open source .com

RADICALLY SIMPLE IT or.. A Strategic Argument for Open Source in Business

David Upton

Chair in Operations Management, Oxford University

david@upton.com

http://david.upton.com

See also "Radically Simple IT," Harvard Business Review, 86(3): 118-124 March 2008

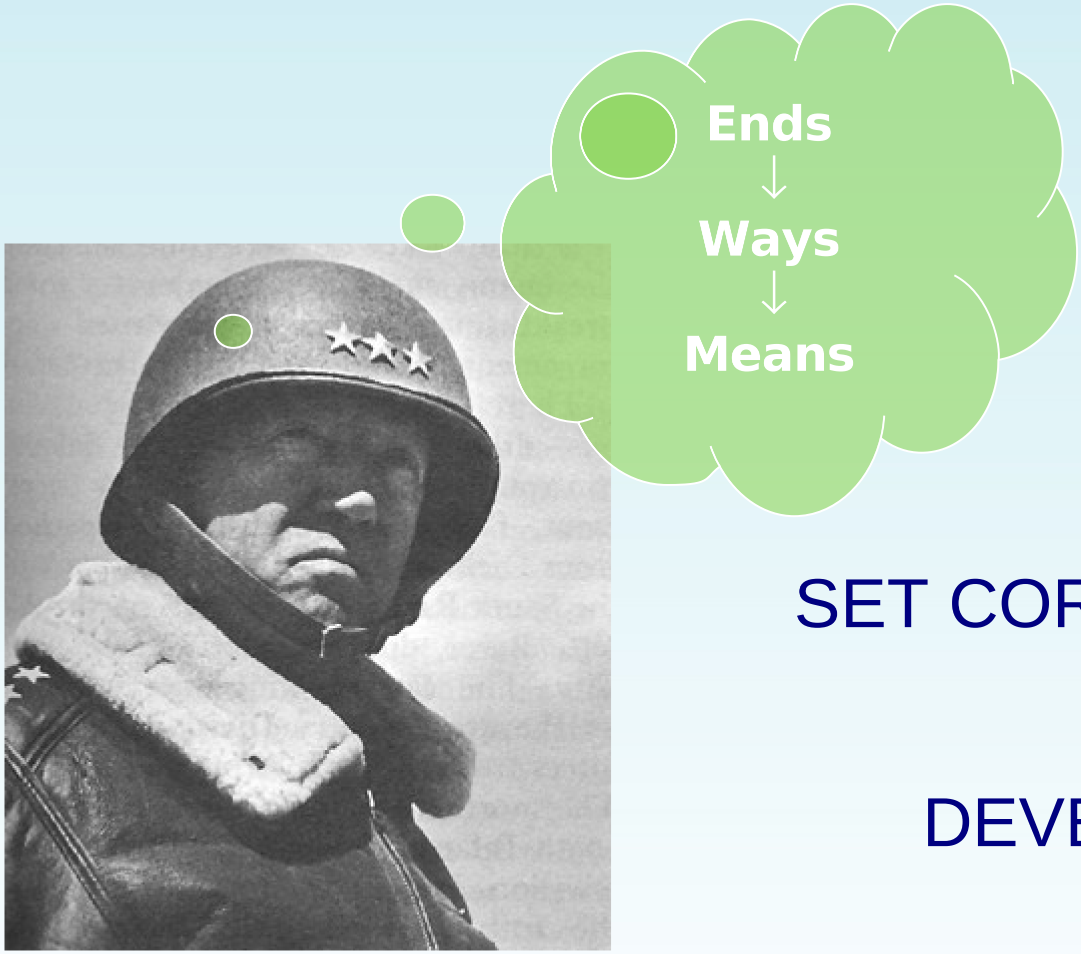
A Red Hat community service

A Path-based Approach: Open Source as a Strategic Advantage

1. My starting point. The Business Problem
   - Legacy Thinking in Strategic Planning
1. How this has messed up IT in the boardroom
2. How open source can provide a solution
3. An example

opensource.com

Strategic Planning: Traditional Approach



SET CORPORATE OBJECTIVES

⇩

DEVELOP STRATEGIES

⇩

ASSEMBLE RESOURCES

Traditional Approach

- Ends
  - Want an ROI of 20%
  - Want 10-15% earnings growth
  - Want to be a $2bn company by 2012
- Ways
  - Install new proprietary system company-wide
  - Install monolithic ERP system

Traditional Approach

- Means
  - Engage systems integrator for two-year WMS implementation
  - Buy the next great package for customer management
  - Re-engineer the business

Problems with this approach:

