

# Rebels vs. Empires: The Cycle of Disruption and Discontinuity in Technology and Business

Neil MacDonald

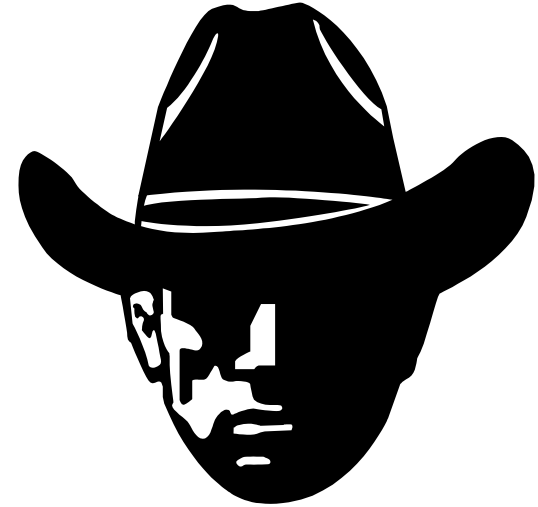


Midsize Enterprise  
Summit. 2007



# 'Something is Happening, and We Don't Know What It Is'

- Governments competing with non-governments
- Companies competing with non-companies
- Products compete with non-products
- Platforms compete with non-platforms
- IT departments compete with rebellious users
  - "Guerilla IT," "Wild West intranets," "cowboy programmers"



- ***Is this something really new, or has it not always been this way?***
- ***Are there different types of disruptive change?***
  - Some based on tech discontinuities
  - Others based on nontech factors

# 'Old School' Competitors Face Off

- Ringing Bros vs. Barnum & Bailey
- Burger King vs. McDonalds
- Toyota vs. Honda
- FedEx vs. UPS
- Coke vs. Pepsi
- Dell vs. Gateway
- Yahoo vs. Google
- Microsoft vs. IBM
- .NET vs. Java

**Competitors in a category evolve to eliminate differentiators.**

# When Do Rebel Movements Arise?

## When an Empire Achieves Total Dominion

- If you "*suck the oxygen out of the system*" (as Microsoft said it was doing to Netscape), then **anaerobic** bacteria will arise.
- If competitors cannot be successful with conventional rules of engagement, then there will emerge unconventional competitors with different rules of engagement.

### Traditional competition:

- Dell vs. Gateway
- IBM vs. Microsoft
- Microsoft vs. Netscape
- Google vs. Yahoo

### Unconventional competition:

- Microsoft vs. Open source
- IBM vs. JBoss
- Microsoft vs. Google
- New York Times vs. Craig's List
- Ringling Bros vs. Cirque du Soleil

# Microsoft vs. Google

## Different Product/Service Mix, Same Rules of Engagement

### Microsoft

- Product company selling CDs
- Coarse-grained license revenue
- Coarse-grained update cycle
- Formal product release
- Installed on client or server
- Built on silicon/circuit substrate
- Mature, broad scope

**Product company evolves to service company**

### Google

- Service company selling ads
- Fine-grained pay-per-click
- Fine-grained continuous updates
- Perpetual beta status
- Bits stream from "The Cloud"
- Built on open-source software
- Initially narrow in scope

**Software-as-service now, desktop footprint coming**

# Microsoft vs. the Open-Source Movement

## Different Rules of Engagement

### Microsoft

- Product company selling CDs
- Coarse-grained license revenue
- Coarse-grained update cycle
- Formal product release
- Installed on client or server
- Built on silicon/circuit substrate
- Mature, broad scope

