eclipsecon™ 2008

Platform Wars: The Next Generation?

Brent Williams
Managing Director, Equity Research
The Benchmark Co., LLC
bwilliams@benchmarkcap.com
brent_williams@hotmail.com

More About Me
Than You Really Wanted To Know

- 10 years' experience as Wall Street stock analyst
- Almost 20 years' experience in software business
  - Unix/C engineer, working on Ingres database in early days of database business (1986-1990).
  - Sales and marketing experience in database and application server industries.
  - Analyst at IDC, in charge of covering Microsoft when Windows NT rolled out.
  - Analyst at Gartner, in charge of covering development tools
- Goal for 2008: Extend current 2-Year streak of avoiding being labeled as MikeM's Love Puppet

Festering with Nostalgia: The Fun We've Had In the Last Year

Industry consolidation continues

- Buying for fundamentally financial reasons. Poster child: Oracle→BEA
  - 4 middleware architectures/stories to meld: Fusion, AIA, BEA WebLogic, BEA AquaLogic
  - Customers increasingly restive about return on maintenance spend. (5 years in, where's the beef?)
  - Market share ≠market dominance or pricing power.
- Defensive Buys
  - Consolidation of 3 diversified BI vendors in 2007
- "Strategic acquisitions" (a.k.a., desperation)
  - MSFT→Yahoo

SaaS becomes a “real” platform

- Now slices value in both dimensions: horizontal and vertical
- Horizontal: Outsourcing infrastructure layers separately
  - Examples: Amazon S3/EC2, Google GFS, Bigtable
- Vertical: outsource standalone apps (CRM), modules (CNQR), transactions (ADP)
  - Salesforce.com becomes an application development platform

Big proprietary vendors buying open source vendors: Powdered Rhino’s Horn or Something More?

- Oracle-Sleepycat (February 2006)
  - Gee, that worked well…
- Citrix-XenSource: catch up with VMWare
  - Citrix needs velocity, squeezed between VMW and MSFT
  - May forget open source component of Xen
- Sun-MySQL
  - Driver for ZFS and other hardware-independent storage?
  - Jury still out on this one
- Nokia-Trolltech
  - Maybe it’s just because they’re both Scandinavian

Mega-Vendors’ “Bambi vs Godzilla” Open Source Moves, A Year Later

- Oracle “Cloning” of Red Hat Linux still stubbornly refusing to generate revenue despite increasing lack of interest on Oracle’s part
  - Oracle’s Xen cloning effort disappeared even faster than the Red Hat attack
- Shows value of branding and customer loyalty on something that’s supposedly a commodity
- Message to mega-vendors: if you don’t contribute to a project, you don’t sell a lot of it

Two things to talk about today

- Platform shifts
- IT user involvement in Eclipse: a key imperative

Platform shifts

Deep Dark Past of Platform Archaeology

- 1st-3rd Generation Proprietary Platforms (1965 - ~1995)
  - Mainframe, mini, Windows PC
  - Extreme architectural & vendor lock-in
  - Lower complexity enabled fantasy of “one vendor could do it all.”
  - Winners harnessed broadest addressable markets
  - Cost of lock-in could be glossed over because IT as % of revenue low, even in IT-intensive industries
- Galactic Architectures (1987-1995)
  - Everything to everyone, solving all problems
  - IBM SAA, Digital AAS, Apple VITAL, ACE Consortium, Sun ONE, etc.
  - Some (SAA) had high vendor lock-in; some (DCE) only platform lock-in
  - Some had neither (Apple VITAL) but also had no raison d’etre or business model
  - Often motivated by a *vendor* problem (defending against a new, cheaper competitor)
  - Collapsed under technology weight or utter lack of customer benefit

Current Dominant Platforms (.NET, Java)

- Current platforms succeeded because they learned from the past
  - Java: (relatively) open licensing model for multi-vendor support
  - .NET: multiple language support, focus on ease-of-development.
- But both have increasingly apparent holes
  - One data storage model and one transaction model at the center
  - One programming model (Java, .NET Framework)
  - One user interface metaphor (though fragmenting somewhat in the case of Java)
  - .NET tied to single (unappealing, high-cost) vendor for most key components
  - Both now seem to be in “defensive” mode, signaling maturity and slowing platform evolution
- By the way, they’re still vertically integrated just like past generations

