# The Open Source Paradigm Shift

**Tim O'Reilly**

**O'Reilly & Associates, Inc.**
*www.oreilly.com*

**June 2003**

O'REILLY®

# Paradigm Shift

A change in world view that calls everything you know into question

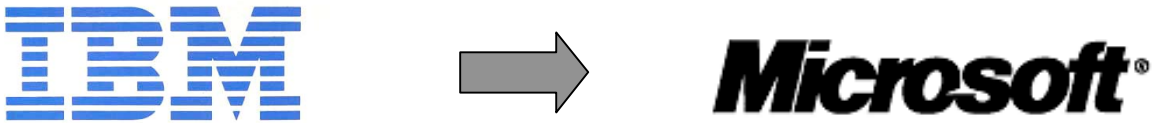
# The PC Paradigm Shift (Hardware)

- Commodity hardware with an open architecture – IBM beats Apple
- Low cost and a pure play commodity hardware business model beat proprietary add-ons – Dell beats IBM and Compaq
- Companies stuck in the old paradigm die off: Digital, Data General, Prime

- Even open architectures have proprietary components – Intel Inside®

# The PC Paradigm Shift (Software)

- Software decoupled from hardware
- Lock-in and competitive advantage move to software – Microsoft beats IBM



# Paradigm Failure at Work!

- Linux critic: “There are no user-friendly applications on Linux”
- Linux advocate: Have you seen the latest release of Gnome, OpenOffice, or the Gimp?

What's being missed here?

# User Friendly Applications on Linux

# What's Wrong with This Picture?

- These applications are being created by open source developers and run on an open source platform, but…
  - Source code is not distributed (and it wouldn't be useful to many developers if it were)
  - Licenses triggered by binary software distribution have no effect
  - The value in these applications is in their data and their customer interactions more than in their software
  - Most are fiercely proprietary

# The Internet Paradigm Shift

- Commodity software with an open architecture
- Information applications decoupled from both hardware and software
- Competitive advantage and revenue opportunities move "up the stack" to services above the level of a single device.
- Lock in is based on data and customer relationships, not proprietary software
- Intel is still Inside, but so is Cisco, and eventually others -- there's plenty of room at the bottom as well as at the top

“I’m an inventor.
I became interested in long term trends because an invention has to make sense in the world in which it is finished, not the world in which it is started.”

-Ray Kurzweil

# What Really Matters: the Three C's

The three deep trends:

- **C**ommoditization of software
- User-**C**ustomizable systems and architectures
- Network-enabled **C**ollaboration

# Software as Commodity

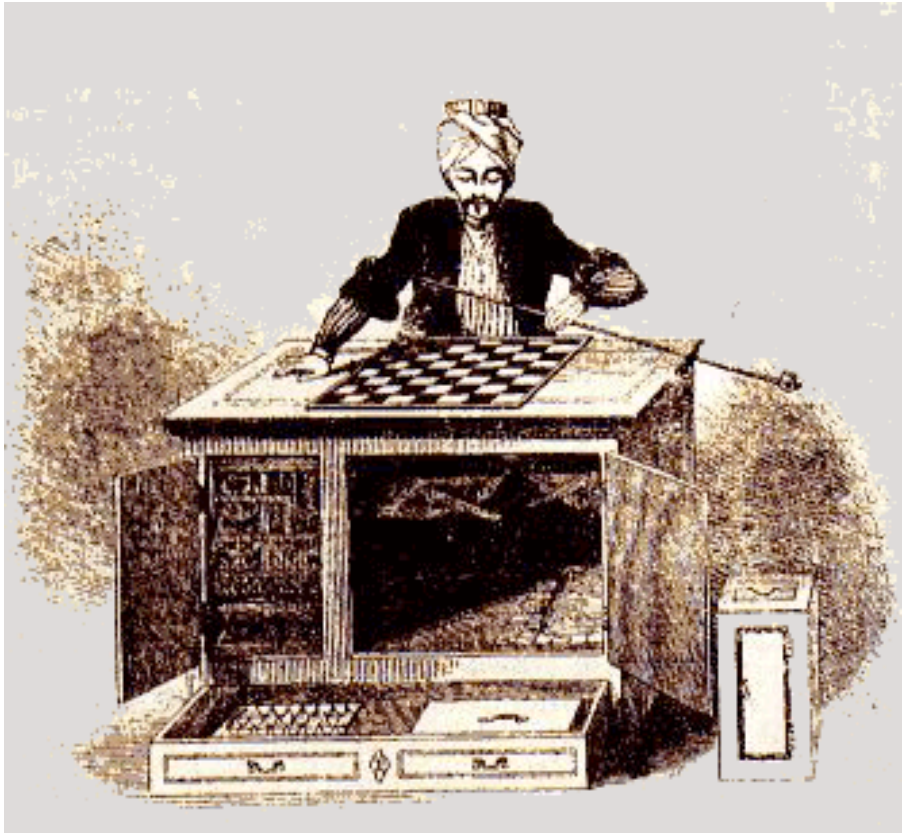
- Linux on Intel gives 10x savings
- Apache means web serving is not a revenue opportunity
- MySQL threatens to do the same for databases
- Open source promotes competition and drives down margins
- Open Unix/Linux/Internet architecture makes "plug-compatible" software the norm
- Proprietary alternatives must become free (as in beer) to compete - usually bundled with added value components