**Product company evolves to service company**

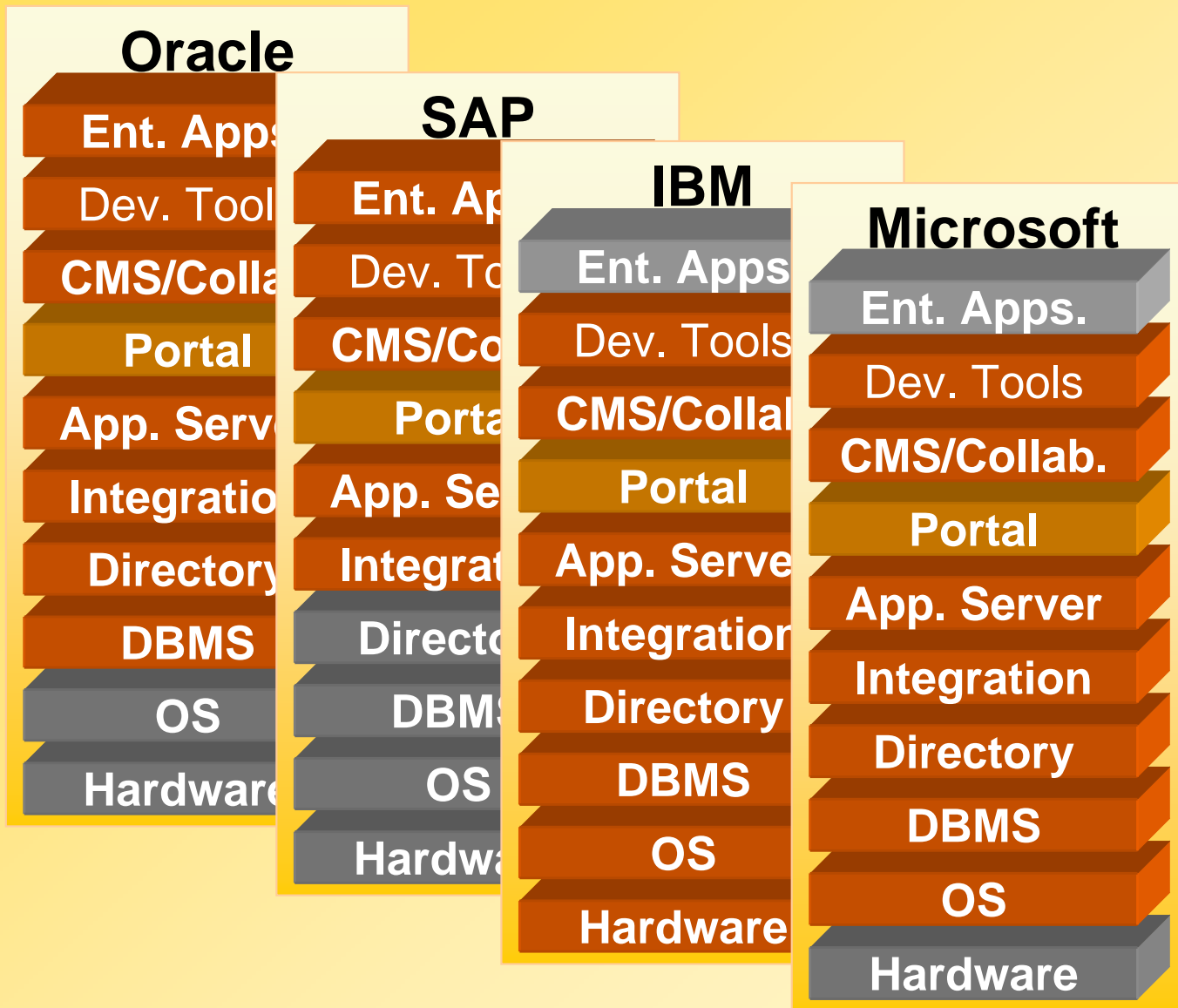
### Open Source

- Simple Web site, free downloads
- Donations via PayPal, AdSense
- Fine-grained continuous updates
- Formal or informal releases
- Installed on client or server
- Built on open-source software
- Initially narrow in scope

**Evolving toward conventional corporate entities**

**Similar analysis can be made for IBM/BEA vs. open-source JBoss.**

# Orderly Vendor Stacks vs. Non-Stacks



**Collection  
of Web APIs**



**Well-Defined Bricks vs. Unshaped Mortar**

# Platforms and Products vs. Nonentities

## Platforms

- Complete, robust, supported
- Enables an ecosystem
- Compatible with context
- Scope is range of apps.