When Platform Shifts Happen: High Level Theory

- Spread between fastest and slowest devices widens 10x-100x
  - 1970s-1980s minicomputer wave, client/server Windows era
- New programming models
  - Structured COBOL, DLLs, OOP, CASE, SOA, AOP, DSLs, ad infinitum
- New user interface technologies: Block mode terminal → character mode terminal, GUI, Web, SOA (machine-to-machine "UI"), Mobile
- New hardware technologies
  - Storage arrays increase reliability of *all* data – narrowing distinction between highly protected data and "ordinary" data
  - Solid-state disks changing game for transaction processing applications – transaction commit window becomes delightfully small, reducing value of specialized transaction engines
  - Creates opening for non-relational mission-critical data, and radically different architectures between OLTP and other systems
- When one-size-fits-all platform no longer fits all, there's an opening to exploit

We are at that point <u>now</u>!

Key Indicators That Something New Is Afoot

- New database vendors (not just MySQL) starting to make inroads
- Google and SaaS starting to put existing players on the defensive
- Facebook, MySpace, LinkedIn as platforms
  - They've got real APIs and *everything*…
- Continuing interest in Web 2.0
- Initial hysteria about "cloud computing" and "mesh"
  - "My PowerPoint architectural vaporware can beat up your PowerPoint architectural vaporware"

What Next-Generation Platform Might Look Like

- Multiple types of data sources at the core
  - New Database/data storage architectures
    - Synchronization and replication of data becoming more reliable and faster: server-server and server-device. Google Gears, JSR170, publishing data on RSS feeds.
    - Holes in Relational Model appear
    - Mixed relational/non-relational designs in applications
  - New transaction models and architectures to store that data reliably ("death to 2PC")
- Increasing architectural spread between platform endpoints and interconnects (clients, network speed/reliability, server performance)
- New user interface models
  - Devices, where Eclipse is well-positioned
  - New browser releases and new XHTML specifications make web apps more competitive
- Machine-to-machine interfaces (Web services/SOA)
- Exploiting continuing re-slicing of value equation
  - Industry standards allow "slicing" of value equation in multiple dimensions.
  - Thus, there is lower perceived value for deep vertical integration in customers' minds

… In other words, we're getting what Redmonk calls the "Stackless Stack"

… And what MikeM (who's Love Puppet I am *not*) calls "runtimes."

How Eclipse stacks up (so to speak) technically

- Leverages existing platforms rather than rejecting them (e.g., extends Java platform rather than introducing something else)
- Architectural room for new tactical technologies (PHP, other languages)
- Beginning of platform-like components (RCP, OSGI, BIRT)
  - Framework is flexible enough to support addition of more platform components over time because it doesn't make core assumptions (database, etc).
- Not nailed down to a particular vertical layering architecture

How Eclipse stacks up economically

- Governance model for shared R&D between competitors is proven. “Manage the chaos” rather than “pick the ultimate winners” reduces politics and maximizes code delivery
  - Should be scalable to include competing customers as well as customers who couldn’t care less about each other
- Spread R&D model allows good returns when platform focused on multiple smaller addressable markets
- Architecture enables leverage of existing technology to re-combine for multiple smaller addressable markets
- Some successes already under way with vertical market consortia (e.g., Autosar)