- ENDS
  - Too short term (~5 years)
    - implies ***buying*** rather than ***building***
    - CEO tenure (median: 5.5 years ptc’s), analysts
  - Overly Quantified
    - Need to measure *but*
    - tends to push out other goals
  - Episodic: “***This too shall pass***” – big problem

The tail on the dog problem

- Strategic planning system is the dog
- … the tail was supposed to wag

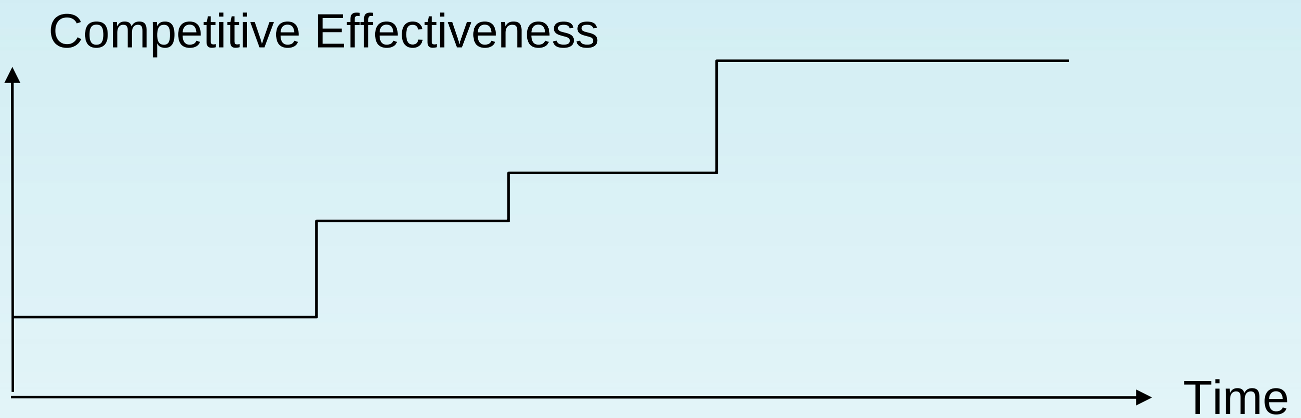
BUT – more and more, value is created in the tail: in IT and operations

Two models of performance improvement

- Strategic-leap versus incremental-approach
- These are archetypes
- Neither exists alone in the wild
- Mark ends of a spectrum

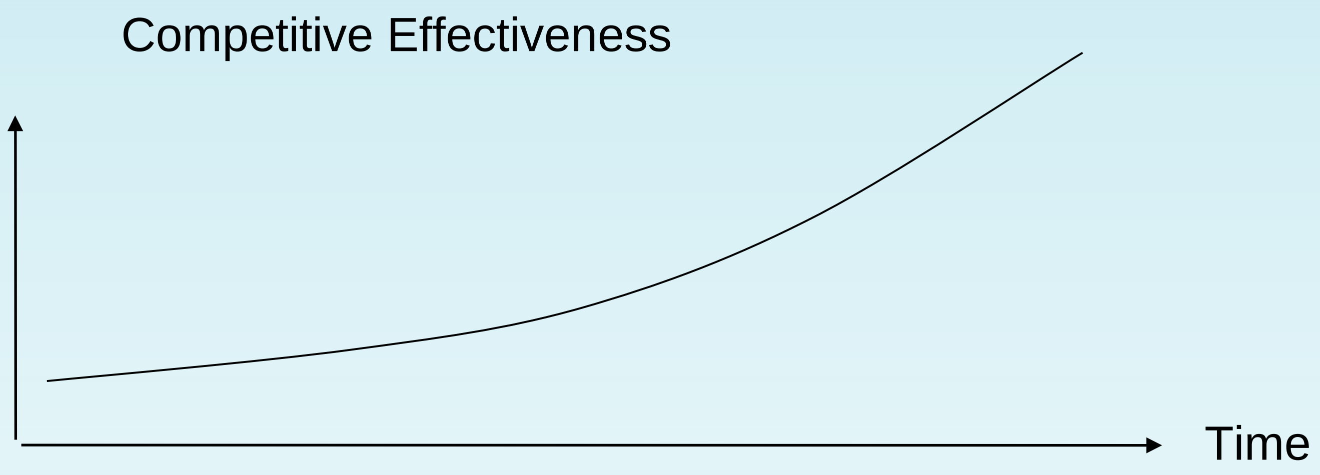
open
source
.com

The Strategic Leap Approach to Performance Improvement



Examples
- Install monolithic IT system
- Introduce blockbuster product
- Attack a new market (or niche)
- Integrate vertically
- Exploit a new technology (product or process)
- Merger/acquisition/strategic alliance
- “Strategic”