*Examples: .NET, Java, ABAP, Flex*  
***Supports many-layered stack***

VS.

## Non-Platforms

- Incomplete, slim support
- Ecosystem is absent or emergent
- Unique — compatibility emergent
- Narrow scope: one type of app.

*Examples: APIs for Delicious, Flickr*  
***One vertical slice in the layer***

## Products

- Discrete unit (SKU)
- Targeted to specific usage scenarios
- Fixed price
- Specific identifiable source/creator
- Distinct releases

*Examples: Microsoft Office, Ajax framework*

VS.

## Non-Products

- Indistinct identity — often one feature
- Use is indirect or nonspecific
- No price — or highly dynamic price
- No clear source or author
- Continuous evolution

*Examples: Writely, Ajax snippets*

# Two Software Development Styles: The 'Cathedral vs. Bazaar' Dichotomy

## Cathedral

- Centralized, top-down, hierarchically organized
- Linear, noncontinuous "chunky" process, "omniscient" management
- Customers at arms length, recipients of a one-way delivery of releases
- Bugs can lurk in deep corners for a long time, long release schedule
- Everybody gets paid, whether they deliver or not

## Bazaar

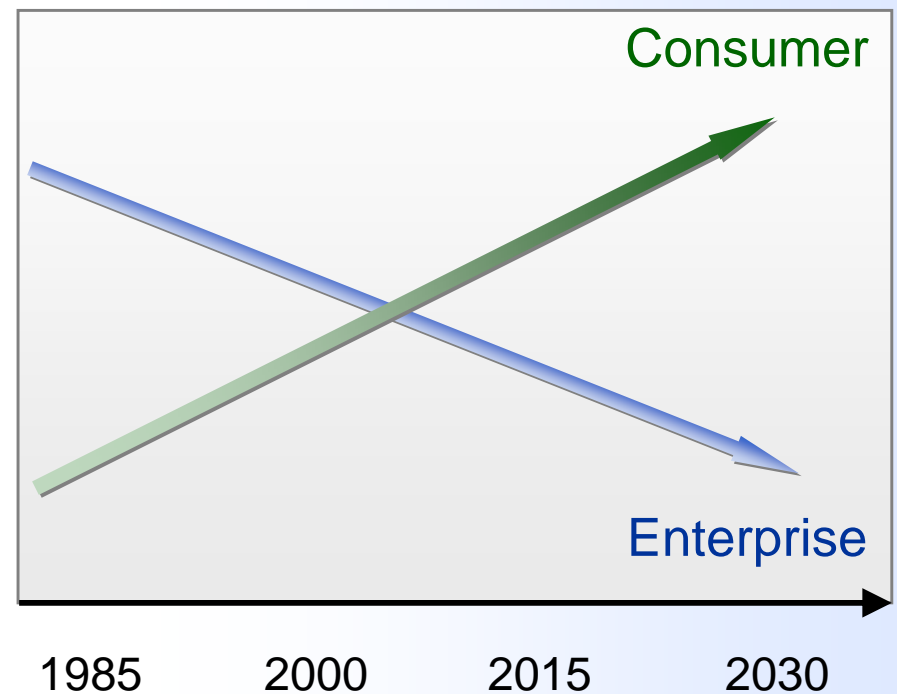
- Decentralized, peer-oriented, "self organized"
- Highly parallel, continuous process
- Rapid feedback is key to rapid evolution and code improvement
- "Customers" are co-developers
- Development is competitive process, meritocratic
- Nobody gets paid, except for nonfinancial incentives
- Of the first 10 people to download Linux, five sent back fixes and new features