End user involvement

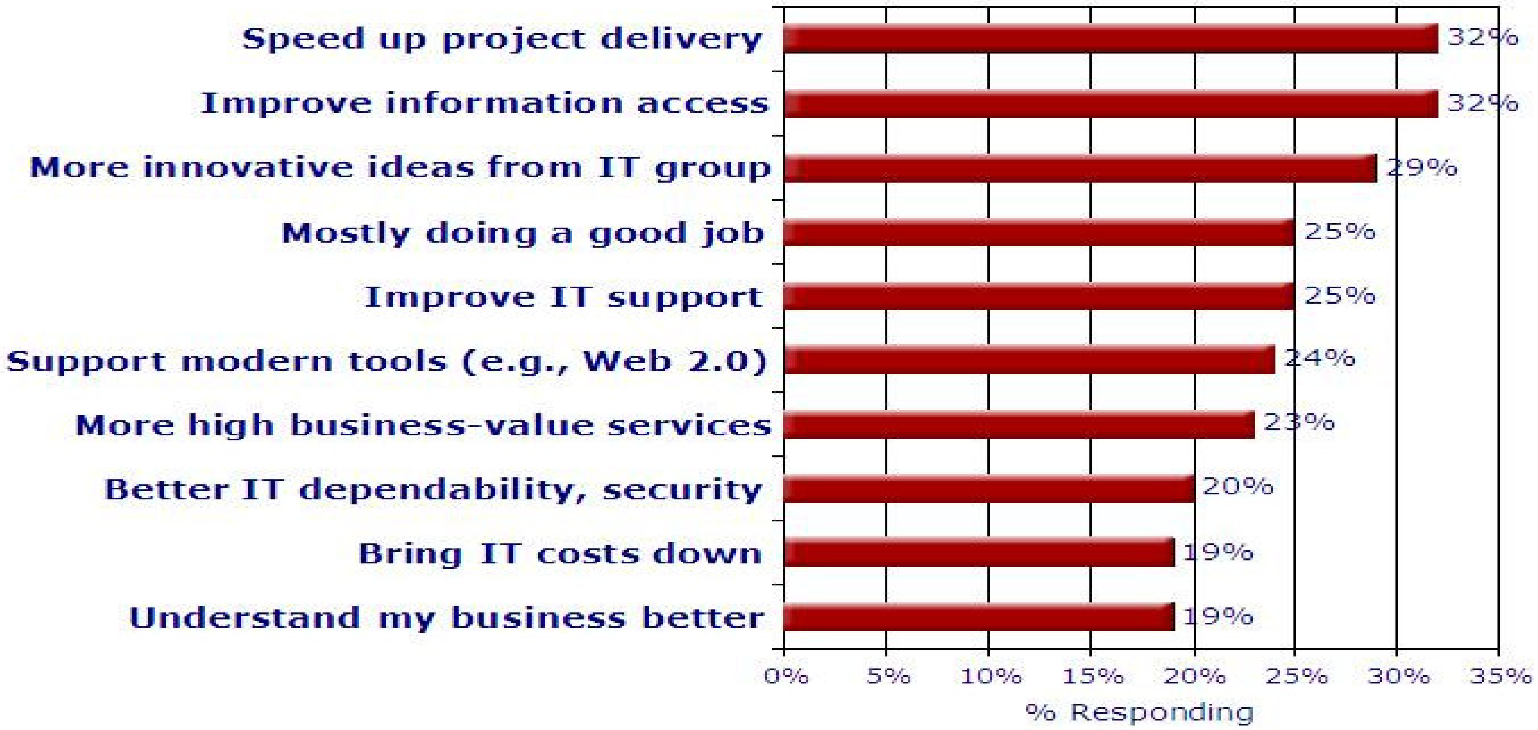
Why???

Not Just About More Dues to Fuel the Eclipse Jet

© Eclipse Aviation, All Rights Reserved

What Business Management Wants from IT (per IDC)

Q: Which of the following are the most important messages you would like to impart to your CIO/senior IT management?



Source: IDC QuickLook Survey, IDC's Enterprise Panel, January, 2008; LOB execs only (n=101); multiple responses allowed

What End Users *Need* From Platform Vendors

(lowest to highest difficulty/importance/immediacy)

