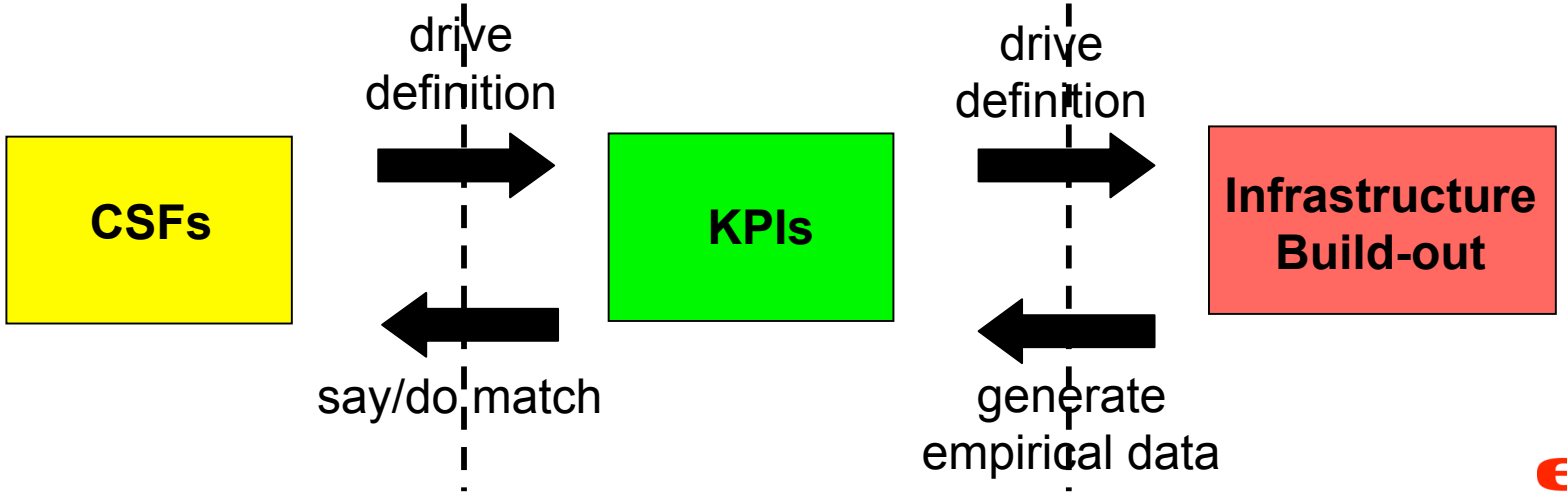
Incident repeat rate

Many-cores

Virtualization

SSD

Model-driven Management



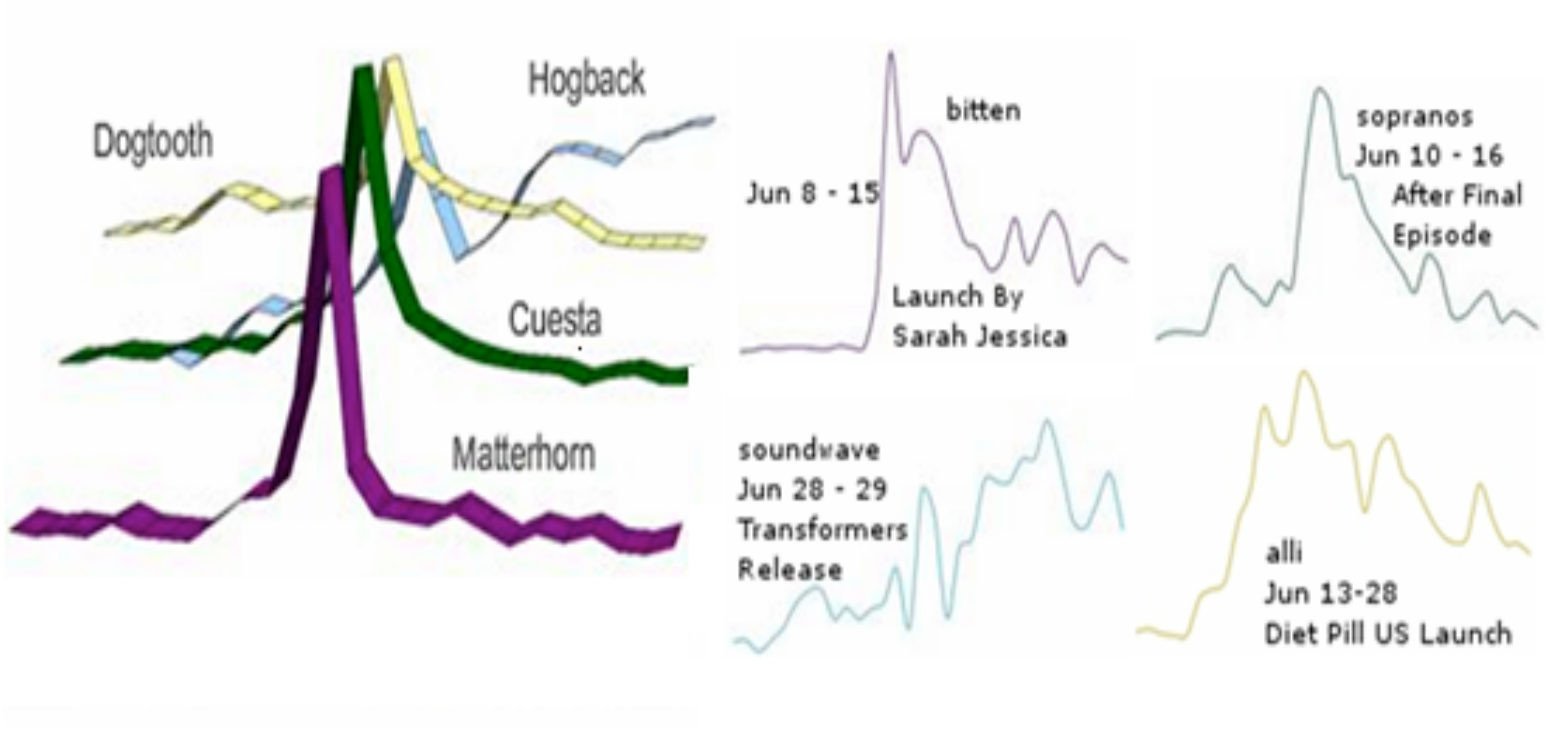
Challenge: Meta-data unstable equilibrium

- We manage relationships rather than isolated thingies
- Syntax is easy, semantics are tough
- Relax requirements for 100% semantic consistency across the site
  - for business meta-data
  - for infrastructure meta-data
- Leap over islands of semantically consistent meta-data
- Specialize in ontology build-outs and code-gens

Challenge: From reactive to proactive-und-predictable

- Model-centric architecture
  - Model reflects relationships, behaviors, constraints
  - Applies to discrete components throughout the whole system of systems
- Failure ≡ gap from model
  - Failure management enacted upon crossing tolerance threshold T
- What's the Erlang-B equivalent of failing a service request by a random user?
  - Once I know what that is, can I actively manage it??

Example: “Surprise Theory”



- Causality between what’s out there and what we see in our Data Centers
- We need real time, stable burst detection system

Nish Parikh, Neel Sundaresan, Scalable and Near Real-time Burst Detection from eCommerce Queries, KDD ‘08, Aug 24-27, 2008 Las Vegas

eBay Research Labs

Challenge: Emergent behaviors

- From a ***complicated system*** to a ***complex system***
- Machine-learning pattern supported by selective feedback loops
  - Damp excess stimuli
  - Skeptics
- Crowd control in the presence of cascading failures
  - Sample use case: an application's failure that takes nested load balancers astray
- Ratchet me to the ***edge of chaos*** and tell me once I get there!