*Concept from The Cathedral & The Bazaar,  
Eric Raymond, O'Reilly Press, 1999*

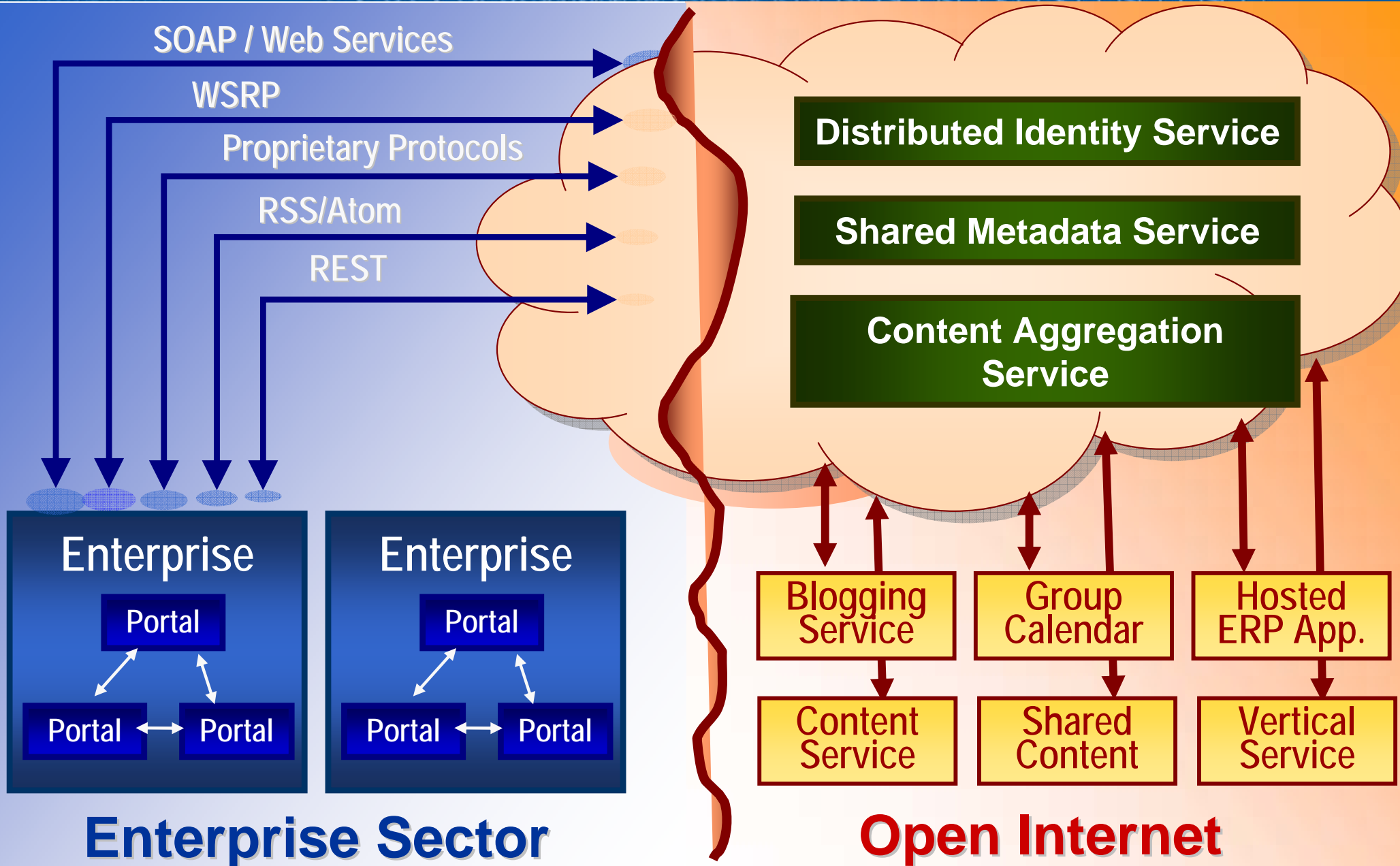
# The Consumer Is King: Trickle Down Becoming Trickle Up