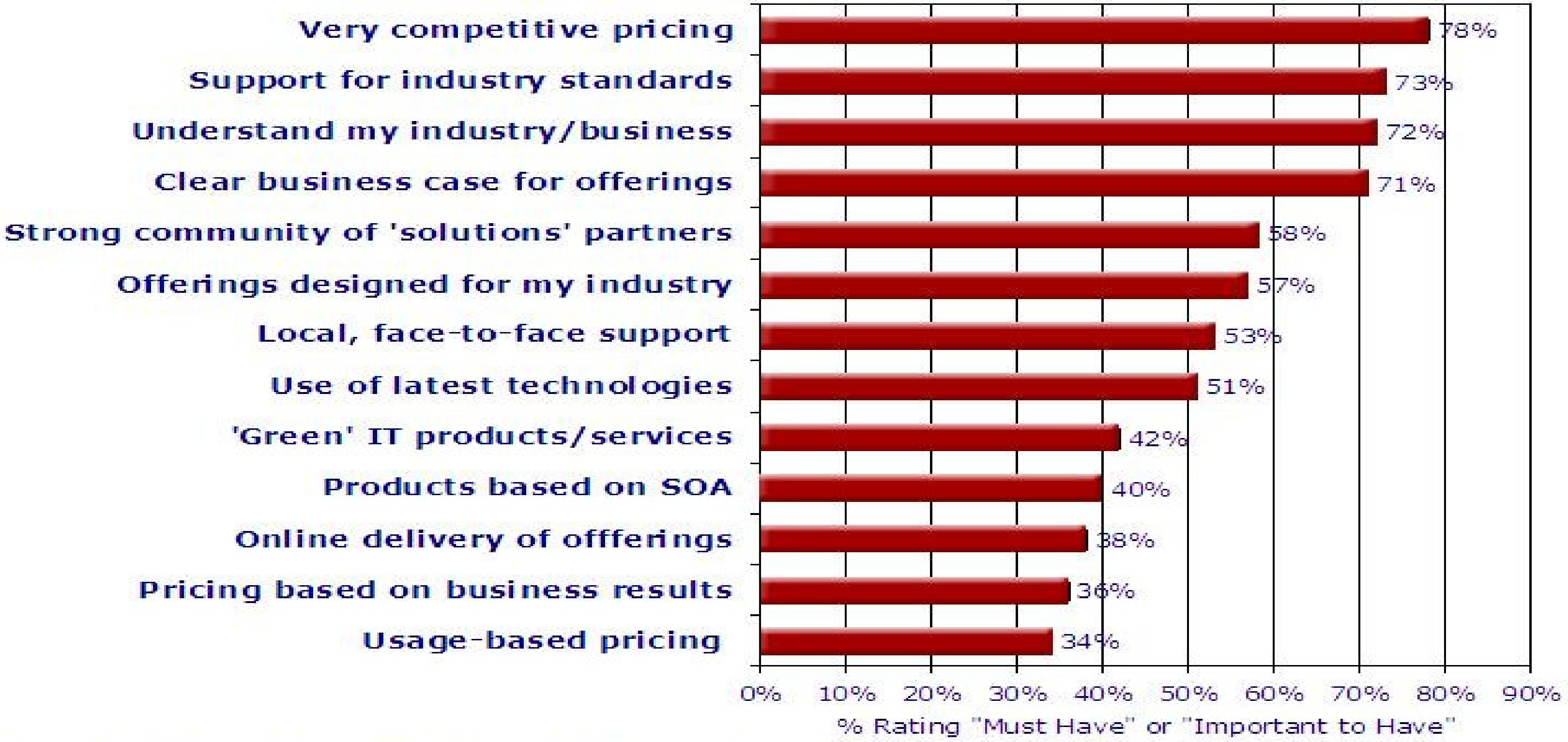
- Bug fixes for current operational problems
- Support for new underlying platform components (i.e., certification on new version of database)
- Specific new features to aid in solving specific issues
- General march of new features to broaden suite of problems that can be attacked with the platform

Traditional ways for customers to get what they need

Call tech support and yell	• Kind words • No action
File a bug report via Web	• No kind words. It *is* a web site. • *Usually:* no action • *Sometimes:* bug fix in next release, at vendors' convenience
Call product manager and yell	• Kind words • No action
Pay for user conference trip to talk directly to engineers	• Cheap canvas or nylon tote bag • Convention center food • (Geeky) kind words • No action
Call salesman and threaten to cancel order if bug not fixed	• Expensive lunch • (Slick) kind words • No action
Get onto "customer advisory board" and yell at CEO (only if you're a big enough manager at a big enough customer)	• Golf with CEO • Expensive leather tote bag • Free lunches *and* dinners • Kind words • No action

What End Users *Want* from Vendors (per IDC)

Q: How important is it for your IT vendors to offer the following, when considering doing business with them?