# The Internet Application Platform

- Commodity Intel hardware
- The Internet protocol stack and utilities like BIND
- LAMP
  - **L**inux (or FreeBSD)
  - **A**pache
  - **M**ySQL
  - **P**HP (or Perl, or Python)
- Platform-agnostic client front ends

# Software Customization, or Why the 'P' Matters So Much

Von Kempelen's Mechanical Turk



# Customizability at Work

- Software is built for use in delivering services, not for sale
- Internet-era applications are updated daily, not yearly
- Interfaces are built with dynamic data, not just software - you might call this "infoware"
- Dynamic languages like PHP, Perl, Python are key to managing infoware interfaces and gluing together software components

# Network-Enabled Collaboration

- Usenet: the real mother of open source
- Software development teams can be distributed, even internationally
- The “Adhocracy” - like-minded developers can find each other and work in ever-shifting groups
- Power shifts from companies to individuals
- Users help to build the application

# What's more...

- Collaborative techniques are increasingly being applied to proprietary software

- With a large-enough development organization, OSS-like behavior emerges

# Collaboration at the Data Layer

- James Kent creates open source gene assembler to keep the human genome in the public domain, assembling the work of hundreds of independent scientists
- Napster/Kazaa users build song swapping network as byproduct of their own self interest
- Google leverages millions of independent linkers via PageRank algorithm
- More people have "contributed" to Amazon than to Linux!

# Business Model Thoughts for Commodity Software

- Linux as the BIOS of the Internet OS
- IBM WebSphere = Compaq
- ??? = Dell
- There are many possible "Intels Inside". Not just LAMP, but:
  - J2EE
  - .Net
  - Specific IOS services, such as digital identity or search (Google replaces the DNS)

# Hidden Service Business Models in Open Source Software

- Not just “professional services,” but services delivered to end users
- UUnet, not RedHat - greatest open source business success to date
- BIND - a monopoly in disguise
- Sendmail and Apache - not software sales but email and web hosting
- Google, Paypal, Amazon et al - the next step on the path to a service-based software economy

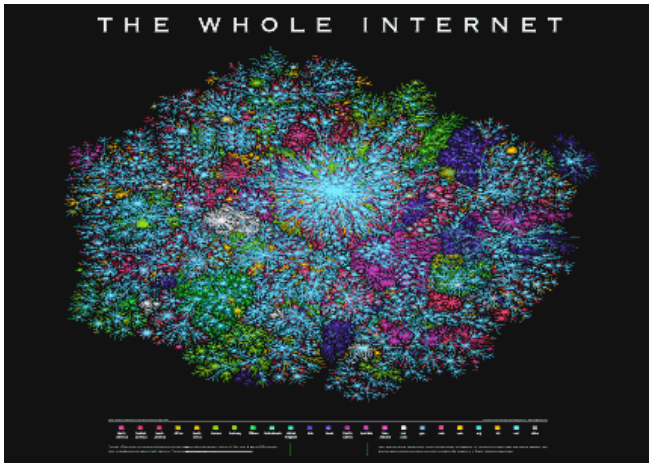
# What Keeps Me Up at Night?