- Consumers driving economy
- Technology not just for geeks
- Consumers getting cutting-edge technology first
- Social norms enable agility
- Markets beat central authority: Navigate the public highways
- It's the ecosystem, not the enterprise: global-class computing
- Consumers driving enterprises
- Second Internet revolution is driven by consumers and drives enterprise adoption
- Growth is the new business mantra
- The largest change in IT will come from consumers
- Barriers: legal, regulatory and security risks, loss of control

## *Trickle Down Becoming Trickle Up*

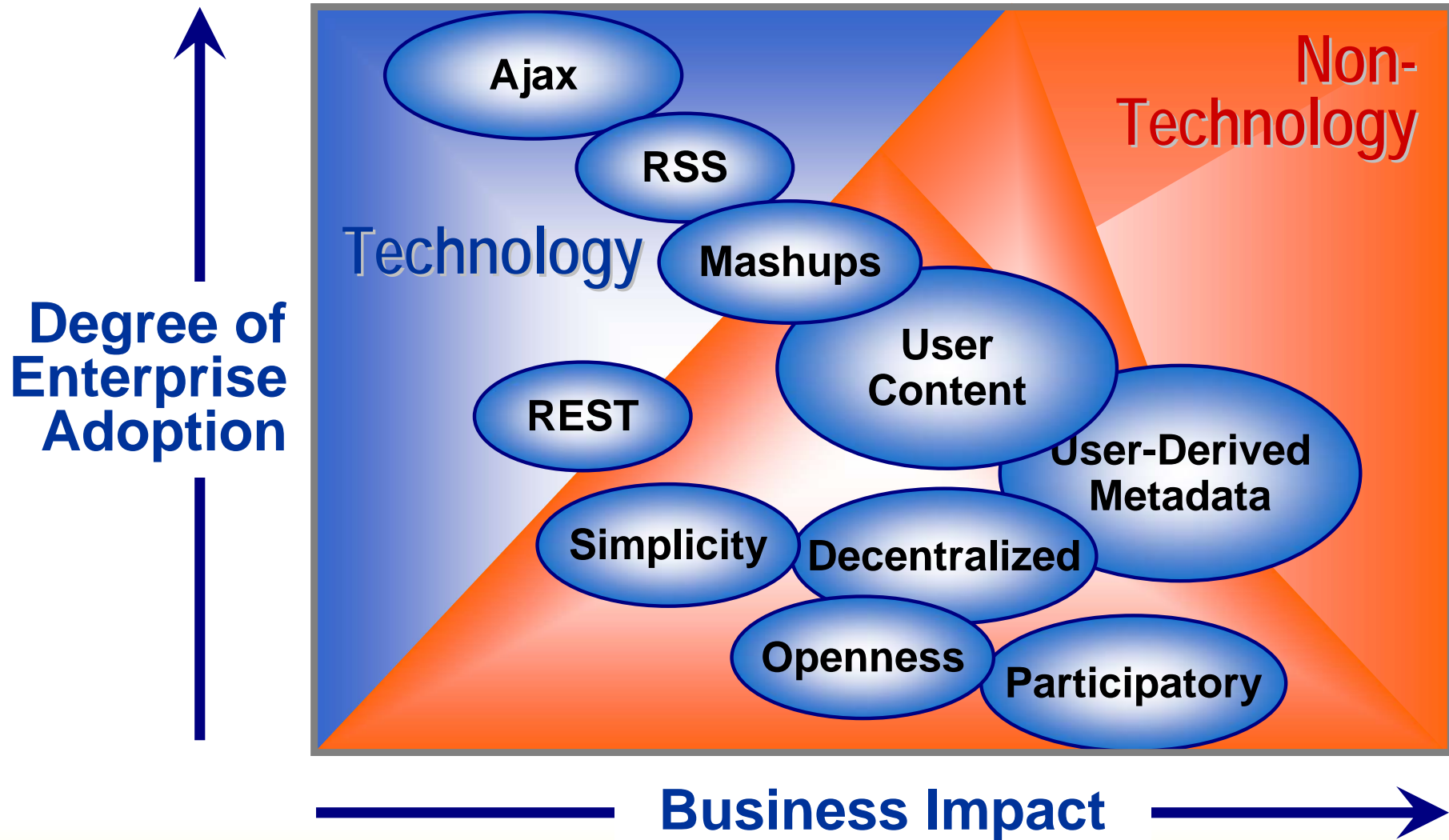


# Enterprise Architecture vs. the Public Cloud



# Web 2.0 Wave Washes Over the Enterprise

## Enterprises Will Adopt the Least-Important Aspects First



**Moral:** High adoption does not necessarily mean high impact

# The Facebook.com: A Case Study in Rebel Technology & Attitude

## "Opportunistic" start

- One college sophomore (psych major)