Source: IDC QuickLook Survey, IDC's Enterprise Panel, January, 2008; LOB and IT execs (n=245)

End users have tried to drive platform evolution before

- Avalanche Consortium
  - Large end-users sharing code amongst themselves (“software flea market”)
  - No commitment to joint R&D (“as is, where-is”) without requirement to contribute evolved code back to original contributors
  - No vendor involvement, meaning no R&D funding to turn cast-off customer-specific code into a “platform” (or even into an application)
- DrKW middleware consortium – OpenAdaptor
  - Died because vendors didn’t see what’s in it for them (originally a Tibco replacement)
  - Niche market too small for vendors
  - Many prospective customers already had legacy solutions
  - Especially hard for vendors when it sort-of-competed with proprietary vendors’ middleware stacks

Open source As End User Savior

- Mainstream reality: Open source increases odds of bug fixes and platform reliability because of multi-vendor environment
  - Vendor you're paying for support is first line of defense
  - Free community assistance from the mailing list
  - Can always find and pay third party to fix a bug
  - Open source licensing model can drive bug fixes back into main line code reducing repeated support costs on same bug
- But that's where it stops today
  - Forward-looking open source organizations can seize opportunity to drive customer involvement in all phases of creating and maintaining new platform (i.e., architecture to release)
  - Huge competitive exclusion in favor of open source: Mega-vendors remain focused on Looking Out for #1, to justify acquisitions and maintain price points/business models

Next Generation Platform Can Resolve Tension Between Verticality and Generality

(for both vendors and end users)

- Some people like Vertical Market Platforms
  - Customers like because
    - Solves unique cross-industry integration problems: industry-specific supply chain issues like trade clearing, bank wire transfers, check clearing, inter-airline reservations, etc.
  - Small/mid-sized specialist vendors like vertical markets
    - Offer some competitive exclusion against big guys because of their unique knowledge, agility, ability to get into a market first
  - Startups should like vertical markets but don't
    - Funding environment tough for focused vendors: VC's want addressable markets of 100% of Internet users or 100% of apps in 100% of Fortune 10,000 companies.
- Some people don't like Vertical Market Platforms
  - Mega-vendors see verticality as unappealing (unless you're talking about padding with services)
    - Addressable market far narrower than core business – even if core business is maturing and growth rate is slowing.
    - Only priced as giveaway to drive for "core" products.
    - Temptation to try to create proprietary "lock in."
- Customers don't inherently get "real" input into platform evolution

  (nothing structural means that they inherently have to have input)
  - Solution: true vendor/customer cooperation with neutral, predictable governance model.
  - Barrier to solution: vendors still have cultural bias over trying to have largest addressable market.
  - Barrier to solution: who could ever come up with a governance model for *that??!!*

Two things to talk about today.

- Platform shifts
- IT user involvement in Eclipse: a key imperative

…Of course, they're deeply interrelated.

How Eclipse Leverages Convergence of Platform and End Users

- Timing of the opportunity is right
  - MSFT having trouble driving platform component innovation into customer base
    - Vista
    - Business applications as platform (“Project Green”)
  - Rise of new programming technologies
    - Stackless Stack a la James Governor of Redmonk (Redmonk quote here)
    - PHP, in particular.
- Shared R&D model makes economics attractive for large vendors to build mix-and-match platforms, and for small vendors to drive verticalization off a common platform base.
- Governance process already in place for managing competing corporate interests
  - Neutral governance mechanism drives structural agility in the marketplace
  - Even big vendors can get agility by participating in Eclipse
  - Enforces loosely-coupled (stackless) architecture, countering tendency to bog down too deep.

Conclusions

- There is a new platform paradigm emerging
  - To be truly different, it should be designed and built with active customer-vendor participation
- Eclipse is uniquely positioned to lead creation of a new platform
  - Take advantage of platform fragmentation trend
  - Governance mechanism in place for broad participation in design and evoluion
- Archipelago of platform components is broadest addressable market of all

- Key challenge is to continue to broaden the platform while involving key strategic users on a large scale.

The Inevitable Shameless Plug

- Benchmark’s unique capability is to tie together the big three: technical issues, market positioning/competitive battlefield, financial aspects including marketing to Wall Street
  - Credibility with CXO-level management, including CEO, CFO
- “Corporate Advisory Services” Group – classic project consulting
  - Unique added Wall Street slant: credible ROI analysis, assessment of positive impact on stock price
  - First client already signed, work in progress for corporate repositioning
- Benchmark offers investment banking services for smaller deals
  - Focusing on Eclipse ecosystem