The “incremental improvement” approach

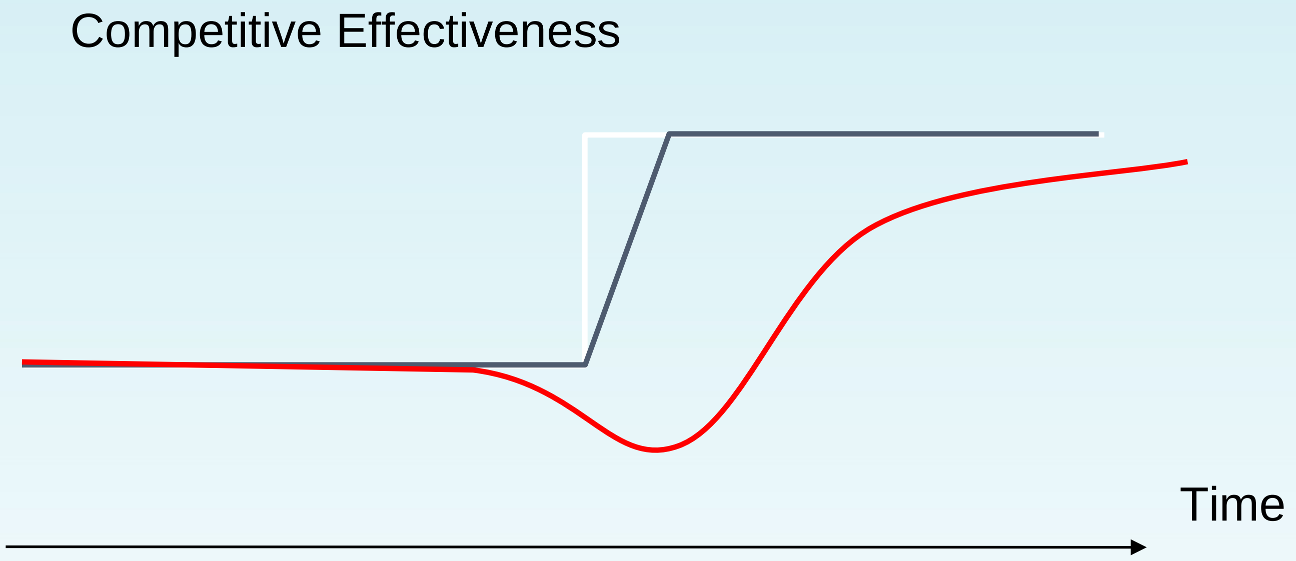


Examples
- Experimentation with on-line information for customers
- New system features
- Feature enhancement
- Inventory reduction
- Reduce set-up/throughput times
- Faster system development
- “Boring”

Strategic Leap Approach to IT:

- Requires only periodic expertise
  - Consultants with episodic (not partnership) engagements
- Each step can have a major financial impact
  - financial experts required: subject to financial constraints
  - timing is critical
- Creates high personal visibility
  - Win big, lose big