  - Wrote code in 10 days during finals
  - Platform: Linux, Apache, MySQL, PHP
  - Launch date: February 2004

## "Systematic" status after 18 months

- 8 million unique visitors
- 5 million registered users
- 5.5 billion page views per month
- Now the 7th most visited site in the U.S.
- 70% of registered users visit **every day**; 93% repeat every month
- Company less than 40 people; most are under 25 years old
- Turned down a buyout offer of \$750M — reportedly asking \$2 **billion**
- Still running on MySQL, PHP, Linux



# What's After Web 2.0: The Big Wheel Just Keeps on Turning

**The  
Open  
Web**

**Web 2.0**

**2006**

**2004**

**The  
Closed  
Web**

**2002**

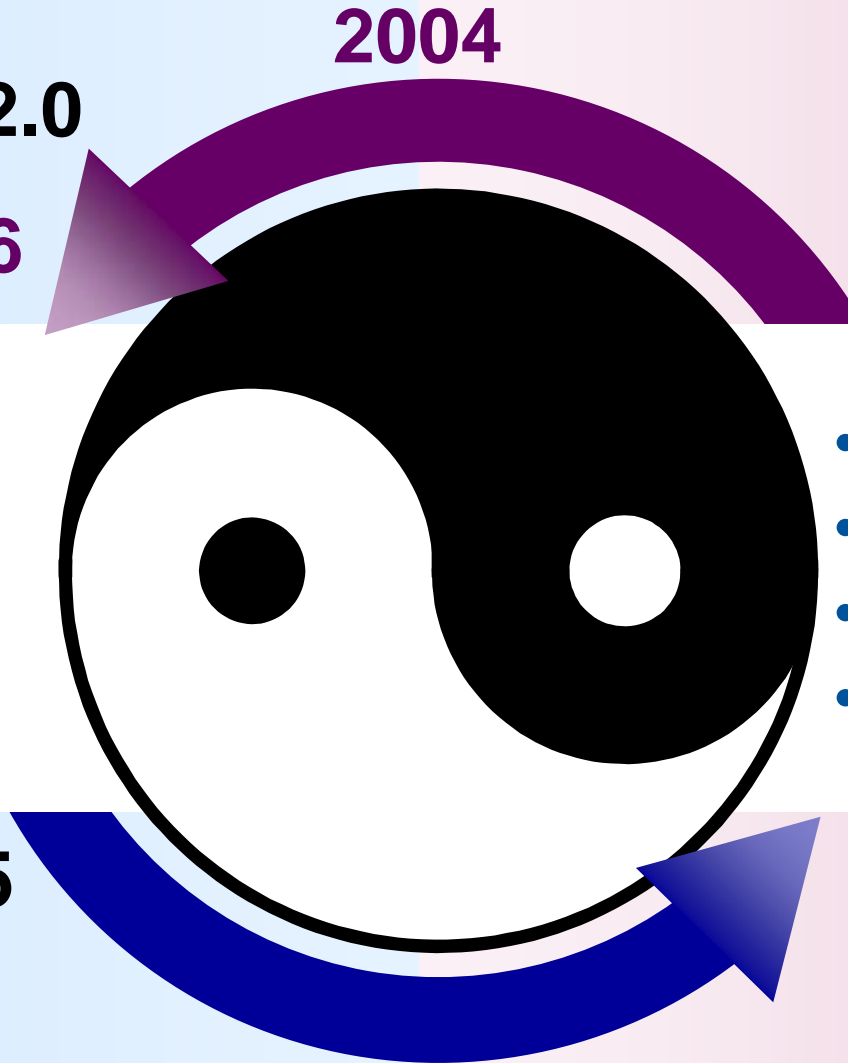
- Open-source
- Collaborative
- Decentralized
- Lightweight
- User-centered