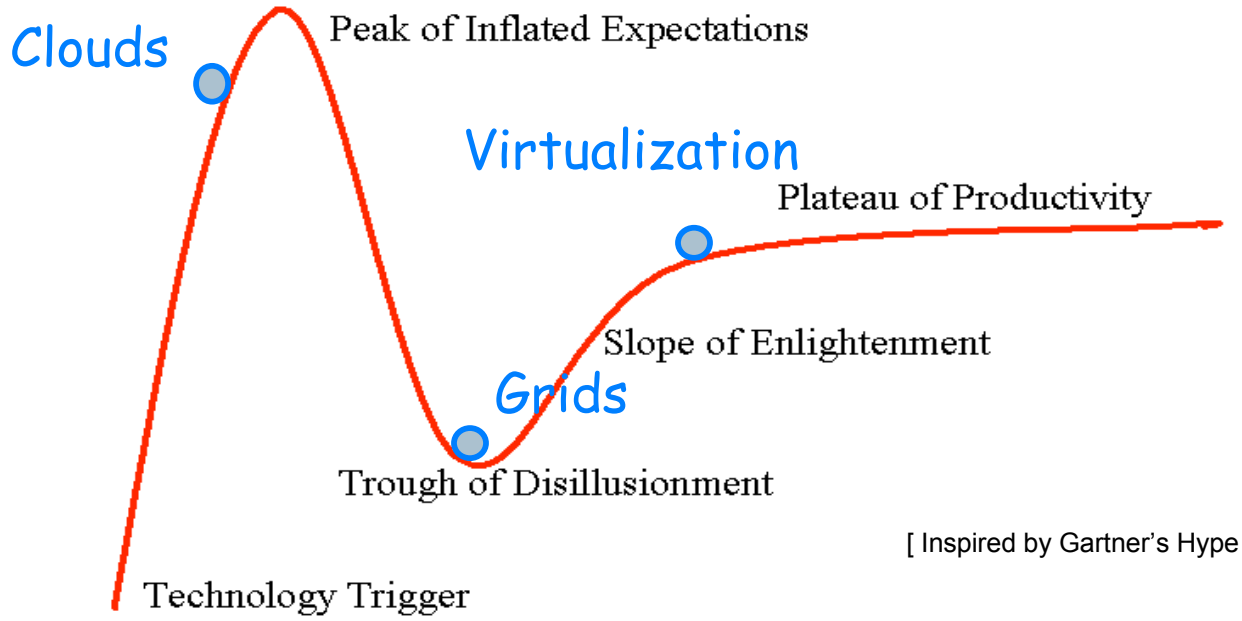
Challenge: Fail fast, fail cheap

- Experience-driven design
  - Design evolves incrementally with experience deriving from a live user community
  - Contrast against controlled conditions and synthetic workloads
  - Just say PlanetLab !
- Improve in a substantive way
  - 3% improvement doesn't cut it
  - It better be worthy of changing engines in a mid-air plane…

Friends or Foes?

Hype Cycles

- System Virtualization
- Grid Computing
- Utility Computing
- SOI, SaaS, PaaS, TWEaaS ;-)



[Inspired by Gartner's Hype Curve™]

Had he had a choice in ‘95…

… would Pierre Omidyar have started eBay.com off a Cloud?

Quite likely.

Will eBay be moving to Clouds

Ronald Coase
Nobel Laureat, 1991

"a firm will tend to expand until the costs of organizing an extra transaction within the firm become equal to the costs of carrying out the same transaction by means of an exchange on the the open market…."
-- "The Nature of the Firm" (1937)

- Within eBay, "utility" paradigms aplenty

Conclusions

- Five principles have been eBay's leading light during its scale-out journey
- We see some new forces at play
  - e.g., power efficiency, feature velocity
- These five principles continue to apply (more so than ever!)
- New challenges hit on a variety of established CS disciplines
- Piercing through Clouds … while avoiding any *Faux PaaS* ;-)

About the Presenters

Randy Shoup has been the primary architect for eBay's search infrastructure since 2004. Prior to eBay, Randy was Chief Architect and Technical Fellow at Tumbleweed Communications, and has also held a variety of software development and architecture roles at Oracle and Informatica.

Franco Travostino is the lead architect for virtualization and cloud computing. Prior to eBay, Franco has enjoyed several journeys in research and development, with features in operating systems, data communication, storage area networks and security. Franco is the first author for the book: “Grid Networks: Enabling Grids with advanced communication technology” (Wiley, 2006).