Murphy’s Law of Strategic Leaps



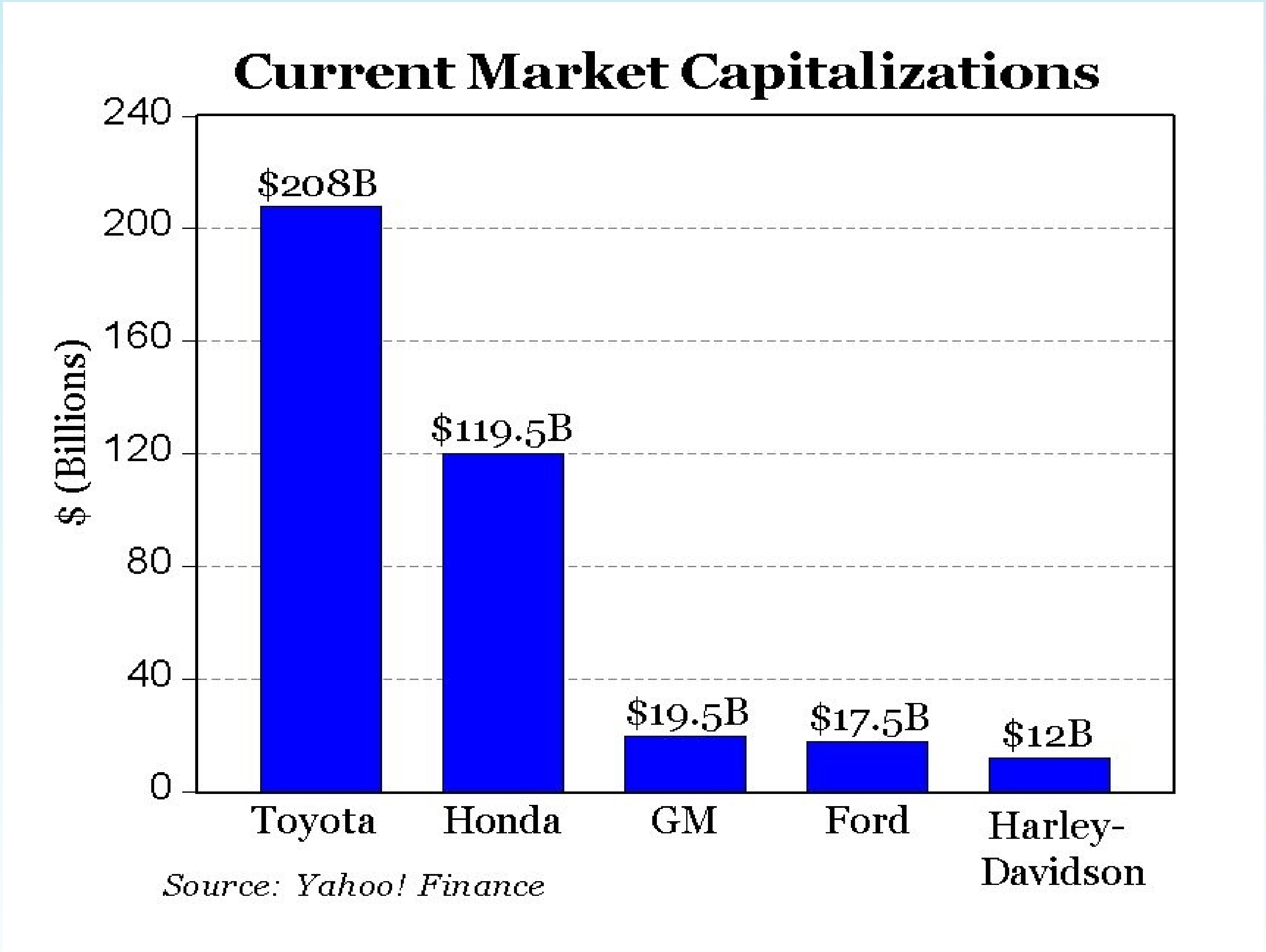
An Old Quote: Japanese successes

"Japanese successes in the auto, semiconductor and consumer electronics markets are primarily due to a determined focus on short-term, incremental gains"

Kenichi Ohmai

Wall Street Journal January 18, 1982

opensource.com



The world continues in that direction

Which approach will you ***favor*** in a world where...

- Technologies, markets and objectives change rapidly
- Forecasts are unreliable (in part because the business changes fundamentally, or more severe network effects)
- New competitors are entering the arena
  - Not the usual suspects
- You learn - and change business goals - <u>as you go along</u>

Our context: CEOs and IT

- Compelling business advantages seen from tireless continuous improvement in manufacturing
  - Creator of Advantage; Specific principles drive the improvement
  - It’s not convergent . There is no endpoint.
- … but where do CEOs see IT?
  - Old Caretaker/installer model
  - Evaluation: 4 stages of enlightenment
    1. Risk avoidance
    2. Cost minimization
    3. Revenue Generation
    4. Option value

The Liquid Concrete Phenomenon: Designing for Improvement

- Big, complex, monolithic System installed
  - The old auto industry model
- Hard to improve
  - Or, improvement rate limited by external suppliers absent a true partnership
- Improvable Systems are: *(think Open Source)*
  - **Modular**: Can experiment locally without global consequences
  - **Accessible**: <u>Able</u> to make change (no obscure skills, languages, protocols)
  - **Inclusive**: (People using it are also involved in the design of it).
    - Lose the concept of ‘users’ vs. ‘IT people’

Open source can unlock the potential

- *Designed* for improvement
- Can improve rapidly (not just at the rate of the software vendor)
- Modularity is built in as part of the development processes. Real modularity. So it’s modular.
- Can live with many technologies
    - Co-exists, plays nicely with others
- Vast network of knowledge
- Opportunity for relentless innovation
- A way out of the strategic leap trap

Example: Shinsei Bank†

- Rapid Business Development through and Open Source
  - Our problem: the plane is flying: change the engine, and improve it while in the air
  - Not just 'Extreme Programming', or 'Customer Satisfaction'
    - These are Business Experiments – not IT/customer experiments
    - Building competitive advantage is not at the internal customer level
    - Not everyone wants to be involved in IT development

† "Radically Simple IT," *Harvard Business Review,* 86(3): 118-124 March 2008 (with Bradley Staats).