- "First mover"
- "Monetize eyeballs"
- "Enterprise Web"
- Property-centered

**Web 0.5**  
**1992**

**1997**

**Web 1.0**  
**1999**



# Commercial Goes Open-Source

# Open Source Goes Commercial

## Open Source

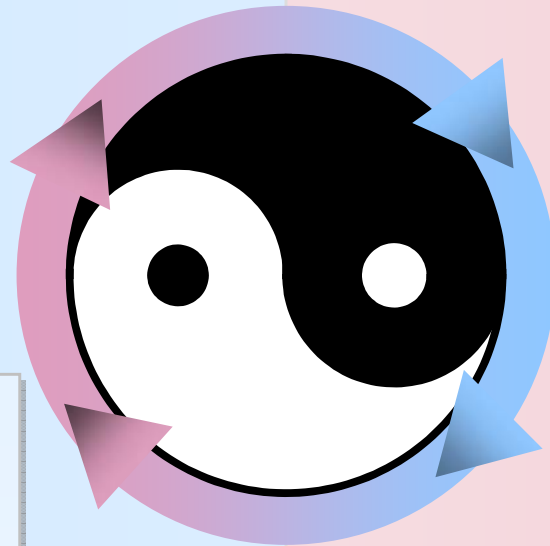
- Red Hat & Novell distros
- Red Hat acquires JBoss
- MySQL licensing
- Sourcefire (SNORT)

### Vendor Motivations

- Transparency
- Counter the insurgency
- Lower-end offerings
- Market disruption

### Your Motivations

- Lower cost
- Reduce vendor lock-in



### Vendor Motivations:

- Food, shelter
- IPO or acquisition \$
- It cost money to support

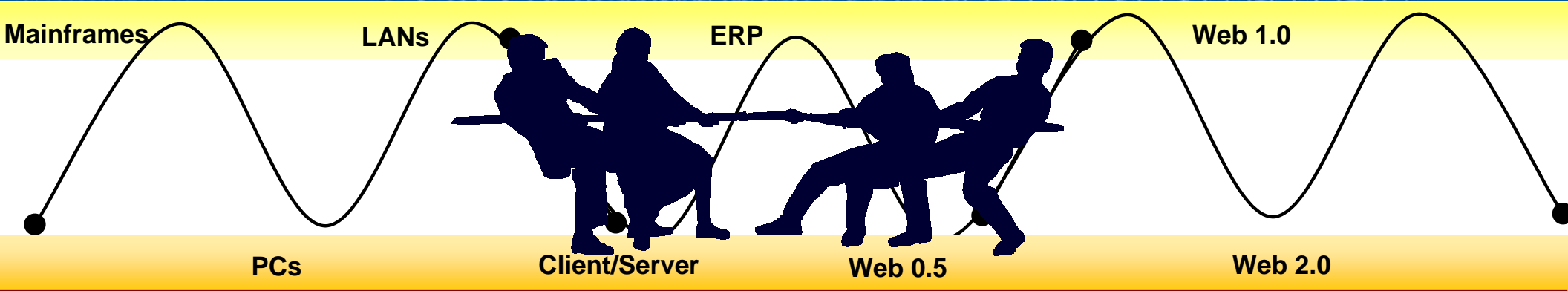
### Your Motivations

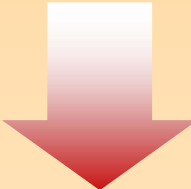
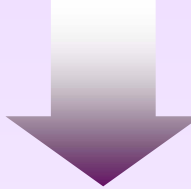
- Predictable support
- Security fixes
- Someone responsible