- Internet application providers have gained from open source, but haven't been trained to give back, and are ignored by OSS advocates
- Meanwhile, owning user data is the new source of lock-in
- What's more, a platform strategy beats an application strategy every time:
  - “Windows is just a bag of drivers.” (Marc Andreesen)
  - “It's just like GUI.  Nobody owns it.” (Jim Allchin)

# Two Types of Platform

- One Ring to Rule Them All

- Small Pieces Loosely Joined



# Small Pieces Loosely Joined

- An architecture of participation means that your users help to extend your platform
- Low barriers to experimentation mean that the system is "hacker friendly" for maximum innovation
- Interoperability means that one component or service can be swapped out if a better one comes along
- "Lock-in" comes because others depend on the benefit from your services, not because you're completely in control

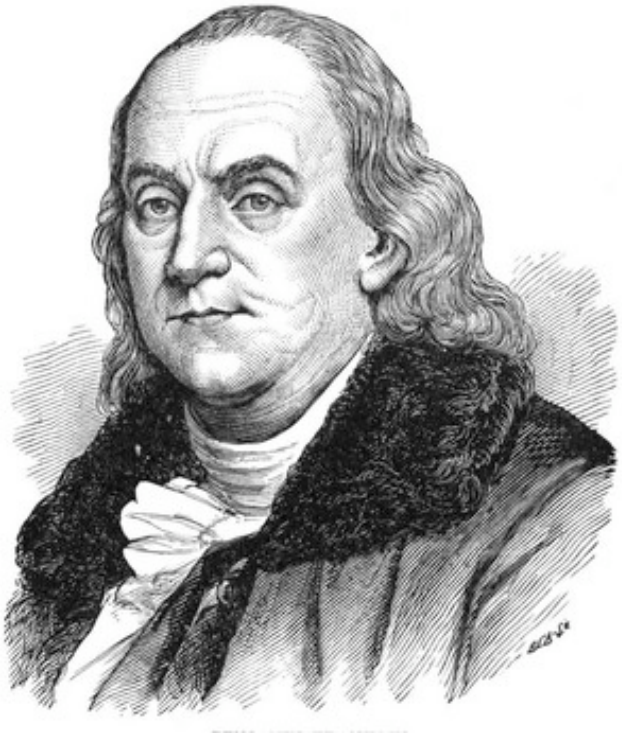
# So What Do We Need to Do?

- Correctly characterize the OSS heritage
  - The native development methodology of the Internet
  - The Internet is OSS's greatest success to date
  - Interoperability and open data formats may be more important than source code availability

# Adhere to Open Standards

"We must all hang together or we will assuredly all hang separately."

—Ben Franklin



# Reinvent the Opportunity for Surprise

- New paradigms usually involve disruptive technologies
  - Poorly understood at first
  - Don’t work as well as existing technologies
  - Don’t have a clear business model
- Low barriers to entry spark innovation
  - Reusable components mean that you can build on the work of others

# Watch the Alpha Geeks

- New technologies first exploited by hackers, then entrepreneurs, then platform players
- Two examples
  - Screen scraping predicts web services
  - Wireless community networks predict universal Wi-Fi

*Rob Flickenger and his potato chip can antenna*

# Embrace the New Paradigm

- Use commodity software components to drive down prices for users
- Give customers increased opportunity for customization
  - Plug-replaceable standards-compliant components
  - Extensible architecture
  - Scripting support
- Look for hidden service business models
- Leverage collaborative development processes and participatory interfaces

# Rethink Open Source in the Context of Web Services

- Google and Amazon APIs treat web applications and their data as programmable components
- Data re-usability may be more critical than source code availability
- Who owns the data?
- Open source represents a kind of "bill of rights" for software developers and users.  What is the bill of rights for web services?

# Final Takeaways

- As developers or investors, you have to think ‘platform’
- Build in extensibility and interoperability
- Think network, think open
- Embrace your users and the people who build on your platform as partners
- Create more value than you capture

# Questions?

- For more information

  *http://www.oreilly.com*

  *http://tim.oreilly.com*

  *http://conferences.oreilly.com*

  *http://www.oreillynet.com*

# Related Reading

- *The Structure of Scientific Revolutions*, by Thomas Kuhn
- *The Innovator's Dilemma*, by Clayton Christenson
- *The Cathedral and the Bazaar*, by Eric S. Raymond
- *Code and Other Laws of Cyberspace*, by Lawrence Lessig
- *The Cluetrain Manifesto*, by Chris Locke, Doc Searls, and David Weinberger
- *Small Pieces Loosely Joined*, by David Weinberger
- *Down and Out in the Magic Kingdom*, by Cory Doctorow