Shinsei Bank

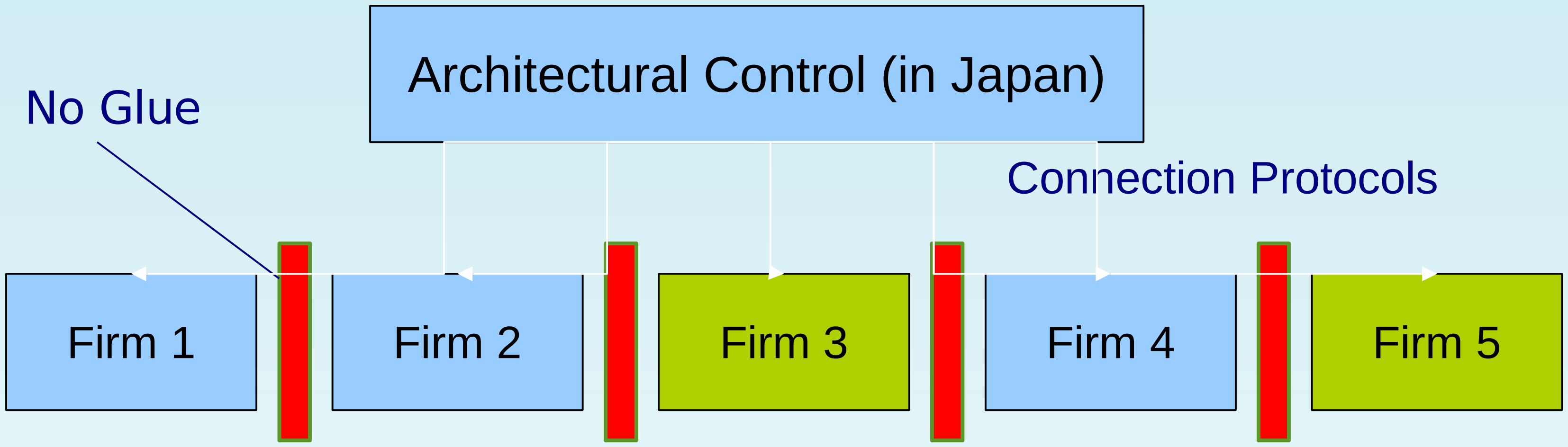
- Rescued from the archaic ashes of LTCB
- Built new banking system (rather than packaged software)
- Mix of open source and off-the-shelf technologies
- Design for **improvement** not immediate **functionality**
- ¼ of the leadtime, 10% of the cost of basic packaged solution
- Reached #1 customer service in Japan in 2 years

Shinsei Principles[†]

- Rapid, incremental improvement rather than big-bang
- Use technology to solve human problems
    - It’s not the technology; it’s not IT
    - Replicate old system as subset of new system
    - Double sided screens for tellers
    - Doesn’t exist in standard packages
- Outsourcing strategy

[†] “Radically Simple IT,” *Harvard Business Review,* 86(3): 118-124 March 2008 (with Bradley Staats).

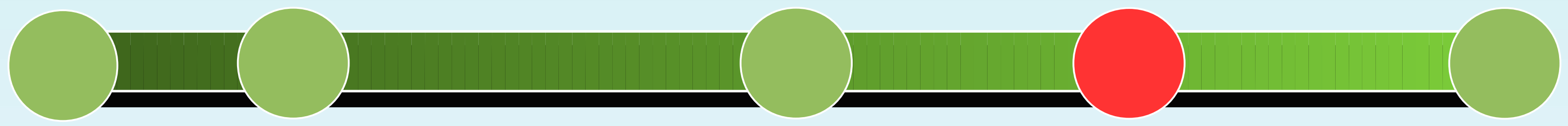
Development Principles at Shinsei



- Maintains control at Shinsei: no individual firm can replicate
- Avoid hold-up by outsourcer
- Allows rapid expansion of features and services
- Incremental rather than big-bang approach
- Provides proprietary advantage
  - Need to use open code is not a risk: tennis racket issue.
  - Capability development; China?

Balance: A Spectrum of Strategic Choice

Complete Package | 'Best of Breed' Solution | Custom Assembly /Open Source | Custom Code



e.g. ERP with APIs

How customized are you?

- ⟶ +

Potential to re-invent the wheel

- ⟶ +

Maintenance required

- ⟶ +

Internal IS skills required

- ⟶ +

Potential to generate competitive advantage?

? ⟷ ?