## Commercial

- Sun Microsystems Solaris 10
- Microsoft's permissive license
- Oracle InnoDB and Sleepycat
- IBM Geronimo

# Confusing the Means With the End: Are You Forcing Your Users to Go Anaerobic?



What the business wants (end 2.0)	What IT thought it wanted (end 1.0)	What IT did (means)
<ul style="list-style-type: none"> <li>• Business value</li> <li>• Managed risks</li> <li>• Flexibility and agility</li> <li>• "Secure enough"</li> </ul> <p><i>Where the business defines its tolerance for risk, <b>not</b> IT</i></p>  <p><b>IT as an enabler of change</b></p>	<ul style="list-style-type: none"> <li>• Reduced cost</li> <li>• Reduced risk</li> <li>• Remove redundancy</li> <li>• Information availability</li> </ul>	<ul style="list-style-type: none"> <li>• "Lockdown"</li> <li>• Rigid standards</li> <li>• (A)rchitecture</li> <li>• Systematic projects</li> </ul>  <p><b>IT as the Empire IT as an inhibitor of change</b></p>

# Opportunistic and Systematic Can Coexist

If You Accept: 'Not Everything Worth Doing Is Worth Doing Well'

	<b>Opportunistic</b>	<b>Systematic</b>
<b>Business driver</b>	tactical	strategic
<b>Approach</b>	tool-driven	model-driven
<b>Code reuse</b>	low	medium
<b>Requirements</b>	simple	complex
<b>Scope</b>	departmental	enterprise
<b>Skills</b>	junior developer	senior developer
<b>Team size</b>	1 to 6	6 to 60+
<b>Code longevity</b>	6 months to 3 years	2 years to 10+ years
<b>Budget</b>	small to medium	large
<b>Methodology</b>	informal	formal, systematic
<b>Risk tolerance</b>	high	low
<b>Project duration</b>	1 to 6 months	6 months to 6 years
<b>Connectedness</b>	low to medium	medium
<b>Code refactoring</b>	low	low
<b>Integration use</b>	low	medium

# Software Architecture as an Unintended Consequence of Social Architecture

## Context

- The structure of a system reflects the structure of the organization that built it.
- An organization with constrained lines of communication may build systems that don't interoperate and are difficult to monitor.
- Attempts to change or improve the structure of the system are often reversed by forces which snap the system back to its original shape (this can be good or bad).
- In the case of dysfunctional organizations, it can mean a continual struggle against design defects (bugs and performance issues).

## Solution

- To fix or refactor the software artifact, refactor or restructure the organization that created it. Of course, this can be very difficult.
- Examine and cultivate process attributes: collaborative, community-based, transparent, repeatable and efficient.
- Implications for SOA and Web 2.0: Will the tail wag the dog?

# Recommendations: Disrupting IT (Defuse the Insurgency)

## "Red Ocean"

- Architect for purpose
- "Lockdown"
- Use standard technology

## • "SOA"

- Enterprise applications
  - Web services
  - Enterprise components
- Enterprise reuse of components

## • IT develops applications

- Heavyweight protocols (SOAP, WS-\*)
- Heavyweight development

## "Blue Ocean"

- Architect for change
- Employee-owned PCs
- Embrace consumer technology
  - Multiple IM clients, blogs
  - Alternative browsers

## • "SOA"

- Think *service orientation*
- Web architectures
- Portals as path to service orientation

- Enterprise mashups
  - Web-based services: RSS, Ajax
  - Enterprise services
- Global reuse of open-source code
  - SourceForge, RubyForge
  - Krugle, sourcebank search engines

## • Users develop applications, modify their own processes

- Departmental portal-based composition
- Application and process orchestration

- Lightweight protocols (REST)
- Lightweight development (SCRUM)

# Rebels vs. Empires: The Cycle of Disruption and Discontinuity in Technology and Business

Neil MacDonald



Midsize Enterprise  
Summit. 2007



# Rebels vs. Empires: The Cycle of Disruption and Discontinuity in Technology and Business

Neil MacDonald



Midsize Enterprise  
Summit. 2007

